CoinShuffle Seminar Report

Andrina Jasmin Eisenegger Tobias Andri Famos

March 2022

Contents

1 Introduction

Blockchain-based currencies are fully, publicly and permanently traceable. Without anonymity, privacy is much worse than in traditional banking. Bitcoin in particular cannot guarantee anonymity of its users (quote). In the Bitcoin system the true identity is only hidden behind pseudonyms (public keys).

This report looks at CoinShuffle, a Bitcoin mixing protocol, to see how unlinkability can be increased in the Bitcoin protocol to increase the level of anonymity.

There are at least four goals to achieve privacy in the blockchain system. First, the anonymity of users: Given a user's blockchain address (i.e. the public key), an attacker should not be able to determine the IP addresses of users. Second, unlinkability: Given two transactions, an adversary cannot determine if they were sent to the same user. Even with two addresses, an adversary cannot determine whether they belong to the same user. Third, untraceability: In the case of a transaction, an attacker cannot determine from which address the transaction originated. And finally, confidentiality: A transaction does not reveal the amount transacted.

In the Bitcoin system, only two goals can be pursued due to the Bitcoin protocol. These two goals are user anonymity and unlinkability.

In order to guarantee the anonymity of a person behind the public keys, there must always be a matching set of subjects that potentially have the same properties (Pfitzmann and Kohntopp, 2001). In concrete terms, this means that as long as the pseudonyms used cannot be linked with a user, the user can hide in the crowd.

But with simple heuristics, the traces left by the user's transactions can be clustered and combined into an identity (Conti et al., 2018).

One way to increase unlinkability is to use a mixer. This involves sending unspent bitcoin transactions plus mixing fees to a mixing address. This address should mix multiple transactions and return the deposited amount. This third-party mixing service must be trusted because the two biggest dangers are that the mixer will steal the deposited amount or reveal the user's anonymity (Bonneau et al., 2014).

A peer-to-peer mixing protocol that is decentralized and perfectly compatible with Bitcoin has been proposed by Maxwell: CoinJoin (2013). This protocol prevents coin theft and is free of mixing fees, but it has weighty drawbacks. These drawbacks are that the mixing server must still be trusted as it learns the relationship between input and output. Furthermore, this protocol is not immune to DoS attacks (Ruffing et al., 2014).

The CoinShuffle protocol is able to deal with the problems and dangers of the two mixed methods presented and still has the advantages of decentralization. This protocol is presented and critiqued in this report. CoinShuffle combines the advantages of CoinJoin (to provide security against theft) and the responsible anonymous group communication protocol Dissent to increase the level of user anonymity as well as robustness against DOS attacks (Ruffing et al., 2014).

In particular, this report critically considers the following features: Unchainability, Verifiability, and Robustness.

2 Background

2.1 Bitcoin

Bitcoin (฿) is a highly volatile digital currency. The so-called cryptocurrency has no physical representation in the form of coins or banknotes and yet is gaining popularity due to its innovative features, simplicity and transparency (Ametrano, 2016).

The idea of cryptocurrencies is that electronic transactions are decentralised and therefore not based on trust. The transactions in the Bitcoin system are stored in a public transaction book (”blockchain”), which is stored in a decentralised peer-to-peer network (Nakamoto, 2008). The blockchain comprises several blocks, each with several transactions. In addition to the transactions, each block contains a timestamp, the hash value of the previous block (”parent”) and a nonce, a random number to verify the hash value. This concept ensures the integrity of the entire blockchain up to the first block (”genesis block”) (Nakamoto, 2008). Hash values are unique and can be effectively protected against fraud, as changes to a block in the chain would change the corresponding hash value.

The security of the blockchain depends on a computationally intensive bitcoin mining algorithm that prevents the double spending of bitcoins and the manipulation of confirmed transactions when honest nodes control a majority of CPU power (e.g. proof-of-work consensus) (Swanson, 2015).

New transactions are therefore not automatically added to the ledger. Rather, the consensus process ensures that these transactions are stored in a block for a certain time (e.g. 10 minutes in the Bitcoin blockchain) before they are transferred to the ledger. After that, the information in the blockchain can no longer be changed. In the case of Bitcoin, the blocks are created by so-called miners who are rewarded with Bitcoins for validating the blocks (Nofer et al., 2017).

2.2 Non-Privacy-Preserving Blockchains

Blockchains protect addresses from being directly linked to real identities in the world. However, this is very difficult to ensure with decentralised identity management, because although no third party needs to be trusted, decentralisation is often achieved through public traceability to increase security (Nakamoto, 2008). This sharing of information means that everyone on the network has access to the addresses' past transactions. In other words: “Consequently, Bitcoin has the unintuitive property that while the ownership of money is implicitly anonymous, its flow is globally visible“ (Meiklejohn et al., 2013).

2.2.1 Privacy Goals

In order to guarantee the anonymity of a person, there must always be a suitable set of subjects who potentially have the same properties (the anonymity set is the set of all possible subjects) (Pfitzmann and Köhntopp, 2001). The definition of anonymity by Pfitzmann and Köhntopp (2001) is: “Anonymity is the state of being not identifiable within a set of subjects, the anonymity set”.

This means that different interactions of the same user with the system should not be associated with his or her identity. So consequently, the combination of pseudonymity and unlinkability (Pfitzmann and Köhntopp, 2001).

There are at least four goals to achieve privacy in the blockchain system, which can be defined as follows:

1. **User Anonymity**: Given a user's blockchain address (i.e., public key), an adversary should not be able to identify users' IP addresses.
2. **Unlinkability**: Given two transactions, an adversary cannot tell if they are sent to the same user. Moreover, given two addresses, an adversary cannot determine whether they belong to the same user.
3. **Untraceability**: Given a transaction, an adversary cannot know which address is issuing that transaction.
4. **Confidentiality**: A transaction does not reveal the transacted amount.

2.2.2 The Anonymity in the Bitcoin System

Within the Bitcoin system, users are only identified by public keys, so-called pseudonymous addresses. This means that the anonymity of users can be removed by an attack in which the attacker constructs i.e., the one-to-many mapping between users and their public keys (Conti et al., 2018). These linked profiles can be de-anonymised through a variety of side channels, or directly, as many Bitcoins services require a real identity (Koshy et al., 2014).

Bitcoin attempts to prevent this attack by storing a user's public key assignment only on that user's node, and each user can create countless numbers of addresses by generating new key pairs (Conti et al., 2018).

This privacy precaution of Bitcoin is not satisfactory for targeted attacks. For example, several studies show that various techniques lead to a weak form of anonymity for Bitcoin addresses (Apostolaki et al., 2021 and Barber et al., 2012).

Meiklejohn et al. (2013) show that even using different addresses does not guarantee optimal privacy. Over time, through multiple transactions, public keys can make a number of declarations. Shared outputs can be evidence of shared control over different addresses. Using some basic heuristics, Bitcoin addresses can be classified that are likely to belong to the same user, and any new transaction with a new address from that user can be added to this cluster (Spagnuolo et al., 2014).

It is even possible to detect the ownership relationship between Bitcoin addresses and IP addresses (Biryukov et al., 2014 and Koshy et al., 2014).

Thus, it can be seen that Bitcoin system cannot provide optimal protection for users' privacy. This problem has been recognised and also heavily discussed in the scientific community and researched for solutions (Conti et al., 2018).

2.3 Add-on Privacy Solutions for Non-Privacy-Preserving Blockchains

2.3.1 Bitcoin Mixing

In the traditional implementation of a peer-to-peer (P2P) mixing protocol, a mixer (mixing address) is required. This anonymous service provider uses mixing protocols to confuse the transaction traces (Conti et al., 2018). This address takes coins from multiple customers, splits the funds into smaller pieces and randomly routes them to a new address for each customer (Conti et al., 2018).

Mixing multiple transactions can provide better privacy protection because it is harder to link the incoming and outgoing addresses of a transaction (Bonneau et al., 2014). The degree of anonymity in P2P protocols depends on the number of clients in the anonymity set (Dingledine and Mathewson, 2006).

Whereas a large transaction volume leads to mixing times of up to 48h and a low transaction volume leads to users frequently getting their own coins back. With the latter, getting your own coins back from a mix is not necessarily a weakness. This is the case with a random permutation of the coins of N participants with a probability of 1/N (Conti et al., 2018).

To use this service, such providers usually charge commissions (1-3 % of the mixed coins) and require manual interaction via a website which is sometimes only accessible via hidden Tor services (Bonneau et al., 2014). Using Bitcoin over Tor is not a good idea, as Biryukov and Pustogarov, 2015 have shown.

To use mixing there are two major threats:

Theft. Malicious mixers may send the funds to their own address instead of transferring it back to the client. Clients can complain publicly about the theft and try to undermine Mixer's reputation, but there is no way to prove the theft accusation, then the money has been sent to a new address with no transaction history (Bonneau et al., 2014).

Deanonymization. Since the mix learns that both addresses (input and output from the client) belong to the same party, the anonymity from the client depends on the mix keeping this mapping secret forever. A mix that is malicious, compromised or subpoenaed could disclose its records and undermine the client's anonymity. Alternatively, the mix could send coins in a non-random way that reveals the connection to observers (Bonneau et al., 2014).

2.3.2 CoinJoin

CoinJoin is a straightforward protocol for P2P mixing that does not require a mixer and is therefore a trustless method as Maxwell claims (2013). This implementation is perfectly compatible with Bitcoin (Maxwell, 2013).

CoinJoin breaks a basic assumption of many blockchain heuristics, which is that multiple Bitcoin payments in a single transaction come from the same party (Meiklejohn et al., 2013). In other words, CoinJoin exploits the fact that a Bitcoin transaction can have many input and output addresses from different parties (Maxwell, 2013).

Multiple Bitcoin users combine multiple Bitcoin payments into a single transaction. The group of users agree on their primary signatures, the inputs and outputs, so no external adversary knows which output is linked to which input, ensuring external unlinkability (Maxwell, 2013).

To ensure that free riders don't stand a chance in the group, CoinJoin users must take turns signing a CoinJoin transaction until all parties agree on the final status of the transaction before it is finally transferred to the blockchain (Atlas, 2014).

So can an adversaries (a group member) perform Denial of Service Attack by refusing to sign their CoinJoin transaction (Chan et al., 2021).

The result is that users can mix their funds without internal or external theft. One party must act as a conductor to manage this process between CoinJoin users (Atlas, 2014).

Maxwell (2013) proposes that to increase privacy, "the N users agree on a uniform output size and provide inputs that are at least equal to that size. The transaction would have N outputs of that size and possibly N more change outputs if some of the users provide inputs that go over the target." This fact can lead to long waits to find a suitable group with the desired outputs. For example, the uniformity of outputs was one reason why many wallet services did not integrate CoinJoin into their services to improve user privacy (Atlas, 2014).

The main drawbacks of CoinJoin are that privacy depends on the number of participants and a loss of privacy occurs because the mixing server learns the relationship between the input and output address (Maxwell, 2013). Also, to create a mix, each participant must share their signature and exit addresses within the group of participants, resulting in internal linkability (Conti et al., 2018). Another drawback is that it almost impossible to identify a misbehaved participant and the requirement that a transaction must be signed by all players makes the process vulnerable to DoS attacks (Chan et al., 2021).

To solve the problems of internal linkability and increase resistance to DoS attacks, Ruffing et al. (2014) propose CoinShuffle, a decentralised protocol that coordinates CoinJoin transactions using a cryptographic mixing technique.

3 CoinShuffle Protocol

CoinShuffle is a decentralized mixing protocol derived from the shuffling phase of the Dissent protocol Ruffing et al., 2014. It runs on top of the bitcoin protocol. The basic mechanism relies on a distributed shuffle approach, where each participant shuffles the output addresses, without knowing whose addresses he is shuffling. To keep the other participants in the dark about the belonging of the addresses the shuffling methodology of the Dissent protocol has been adopted. In the protocol, N participants create a shared transaction in which they shuffle a certain amount of bitcoins.

The CoinShuffle protocol can be split into three temporal blocks to simplify the explanation: (Ruffing et al., 2014)

1. **Announcement**: The Announcement phase is used to share the needed information (e.g. input addresses, public keys) for the transaction between the participants.
2. **Shuffling**: In the shuffling phase, a list of output addresses is passed through all the participants. Each participant adds his output address, shuffles the list and passes it on.
3. **Transaction Verification**: Before signing the transaction and committing it to the bitcoin network, each participant can verify the integrity of the transaction. He checks for his output address in the Transaction and signs.

During the whole protocol, any participant can invoke a transition to the blame phase (Ruffing et al., 2014). The blame phase is started when one or more participants are not complying with the protocol. The participant or participants having deviated from the protocol get pointed out (blamed) and a new run of CoinSuffle is started without them.

3.1 Goals

CoinShuffle aims at fulfilling the following properties as a mixing protocol (Ruffing et al., 2014):

- **Verifiability**: The control of the bitcoins must always remain with the participant. There must be no way to steal any participant's bitcoin during a run and no possibility of loosing any bitcoins.
- **Robustness**: Every run of a protocol must eventually succeed. Neither an adversary nor a honest participant with a technical error can have the possibility to prevent a successful run of CoinShuffle. This obviously only holds as long as the communication between the participants can be kept up.
- **Unlinkability**: After a run of the protocol, neither an adversary using public information nor a participant of the protocol can be know the link between the output addresses and the participants or the input addresses.

3.2 Protocol Phases

A run of the CoinShuffle Protocol can be split into six phases. For a run, we make the following assumptions:

- The input addresses of all participants are already known by the other participants
- The agreed upon amount of bitcoins to shuffle is ฿m
- Most of the steps in the protocol are executed in turns. This requires a agreed upon order which we assume to be known by all participants.

When a participant is sending a message in the run of the protocol, he signs it using the signing key of his bitcoin input address. For simplicity we only state this once in the protocol description.

3.2.1 Phase 1: Announcement

In the announcement phase, each participant announces a new public key for the encryption in phase 2 and proves he is the owner of the private key corresponding to his input address. This is done in turns. Participant i sends a message containing his public key signed with the private key for his input address.

Before sending his singed message, participant i checks all the previous messages $j < i$, validates the signature and checks if the input address of j contains at least ฿m.

3.2.2 Phase 2: Shuffling

At the beginning of the shuffling phase, each participant creates a fresh bitcoin address which will be used as an output address on the derived transaction of the protocol. To ensure participant i does not know which output address corresponds to which participant, a layered encryption approach is used.

The first participant creates a layered asymmetric encryption on his output address with the public keys of all the other participants (in the agreed upon order). He then sends it to the next participant.

Any participant i receives a randomly shuffled list of output addresses of the previous participants. Each output address in the list has layers of encryption where the top encryption layer

was produced with the public key of participant i. He encrypts his output address with the remaining $N - i$ encryption layers, strips the output addresses of the encryption layer i and adds his encrypted output address to the list. He then shuffles the list randomly and communicates it to the next participant.

In the end, a permutation of all the output addresses is derived. But no participant knows which addresses correspond to which participant, as he has not seen the addresses in clear text in his shuffling run.

3.2.3 Phase 3: Broadcast of the Output

After the last participant having stripped the last layer of encryption and having shuffled the clear text addresses, he then broadcasts the whole list of addresses to the rest of the participants. Now each participant checks for his own output address in the list to ensure, nobody has tampered with the addresses during the process. If a participant identifies a missing address, he forces an enter to the blame phase.

3.2.4 Phase 4: Equivocation Check

The last participant in this scenario has quite a powerful role. He is in charge of sending the shuffled list to the rest of the participants, thus he could tamper with the list. The participants have already checked for their address in the lists, but now in the equivocation check, they check for the equivocation.

Each participant computes the has of the encryption keys and the list of output addresses he received. He then broadcasts the hash to all the other participants and compares all the received hashes for equality with his own. Once again, if any participant receives a hash different from his, the protocol enters the blame phase.

3.2.5 Phase 5: Transaction Verification and Submission

Each participant creates his own mixing transaction for the bitcoin network. The transaction sends $฿m$ from each input address to each output address.

After signing the transaction, he then broadcasts his signature to the other participants, takes the received signatures and adds them to the transaction. He checks once again, if the bitcoins of the other participants have been spent in the meantime and if not submits the transaction to the bitcoin network. Once again, if the bitcoins have already been spent, the blame phase is entered.

Once the transaction has been submitted to the bitcoin network and no participant has entered the blame phase, the run of the protocol has successfully terminated.

3.2.6 Phase 6: Blame

The blame phase can be invoked by any participant and is only invoked if in a step of the protocol a misbehaviour of a participant is detected. Depending on the reason for invocation, additional information is sent to the blame phase. Deterministically, a misbehaving participant is then designated. The remaining participants restart a run of the protocol. By removing a misbehaving participant, CoinShuffle prohibits anyone from blocking the protocol by always submitting fraudulent messages.

3.3 Analysis

In this section we will give an analysis of the stated goals unlinkability, verifiability and robustness.

3.3.1 Unlinkability

The main property to achieve is the unlinkability. One can distinguish between two scenarios of unlinkability, the external unlinkability, only relying on public information (e.g. the blockchain) and the internal unlinkability with non public information (e.g. messages collected through traffic analysis). The external unlinkability is given through the fact that a transaction is created with multiple input and output addresses. Bitcoin does not include any explicit mapping of the input addresses to the outputs. Additionally, the amount transacted is equal for each address, thus no mapping can be derived from the amounts either (Ruffing et al., 2014).

The internal unlinkability is also given, through the use of the layered encryption in the shuffling phase. Even when having all the messages, neither an adversary nor a participant can derive which participant has added an output address. This holds as long as the encryption keys are kept private. As the cipher text of all the output addresses change at each step, there is no way to derive which item has been added to the list and which ones have just been stripped by a layer of encryption (Ruffing et al., 2014).

Note that the unlinkability is only given as long as there are at least two honest participants (Ruffing et al., 2014).

3.3.2 Verifiability

The verifiability is given, as each participant will only sign the derived transaction if he finds his output address in the list of output addresses (Ruffing et al., 2014). Thus she controls the flow of her bitcoin and can not loose it unless she does not keep her private signing key secret.

3.3.3 Robustness

Given a live communication, an adversary can try to prevent a successful run of the protocol. When a participant deviates from the protocol, the blame phase is reached. By sharing the contextual information depending on the state of the protocol before entering the blame phase, each participant can rerun all the steps that lead to the blame phase. As each message is signed with the private signing key of the input addresses, a forgery of the messages is not possible. Thus any participant can retrace the steps and point out the person to blame.

Every run of the protocol thus results either in a valid transaction being submitted to the bitcoin network or in the removal of a deviating participant and a rerun of the protocol. If there are at least two honest participants, CoinShuffle will therefore always derive at a valid transaction (Ruffing et al., 2014).

3.4 Extensions and description of practical details

In order to arrive at a usable protocol, a few special cases have to be considered and a few practical details explained. We will touch on the transaction fees and the extension of the protocol to allow for unequal bitcoins on the input addresses as propose by Ruffing et al., 2014. Additionally we propose a adaption of the protocol to offer the possibility to shuffle unequal amounts.

3.4.1 Transaction Fees

To ensure a transaction is processed by the miners, a transaction fee must be provided (Schär and Berentsen, 2020). The fee for the miners per transaction is the difference between the input Ƀand output Ƀand incentivises the miners to take the transaction into consideration. In CoinShuffle, the transaction fee can be implemented easily. The optimal transaction fee is derived at the beginning of a run. Then the fee is split into N parts (with N participants) and $\frac{1}{N}$Ƀis subtracted from each output. Thus the transaction fee is split equally among the participants.

3.4.2 Unequal balances

Typically, not every user will hold an input address with the exact amount of Ƀto mix. Thus Ruffing et al. introduce the notion of a change address (Ruffing et al., 2014. To prevent deanonymization attacks the output amount has to be equal for each of the mixed output addresses. Additionally each participant can announce a change address in the Announcement phase. The change address will recceive the difference from a users input address and the amount to mix Ruffing et al., 2014. All participants then store the change addresses of the others and add them to the transaction in the transaction verification phase.

Change addresses are not mixed. As the transmission of the change addresses is signed with the private key of the input address, the change addresses are internally likable to a participant.

Additionally to the notion of a change address, we figured the notion of pooling from multiple input addresses might be interesting as well. As we can see the case of a participant having only a address holding more than m Ƀthere is also the case of a participant having multiple input addresses summing up to m Ƀ. To make the protocol more general, the case of participants announcing not only one input address but multiple could be considered. Such a functionality would only entail small changes such as checking multiple input addresses instead of one for sufficient funding and transmitting multiple signatures in the verification and submission phase.

3.4.3 Unequal amounts to mix

Although not mentioned by Ruffing et al. we also figured the protocol could be extended to handle unequal amounts to mix. Given multiple participants with the need to mix different amounts, a run of the protocol could be adopted in the following way: The agreed upon m Ƀwould be set at the smallest common denominator of all the amounts to mix. Each participant would then add multiple output addresses to the list in the shuffle phase in order to get his intended balance in all the addresses. The unlinkability would be given as still no participant can see the list of output addresses in plain text.

3.5 Advantages

The advantage of the CoinShuffle protocol is that it provides internal unlinkability in a distributed fashion. There is no need for trust in any participants or entities in order to mix coins. It requires no transaction fees, as there is no entity to be paid. Also it is compatiable with the current state of bitcoin and all its derived coins (e.g. Litecoin). No changes to the base protocols are required. (Ruffing et al., 2014)

3.6 Downsides

The major downside might be the run time and scalability of CoinShuffle. Although it is a secure approach, one single run of the protocol with 50 participants took 180 seconds in a simulation (Ruffing et al., 2014). When adding malicious actors and participants with lagging internet connections, we think the run time would grow to an impractical number.

Also, the process of finding people to mix coins with might be tedious. Added to the long run time might come some long waiting times to find participants with the need to mix the same amount of bitcoins. The limitation of mixing only same amounts is inherent to mixing in one transaction. We have stated a possible way to circumvent this requirement.

4 Conclusion

Ruffing et al. proposed an interesting approach to refine the CoinJoin by adding internal unlinkability through the usage of an anonymous group messaging protocol. A prototype has been built by Ruffing et al. Ruffing et al., 2014 but there has been no real world usage as far as we can tell. It seems like CoinShuffle only served as an initial approach and has been developed further into CoinShuffle++ (Yocom-Piatt, 2019). With CoinShuffle++ Ruffing et al. have addressed the problem of the high runtimes with high numbers of participants. They swaped the anonymous group messaging protocol based on Dissent to a new approach called DiceMix. It is worth noting that ConinShuffle++ is also used on crypto curencies such as Decred (Yocom-Piatt, 2019)

References

Ametrano, F. M. (2016). *Hayek Money: The Cryptocurrency Price Stability Solution* (SSRN Scholarly Paper No. 2425270). Social Science Research Network. Rochester, NY. https://doi.org/10.2139/ssrn.2425270

Apostolaki, M., Maire, C., & Vanbever, L. (2021). Perimeter: A network-layer attack on the anonymity of cryptocurrencies. In N. Borisov & C. Diaz (Eds.), *Financial cryptography and data security* (pp. 147–166). Springer Berlin Heidelberg.

Atlas, K. (2014). Advisory — CoinJoin Sudoku. Retrieved April 25, 2022, from http://www.coinjoinsudoku.com/advisory/

Barber, S., Boyen, X., Shi, E., & Uzun, E. (2012). Bitter to better - how to make bitcoin a better currency. *Financial Cryptography*.

Biryukov, A., Khovratovich, D., & Pustogarov, I. (2014). Deanonymisation of clients in bitcoin p2p network. *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, 15–29. https://doi.org/10.1145/2660267.2660379

Biryukov, A., & Pustogarov, I. (2015). Bitcoin over tor isn't a good idea, 122–134. https://doi.org/10.1109/SP.2015.15

Bonneau, J., Narayanan, A., Miller, A., Clark, J., Kroll, J. A., & Felten, E. W. (2014). Mixcoin: Anonymity for Bitcoin with Accountable Mixes. In N. Christin & R. Safavi-Naini (Eds.), *Financial Cryptography and Data Security* (pp. 486–504). Springer. https://doi.org/10.1007/978-3-662-45472-5_31

Chan, W. K., Chin, J.-J., & Goh, V. T. (2021). Simple and scalable blockchain with privacy. *Journal of Information Security and Applications*, *58*, 102700. https://doi.org/https://doi.org/10.1016/j.jisa.2020.102700

Conti, M., Sandeep Kumar, E., Lal, C., & Ruj, S. (2018). A survey on security and privacy issues of bitcoin. *IEEE Communications Surveys Tutorials*, *20*(4), 3416–3452. https://doi.org/10.1109/COMST.2018.2842460

Dingledine, R., & Mathewson, N. (2006). Anonymity loves company: Usability and the network effect. *WEIS*.

Koshy, P., Koshy, D., & McDaniel, P. (2014). An Analysis of Anonymity in Bitcoin Using P2P Network Traffic. *Financial Cryptography and Data Security*, 469–485. https://doi.org/10.1007/978-3-662-45472-5_30

Maxwell, G. (2013). CoinJoin: Bitcoin privacy for the real world. Retrieved April 25, 2022, from https://bitcointalk.org/index.php?topic=279249

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. *Proceedings of the 2013 Conference on Internet Measurement Conference*, 127–140. https://doi.org/10.1145/2504730.2504747

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, 21260.

Nofer, M., Gomber, P., Hinz, O., & Schiereck, D. (2017). Blockchain. *Business & Information Systems Engineering*, *59*. https://doi.org/10.1007/s12599-017-0467-3

Pfitzmann, A., & Köhntopp, M. (2001). Anonymity, Unobservability, and Pseudonymity — A Proposal for Terminology. In H. Federrath (Ed.), *Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability Berkeley, CA, USA, July 25–26, 2000 Proceedings* (pp. 1–9). Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-44702-4_1

Ruffing, T., Moreno-Sanchez, P., & Kate, A. (2014). CoinShuffle: Practical Decentralized Coin Mixing for Bitcoin [Series Title: Lecture Notes in Computer Science]. In M. Kutyłowski & J. Vaidya (Eds.), *Computer Security - ESORICS 2014* (pp. 345–364). Springer International Publishing. https://doi.org/10.1007/978-3-319-11212-1_20

Schär, F., & Berentsen, A. (2020). *Bitcoin, Blockchain, and Cryptoassets: A Comprehensive Introduction*. MIT Press.

Spagnuolo, M., Maggi, F., & Zanero, S. (2014). Bitiodine: Extracting intelligence from the bitcoin network. *Financial Cryptography*.

Swanson, T. (2015). Consensus-as-a-service: A brief report on the emergence of permissioned, distributed ledger systems. https://vdocuments.mx/consensus-as-a-service-a-brief-report-on-the-emergence-of-permissioned-distributed.html

Yocom-Piatt, J. (2019). Iterating Privacy [Section: posts]. Retrieved May 30, 2022, from https://blog.decred.org/2019/08/28/Iterating-Privacy/