Bitcoin Privacy - A Survey on Mixing Techniques[⋆,⋆⋆]

Simin Ghesmati[a,b], Walid Fdhila[a] and Edgar Weippl[a,c]

[a]SBA Research
[b]Vienna university of technology
[c]University of Vienna

ARTICLE INFO

Keywords:
Blockchain
Privacy
Mixing
Tumbling
Bitcoin
Distributed ledger
Anonymity
Deanonymization WGM
BEC

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 rapidly proved to be a double-edged sword that can put user privacy at risk. Unlike 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 the financial privacy of users and reveal their actual identities using advanced heuristics and techniques to identify eventual links between transactions, senders, receivers, and consumed services (e.g., online purchases). In this regard, a multitude of approaches has been proposed in an attempt to overcome financial transparency and enhance user anonymity. These techniques range from using mixing services to off-chain transactions and address different privacy issues. In this survey, we particularly focus on comparing and evaluating mixing techniques in the Bitcoin blockchain, present their limitations, and highlight the new challenges.

1. Introduction

In recent years, there has been an increasing interest in blockchain technology. The first design emerged by by Satoshi Nakamoto [55] in late 2008. the number of use cases and applications of blockchain technology beyond cryptocurrencies has increased tremendously. Examples of that are supply chains, industry 4.0, healthcare, and identity management. In the literature, several studies have been focusing on Bitcoin privacy and were able to analyze the chain of interactions between users, identify relationships and reveal user real identities [59, 52, 36, 43]. This, in turn, urged research in both academia and industry to find solutions and methods to overcome such privacy leaks. This leads to a plethora of either (i) new separate project proposals that have inherent privacy such as Zcash, Monero, and Dash, or (ii) privacy improvement proposals to Bitcoin. Both approaches were respectively categorized as (i) built-in data privacy and (ii) add-on data privacy [65]. In this paper, we solely consider privacy methods proposed for Bitcoin, and more specifically mixing-based techniques [35]. In particular, we aim to evaluate and compare existing mixing approaches by analyzing their privacy, security and, efficiency and studying their applicability to the Bitcoin blockchain. The research questions investigated in this study are as follows.

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

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

(RQ3) How do existing mixing techniques compare in terms of efficiency e.g., interaction with input users, interaction with the recipient, Bitcoin compatibility, sending the coins directly to the recipient, number of transactions, minimum mixing time (Block)?

The contribution of the paper is two-fold, a systematic review of the literature, and the evaluation of mixing techniques. We introduces the main concepts in section 2 and outline several de-anonymization attacks in section 2.1. In section 3 we discuss mixing techniques, while in section 4, we evaluate them according to predefined criteria. We conclude the work and summarize the challenges in section 6.

2. Background

In 2008, Satoshi Nakamoto [55] published the Bitcoin white paper as 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 [2]. In the following, we use the term bitcoin to refer to the cryptocurrency, and Bitcoin to refer to the underlying blockchain. Bitcoin consists of a sequence of chained blocks, each identified by a block header, and refers to a previous block (the block parent). This forms a chain of blocks that ties to the first block; i.e., the "genesis block" [2]. Transactions are recorded in a distributed, permanent, and verifiable manner. The ledger is immutable, and no data within can be edited or deleted.

Address. Asymmetric cryptography is applied in Bitcoin in which public and private keys are used. The addresses are the hash of public keys and the users are able to unlock the coins associated in that address by the signature which is computed by the corresponding private key. Script hash is another type of address that allows transactions to be sent to a script hash [2]. Note that Bitcoin uses Elliptic Curve Digital Signature Algorithm (ECDSA) [11] to generate sig-

⋆This document is the results of the research project funded by COMET.

sghesmati@sba-research.org (S. Ghesmati); wfdhila@sba-research.org (W. Fdhila); eweippl@sba-research.org (E. Weippl)
ORCID(s):

natures.

Transaction. In Bitcoin, transaction is transferring the value associated with an address (input) to another address (output) [14]. An unspent transaction output (UTXO) which belongs to a sender is used as an input of the transaction and the recipient address is considered as an output. A new so-called "change address" is created and used as an output to get the remainder of the coins to the sender. A transaction contains the following attributes: transaction id, version, inputs, outputs, and nLockTime (a parameter that specifies the time before which a transaction cannot be accepted into a block). Each input refers to an output of a previous transaction. To avoid double-spending [79], Bitcoin stores a list of unspent transaction outputs (UTXOs) [81]. Once an output is spent, it is automatically removed from the list.

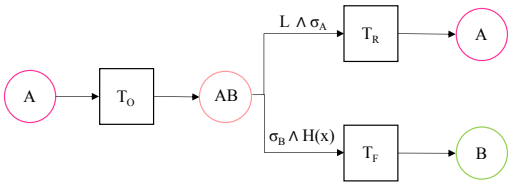
Transaction fee. To overcome flooding attacks [82], Bitcoin requires paying a transaction fee to miners to be included in a block. It is calculated by subtracting the sum of the input values from the sum of the output values [83]. Note that large transactions often require higher fees to be confirmed [38].

Transaction scripts. Bitcoin script [76] is a Forth-like [80] 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 [76]. The Bitcoin script was designed to be simple and executable in most of the hardware while requiring small processing [2]. Transactions use scripts to specify the conditions under which the coins can be spent [38]. A vast majority of transactions in Bitcoin employ Pay-to-Public-Key-Hash (P2PKH) script [9]. Other forms of scripts can enable more complex conditions for spending the coins, e.g., Pay-to-Script-Hash (P2SH) [10] and Multisig [73, 8].

Timelock transaction. Timelock transaction [74] restricts spending the coins until the specified time, which can be used for refund. The transaction will be valid at the time set in a field called nLockTime which is one of the fields in every Bitcoin transaction.

Hashlock transaction. Hashlock transactions [75] is locked by a hash and can be spent by providing a pre-image of the hash. 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, and consequently, all the transactions that were locked with this hash can be redeemed [26]. To prevent redeeming the coins by other users in the blockchain, Hashlock transactions are locked by both signature and the pre-image of the hash.

Hash time locked contracts (HTLC). HTLC [77] is a script that employs both Hashlock [75] and timelock [74] transactions. The output is locked by a hash and if the recipient is unable to unlock it in a specific time, the coins are returned to the sender. Figure 1 illustrates an HTLC transaction, where Bob can fulfill the transaction by providing pre-image (x) as well as his signature, and Alice can get the refund via T_R after the locktime.



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)

2.1. Bitcoin Privacy Threats

The public availability property of the blockchain introduce privacy issue for blockchain users. The previous research indicated that heuristics along with information from other resources such as forums, online shops, etc, can effectively cluster the transactions and identify the users. In the following subsections, we will review the common heuristics used to de-anonymize blockchain users, and then provide recent research into the known attacks on the Bitcoin blockchain in which anonymity has been compromised.

2.2. De-anonymization in Bitcoin

There is a degree of uncertainty around the terminology of "Anonymity" in the blockchain area. Considering the anonymity definition proposed by [58]: "The subject is not identifiable within a set of subjects, the anonymity set". Bitcoin is not fully anonymous, and a multitude of studies [59, 52, 36, 22, 43] have proved possible deanonymization by mapping Bitcoin addresses to their real entities. In the following, we present some of the most prominent techniques [15, 52].

Common input ownership. When a transaction has multiple inputs, each of the inputs should be signed by its associated signature. It is assumed that all the inputs in a transaction belong to the same user since it is not common that multiple users join to create a transaction [15]. Considering this, the common input ownership heuristic considers all the inputs of a transaction to one user. According to [36], the heuristic is able to identify almost 69% of the addresses stored in the clients' wallets.

Detecting change addresses. When the sum of a transaction inputs is larger than the sum of its outputs, a first use address called change address is created which returns the remainder of the coins to the sender [24]. This heuristic considers that the change address is controlled by the owner of the input addresses [15]. The following is a list of common heuristics that are employed to identify change addresses.

- Address reuse: A fresh address output can be a change address [52].
- Script types: The output with a similar script, if all the inputs have similar scripts (P2PKH or P2SH) can be a change address [44].

- Same input and output: An input address that is also an output can be a change address [44].
- Unnecessary input: An output that has a smaller amount than all the inputs can be a change address [44].
- Consumer heuristic: The output that is later used in the batching transactions is not likely to be a change address [78].
- Round numbers: The non-round number output value can reveal a change address [78].
- Wallet fingerprinting: Wallets create transactions in a different manner, which can be used to reveal change addresses [78].

Transaction graphs. A transaction graph can effectively demonstrate bitcoins flow between users. In this graph, Bitcoin addresses represent the graph nodes, and transactions linking input addresses to output addresses are the graph edges [25]. As a transaction input is related to an output of a previous transaction, it becomes possible to identify relationships between the transactions [53]. Moreover, most of the transactions have change addresses which are under the control of the input entities (pseudonymous user) and, therefore, can be linked to the same entity in the transaction graph. By interacting with services that require a user's real identity, a user public key can be linked to her identity, and consequently, the relationship between her previous transactions can be obtained in the transaction graph. Therefore, as also stated in [39], using a fresh address for every transaction can not prevent a privacy leakage. Another issue is that the sender knows the recipient address and consequently can identify further transactions performed by the recipient and find with whom she transacted.

Linking similar addresses / Address reuse. The addresses that are reused in the transactions can be linked together in the blockchain belonging to the same entity.

Side-channel attacks. Side-channel attacks [15] such as time correlation, amount correlation, and network-layer [13] information can reveal the transactions and users.

2.3. Related works on De-anonymization

In recent years, several works have addressed user de-anonymization in blockchain using the techniques in previous subsection. Meiklejohn et al. [52] clustered Bitcoin wallets based on evidence of shared authority and then utilized re-identification attacks to classify the users of the clusters. They have concluded that the information collected by Bitcoin businesses such as exchanges along with the ability to label monetary flows to those businesses alleviate the willingness of using Bitcoin for illicit activities. Reid and Harrigan [59] analyzed the 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 has the potential to contribute to de-anonymize Bitcoin's users. They employed flow and temporal analysis in the research and identified more than 60% of the users in the visualization and found their relationships. Harrigan and Fretter [36] explored the reasons for the effectiveness of simple heuristics in Bitcoin, they considered the impact of address reuse, avoidable merging, superclusters with high centrality and the growth of address clusters. Ermilov et al. [22] utilized off-chain information as votes for address separation and considered it together with blockchain information in their clustering model. They applied blockchain-based heuristics such as common input ownership, detecting the change address, along with off-chain information for clustering. Jourdan et al. [43] defined features to classify entities from a graph neighborhood perspective, as well as centrality and temporal features in the Bitcoin blockchain, they classified addresses into exchanges, gambling services, general services, and the darknet categories. The results of the above-mentioned research heighten the necessity to enhance blockchain privacy through powerful techniques. In the next section, we will deep dive into the existing mixing techniques.

2.4. Research methodology

In this study, we have followed common guidelines for research synthesis comprising (i) the research questions identification, (ii) the literature search and selection, 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 identify unpublished papers. Moreover, we conducted a direct search on "Github" for existing implementations of mixing techniques. In total, we obtained 869 research papers. All the titles were read, and papers with no relevant content were dropped. Next, all abstracts were read and papers with no relevant content according to the abstracts were also removed. Only papers published between 2009 and 2020 were considered in our research. Duplicates, and items not focusing on mixing methods, or not related to Bitcoin were also excluded. The literature for the systematization is selected based on the following criteria which are inspired by [1]: (i) **Merit**. The technique is unique and among the first to explore privacy solutions. (ii) **Scope**. The technique is compatible with the Bitcoin blockchain at least via soft-fork, and (iii) **Disruption**. The paper technique is a novel area that the community is investigating. Our second contribution is the evaluation of the selected mixing techniques using the criteria defined in the following paragraphs. The mixing techniques have been evaluated over three main categories: Security, privacy, and efficiency. Several criteria have been proposed in the literature, and our selected criteria inspired by commonly used criteria in the recent research.

3. Mixing techniques

Transactions consist of multiple inputs and outputs which can be traced using sophisticated analytical tools. The mixing 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 addresses as well as the value of the transaction can also be hidden using enhanced techniques. Various mixing techniques exist and differ in terms of privacy, security, and efficiency. These techniques can be categorized into centralized mixers, atomic swap, CoinJoin based and threshold signatures which will be discussed in the following subsections. But first, we outline the research methodology we adopted for the identification, selection and synthesis of the research items included in this study.

3.1. centralized mixers

In this subsection, we investigate the methods that consider a centralized party to mix the coins, where in most cases the recipient sends the coins to a central mixer, and then the mixer forwards the coins to the recipient. Ruffing et al. [62] pointed out that the central mixer can be a single point of failure and the target of DoS attacks.

3.1.1. Mixing websites

The mixing idea was initially proposed by Chaum [19] in 1981 to ensure anonymous email communication without relying on a universal trusted authority. Similar techniques have been lately employed to address anonymity on the blockchain. The latter employ mix 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. The latter receives senders' coins in equal amounts, mix 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'. In most mixing websites, users are asked to fill out a form in which they fill the recipient's address and select preferred mixing delay. Subsequently, a fresh address is generated by the mixer to receive the coins from the sender. The fresh address is demonstrated along with the mixing fee, transaction fee, and the conditions to the user.

Issues. The mixer can steal the coins and never send them to the recipient. The mixer is able to store the user's data. The mixer is also able to link the sender and recipient. The users are able to mix different amounts instead of fixed denominations, and it has the potential to compromise the mixing transaction in the blockchain. Reasonable anonymity set requires a high waiting delay. The mixing fees are high in practice (1% to 5% of the transaction amount). Moreover, [15] discussed that fixed mixing fees can reveal information in mixing transactions when the mixer always deducts a specific percentage of the coins as a fee.

3.1.2. MixCoin

MixCoin [15] was proposed by Bonneau et al. as a Bitcoin mixer to prevent theft in mixing services using the mixer signature as a warranty in case it acts maliciously. According to the protocol, the mixer has to sign senders' 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 recipients, the sender has the possibility to publish the warranty. This way, anyone can verify that the mixer acted maliciously, causing negative impacts on its reputation. In MixCoin, mixing fees is all or nothing, as the constant mixing fee can reveal mixing transactions in sequential mixing. To improve anonymity, mixing transactions have a standard chunk size among all the mixers to make uniform transactions. It is also demonstrated that chaining multiple mixing together can provide strong anonymity.

Issues. Although theft can be detected, it is not prevented. Such as mixing websites, the mixer is able to store user's data as well as linking senders and recipients. Additionally, using multiple mixing achieves a reasonable anonymity set leads to extra fees and delays.

3.1.3. BlindCoin

BlindCoin [68] adds Blind signatures to Mixcoin. Blind signature was proposed by Chaum [18] to sign a message without revealing the content to the signer. Using blind signature in Blind coin, hide the relationship between input and output addresses from the mixer itself, where the mixer puts her warranty by blindly signing the recipient's address. Later, the sender anonymously submits the unblinded recipient's address to the mixer using a new identity. The mixer will send the coins to the recipient's address as it can see its signature on the recipient's address.

Issues. Such as MixCoin, theft can be detected, but it is not prevented. The protocol needs at least four Bitcoin blocks, two blocks to publish public log messages, and two blocks to perform the transactions which leads to extra fees and delays.

3.1.4. LockMix

LockMix [3] is a central mixer that improves BlindCoin [68] by preventing the mixer from stealing the coins using multi-signature. 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 along with 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 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 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 will lose their coins if they misbehave, which is a lose-lose scenario.
Issues. It is possible that the sender and the mixer lose their coins, in the case of a lose-lose scenario. The protocol needs at least six blocks to be confirmed, two blocks to publish public log messages and four blocks to perform the transactions which leads to high fees and delays. Too many parameters should be set to run the protocol.

3.1.5. Obscuro

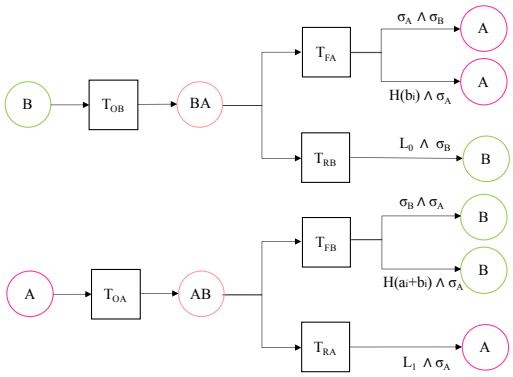
Obscuro [67] is a central mixer that employs a trusted execution environment (TEE). To run the protocol, the mixer generates the key in TEE and publishes the public key and Bitcoin address. All the users send their coins to a single address of the mixer and publish their transactions in the network. Encrypted recipient addresses along with a transaction refund script are included in the transaction to get the coins back 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 to perform the mixing transaction. Specifying the minimum participants ensures users about the mixing set size before they participate in the protocol.
Issues. Various amounts in the transactions have the potential to de-anonymize the transactions unless the users send a fixed denomination to the mixer address. It has also not been specified how to incentivize the mix service provider as it is mentioned in the paper that the mixing fee can be burned or deposited to charities. The anonymity set is restricted by the Bitcoin transaction size. Obscuro's transactions have a specific characteristic in transaction graph, a couple of transactions that send the coins to one single address containing encrypted recipient's addresses in OP_ RETURN field, and a bunch of inputs from that single address to the recipients which has the potential to compromise plausible deniability [83] in which users cannot deny participating in mixing transactions.

3.2. Atomic swaps

An atomic swap is a decentralized technique that enables users exchange their coins without relying on a centralized intermediary.

3.2.1. FairExchange

This protocol proposed by Barber et al. [4] enables two users to swap their coins. As such, Alice sends the coins to Bob's recipient address and Bob does this to Alice's recipient address. In the first step, Alice and Bob create two key pairs to use in different transactions, then they generate secret values a_i and b_i and engage in a cut and choose protocol



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

to provide $H(a_i+b_i)$ and $H(b_i)$ that will be included in offer transactions.

As illustrated in figure 2, in order to offer the coins, Bob creates T_{OB} that can be redeemed by either Alice and Bob signatures or Alice signature and b_i. Bob also creates a transaction refund T_{RB} to ensure that he can get back his coin if the protocol is aborted. He waits for Alice to sign T_{RB} and publish T_{OB} and T_{RB}. Alice does the same, where T_{OA} can be redeemed by both Bob's and Alice's signatures or Bob's signatures and $a_i + b_i$. Bob fulfills Alice's transaction by providing his signature and a_i+b_i. Alice then subtracts a_i and obtains b_i to fulfill Bob's transaction.
Issues. The protocol consists of four transactions which lead to additional costs and delays. The protocol has special output scripts that can be distinguished from other transactions in the blockchain [15, 54]. Both fulfill transactions contain similar values and the anonymity set is restricted to the transaction with those values [54]. On one hand, the protocol supports two-party mixing which requires many transactions to achieve the desired anonymity set [38], on the other hand, the timelock transactions curb the anonymity set [54].

3.2.2. Xim

. Xim [7] proposes a novel approach to find a user to perform FairExchange transactions (3.2.1). 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 to contact the partners (e.g. Onion address or Bulletin board). She can then be contacted by several participants and choose one for partnership. The selected participant should then pay τ coin to the miner (to prevent Sybil and DoS attacks) to start creating the transactions. Otherwise, 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.
Issues. FairExchange issues apply in this protocol. It takes almost hours to find partners and perform the transactions.

3.2.3. *CoinSwap*

CoinSwap [51] prevents stealing the coins by the intermediary. 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 3). 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}. To prevent stealing the coins 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 misbehaving 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) and 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) and sends it to Carol to sign and publish 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 get back her coins from T_{0R}, while Bob can redeem the coins from hashlock simultaneously. Carol should use a different identifier for the transactions between Carol and Bob [54]. The key point in CoinSwap is that the transactions are in two different paths which makes it possible to have them in two different blockchains that support timelock and hashlock transactions (e.g. Bitcoin and LiteCoin) [28].

Issues. The mixer is able to store the user's data. The mixer can link the sender and the recipient. The protocol needs to wait for the confirmation of the initial transactions to continue the protocol. It requires four transactions which in turn lead to additional costs and delays. The participants should create eight transactions in total. Möser and Böhme [54] mentioned that due to the use of timelock transactions, the anonymity set is restricted to multi-signature transactions within a specific time frame. The same value in transaction T_0 and T_1 has the potential to link these transactions in the blockchain. If the protocol goes to hashlock transactions, it reveals the link between two transactions. CoinSwap has a malleability problem without Segwit [47], as the transaction id of the initial transaction can be malleated by a malicious user, as a result, the hashlock and time lock transactions can not be redeemed because they have signatures on inputs that do not exist [28]. The liveliness of the network/ Internet and non-censorship by the miners are required to ensure getting back the coins from timelock transactions.

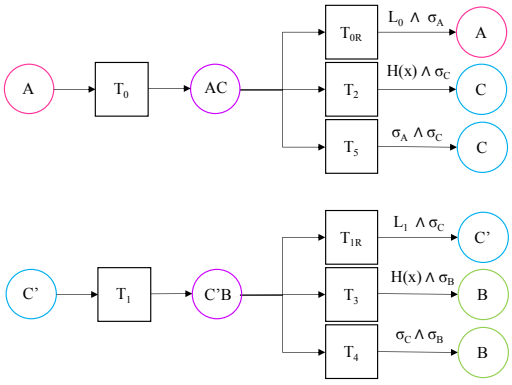


Figure 3: CoinSwap

3.2.4. *New CoinSwap*

Since CoinSwap was proposed before the advent of check lock time verify (CLTV) [66] or check sequence verify (CSV) [16] Opcodes, it used nlocktime feature to create refund transactions. The hashlock transactions were also kept as backouts and should not be published. New CoinSwap [28] uses CLTV to create hash time locked transactions which needs creating six transactions in total by the participants as well as using segwit to prevent malleability and as a result, the protocol is theft-resistance. In this protocol, Alice gets 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 then can use mixed coins to send the real recipient.

Issues. New CoinSwap comes up to malleability and prevents theft and needs fewer transactions in comparison with CoinSwap, however other issues in CoinSwap can be considered in the New CoinSwap protocol.

3.2.5. *PaySwap / Design for a CoinSwap Implementation*

PaySwap [6] improves over CoinSwap. PaySwap utilizes two-party ECDSA to create 2-of-2 multi-signature addresses. These kinds of addresses are similar to regular single-signature addresses. Alice1 pays Bob1 and Bob2 pays Alice2 by CoinSwap transactions. Such as Joinmarket wallet [5]) , Alice can be a market taker and Bob can be a market maker. Alice pays a fee to Bob as a market taker. To prevent the amount correlation in which an attacker can search the transaction values and find Alice2. PaySwap proposed multi-transactions in which Alice sends the coins to Bob and receives three transactions with different amounts with a total of those coins. To address internal traceability in which Bob can trace Alice coins flow, Alice can route her coins through many market takers (Bob, Carol, and Dave). In this route, a market taker only knows the previous and the next

addresses. Alice will inform every market taker about the CoinSwap incoming address 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 to enhance the privacy of the transactions. The combination of CoinSwap with PayJoin [32] is also considered as a possible solution to break multi-input transactions heuristics in which Bob's input can be added to Alice's inputs in the transaction.

Issues. The protocol contains four transactions that increase fees. PaySwap multi-transactions increase fees and consequently delays, e.g. getting back the coins via three PaySwap multi-transactions needs eight bitcoin transactions (two from Alice to Bob, six from Bob to Alice three different addresses).

3.2.6. Blindly signed contract (BSC)

The protocol [39] is proposed in two schemes; (i) on-blockchain and (ii) off-blockchain. The former uses untrusted intermediaries while the latter utilizes micro-payment channel networks where transactions are performed off-chain and their confirmation on-chain. The on-chain scheme consists of two FairExchange transactions, which means four transactions should be submitted on the blockchain, where Alice sends the coins to Bob via an intermediary (Carol). To initiate the protocol, Carol posts public parameters including blind signatures parameters, transaction fee, reward value (w is considered as a mixing fee in the protocol), 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 beforehand. 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 $TF_{A->C}$ by providing a blind signature ($\overline{\sigma}$) on ($\overline{sn}$). Alice can obtain σ and send the voucher to Bob to fulfill the transaction $TF_{C->B}$ that sends one coin from Carol to Bob.

Issues. Soft-fork is required to add the blind signatures to the Bitcoin blockchain. The protocol consists of four transactions confirmed in three blocks (at least thirty minutes) which leads to additional costs and delays. The mixing fee should be set more than the transaction fee to incentivize the mixer to prevent aborting the protocol while the transaction fee gets paid to the mixer.

3.2.7. TumbleBit

TumbleBit [38] uses the puzzle solution to provide privacy in the case of untrusted central tumblers. The protocol needs four transactions confirmed in two blocks. As a high-level description, the tumbler gives the coins to Bob if he solves a puzzle and then the tumbler sells a solution to the puzzle provided by Alice to the same amount of coins. To run the protocol, the tumbler creates an Rivest–Shamir–Adleman (RSA) [60] puzzle for the solution ϵ, and 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 4, 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 (which we will explain below), gets the blinded solution, and sends it to Bob. Afterward, Bob unblinds this puzzle, decrypts the ciphertext to get the signature (σ), and gets the coins. In the transaction between Alice and tumbler, Alice sends the blinded puzzle to the tumbler, who can solve the puzzle and get ϵ^*. Alice then sends ϵ^* to Bob to unblinds it to ϵ and receive the coins. **Issues**. The protocol requires more fees to perform four transactions. The users should transfer fixed denominations to prevent linking the sender and the recipient.

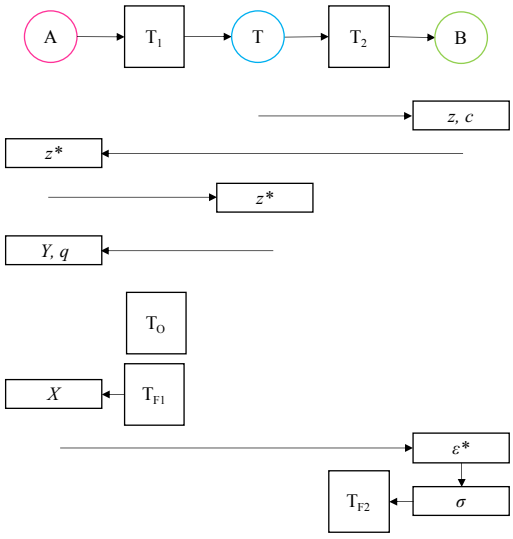


Figure 4: TumbleBit

3.3. CoinJoin based

CoinJoin based mixing employs techniques that 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.

3.3.1. CoinJoin

CoinJoin was proposed by Maxwell in 2013 [50]. Since the inputs of the Bitcoin transactions should be separately signed by the associated signatures, the users are able to jointly create one transaction with their inputs. In this manner, they can not only break the common input ownership heuristic but also hide the relation of inputs and outputs of a transaction if they send similar coins to the output addresses.

To create the transaction, users provide their input, output, and change addresses and separately sign the transaction. One of the users combine the signatures and broadcasts the transaction to the network.
Issues. There is an internal traceability between users. Transaction size is confined in Bitcoin which results in a small anonymity set. Large anonymity sets increase the risk of DoS and Sybil attacks and boost communication overhead. The users should spend the same amount of coins which reduces its practicability and usability. Consequently, outputs with similar amounts make the transaction distinguishable in the blockchain and can reveal the change addresses. Additionally, it prevents plausible deniability. It is not specified how the users are selected in the protocol [7].

3.3.2. CoinShuffle

CoinShuffle [62] is an improvement over CoinJoin to reach untraceability against mixing users, inspiring Dissent protocol [21]. The users find each other via a peer-to-peer protocol, after which each user (except 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. Bob decrypts it and creates layered encryption of his own output address B' with the remaining key. Afterward, he shuffles these two ciphertexts and sends them to the next one who repeats the same. The last user gets all outputs, adds her own output, shuffles them, and sends the 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 list as output, signs it, and broadcasts it to 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 rerun. **Issues**. Similar to CoinJoin, small anonymity, distinguishability of the transactions in the blockchain, and revealing the change addresses are considered as CoinShuffle's issues. The protocol is vulnerable to DoS and Sybil attacks. Users' interaction via peer-to-peer networks curbs scalability [38] (Mixing with 50 users takes 3 minutes). The last user controls the shuffled list and might shuffle the addresses in a manner that benefits (e.g. choose preferred input to send the coins to her own output address) [40]. Furthermore, having the last user controlling the shuffle prevents plausible deniability for the other users [83]. The user should first send the coins to her own address and then to the recipient as she should provide a new fresh address if blame occurs and a rerun of the protocol is needed. Although DoS attack can be detected, it is not prevented. Users cannot be forced to pay fees upfront to prevent DoS and Sybil attacks [39].

3.3.3. CoinShuffle++

CoinShuffle++ [63] uses DiceMix protocol [63] that requires sequential processing. Its predecessor CoinShuffle [62] requires a number of communication rounds linear in the number of users. CoinShuffle++ utilizes DiceMix to process mixing in parallel that 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.
Issues. Except for scalability and anonymity set, other CoinShuffle's issues are considered in CoinShuffle++.

3.3.4. ValueShuffle

ValueShuffle [61] 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 perfect improvement in comparison with the predecessors that not only hides transaction value from prying eyes in the blockchain 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 address. A Stealth address is a unique one-time address that is generated by the sender to improve the recipient's privacy.
Issues. Soft-fork is required to integrate Confidential transactions in the Bitcoin protocol. The protocol is vulnerable to DoS and Sybil attacks. The anonymity set is moderate (Mixing with 50 users takes 8 seconds). Due to the unique structure of stealth addresses, the anonymity of these addresses is confined to the set of the users that use such addresses.

3.3.5. CoinJoinXT

CoinJoinXT [31] proposes a form of CoinJoin transaction in which users first send the funds to a funding address that they jointly control using multi-signatures. Then, they sign a set of spending transactions from this address in advance (using Segwit solves transaction malleability). All the spending transactions should be given a specified time lock to prevent publishing them 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 transactions. However, to prevent double-spending, they should also create and pre-sign a backout transaction for each round, which has a specified time lock. To prevent subset-sum attack, participants can use off-chain privacy e.g. channel in Lightning network to send part of the outputs to the channel and shift the balance over time. Distinguishability of multi-sig in P2SH script can be solved by Schnorr signature [64] or Scriptless ECDSA-based Construction [48] to form 2-of-2 multisig transactions such as P2PKH transactions.

Issues. The protocol needs several transactions which lead to high fees and delays.

3.3.6. Snicker

Snicker [29] is a simple non-interactive CoinJoin where the keys for encryption are reused. 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 by the public key of the other participant (Bob) to create a CoinJoin proposal. Alice's message contains her UTXO, the desired recipient address, the amount, transaction fee, the full transaction template by UTXOs of Alice and Bob, Alice's signature on the transaction, and the recipient address of Bob which is created by adding k'G to either Bob's existing reused public key (Version-1) or R value in one of the Bob's signature (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, and sign and broadcast the transaction to the network. Alice should scan the Blockchain to find potential participants according to the amount of his UTXO and the age of UTXO. Scanning the blockchain can be done by one of the block explorers and Alice only downloads the data to find the active users.

Issues. The large anonymity set causes huge data of the proposals which in turn takes time and needs computation requirements for participants to decrypt the proposals. Junk proposals can waste the participants' time. Bob may receive the proposal when his input is already spent. The anonymity set for more than two users can not be easily achieved. The wallets should keep the record of the past transactions in order to derive the private key of the newly generated public key in Version-2. Sybil attacks are possible on the encrypted message.

3.3.7. PayJoin

PayJoin [32] (similar to bustapay [37] and P2EP [12]) solves the distinguishability of 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 this kind of transaction, there should be no input bigger than the biggest output to look like ordinary transactions [33]. To run the protocol [30], Bob sends the recipient's address and amount. Alice creates and signs a transaction in which she sends the specified amount to Bob's address and provides her change address to receive the remainder and then sends the 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 and adds the coins he added by inputs to the final amount. He signs his inputs and sends this new transaction to Alice. Alice checks and signs the transaction and broadcasts it to the network. Figure 5 illustrates a simple form of PayJoin where Ch_A is Alice's change address.

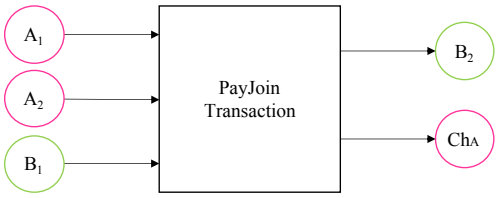


Figure 5: PayJoin

Issues. It needs interaction with the recipient. The division of the transaction fee is an open problem in the protocol which can be solved by paying a fee based on UTXO contribution. The unnecessary input heuristic can be applied to find the change address. Selecting the recipient's UTXOs to be appended as inputs should be done in a way that forms the transaction similar to ordinary transactions. PayJoin fails when the recipient does not have any UTXO to add to the inputs.

3.4. Threshold signature

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

3.4.1. CoinParty

CoinParty [83] employs mixing peers instead of group transactions in CoinJoin based protocols to provide plausible deniability. It employs a threshold variant of ECDSA (inspired by [41]) using secure multi-party computation (SMC). In the first phase, mixing peers generate 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 send a different escrow address to each input peer. Input peers commit their coins into the escrow addresses. The coins in the escrow addresses can only be redeemed if the majority of mixing peers sign the transactions. In the shuffling phase, input peers utilize layered encryption to encrypt their output addresses by the public keys of the mixing peers. Then, they broadcast the encrypted output along with the hash of their output to mixing peers (to be used in checking the final shuffled addresses by mixing peers). Each mixing peer decrypts the output and shuffles the address and sends it to the next peer. The last mixing peer shuffles the output addresses and broadcasts it to the mixing peers. All mixing peers check the shuffled addresses and seeds a pseudo-random number generator (PRNG) to obtain a final permutation of outputs to prevent the final peer from controlling the last permutation and ensures random shuffling. Finally, 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.

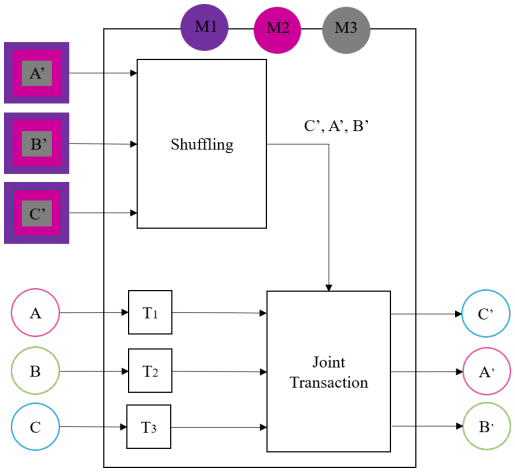


Figure 6: CoinParty, inspired by [83]

Issues. The protocol is secure only if 2/3 of mixing peers are honest which cannot be easily achieved in a peer-to-peer network without strong identities [63] and makes the protocol prone to theft and DoS attack [20]. As mixing peers have no share in the escrow transactions, they may leave when the coins are in the escrow addresses. It is possible to incentivize them using mixing fees, which in turn, increases the fees by input peers [40]. Every transaction requires the threshold ECDSA key generation and the threshold ECDSA digital signature, which leads to high computation [40].

3.4.2. SecureCoin

SecureCoin [40] uses the CoinJoin technique along with the 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. In the next step, they generate fresh recipient addresses and shuffle the addresses. Address shuffling and blame phase (accusation in SecureCoin) are almost similar to CoinShuffle in which the users encrypt their recipient addresses and send them to the next user to decrypt the message and add her recipient address. If a user misbehaves the protocol enters the accusation phase in 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 kicked out users in the accusation phase. Therefore, a user gets her coins back even if she behaves maliciously in the protocol. To complete the protocol, the majority of the users should jointly sign the transaction.
Issues. The protocol is secure if 3/4 of the users are honest or 2/3 of the users are honest at the cost of computation complexity by employing polynomial degree reduction (see computation complexity in [40] for more details). The final transaction can be identified in the blockchain as a mixing transaction since it has only one input and a bunch of outputs with the same value amounts, which compromise plausible deniability.

3.4.3. Secure Escrow Address (SEA)

Secure Escrow Address (SEA) [71] is a decentralized protocol that employs distributed key generation proposed by [27] to send the coins first to a temporary address in the joint control of the users and then to the recipients' addresses. To do this, all the users jointly create a public key of a joint address (J), where 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 [62], then they shuffle the recipients' addresses using layered encryption and the last user broadcast the shuffle list. Each user checks the list to verify whether her recipient's address is included or not. If everything goes right, they create a transaction and transfer the coins from address J to the recipients' addresses. To redeem the coins users should sign the transaction by their own share of the secret. The protocol has not been implemented and there are no test results to see how the distributed key generation works in ECDSA scheme.
Issues. The coins can be spent if half of the users sign the transaction. Therefore, the attackers who control half of the inputs can spend the coins in the joint address. It is stated in the protocol to exclude the dishonest users in the shuffling phase, but this limits the minimum number of users to sign the joint transaction. The final transaction can be detected in the blockchain as a mixing transaction since it has only one input from a temporary address and a bunch of outputs with the same value amounts, which also compromises plausible deniability.

4. Evaluation

In this section, we evaluate the techniques previously presented in terms of privacy, security, and efficiency criteria. Figure 7 outlines the most addressed criteria in the literature. In table 1, we evaluate the techniques into four main categories (centralized mixers, atomic swap, CoinJoin based, and threshold signatures).
Privacy criteria.
Anonymity set. The set of participants in the mixing transaction required to enhance the latter indistinguishability.
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" [69].
Untraceablity. "Given a transaction, all senders are equiprobable. One cannot figure out who is the sender among a transaction input addresses" [69].
Payment value privacy. The transaction value is protected from blockchain data analysis.
Security criteria. *Theft resistance*. The coins cannot be stolen during the protocol execution.
Dos resistance. The participant cannot refuse to compute

Figure 7: Mixing techniques criteria

the transaction (considered only for decentralized peering to create a transaction).
Sybil resistance. The attacker cannot take part in the protocol with different identities to identify the recipient addresses with which it is paired.
Efficiency criteria.
No interaction with input users. There is no interaction with other participants for peering to create a transaction.
No interaction with the recipient. There is no interaction with the transaction recipient.
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 receiving the coins to a new address of the sender and then send it to the recipient.
Number of transactions. The minimum number of transactions to complete the protocol.
Minimum mixing time (Block). The minimum number of blocks to complete the protocol (i.e., currently, 10 minutes for Bitcoin).

Figure 8 illustrates the categorization of the techniques while it indicates the evolution of the them.

5. Discussion

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.
Privacy. As can be seen in the table 1, the techniques compared in terms of large, medium, and small anonymity sets. Most of the techniques except some of 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 coordination between a large set of users to create a CoinJoin transaction can not be easily achieved in practice 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 using Dicemix, in which 50 participants can create a transaction in 8 seconds, considered as a reasonable time for this size of anonymity set. Coinswap techniques can be hidden among all the transactions with the same value in the blockchain (implementation of two-party ECDSA where multi-signature transactions are form such as single signature transactions can effectively provide anonymity for these transactions). Although the anonymity set in atomic swap techniques is large, the timelock transactions in these techniques curb the anonymity set [54]. Although in all techniques, users should create fresh addresses to receive mixed coins, 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 in turn can be used in the transaction graph. ValueShuffle can meet unlinkability as stealth addresses are one-time use payment addresses, however, the stealth addresses can be applied in other techniques too to improve those techniques over unlinkability. For instance, Darkwallet which has not been update since 2015 [17] was the 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 that use such addresses [54]. All the techniques tried to improve the untraceablity of transactions in the blockchain, however, the techniques that have partial coverage of this feature are those that have internal traceability in which the relationship between inputs and outputs is traceable among the participants of the mixing techniques. When a technique is internally

		Anonymity set	Unlinkability	Untraceability	Payment value privacy	Theft resistance	DoS resistance	Sybil resistance	No interaction with input users	No interaction with recipient	BTC Compatible	Direct send to recipient	no. Trx	Min Block
		Privacy				Security			Efficiency					
Third party services	Mixing websites	Large	○	◑±	○	○	●	●	●	●	●	●	2	2
	MixCoin [15]	Large	○	◑±	○	○×	●	●	●	●	●	●	2	2
	BlindCoin [68]	Large	○	●	○	○×	●	●	●	●	●	●	2	4 △
	LockMix [3]	Large	○	●	○	◑•	●	●	●	●	●	●	4	6
	Obscuro [67]	Large	○	◑±	○	●	●	●	●	○	◑	●	2	2
Atomic swap	FairExchange [4]	Large	○	◑±	○	●	●	○	○	●	●	●	4	3
	Xim [7]	Large	○	◑±	○	●	●	●	○	●	●	●	7	X *
	CoinSwap [51]	Large	○	◑±	○	◑°	●	●	●	○	●	●	4	2
	New CoinSwap [28]	Large	○	◑±	○	●	●	●	●	●	●	○	4	2
	PaySwap [6]	Large	○	●	◑	●	●	●	●	●	●	○	4	2
	BSC [39]	Large	○	●	○	●	●	●	●	○	◑◇	●	4	3
	TumbleBit[38]	Large	○	●	○	●	●	●	●	○	●	●	4	2
CoinJoin based	CoinJoin [50]	Small	○	◑±	○	●	○	○	○	●	●	○	1	1
	CoinShuffle [62]	Small	○	●	○	●	◑‡	○	○	●	●	○	1	2
	Coinshuffle++ [63]	Moderate	○	●	○	●	◑‡	○	○	●	●	○	1	2
	ValueShuffle [61]	Moderate	●	●	●	●	◑‡	○	○	●	◑◇	●	1	1
	CoinJoinXT [31]	Large	○	◑±	◑	●	●††	●	○	●	●	●	X	X
	SNICKER [29]	Small	○	◑±	○	●	○	○	●	●	●	○	1	1
	PayJoin [32]	Large	○	◑±	◑	●	●	●	○	○	●	●	1	1
Threshold signatures	CoinParty [83]	Large	○	●	○	◑⊕	◑‡	○	○	●	●	○	2	2
	SecureCoin [40]	Moderate	○	●	○	◑⊕	◑‡	●	○	●	●	○	2	2
	SEA [71]	Moderate	○	●	○	◑	◑‡	●	○	●	●	○	2	2

●Full coverage ◑Partial coverage ○No coverage

± Internal traceability.
× Theft is detected, but it is not prevented.
• It is possible in lose-lose or get nothing scenarios.
° In the case of malleability of initial transaction.
†† In two-party case.

⊕ If 2/3 of users are honest.
‡ Prevented by finding the malicious participant and excluding her.
◇ Soft-fork is required.
△ Two blocks for public log messages plus two blocks for two transactions.
* It is a two-party transaction, so needs many mixing transactions to achieve a large anonymity set.

Table 1: Evaluation of mixing techniques

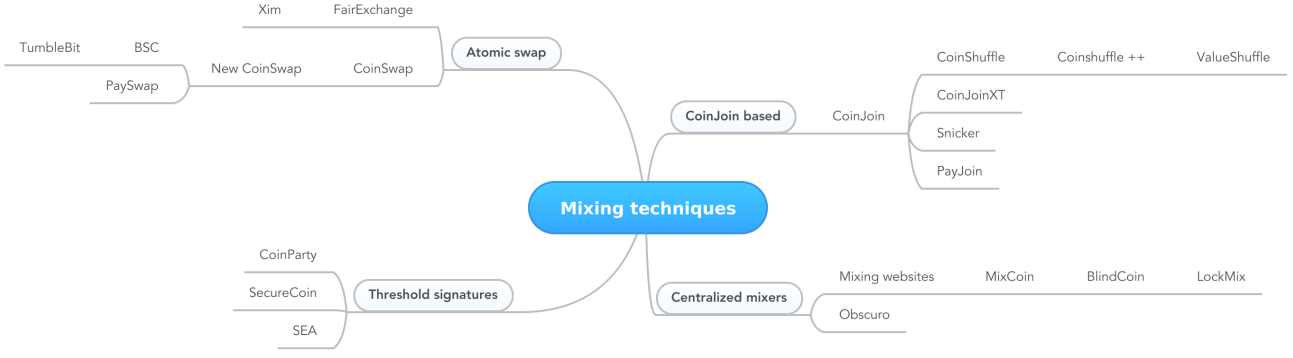


Figure 8: Evolution of mixing techniques

traceable the involved participants are able to store the other user's data which can lead to information leakage. It should be mentioned even if they are traceable among the participants, they provide privacy against blockchain analysts [54]. To prevent tracing the transactions by precise value attacks, providing transaction value privacy, or hiding the actual payment value is one of the features that boost transaction privacy. Among the techniques, ValueShuffle proposed 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 from that and by this, there is no need for the fixed denomination in the proposed techniques which consequently improves the usability and liquidity in other techniques where the users can mix their desired number of coins. [57] compares the implementation of CT in Tumble bit 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, considering that applying CT in Bitcoin transactions increases nine times the transaction fee. CoinJoinXT can provide a level of value privacy, however, the subset-sum may break this criteria.

Security. One of the most prominent criteria in payment networks is to prevent the coins from being stolen or lost. This criteria is crucial in blockchain as there is no practical solution to claim the coins back (except hard-fork). Most of the techniques tried to address this criteria. While mixing websites are not theft-resistance, the mixer is only accountable in MixCoin and BlindCoin, although theft can be detected it cannot be prevented. The previous exit scams [46] in the mixing websites[46] makes trusting those services hard. The techniques that are based on threshold signatures also can not greatly provide this feature as they need the majority of the users to be honest which can not be easily achieved in a peer to peer network. Atomic swap and CoinJoin based techniques can provide this feature in the envisioned protocols. According to our definition for Dos-resistance, most of the CoinJoin based and threshold signatures techniques lack this feature as they need the users to behave honestly during the protocol. To prevent Dos attacks, finding and kicking out the malicious users and rerun the protocol, and also locking the malicious user's UTXO have been proposed in some of the techniques. PayJoin is DoS resistance 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 as none of the participants can abort the protocol and affect others. Sybil attacks are prevented by receiving the fee upfront in most of the techniques. The CoinJoin techniques that are considered as no coverage for Sybil-resistance are those that do not propose preventing such attacks in their protocols.

Efficiency. In terms of the 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 requires the availability of the users during the protocol, even if the user is not malicious, the connection lost may result into the protocol failure. This can effectively delay creating CoinJoin transactions while most techniques can be performed without interaction with the input users. Obscoro, PayJoin, and PaySwap require interaction with the recipients. CoinSwap, BSC, and TumbleBit require interaction with the recipient in their original protocol, however, the sender can play the recipient role with different identities in these protocols to omit interaction with the recipient. In this scenario, the sender receives the coins in her own new address and needs one more transaction to send the mixed coin to the desired destination address. Directly sent to the recipient address columns intends to show that in some of the proposed techniques the user needs to first the coins to her own address and next to the desired destination address. This problem exists in CoinJoin based and threshold signatures techniques where the participant should provide a new output if the protocol goes to the blame phase. Valueshuffle uses stealth addresses to overcome this problem, application of stealth addresses in other techniques can solve this problem for those techniques too. Most of the techniques are compatible with the current implementation of the Bitcoin blockchain, however, ValueShuffle requires CT, BSC requires blind signatures to be implemented in the Bitcoin blockchain via soft-fork, Obscuro also requires some changes in the Bitcoin Core implementation. The last two columns indicate the number of transactions and the minimum number of blocks to run one round of the protocol which are great insights for delays and transaction fees that should be paid by the participants. It is really important to consider the cost that should be shoulder by participants to do mixing, apart from mixing fee, additional transaction fees in the mixing techniques would be the main barrier for the adoption of those techniques in practice. Even in CoinJoin based techniques, the participants should pay at least one additional transaction fee for mixing the coins and then transfer the coins to the destination address. Considering the point 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. Atomic swap techniques also require four transactions in at leas two blocks which in turn lead to additional costs and delays.

Implementation in practice. Most of the aforementioned techniques have not been implemented in practice or there is a significant delay between the protocol and its implementation in practice. Table 2 lists the implementation of the techniques in practice. As can be seen most of the implementation are centralized mixing websites. According to [54], Fairexhange transactions can not be found in the blockchain and most of the applications of stealth addresses back to the time of Darkwallet implementation. Dumplings [56] indicates an increment of CoinJoin transactions in the past two years (since 2018). It should be mentioned that the number of CoinJoin can be as a result of multiple mixing rounds to achieve a better anonymity set. Joinmarket, Wasabi, and Samourai are the implemented CoinJoin wallets. Joinmarket uses a taker-maker model where the taker announces her willingness to perform a CoinJoin transaction and makers participate with her in the CoinJoin transaction by receiving fees. In this approach, privacy is for the taker who creates the CoinJoin transaction [34]. Wasabi uses Chaumian CoinJoin where the participants register their inputs and blindly sign the outputs to the coordinator to create a CoinJoin transaction. Samourai proposed Whirlpool which has specified pools where the users can join to mix their coin with other participants and create CoinJoin transactions. SharedCoin which was a CoinJoin service by Blockchain.info in which Blockchain.info was able to find the inputs and outputs relationships, and also Darkwallet that provided creating CoinJoin transactions and used Stealth addresses discontinued, probably due to the legal reasons. In 2020, BTCpay added PayJoin to allow merchants to create their stores that accept PayJoin transactions. Currently, Wasabi, Samourai, Joinmarket, and Bluewallet support PayJoin transactions. How-

Mixing websites adopted from [46]	CoinJoin	Coinshuffle	PayJoin	TumbleBit
ChipMixer.com	Joinmarket	Shufflepuff	Samouraiwallet (Stowaway)	NTumbleBit
BitMix.Biz	Wasabiwallet	NXT	BTCPay	Breeze
Bitcloak43blmhmn.com	Samouraiwallet (Whirlpool)		Wasabiwallet	
Mixer.money	Darkwallet (until 23.01.2015) [17]		Joinmarket	
MixTum.io	Sharedcoin (until 02.09.2016) [70]		Bluewallet	
Blender.io				
FoxMixer.com				
MixerTumbler.com				
CryptoMixer.io				
MyCryptoMixer.com				
tumbler.to				

Table 2: Mixing techniques adoption in practice

ever, creating PayJoin transactions between the users implemented before in Joinmarket and Samourai wallet (Stowaway). Shufflepuff [72] is an alpha version in Github and its last updates back to 2016 and Nxt [42] has been activated since block 621,000 (09.03.2020) on mainnet. At the time of writing, there is no commercial implementation of atomic swap techniques. Recently [6], developing a new CoinSwap design/PaySwap wallet has been proposed. There are also some alpha implementations of TumbleBit in Github (NTumbleBit and Breeze) which are not commercial at the time of writing the paper.

Future research. Future research can be done in three areas including usability, law enforcement, and practicality of the techniques.

Usability. In the usability area the following questions can be considered: To what extent the users are aware of add on and built-in privacy techniques and their implementations in practice? I. Do they trust third party privacy-preserving services? II. Which would be preferred by users, using add on techniques implemented by wallets and services or using built-in techniques such as privacy coins to achieve stronger anonymity? Do the users accept the extra fees and delays to achieve stronger privacy in the blockchain? I. Is there any significant difference in paying for privacy between privacy-aware and privacy-unaware users? 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, no interaction with other users)? Do the current implementations of the techniques allow the users to realize what needs to be done and do they understand how to do it? Do the implementations of privacy wallets have any effects on the number of private transactions?

Law enforcement. While there are some footprints of using privacy techniques in the dark web which cause the thought that privacy-preserving techniques are used for illicit activities, privacy techniques may be used by users who are aware of the catastrophic problem because of deanonymization in the blockchain. To the best of our knowledge, there is no research in the state of the art which can categorize the destination addresses of the CoinJoin transactions which are commonly used privacy techniques that have been implemented by privacy wallets (JoinMarket, Wasabi, Samourai). Although it can not be easily figured out the recipient of those CoinJoin transactions, categorizing these addresses by using ground truth can shed light on the usage of the CoinJoin techniques in the Bitcoin Blockchain. To this reason the following research question would be a good starting point. Is it possible to categorize the destination of the CoinJoin transactions to find the percentage of its application in the illicit activities? There is always a trade-off between privacy and law enforcement rules in the cryptocurrency environment. Achieving privacy for most of the users while preventing the technology to be a good place for criminal activities and the dark market is still an open problem in the field. [45] proposed a model to enable law enforcement agency to collaborate with a involved parties in CoinJoin transactions to find criminals which can be a good starting point.

Practicality. Accepting PayJoin technique in the market can 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 can not 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 their effectiveness on tagging the PayJoin transactions. Further research also can be done in non-equal amount CoinJoin transactions. As of now, distinguishability of equal size CoinJoin transactions has the potential to get the users into a problem, they may encounter the services that refuse to accept the output of CoinJoin transactions. Knapsack which proposed in [49] and Wabisabi [23] would be great future work to improve the indistinguishability of these types of transactions.

6. Conclusion

The aim of the current study is to review and evaluate mixing techniques in Bitcoin. The study has compared a multitude of selected proposals according to a set of criteria. These proposals offer different guarantees in terms of privacy, security, and efficiency. Strong privacy affects efficiency, scalability, and usability. Among atomic swap techniques, New CoinSwap and its predecessors can meet most of the criteria while they require 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 to these techniques. The recently proposed PayJoin method, which is based on CoinJoin

can indeed solve the distinguishability and improve anonymity. One of the main advantages of CoinJoin based techniques is the reduced number of transactions to run the protocol, which makes them affordable. However, most of these techniques fail to provide a large anonymity set and plausible deniability. Confidential transactions, proposed in ValueShuffle, can efficiently solve this problem and provide indistinguishability for CoinJoin based techniques. Mixing 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, it also comes with a cost, i.e., transaction fees. Even though the mixing fee can be negligible, an additional number of transaction fees may limit the techniques adoption by users. According to our results, except centralized mixers and threshold based techniques, theft resistance criteria has been met by most of the techniques. Although the initial intention of guaranteeing strong privacy was to hinder user details from malicious adversaries and criminals, such privacy-preserving techniques can be employed to conduct illicit activities. Therefore, new methods able to distinguish transactions used for illicit activities from regular mixing transactions (e.g., financial privacy) are required.

References

[1] Alrawi, O., Lever, C., Antonakakis, M., Monrose, F., 2019. Sok: Security evaluation of home-based iot deployments, in: 2019 IEEE Symposium on Security and Privacy (SP), IEEE. pp. 1362–1380.

[2] Antonopoulos, A.M., 2017. Mastering Bitcoin: Programming the open blockchain. " O'Reilly Media, Inc.".

[3] Bao, Z., Shi, W., Kumari, S., Kong, Z.y., Chen, C.M., 2019. Lockmix: a secure and privacy-preserving mix service for bitcoin anonymity. International Journal of Information Security , 1–11.

[4] Barber, S., Boyen, X., Shi, E., Uzun, E., 2012. Bitter to better—how to make bitcoin a better currency, in: International conference on financial cryptography and data security, Springer. pp. 399–414.

[5] Belcher, C., (2018) Last accessed 24 March 2021. Joinmarket. https://github.com/JoinMarket-Org/ joinmarket-clientserver .

[6] Belcher, C., Last accessed 16 July 2020. Design for a coinswap implementation for massively improving bitcoin privacy and fungibility. https:// gist.github. com/ chris-belcher/ 9144bd57a91c194e332fb5ca371d0964 .

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

[8] bitcoin.org, Last accessed 21 July 2020a. Multisig. https:// developer.bitcoin.org/ devguide/ transactions.htmlmultisig .

[9] bitcoin.org, Last accessed 21 July 2020b. Pay-to-public-key-hash. https:// developer.bitcoin.org/ devguide/ transactions.htmlp2pkh-script-validation .

[10] bitcoin.org, Last accessed 21 July 2020c. Pay-to-script-hash. https:// developer.bitcoin.org/ devguide/ transactions.htmlp2sh-scripts .

[11] Blake, I.F., Seroussi, G., Smart, N.P., 2005. Advances in elliptic curve cryptography. volume 317. Cambridge University Press.

[12] 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/ .

[13] Bojja Venkatakrishnan, S., Fanti, G., Viswanath, P., 2017. Dandelion: Redesigning the bitcoin network for anonymity. Proceedings of the ACM on Measurement and Analysis of Computing Systems 1, 1–34.

[14] Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J.A., Felten, E.W., 2015. Sok: Research perspectives and challenges for bitcoin and cryptocurrencies, in: 2015 IEEE symposium on security and privacy, IEEE. pp. 104–121.

[15] Bonneau, J., Narayanan, A., Miller, A., Clark, J., Kroll, J.A., Felten, E.W., 2014. Mixcoin: Anonymity for bitcoin with accountable mixes, in: International Conference on Financial Cryptography and Data Security, Springer. pp. 486–504.

[16] BtcDrak, M.F., Lombrozo, E., 2015. Bip 112: Checksequenceverify. URl: https://github. com/bitcoin/bips/ blob/master/ bip-0112. mediawiki .

[17] caedesvvv, 2015. Darkwallet. https:// github.com/ darkwallet/ darkwallet/ releases/ tag/ 0.8.0 .

[18] Chaum, D., 1983. Blind signatures for untraceable payments, in: Advances in cryptology, Springer. pp. 199–203.

[19] Chaum, D.L., 1981. Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM 24, 84–90.

[20] Conti, M., Kumar, E.S., Lal, C., Ruj, S., 2018. A survey on security and privacy issues of bitcoin. IEEE Communications Surveys & Tutorials 20, 3416–3452.

[21] Corrigan-Gibbs, H., Ford, B., 2010. Dissent: accountable anonymous group messaging, in: Proceedings of the 17th ACM conference on Computer and communications security, pp. 340–350.

[22] Ermilov, D., Panov, M., Yanovich, Y., 2017. Automatic bitcoin address clustering, in: 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE. pp. 461–466.

[23] Ficsór, , Kogman, Y., Seres, I.A., Last accessed 3 Feb 2021. Wabisabi. https://github.com/zkSNACKs/WabiSabi/releases/download/build-70d01424bbce06389d2f0536ba155776eb1d8344/WabiSabi.pdf .

[24] Filtz, E., Polleres, A., Karl, R., Haslhofer, B., 2017. Evolution of the bitcoin address graph, in: Data science–Analytics and applications. Springer, pp. 77–82.

[25] Fleder, M., Kester, M.S., Pillai, S., 2015. Bitcoin transaction graph analysis. arXiv preprint arXiv:1502.01657 .

[26] Franco, P., 2014. Understanding Bitcoin: Cryptography, engineering and economics. John Wiley & Sons.

[27] Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T., 1999. Secure distributed key generation for discrete-log based cryptosystems, in: International Conference on the Theory and Applications of Cryptographic Techniques, Springer. pp. 295–310.

[28] Gibson, A., 2017 (Last accessed 23 August 2020). New coinswap. https:// joinmarket.me /blog /blog /coinswaps/ .

[29] Gibson, A., 2017 (Last accessed 31 August 2020). Snicker - simple non-interactive coinjoin with keys for encryption reused. https://joinmarket.me /blog /blog/snicker/ .

[30] Gibson, A., 2018 (Last accessed 23 August 2020). Basic payjoin / p2ep protocol for joinmarket wallets. https:// gist.github.com/ AdamISZ/ 4551b947789d3216bacfcb7af25e029e .

[31] Gibson, A., 2018 (Last accessed 31 August 2020). Coinjoinxt - a more flexible, extended approach to coinjoin. https://joinmarket.me /blog/blog /coinjoinxt/ .

[32] Gibson, A., 2019 (Last accessed 23 August 2020)a. Payjoin. https:// joinmarket.me /blog /blog /payjoin/ .

[33] Gibson, A., 2019 (Last accessed 23 August 2020)b. Payjoin - a basic demo. https:// joinmarket.me /blog /blog /payjoin-basic-demo/ .

[34] Gibson, A., Last accessed 3 Feb 2021. From mac to wabisabi. https://joinmarket.me/blog/blog/from-mac-to-wabisabi/ .

[35] Halpin, H., Piekarska, M., 2017. Introduction to security and privacy on the blockchain, in: 2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), IEEE. pp. 1–3.

[36] Harrigan, M., Fretter, C., 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. pp. 368–373.

[37] Havar, R., Last accessed 20 September 2020. Bustapay bip: a practical sender/receiver coinjoin protocol. https://lists. linuxfoundation.org/pipermail/bitcoin-dev/2018-

August/016340.html .
[38] Heilman, E., Alshenibr, L., Baldimtsi, F., Scafuro, A., Goldberg, S., 2017. Tumblebit: An untrusted bitcoin-compatible anonymous payment hub, in: Network and Distributed System Security Symposium.
[39] Heilman, E., Baldimtsi, F., Goldberg, S., 2016. Blindly signed contracts: Anonymous on-blockchain and off-blockchain bitcoin transactions, in: International conference on financial cryptography and data security, Springer. pp. 43–60.
[40] Ibrahim, M.H., 2017. Securecoin: A robust secure and efficient protocol for anonymous bitcoin ecosystem. IJ Network Security 19, 295–312.
[41] Ibrahim, M.H., Ali, I., Ibrahim, I., El-Sawi, A., 2003. A robust threshold elliptic curve digital signature providing a new verifiable secret sharing scheme, in: 2003 46th Midwest Symposium on Circuits and Systems, IEEE. pp. 276–280.
[42] Jelurida, Last accessed 11 August 2020. Nxt. https://nxtdocs.jelurida.com/ Coin Shuffling .
[43] Jourdan, M., Blandin, S., Wynter, L., Deshpande, P., 2018. Characterizing entities in the bitcoin blockchain, in: 2018 IEEE International Conference on Data Mining Workshops (ICDMW), IEEE. pp. 55–62.
[44] Kalodner, H., Last accessed 23 July 2020. Privacy. https:// citp.github.io/ BlockSci/ reference/ heuristics/ change.html .
[45] Keller, P., Florian, M., Böhme, R., 2020. Collaborative deanonymization. arXiv preprint arXiv:2005.03535 .
[46] LeGaulois, Last accessed 11 August 2020. 2020 list bitcoin mixers bitcoin tumblers websites. https:// bitcointalk.org/ index.php?topic=2827109.0 .
[47] Lombrozo, E., Lau, J., Wuille, P., 2015. Bip 141: Segregated witness.
[48] Malavolta, G., Moreno-Sanchez, P., Schneidewind, C., Kate, A., Maffei, M., 2019. Anonymous multi-hop locks for blockchain scalability and interoperability., in: NDSS.
[49] Maurer, F.K., Neudecker, T., Florian, M., 2017. Anonymous coinjoin transactions with arbitrary values, in: 2017 IEEE Trustcom/BigDataSE/ICESS, IEEE. pp. 522–529.
[50] Maxwell, G., 2013a. Coinjoin: Bitcoin privacy for the real world, 2013. URl: https:// bitcointalk. org/ index. php .
[51] Maxwell, G., 2013b. Coinswap: transaction graph disjoint trustless trading (2013). URL: https:// bitcointalk. org/ index. php .
[52] Meiklejohn, S., Pomarole, M., Jordan, G., Levchenko, K., McCoy, D., Voelker, G.M., Savage, S., 2013. A fistful of bitcoins: characterizing payments among men with no names, in: Proceedings of the 2013 conference on Internet measurement conference, pp. 127–140.
[53] Moser, M., 2013. Anonymity of bitcoin transactions .
[54] Möser, M., Böhme, R., 2017. Anonymous alone? measuring bitcoin's second-generation anonymization techniques, in: 2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), IEEE. pp. 32–41.
[55] Nakamoto, S., 2008. A peer-to-peer electronic cash system. Bitcoin.–URL: https:// bitcoin. org/ bitcoin. pdf .
[56] nopara, ., Last accessed 3 Feb 2021. Dumplings. https://github.com/nopara73/Dumplings .
[57] nopara73, Last accessed 3 Feb 2021. Tumblebit vs coinjoin. https://nopara73.medium.com/tumblebit-vs-coinjoin-15e5a7d58e3 .
[58] Pfitzmann, A., Hansen, M., 2010. A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management .
[59] Reid, F., Harrigan, M., 2013. An analysis of anonymity in the bitcoin system, in: Security and privacy in social networks. Springer, pp. 197–223.
[60] Rivest, R.L., Shamir, A., Adleman, L., 1978. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM 21, 120–126.
[61] Ruffing, T., Moreno-Sanchez, P., 2017. Valueshuffle: Mixing confidential transactions for comprehensive transaction privacy in bitcoin, in: International Conference on Financial Cryptography and Data Security, Springer. pp. 133–154.
[62] Ruffing, T., Moreno-Sanchez, P., Kate, A., 2014. Coinshuffle: Practical decentralized coin mixing for bitcoin, in: European Symposium on Research in Computer Security, Springer. pp. 345–364.
[63] Ruffing, T., Moreno-Sanchez, P., Kate, A., 2017. P2p mixing and unlinkable bitcoin transactions., in: NDSS, pp. 1–15.
[64] Schnorr, C.P., 1989. Efficient identification and signatures for smart cards, in: Conference on the Theory and Application of Cryptology, Springer. pp. 239–252.
[65] Tasca, P., Tessone, C.J., 2017. Taxonomy of blockchain technologies. principles of identification and classification. arXiv preprint arXiv:1708.04872 .
[66] Todd, P., 2014. Bip 65: Op checklocktimeverify. Github (accessed 18 October 2015) https://github. com/bitcoin/ bips/blob/master/bip-0065. mediawiki .
[67] Tran, M., Luu, L., Kang, M.S., Bentov, I., Saxena, P., 2018. Obscuro: A bitcoin mixer using trusted execution environments, in: Proceedings of the 34th Annual Computer Security Applications Conference, pp. 692–701.
[68] Valenta, L., Rowan, B., 2015. Blindcoin: Blinded, accountable mixes for bitcoin, in: International Conference on Financial Cryptography and Data Security, Springer. pp. 112–126.
[69] Van Saberhagen, N., 2013. Cryptonote v 2.0.
[70] Ver, R., 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/ .
[71] Wang, Q., Li, X., Yu, Y., 2017. Anonymity for bitcoin from secure escrow address. IEEE Access 6, 12336–12341.
[72] Weigl, D., 2016 (Last accessed 11 August 2020). Mycelium shufflepuff. https:// github.com/ DanielWeigl/ Shufflepuff .
[73] Wiki, Last accessed 16 July 2020a. Multisignature. https:// en.bitcoin.it/ wiki/ Multisignature .
[74] Wiki, Last accessed 16 July 2020b. Timelock. https:// en.bitcoin.it/ wiki/ Timelock .
[75] Wiki, Last accessed 21 July 2020a. Hashlock. https:// en.bitcoin.it/ wiki/ Hashlock .
[76] Wiki, Last accessed 21 July 2020b. Script. https:// en.bitcoin.it/ wiki/ Script .
[77] Wiki, Last accessed 22 July 2020. Htlc. https:// en.bitcoin.it/ wiki/ Hash Time Locked Contracts .
[78] Wiki, Last accessed 23 July 2020. Privacy. https:// en.bitcoin.it/ wiki/ Privacy .
[79] Wikipedia, Last accessed 21 July 2020a. Double-spending. https:// en.wikipedia.org/ wiki/ Double-spending .
[80] Wikipedia, Last accessed 21 July 2020b. Forth (programming language). URL: https:// en.wikipedia.org/ wiki/ Forth (programming language) .
[81] Wikipedia, Last accessed 21 July 2020c. Unspent transaction output. https:// en.wikipedia.org/ wiki/ Unspent transaction output .
[82] York, D., 2010. Seven deadliest unified communications attacks. Syngress.
[83] Ziegeldorf, J.H., Grossmann, F., Henze, M., Inden, N., Wehrle, K., 2015. Coinparty: Secure multi-party mixing of bitcoins, in: Proceedings of the 5th ACM Conference on Data and Application Security and Privacy, pp. 75–86.