

# Improving Privacy of Blockchains

Emily Schaefer  
University of Minnesota, Morris  
Morris, Minnesota, USA 56267  
schae586@morris.umn.edu

## ABSTRACT

Bitcoin is a popular cryptocurrency that uses blockchain technology to create a secure, public, immutable, peer-to-peer distributed currency system. Since this system does not rely on trusted third parties, blockchains use many cryptographic techniques. This increases users privacy by eliminating third parties from having access to their information. Users interact with this system using pseudonyms, but there are concerns on how well blockchains protect users privacy. This paper will explore the cryptographic techniques in Bitcoin's blockchain and address the privacy concerns of blockchains, while discussing a proposed solution to increasing user privacy on the public blockchain.

## Keywords

Blockchain, Privacy, Bitcoin, Cryptography, Cryptocurrency, CoinShuffle

## 1. INTRODUCTION

In traditional financial industries, third parties such as banks handle currency and transactions [8]. Cryptocurrencies eliminates the need for a trusted intermediary to handle financial transactions by using cryptography [7]. Cryptocurrencies are increasing in popularity [1]. The first and most successful such system was proposed by Satoshi Nakamoto in 2008 in the article, "Bitcoin: A Peer-to-Peer Electronic Cash System." Bitcoin allows payments to be transferred between parties directly without intermediaries [8]. These payments called transactions are in Bitcoins, which is the digital currency used by Bitcoin denoted  $\text{₿}$  [1].

Blockchains are the technology behind this decentralized Bitcoin system. A blockchain is a public ledger that is made up of a chain of blocks that contain transactions in a chronological order [7]. Each block links to the previous block and the chain grows as new transactions are created and added to the blockchain [11]. Blockchains are public, pseudonymous, distributed, peer-to-peer and immutable [8]. Blockchains are public so all transactions are public. Therefore, transaction information such as who was involved in the transaction, the number of Bitcoins paid, and the transaction time are public information. Even though blockchains are public, users interact in Bitcoin with pseudonyms. Pseudonyms are false

names, that do not reveal any identities. Therefore, the public does not know who was involved in the transaction. The Bitcoin network is all of the users using Bitcoin and a node is a user in the Bitcoin network. The blockchain is distributed in that each node in the Bitcoin network stores a copy of the entire blockchain [8]. Blockchains are peer-to-peer because all of the nodes are connected. In addition, all nodes must agree on the transactions [8]. This makes it difficult to corrupt or alter blockchains. The blockchain is immutable meaning transactions cannot be changed once they are added to the blockchain because otherwise all the blocks after would need to be redone [11].

This decentralized blockchain technology increases user privacy by eliminating third parties access to users information [8]. However, there are concerns that the public aspect of blockchains threatens user privacy [10]. Users interact on the blockchain using pseudonyms called addresses [1]. While the use of pseudonyms protects user privacy, it is possible to link address to people, thus gaining their transaction information [11]. To address this issue, a method called Bitcoin mixing is proposed that hides how a user's address spends Bitcoins. This paper will present CoinShuffle, which uses Bitcoin mixing to improve user privacy on the public blockchain.

Blockchains were first developed in the financial industry, but they have the potential to transform many other industries such as healthcare, voting, education, among several others. There are numerous variations of blockchain, but this paper will present the Bitcoin implementation of the blockchain.

The paper will provide a description of Bitcoin's blockchain protocol, address the privacy of Bitcoin's blockchain and present an improved protocol called CoinShuffle to improve user privacy. Section 2 presents terminology used throughout the paper and cryptographic techniques used in the blockchain protocol. Section 3 discusses the blockchain protocol with addresses, transactions, and Bitcoin mining. Section 4 presents privacy concerns and CoinShuffle as an improvement to Bitcoin's protocol in protecting users privacy.

## 2. BACKGROUND

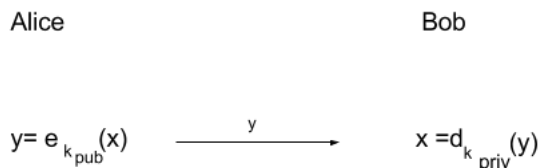
In order for blockchains to be decentralized, cryptography is used to make them secure. This section will discuss cryptographic concepts that are involved in the Blockchain protocol such as public-key cryptography, hash functions, and digital signatures.

## 2.1 Public-Key Cryptography

Cryptography is the study of secure communication of messages between two parties to prevent third parties from viewing the message. This can be done with public-key cryptography. Public-key cryptography has two functions. The first function is for encryption and decryption of a message to prevent third parties from viewing the message. The second use is for verification of identity, which is discussed in section 2.3.

To prevent unauthorized listening called eavesdropping, a message needs to be encrypted in transmission. Encryption  $e$  means converting a message in plaintext (unencrypted text) to a ciphertext (encrypted text). In order for the intended recipient to read the encrypted message, it needs to be decrypted. Decryption  $d$  means decoding the ciphertext back to plaintext. Public-key cryptography involves a public key denoted  $k_{pub}$  and a private key denoted  $k_{priv}$  that form a key pair. If a message is encrypted with one of these keys, it can only be decrypted with the other key. The public key is distributed to everyone and the private key is only known to the owner of the key pair. For sending a message the public key is used for encryption while the private key is used for decryption. No third party can read the message because only the person who has the private key can decrypt the message in ciphertext. The key pair is related by rigorous mathematical relationships which is out of the scope of the paper. However, it is computationally impossible to calculate the private key based on the public key. The only way to compute the private key based on the public key is brute force. However, this could take hundreds or thousands of years with modern technology. This makes it secure for the public key to be public. An analogy of public-key cryptography is that anyone can put a letter in a locked mailbox, but only the person with the key can get the letters from the mailbox.

Figure 1 shows the protocol for Alice sending Bob a message using public-key cryptography. Assume Bob has generated the key pair consisting of  $k_{pub}$  and  $k_{priv}$ .  $k_{pub}$  is known to both Alice and Bob, while  $k_{priv}$  is only known to Bob. Alice encrypts her message  $x$  in plaintext with the public key  $e_{k_{pub}}(x)$  to obtain a ciphertext, which she then sends to Bob. Bob decrypts the ciphertext using his private key  $d_{k_{priv}}(y)$  to get the original message  $x$ . Because Bob only knows his private key, no one else can read Alice's message in transmission. The public key is sent to Alice, but if a third party is eavesdropping and obtains the public key, they cannot decrypt the message because they do not have the private key. [9]

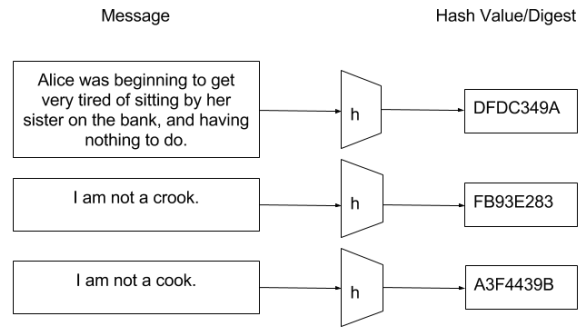


**Figure 1: Public-key Cryptography Protocol based on figure from Paar and Pelzl [9]. The keys  $k_{pub}$  and  $k_{priv}$  are Bob's generated keys.**

## 2.2 Hash Function

A hash function takes an arbitrary sized input and produces a string of a fixed length. A hash value or digest is created by a deterministic hash function. Using the notation where  $h(x)$  is the hash function and where  $x$  denotes the message, some significant features are as follows:

1. It is easy to compute  $h(x)$  from  $x$ .
2. Given the hash value  $z$  where  $h(x) = z$ , it is computationally infeasible to find  $x$ . This means that  $h(x)$  is a one way function.
3. Given an input of  $x_1$  and  $h(x_1)$ , it is computationally infeasible to find another message  $x_2$  where  $h(x_1) = h(x_2)$ .
4. Given two distinct messages  $x_1$  and  $x_2$  such that  $x_1 \neq x_2$ , it is highly improbable for  $h(x_1) = h(x_2)$  to occur.



**Figure 2: Hash Function based on figure from Paar and Pelzl [9]**

Figure 2 demonstrates the functionality of a hash function. Specifically this example shows how the length of the hash value is the same regardless of the message length. It is also important to note how two messages with a difference of one character will likely to be very different from one another. [9]

## 2.3 Digital Signature

A digital signature is similar to a handwritten signature to show approval of a transaction. This can be achieved by using public-key cryptography for verification of identity. The general overview of a digital signature is: a sender signs the transaction with their private key and the receiver verifies the signature by using the corresponding public key. Assuming each user has a private and public key pair, a digital signature consists of a signing algorithm and a verifying algorithm. In the signing algorithm given a message that is public, the hash value of the message is signed with the private key. The verifying algorithm authenticates the signature based on the public key. The following example of Bob digitally signing a message to Alice will demonstrate the specifics of digital signatures. Assume Bob has a private  $k_{priv}$  and public key  $k_{pub}$  that form a key pair.

1. Signature Algorithm: The signature algorithm takes Bob's private key and the message as input to produce a digital signature denoted  $sig_{k_{priv}}(x)$ . Commonly the digital signature is a signature of the hash of the message  $sig_{k_{priv}}(h(x))$ .

The fixed size of hash functions are desirable for digital signatures because less time is needed to sign a smaller message. In addition, less time is needed to compute the hash value than signing by encryption. The message  $x$  and Bob's digital signature is then sent to Alice.

2. Verifying Algorithm: Now Alice needs to verify the signature to ensure it was signed by Bob by verifying with his public key. Since Bob encrypted the message with his private key, anyone can decrypt it with his public key, thus validating the signature. This algorithm takes in the message, Bob's digital signature and Bob's public key as inputs. To verify, Alice creates a hash of the message  $x$  and decrypts the digital signature using Bob's public key by  $d_{pub}(sig_{k_{priv}}(h(x))) = h(x)$ . If the hash values are equal, the verification algorithm returns true. If the algorithm returns true, it ensures that Bob signed the message and the message  $x$  was not altered in transmission. [9]

There are different implementations of digital signatures such as the Elliptic Curve Digital Signature Algorithm (ECDSA), which creates a digital signature by using elliptic curves [9]. This is the digital signature used by Bitcoin [5].

### 3. BLOCKCHAIN PROTOCOL

There are several variations of blockchains. This paper will examine Bitcoin's protocol, which is the first implementation of blockchain technology. The foundation of the blockchain protocol can be split up into (1) addresses (2) transactions and (3) Bitcoin mining. A transaction occurs when money is transferred between two addresses [1]. Bitcoin mining is the process of verifying the transactions and adding them to the blockchain [7].

#### 3.1 Addresses

Users perform transactions by using pseudonymous addresses [1]. A user's address is a double hash of the public key [5]. But throughout the rest of the paper, there will be no differentiation between an address and a public key. The address is public and is used to make payments and receive payments [5]. Users also have a corresponding private key that is used to sign transactions to verify ownership [7]. The public/private key pair is an Elliptic Curve Digital Signature Algorithm (ECDSA) key pair in which the two keys are in a mathematical relationship [5]. All addresses are public, but the address is a pseudonym (a number) created by the double hash of the public key. Therefore, no identity is linked to the address. In addition, most users have multiple addresses [1].

#### 3.2 Transactions

A transaction is the exchange of money between the sender and the receiver [7]. Transactions are from the sender's address to the receiver's address [1]. Transactions are made up of an array of inputs and an array of outputs [2]. Therefore, transactions can have multiple inputs and multiple outputs [5]. An input is the address of the Bitcoin being spent and an output is the address for sending the Bitcoin [10]. Thus a transaction transfers Bitcoins from the array of inputs to the array of outputs [5].

Each input is digitally signed using the sender's private key [5]. This ensures that only the owner of that address can spend the Bitcoins at that address [5]. The whole blockchain is dependent on the state of these transactions [5]. Figure 3 shows multiple input and output addresses in a transaction

Input Addresses	Output Addresses
A: ₿2	X: ₿3
B: ₿7	Y: ₿2
	Z: ₿4
Sig <sub>K<sub>priv</sub></sub> (A)	
Sig <sub>K<sub>priv</sub></sub> (B)	




Figure 3: Input and Output Addresses modified from Ruffing et al [10]

with the Bitcoin value (₿ $v$ ). The input addresses are  $A$  and  $B$  while the output addresses are  $X$ ,  $Y$ , and  $Z$ . It also demonstrates that each input address is digitally signed with the sender's corresponding private key. For example,  $Sig_{k_{priv}}(A)$  is the input address  $A$  being digitally signed with the corresponding private key. This is evidence that the user at the address  $A$  has authorized the transaction since they are the only one who has the private key for that address.

Transactions must go through the mining process in order to be added to the blockchain.

#### 3.3 Bitcoin Mining

Bitcoin mining refers to validating transactions and adding transactions to the blockchain [7]. If the transaction is validated, a block is generated that contains the transaction [7]. Mining is necessary to ensure the sender is using Bitcoins that belong to them and/or not double spending the Bitcoin since ownership of Bitcoins is public information given in the transactions [7].

Bitcoin mining uses proof of work, which is an algorithm [11]. In proof of work, users called miners complete resource-intensive tasks [7]. Resource-intensive task means it requires a computer a significant amount of resources such as time or memory.

The miners compete to complete the task and the miner who completes the task first earns Bitcoins as a reward which acts as an incentive to complete the task [7]. It has to be a resource-intensive task to prevent too many blocks from being added to the blockchain at once. This is to limit the amount of new currency being introduced into the system as the Bitcoins earned are new Bitcoins [7]. Two terms used in the mining process are a nonce value and a Merkle tree root hash. A nonce value is a number used in the mining process that is usually random and is used only once [7]. A Merkle tree root hash is the hash value of all the transactions in the block [11]. The resource-intensive task consists of finding a nonce value that outputs a hash value less than a target number when hashed with the hash value of a previous block, the Merkle tree root hash of all transactions [1]. This is shown in Figure 4. The only strategy to find a nonce value is to trying random nonce values [2]. If the desired nonce value is found, then the block (which contains the transaction) is validated, and then it needs to be verified by the Bitcoin network [1]. The hash computation with the nonce value

block contents				
prev block	transactions	(nonce)	hash result	target
Hash				
H (#78A...	tx#839, tx#a76,...	3001	= 438...	< 100...
H (#78A...	tx#839, tx#a76,...	3002	= 988...	< 100...
H (#78A...	tx#839, tx#a76,...	3003	= 587...	< 100...
H (#78A...	tx#839, tx#a76,...	3004	= 087...	< 100...

Figure 4: Proof of work modified from [4]

is the proof of work [1] and the block is broadcasted to all nodes on the Bitcoin Network [7]. All of the nodes check the validity of the block by checking the hash computation from the proof of work [7]. If they come to the consensus that it is valid, the block is added to the growing blockchain [7].

## 4. PRIVACY

This section will discuss privacy concerns on the public blockchain of tracing transactions given an address. It will also explore the proposed CoinShuffle as a solution to this privacy concern.

### 4.1 Privacy Issues

Blockchains are public, which is a threat to user privacy. Because blockchains are public, transactions are public. Therefore, anyone can view transaction information such as the Bitcoin addresses involved in the transaction, how many Bitcoins were spent, and transaction times. The current privacy on the public blockchain is pseudonyms to protect the privacy of its users, which makes it cryptographically impossible to identify users [10]. Just by looking at a Bitcoin address, the specific individual cannot be determined. However, there are ways to link Bitcoin addresses through specific techniques. These techniques are out of the scope of the paper. But one way is linking an address to a specific person with IP addresses [11]. Transaction information is traceable given a Bitcoin address [10] and therefore if someone links an address to a specific person, they can track the specific persons transaction information. The following sections will discuss a technique called Bitcoin mixing to make transactions less traceable. It will also discuss a Bitcoin mixing protocol called CoinShuffle.

### 4.2 Bitcoin Mixing

Bitcoin mixing is a technique that is used to make transactions less traceable for an address [10]. This is done by combining multiple transactions into one transaction by mixing Bitcoins with other users to make input and output addresses unlinkable [10]. Given a single transaction with Alice's input address  $A$  and the output address  $X$ , it is clear the link between the input and output addresses. The public knows that Alice sent her Bitcoin to the output address  $X$ . The output address  $X$  could be another one of Alice's ad-

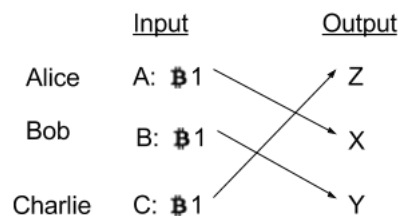


Figure 5: Combined transaction with shuffled outputs

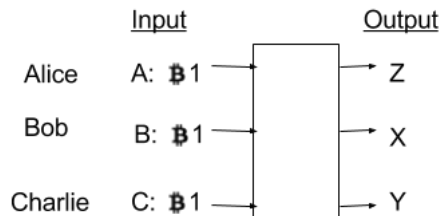


Figure 6: Combined transaction from outside perspective

resses or a different address that she does not own. Either way, there is a link between the input and output address. Similarity, given another single transaction with Bob's input address  $B$  and the output address  $Y$ , the input and output addresses are linked. And the same goes for Charlie's input address  $C$  sending to output address  $Z$  in another separate transaction. If these three transactions are combined into one transaction with the output addresses shuffled, there is less correlation between the input and output addresses. Figure 5 shows a combined transaction of the three individual transactions mentioned above. Figure 6 demonstrates what the transaction looks like from the outside perspective. It is an array of input and output addresses. But given Alice's input address, it is unknown if her output address is  $X$ ,  $Y$ , or  $Z$  hence decreasing the traceability of an address.

### 4.3 CoinShuffle

CoinShuffle is a proposed solution to protecting the privacy of users by Tim Ruffing et al [10]. It does not require any changes to the Bitcoin protocol as it is compatible with the current Bitcoin system. CoinShuffle builds on CoinJoin [6] and Dissent [3].

The CoinShuffle protocol is split into the following parts: announcement, shuffling, transaction verification and blame. This protocol can be done with any number of  $N$  participants, but for simplicity this paper will explore the protocol with  $N = 3$  participants. The participants will be named Alice, Bob, and Charlie. Figure 7 provides a figure outlining all four parts in the protocol.

Assume that each participant knows the order of all other participants. Alice is participant 1, Bob is participant 2, and Charlie is participant 3. Also assume each user who wants to participate in the mixing process possess a public/private key pair  $k_{pub}$  and  $k_{priv}$ . The public key is published and is the users Bitcoin address. The public key (i.e.

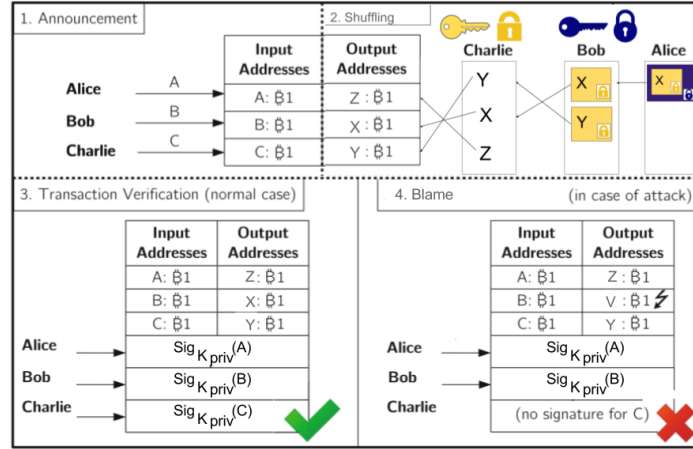


Figure 7: CoinShuffle Protocol modified from Ruffing et al [10]

address) is one of the input addresses. Alice's input address is  $A/k_{pub_A}$ , Bob's input address is  $B/k_{pub_B}$ , and Charlie's input address is  $C/k_{pub_C}$ . Alice's private key is  $k_{priv_A}$ , Bob's private key is  $k_{priv_B}$  and Charlie's private key is  $k_{priv_C}$ . Alice, Bob, and Charlie all broadcasts their input addresses so all participants are aware of all input addresses. A message in CoinShuffle is a protocol message that users must sign throughout to approve of certain iterations in the protocol. In the following protocol, many new public/private key pairs are generated to keep the mixing key pairs separate from the transaction key pairs. This is to prevent users from being tricked into signing Bitcoin transactions.

### 4.3.1 Announcement

Each participant except for the first participant (Alice in this case) creates a new public and private key pair. Since the public key is used for encryption and the private key is used for decryption, this new key pair will be denoted  $(e, d)$ . This new key pair is used for encryption and decryption of each participant's output address. Alice being participant 1 does not have an encryption/decryption pair. Bob has the key pair  $(e_B, d_B)$  and Charlie has the key pair  $(e_C, d_C)$ . In Figure 7, Charlie's new public encryption key is shown by the yellow lock and his private decryption key is shown by the yellow key. Bob's new public encryption key is shown by the blue lock and his private decryption key is shown by the blue key.

Then the participants digitally sign their new public encryption key and the amount of Bitcoins  $\beta v$  to be shuffled with the private key corresponding to their input addresses. For example, Bob's signature is  $Sig_{k_{priv_B}}(e_B, \beta v)$ , and Charlie's signature is  $Sig_{k_{priv_C}}(e_C, \beta v)$  which are then broadcasted. Therefore, all of the participants know all of the encryption keys  $(e)$ . All of the participants in the figure have a value of  $\beta_1$  in their input address for simplicity, but any number of Bitcoins can be mixed. When each participant receives a signature from all other participants, the participant ensures that the input address  $k_{pub}$  has at least  $\beta v$  to spend. For instance, Alice verifies that Bob and Charlie have at least  $\beta v$  at  $k_{pub_B}$  and  $k_{pub_C}$  respectively upon receiving the signature. If this check fails, the blame

phase is entered which is discussed in section 4.3.4.

### 4.3.2 Shuffling

Next, each participant creates another new public/private key pair  $(k'_{pub}, k'_{priv})$  and therefore the new corresponding Bitcoin address  $k'_{pub}$ . The public key  $k'_{pub}$  is the output address in the mixing transaction. The private key is kept secret and is reserved for spending the mixed Bitcoins corresponding to the new output address. At the end of the transaction these addresses are public, but no one knows which one belongs to which of the users since they are shuffled in an arbitrary order. In the figure, Alice's new output address is  $X/k'_{pub_A}$ , Bob's new output address is  $Y/k'_{pub_B}$ , and Charlie's new output address is  $Z/k'_{pub_C}$ . All of the output addresses are kept private and only known to the owner of the output address. Let  $L_i$  denote the shuffled list of output addresses, at the  $i$ th step.

First, participant 1 uses the published encryption keys  $e$  from all the other participants to encrypt their output address. Alice's output address is  $X$ , which she first encrypts with Charlie's encryption key  $e_C$ . This is shown by  $e_{e_C}(X)$ . Then it is encrypted once more with Bob's encryption key  $e_B$  giving  $e_{e_B}(e_{e_C}(X))$ . Now,  $L_1 = e_{e_B}(e_{e_C}(X))$ . Alice then signs  $L_1$  with her private key corresponding to her input address  $(k_{priv_A})$  denoted by  $Sig_{k_{priv_A}}(L_1)$ . Alice then sends it to participant 2, who is Bob.

When Bob receives  $L_1$  from Alice, he decrypts one layer from each output address in  $L_1$  with his private decryption key  $d_B$ . This is shown by  $d_{d_B}(e_{e_B}(e_{e_C}(X))) = e_{e_C}(X)$ . Bob does not know what Alice's output address is because it is encrypted with Charlie's key. Then Bob encrypts his output address  $Y/k'_{pub_B}$  with Charlie's public encryption key  $e_C$  to get  $e_{e_C}(Y)$ , which is then added to  $L_1$ . Now,  $L_2 = e_{e_C}(X)$  and  $e_{e_C}(Y)$ . Then every item in  $L_2$  is randomly shuffled and  $L$  is digitally signed  $Sig_{k_{priv_B}}(L_2)$  with Bob's private key corresponding to his input address.  $L_2$  is then sent to Charlie, who is the last participant.

When Charlie receives  $L_2$  from Bob, he decrypts one layer from each output address in  $L_2$  with his private decryption key  $d_C$ . This is shown by  $d_{d_C}(e_{e_C}(X)) = X$  and  $d_{d_C}(e_{e_C}(Y)) = Y$ . Charlie knows the output addresses  $X$

and  $Y$ , but he does not know which output address belongs to who. Then Charlie adds his output address  $Z$  to get  $L_3 = X, Y$  and  $Z$ . Then every item in  $L_3$  is randomly shuffled and  $L_3$  is digitally signed  $Sig_{k_{priv_C}}(L_3)$  with Charlie's private key corresponding to his input address so the shuffled list  $L$  is broadcasted to all other participants.

Each participant verifies that their output is included in the list of shuffled output addresses. Alice ensures her output address  $X$  is in the shuffled list, Bob ensures his output address  $Y$  is included in the shuffled list and Charlie ensures his output address  $Z$  is included in the shuffled list.

### 4.3.3 Transaction Verification

If each user's output is included in the list, a new transaction is created that spends  $\text{€}v$  from the input addresses  $T_{in} = (A, B, C)$  and sends  $\text{€}v$  to the new shuffled output addresses  $T_{out} = (Z, X, Y)$ . Then every participant digitally signs the transaction with their Bitcoin signing key  $k_{priv}$  and then shares the signature among participants. Once the signatures are received, the input addresses are double checked to have at least  $\text{€}v$  to ensure none have been spent in the shuffling process. The signatures are added to the transaction, which can then go to the mining process discussed in section 3.3.

### 4.3.4 Blame

This phase protects honest users from malicious users. If any suspicious activity is detected in the CoinShuffle protocol, the blame phase is entered. Suspicious activity includes, but is not limited to, not enough Bitcoins in the input address, a missing signature, or a missing output address. In this phase, the malicious user is identified depending on the error. In the last part of Figure 7, Charlie is missing a digital signature and therefore he would be identified as the malicious user as he tried to spend Bitcoins at an address that does not belong to him. If a participant does not have enough Bitcoins at their input address, they would be identified as the malicious user. There are several other ways a malicious user is identified and the article goes into detail based on the error, but this is beyond the scope of the paper. Once the malicious user is identified, the remaining honest users start the mixing process from the beginning without the malicious user. [10]

## 4.4 CoinShuffle Privacy Analysis

In the protocol, the shuffling participants do not learn the relationship between an input address to its corresponding output address. The only thing the other participants knew about each other is the input address, the amount of Bitcoins they want to mix, the public encryption key, and the list of shuffled output addresses. The input address and the amount of Bitcoins is already public information. The keys used in encryption during the protocol are never used again. The list of shuffled output addresses is also public information but the participants do not know which output address is whose. Overall, this protocol allows users to mix Bitcoins to decrease the correlation between input and output addresses in a decentralized way without giving away any additional details that would threaten their privacy.

## 5. CONCLUSION

Blockchains are growing in popularity with its well-known characteristics such as decentralization, security, immutabil-

ity, anonymity and peer-to-peer model. It is a developing technology with some challenges where possible solutions to those challenges are evolving. In this paper, we discussed Bitcoin's blockchain implementation in the financial industry and the concern on protecting user privacy, with the proposed CoinShuffle solution. It is important to address Bitcoin's blockchain limitations to improve upon them. This emerging blockchain technology can be applied to other industries due to its desirable characteristics [11] and opens the possibility to new blockchain uses.

## Acknowledgments

Thank you to Elena Machkasova, K.K. Lamberty, and Humza Haider for their guidance and feedback.

## 6. REFERENCES

- [1] E. Androulaki, G. O. Karame, M. Roeschlin, T. Scherer, and S. Capkun. Evaluating user privacy in bitcoin. In *International Conference on Financial Cryptography and Data Security*. IEEE, 2015.
- [2] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten. Sok: Research perspectives and challenges for bitcoin and cryptocurrencies. In *2015 IEEE Symposium on Security and Privacy*, 2015.
- [3] H. Corrigan-Gibbs and B. Ford. Dissent: Accountable anonymous group messaging. In *Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS '10*, pages 340–350, New York, NY, USA, 2010. ACM.
- [4] S. Driscoll. How bitcoin works under the hood, 2013.
- [5] D. D. F. Maesa, A. Marino, and L. Ricci. Uncovering the bitcoin blockchain: An analysis of the full users graph. In *2016 IEEE International Conference on Data Science and Advanced Analytics*, 2016.
- [6] G. Maxwell. Coinjoin: Bitcoin privacy for the real world, August 2013.
- [7] U. Mukhopadhyay, A. Skjellum, O. Hambolu, J. Oakley, L. Yu, and R. Brooks. A brief survey of cryptocurrency systems. In *Privacy, Security and Trust (PST), 2016 14th Annual Conference*. IEEE, 2016.
- [8] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.
- [9] C. Paar and J. Pelzl. *Understanding Cryptography*. Springer, 2010.
- [10] P. M.-S. Tim Ruffing and A. Kate. Coinshuffle: Practical decentralized coin mixing for bitcoin. In *European Symposium on Research in Computer Security*. Springer, Cham, 2014.
- [11] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang. An overview of blockchain technology: Architecture, consensus, and future trends. In *2017 IEEE 6th International Congress on Big Data*. IEEE, 2017.