MONASH University

Anonymity in Bitcoin

This thesis is presented in partial fulfilment of the requirements for the degree of Master of Networks and Security at Monash University

By:

Dimaz Ankaa Wijaya

Supervisors:

Dr. Joseph Liu

Dr. Ron Steinfeld

Year:

2016

Declaration

I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institute of tertiary education. Information derived from the work of others has been acknowledged.

Signed by

[illegible]

Name	Dimaz Ankaa Wijaya
Date	17 June 2016

Acknowledgement

I would like to express my sincere gratitude to my supervisors Dr. Joseph Liu and Dr. Ron Steinfeld for providing their expertise and vast experience during the research.

I thank my course coordinator Dr. Nandita Bhattacharjee for the academic advices and recommendations since the first day I arrived at Monash University.

I would also thank Dr. Phu Dung Le who has been an exemplary mentor, allowing me to improve my skill sets, my way of thinking, and my way of conveying ideas to others.

I acknowledge Indonesia Endowment Fund (Lembaga Pengelola Dana Pendidikan) of Ministry of Finance, Republic of Indonesia, for the full scholarship granted to finance my study.

I will always be in debt to my parents and my mother-in-law, for the limitless love and support, allowing me pursuing my dream.

I also owe my life to my wife Indah Kurniawati and my beloved daughter Sophia Esther Wijaya for trusting me and becoming faithful companions along the way.

Abstract

Bitcoin is a pioneer in cryptocurrency in which no trusted party is needed to run the system. Instead, it uses cryptographic techniques to verify and validate the transactions in the system. Bitcoin uses peer-to-peer network and each node in the system maintains a copy of the database of all transactions in the Bitcoin system. As a consequence of the absence of a central authority to organize the system and the transactions within the system, every transaction can be seen, verified, and validated by the nodes interconnected through the Internet constructing a decentralized system. By implementing such system, any unauthorized change towards the transactions must face a difficult task of changing the information not only in 1 or 2 nodes but in at least 51% of the nodes for the change to be approved.

The decentralized system comes with a price. All of the transactions are visible to public. Although the original privacy model of Bitcoin ensures that there is no direct relationship between the Bitcoin addresses and the Bitcoin users, many studies show that these relationships can be identified by conducting analysis of the transactions. Solutions have been offered to increase the anonymity of Bitcoin transaction. However, weaknesses have been identified to be exist in these solutions.

This thesis presents an improved protocol for anonymizing Bitcoin transactions. The protocol relies on a group of middlemen and creates several transactions among these parties which do not hold any specific information that correlates all of the transactions. The protocol maintains the anonymity of the payer from the payee even if the payee colludes with at most one middleman while trying to recover the identity of the payer. The protocol is secure against money stealing cheating by the participants while it maintains compatibility with the existing Bitcoin infrastructure. The proposed protocol is the first to have all of the desirable properties.

Table of Contents

List of Tables

List of Figures

Chapter 1 – Introduction

1.1. Preamble

Bitcoin is a digital payment system introduced by Satoshi Nakamoto (Nakamoto, 2008). The system is a technology breakthrough that enables people to pay each other without the need of trusted party. Eliminating a trusted party in a payment system means that the verification of the validity of the transaction shall be done in a different way, which is where cryptography methods take part. Bitcoin does not require any trusted party; it runs in a peer-to-peer network and the system is built over a decentralized mechanism. To make sure that only valid transactions are confirmed in the system, Bitcoin system verifies these transactions before they are confirmed.

Bitcoin utilizes a database called blockchain to store the confirmed transactions. Inside the blockchain, there are blocks containing transactions which are then hashed and linked each other in a Merkle Tree (Merkle, 1980). To create links between blocks, the hash value of previous block is included inside the current block and then hashed. The hash value needs to satisfy certain difficulty level to be considered as a valid block. If the block is valid, then the network includes the block into the blockchain. The information stored in the blockchain is protected by a scheme called Proof of Work (PoW) to avoid the information from being altered (Nakamoto, 2008).

Bitcoin does not require any centralized organization and thus it can be called as a bank less payment system. It also does not require the involvement of any government in the world; the rules are determined in a voting system by Bitcoin miners. In the Bitcoin system, the generation of new bitcoins are limited to a maximum of 21 million bitcoins. The new bitcoins are minted by those who have hashing power called Bitcoin miners. For each blocks produced, the miners will be rewarded 50 bitcoins. The reward of the bitcoin will be cut in half every roughly 4 years, and by the time this paper is written, the reward is 25 bitcoins for every blocks produced.

Bitcoin is not only popular because of its simplicity but also its property of enabling users to anonymously create financial transactions. However, this is not always the case. Bitcoin is no longer considered as a fully anonymous system but rather as a pseudo-anonymous; some properties can still be exploited to reveal some information about the users and the transaction itself. The properties of the Bitcoin transactions can be analyzed to reveal information about the users involved in the transactions. The goal of the research is to design an improved protocol to increase the anonymity of Bitcoin transactions.

1.2. Problem

The issue of privacy has been around since Chaum (1983, 1985) proposed his ideas of protecting the information of individual’s financial records in the computerized world. The banking system collects all of the users’ records and by wrong intention it could be used to monitor the activity of the money owners. Those who do not want to be monitored prefer to choose cash and coins, but this physical method also creates problems such as robbery, theft, and inconvenience in creating transactions especially when they involve a large sum of money.

The privacy of financial activities has always been a concern for the cryptography community (Bonneau et al., 2014). The concern is reflected in the construction of Bitcoin, in which the inventor of Bitcoin asked for suggestions from the community members and even tested the system within the community members. Bitcoin is originally designed to have a degree of anonymity. A user does not need to provide any identification card to start using Bitcoin in daily life. To start using the Bitcoin, a user only need to generate a public key pair. Its public key will be the Bitcoin address to receive money and the private key will be the secret value to spend the money. The user also needs to connect to the Bitcoin network which is available through the Internet.

However, many researchers have shown that some aspects of Bitcoin transaction and policies enforced by government bodies can be used to reveal the identity of Bitcoin users. Anyone connected to Bitcoin system can investigate every transaction created by a user, including the transaction history of the money contained in an address, allowing unauthorized monitoring of financial transaction of the user. The problem gets worse because the bad guys are capable of doing the same thing assuming that the same information is obtained by the bad guys.

For those Bitcoin users requiring a higher level of privacy, several efforts can be made to increase the anonymity. In anonymizing the coins, the users must be careful because they will face several problems and risks in using coin mixing services. There is no guarantee that the services will increase the anonymity of the users’ bitcoin and some characteristics may still persist in the coins after using the services.

1.3. Aim

This research aims to provide a new approach of anonymizing Bitcoin transaction. By studying related research about how Bitcoin transactions are analyzed and distinguished between one and another, the new approach minimizes the characteristics of those transactions which are used to analyze. Existing systems was studied to discover the strengths and weaknesses, so that improvements could be made over those systems. The proposed approach provides an improved way to create more anonymous Bitcoin transactions.

1.4. Goals

The goal of this research is to construct an improved model of anonymizing Bitcoin transaction which will give a higher degree of anonymity for the Bitcoin users. After investigating the nature of Bitcoin transactions and how they can be analyzed to gather information about the pattern and the users, it is important to create a mechanism to protect these piece of information from unwanted event which may happen to the Bitcoin users.

The research goals of this work can be specified as follows:

- Create an anonymizing protocol which is compatible with the current Bitcoin network.
- The anonymizing protocol must protect the information of the payer. The payee cannot identify which Bitcoin addresses belong to the payer.
- No participant should be able to get information on the transactions within the protocol.
- It must be infeasible for any adversary outside of the protocol to link the transactions created by the protocol.
- No participant can steal the money.

As a final result, a protocol is constructed in order to fulfil the requirements above. The protocol helps the Bitcoin users to anonymize the transactions created in the current Bitcoin system.

1.5. Thesis structure

The thesis is structured as follows.

- In Chapter 2 (Literature Review), the detail about the Bitcoin system is described, as well as several technologies that underpin the system.
- In Chapter 3 (Security Model), the preferred characteristics of the anonymity and the cheating scheme are presented. These 2 characteristics are expected to be fulfilled by the outcome.
- In Chapter 4 (The Protocol), the detailed research outcomes are presented, including several transaction protocols which were considered as potential candidates to be used in the protocol and their evaluation, leading to the final proposed protocol.
- In Chapter 5 (Implementation), an implementation of the protocol in real Bitcoin transactions is presented, using tools which will also be discussed in this chapter.
- In Chapter 6 (Evaluation), the protocol is evaluated against the security models presented in Chapter 3. The trade-offs are also discussed. These trade-offs are taken in exchange of features required by the solution.
- In Chapter 7 (Conclusion), the summary of the research is provided. The contribution of the research, challenges and limitations, and suggestion for future research are also highlighted.

Chapter 2 - Literature Review

2.1. Introduction

This chapter describes the Bitcoin as a system which provides the capability of transferring values between users and identifying the research gaps. The literature review contains studies about the Bitcoin, the technology that constructs the system, and how a Bitcoin transaction is created. The chapter is structured in several sections: (a) Bitcoin system; (b) cryptographic technology; (c) Bitcoin transaction; and (d) privacy issue in Bitcoin.

2.2. Bitcoin System

Bitcoin replaces trusted party by cryptographic techniques to verify transactions. The idea of employing cryptographic technique in an electronic money was first proposed by Dai (1998) in a concept called B-money. Several cryptographic techniques used in the Bitcoin core are hashing algorithms, public key cryptography, and digital signature. More cryptographic concepts are later added into the system such as zero knowledge proof (Bitcoin Wiki, 2011e) and Shamir secret sharing (Shamir, 1979) for the users to get more features out of the payment system.

Despite Bitcoin does not need a central authority, some parties are needed to run the system. These parties can be explained as follows.

a. Nodes

Nodes are the ones that keep the blockchain for the system. They are involved in peer-to-peer Bitcoin network by relaying transactions from other nodes. These nodes are provided by different parties to support the continuity of Bitcoin. Big vendors run their own nodes to verify transactions before publishing them into the network and to receive valid confirmation regarding to their financial activities with their clients.

b. Miners

Miners are the ones that dedicate their resources (time, CPU power, electricity) to add the transactions into blocks and add blocks into the blockchain. The miners are rewarded fresh-minted bitcoins each time they managed to create a valid block which is then added into blockchain. The reward was started at 50 bitcoins and will be cut to half every 4 years. By the time this research is conducted, the reward is 25 bitcoins and it will become 12.5 bitcoins in the next few months. The miners also receive transaction fee paid by users. The transaction fees may vary depending on the size of the transaction; the bigger the size, the higher the transaction fee. The miners can mine the blocks on their own or they can also join a mining pool. A mining pool distributes jobs to each of the miner and shares the reward to the miners based on their contributions.

c. Users

Users are the ones that utilize the Bitcoins to create payments from one to another. Mostly they only need wallets to store information about their Bitcoin addresses and the corresponding private keys used to spend the fund held in the addresses they own. The wallets may also serve to keep record of total fund of the owners for the convenience.

2.2.1. Distributed Database

One way to achieve a trustless electronic cash is by distributing the database, so everyone can verify the validity of the transactions (Nakamoto, 2008). The transactions are grouped in blocks generated every 10 minutes by miners. A new block is added into the top of the chain and linked to the last block by adding the hash value of that last block into the hash value of the new block, therefore if someone wants to change transactions from previous blocks, then he/she has to redo all the works of the next blocks until the last blocks in the blockchain (Nakamoto, 2008). The blockchain is a distributed database; Bitcoin nodes keep and maintain identical copies of the full transaction records in the blockchain. By applying this method, the system becomes more robust and has avoided a single point of failure (Franco, 2015). Beside the blockchain, there is also another database kept by the nodes called Unspent Transaction Output (UTXO). The UTXO maintains all transaction outputs that are not yet spent. These records are kept in the RAM because they need to be accessed in a fast fashion (Franco, 2015).

2.2.2. Bitcoin Users

Bitcoin was first introduced in a cryptographic mailing list (Bartlett, 2014) and gained wide attention after many people realized that its characteristics are sufficient to fulfill their need of an anonymous payment method. Because of its rising popularity, Bohr and Bashir (2014) conducted a research to determine the profiles of Bitcoin users. By using datasets collected by Smyth (2014), they carried out statistical analysis. The datasets mapped users' demographic information such as age, location, and gender, as well as other information such as political ideology. The result provides an insight that most Bitcoin users (at the time the research was conducted) reside in United States and they have average age of 25 to 35 years old. The research also points out users' intention of buying illicit goods (drugs, narcotics) with their bitcoins. The finding mentions about the connection between political view and Bitcoin users, in which the majority of the users are libertarian, and by looking at the comments of those users during the data gathering activity, bitcoins are used by those who want to be free from the current monetary system controlled by banks.

The anonymity of Bitcoin has attracted people to create transactions of illicit goods as expressed by J. Martin (2014) during his research in popular drug website which only existed in dark net, Silk Road. By accepting bitcoins, certain degree of anonymity will be in favor of the users and they can pay the goods without worrying of being traced back to their real identity. Bryans (2014) determined the connection between Bitcoin and its usage in money laundering scheme because of its decentralized system and anonymity. While the paper describes the similar level of anonymity between Bitcoin and cash, Bitcoin itself can be used without face-to-face transaction because it relies on peer-to-peer network technology.

2.2.3. Bitcoin Network

There are at 2 types of network in Bitcoin. The first one is called Mainnet; it is where the Bitcoin system is run. The other network is called Testnet ,which can be used to test new applications or new features related to Bitcoin without disturbing the existing system (Bitcoin Wiki, 2011c). The current testnet is Testnet3. There are differences between Mainnet and Testnet; some of them are be described in Table 1.

Table 1. Differences between Mainnet and Testnet

Feature	Mainnet	Testnet3
Listen Port	8333	18333
RPC Connection Port	8332	18332
Address Version	0x00	0x6F
Message Header	0x0B110907	0xF9BEB4D9
IsStandard Check	Yes	No

Despite Testnet3 seems to support nonstandard transaction by not applying IsStandard checking procedure, based on experiments Testnet3 does not relay the nonstandard transaction as expected. Moreover, the nodes running Testnet3 do not seem to be properly updated to the latest version of Bitcoin core software, and therefore it is impossible to construct a transaction utilizing new features of Bitcoin protocol in Testnet3.

2.3. Cryptographic Technology

To eliminate the need of trusted party such as banking institution, Bitcoin system employs cryptographic techniques within its system, including public key cryptography, digital signature, and hash functions. Public key cryptography is used to proof the ownership of bitcoins by means of digital signature. Hash functions are also used in some parts of the system to validate information.

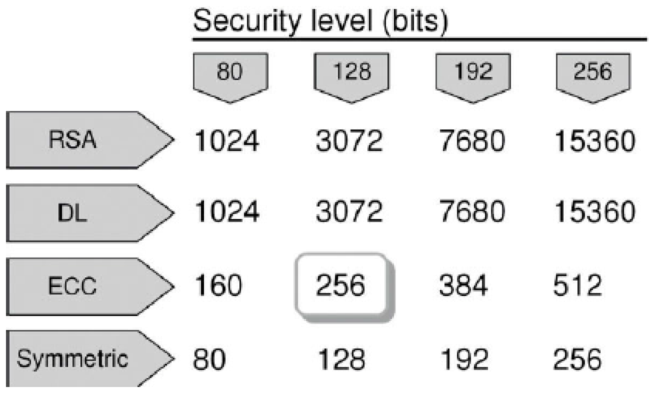
2.3.1. Public Key Cryptography

Public key cryptography is developed to answer the challenge of exchanging the keys within private-key systems in a secure manner (Stinson, 1995). In private-key systems, a communication between 2 parties, say Alice and Bob, shares the same secret key K. Whenever Alice wants to send a message M to Bob, she encrypts the message by using an encryption function Q = E(M, K) and sends the encrypted message to Bob. To recover the encrypted message, Bob uses decryption function M = D(Q, K). The biggest problem in the shared secret key is the secret key distribution. For Alice and Bob to communicate securely by using a private key system, they first need to exchange the same secret key. This can be a problem because sharing the secret key securely is a hard task. If the secret key K is exposed to other party, the scheme becomes insecure. The encrypted message Q can be decrypted by an adversary to get the original message M.

To eliminate such problem, the first idea of public key cryptography was proposed by Diffie and Hellman (1976). Public key cryptography uses two keys instead of one as in private-key cryptography, named private key and public key. In public key cryptography, given the public key, it is infeasible to calculate the private key. While the public key can be published freely, the private key is intended to be kept secret by the owner. By using this scheme, if Alice wants to send a message M to Bob, she uses Bob's publicly available public key to encrypt the message and later Bob can decrypt the message by using his private key. Rivest, Shamir, and Adleman (1978) introduced a public key cryptography called RSA cryptosystem based on integer factorization. RSA is the first implementation of public key cryptography scheme. Some other implementation of public key cryptography are ElGamal (ElGamal, 1985) and Elliptic Curve.

2.3.2. Elliptic Curve Cryptography

Bitcoin system implements Elliptic Curve Cryptography (ECC) because of some reasons. Compared to other public key cryptography, ECC has a small keysize as shown in Figure 1. Because of the small keysize, the computation can be done faster than the ones with bigger keysize (Franco, 2015). ECC with 256 bits keysize as used in Bitcoin system has an equal security level to 128 bits keysize of symmetric cryptography and 3072 bits keysize of RSA system.



	Security level (bits)			
	80	128	192	256
RSA	1024	3072	7680	15360
DL	1024	3072	7680	15360
ECC	160	256	384	512
Symmetric	80	128	192	256

Figure 1. Keysize Comparison Between Public Key Cryptosystems (Franco, 2015)

In elliptic curve, the public key is seen as a point, while the private key is a number which defines steps from a starting point called generator to the point where the public key points. In the elliptic curve protocol, it is easy to calculate the public key given the private key, but it is infeasible to calculate the private key given the public key.

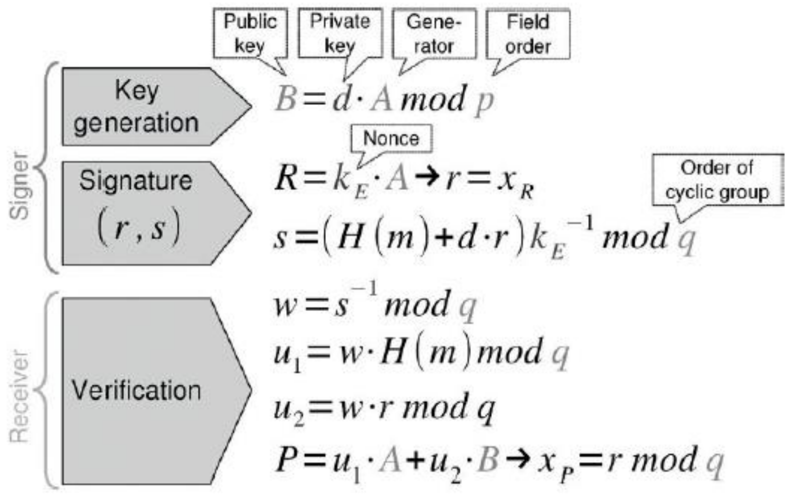
For the implementation of Bitcoin system, Satoshi chose Secp256k1 (Brown, 2010) as the standard for the Elliptic Curve Cryptography. It specifies the parameters of elliptic curve (p,a,b,G,n,h) as follows:

p	=	FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFC2F
a	=	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
b	=	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000007
G	=	04 79BE667E F9DCBBAC 55A06295 CE870B07 029BFCDB 2DCE28D9 59F2815B 16F81798 483ADA77 26A3C465 5DA4FBFC 0E1108A8 FD17B448 A6855419 9C47D08F FB10D4B8
n	=	FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE BAAEDCE6 AF48A03B BFD25E8C D0364141
h	=	01

2.3.3. Digital Signature

Digital signature is a mean of proving the identity of a user which can be verified by anyone else. Digital signature makes use of public key cryptography. Consider Alice wants to send message to Bob via insecure communication. First, Alice creates a public key pair which consists of private key and public key. She sends her public key to Bob and keeps the private key to herself. Then, Alice creates the digital signature by using a digital signature algorithm, her public key, and the message to be sent. The digital signature is then appended in her message and sent to Bob. Bob can verify the digital signature by using Alice’s public key. If the verification result is the same as the message, then the message is proven to be sent by Alice.

Bitcoin uses Elliptic Curve Digital Signature Algorithm (ECDSA) to provide a digital signature scheme. The signature generation and verification are explained Figure 2.



Public key
Private key
Gene-rator
Field order
Key generation
$B = d \cdot A \bmod p$
Nonce
Signature (r, s)
$R = k_E \cdot A \rightarrow r = x_R$
Order of cyclic group
$s = (H(m) + d \cdot r) k_E^{-1} \bmod q$
Signer
Verification
$w = s^{-1} \bmod q$
$u_1 = w \cdot H(m) \bmod q$
$u_2 = w \cdot r \bmod q$
$P = u_1 \cdot A + u_2 \cdot B \rightarrow x_P = r \bmod q$
Receiver

Figure 2. Bitcoin Digital Signature Scheme (Franco, 2015)

2.3.4. Hash Functions

A Hash function is an algorithm which calculates an input of any length to an output of specific length. It has some characteristics. Firstly, although the length of the output can be any length, the output remains the same, and the calculation requires small amount of time to produce the result. Secondly, with the same input, the same hash function always produces the same hash value. In Bitcoin, there are several hash functions that are used to serve its function, including SHA256 and RIPEMD160.

2.3.4.1. SHA-256

SHA256 is a standard hash function published by NIST in 2001 as one of the SHA-2 family. It defines the hash value it produces to have a fixed length of 256 bits. Bitcoin uses SHA256^2, which is defined as performing the SHA256 twice. It is done because of Satoshi's awareness of length extension attack which can be used to append a new message into the existing message and calculate the correct hash value of the combined message. The SHA256^2 operation is used in several parts of the system, including in the mining process of a Bitcoin block.

2.3.4.2. RIPEMD-160

RIPEMD (RACE Integrity Primitives Evaluation Message Digest) is a hash algorithm based on MD4. It generates 160 bits of hash value. In Bitcoin, RIPEMD160 is used in address generation which requires a public key as an input (Bitcoin Wiki, 2014).

2.3.5. Shamir Secret Sharing

Shamir (1979) introduced a secret sharing mechanism by using a mathematical equation called (k,n) threshold scheme where k < n. In the scheme, a secret value D is divided into n pieces and can be retrieved back by only using k pieces of total n available. To create the scheme, a polynomial of degree k-1 is used with

$$q(x) = a_0 + a_1x + \cdots + a_{k-1}x^{k-1}$$

where $D = a_0$ is the secret value which will be destroyed once n values are created. In the scheme, q(1), q(2), ..., q(n) are defined and distributed to different parties. If only k number of different parties agree to reveal the secret value D that the secret value D can be calculated by using polynomial interpolation. By using k number of different q(x), $a_1, a_2, \dots, a_{k-1}$can be determined and then a_0 can be calculated.

Shamir Secret Sharing is in Bitcoin system to protect a private key by dividing the information into n number of pieces and distributed among several parties. In order to reveal the private key, k number of pieces must be obtained. This method offers more security rather than storing the private key in one place.

2.3.6. Secure Multiparty Computation

The idea of secure multiparty computations were first introduced by Yao (1982) to enable information multi-party computation without worrying that the involved parties learn about the inputs. Secure multiparty computations isused to solve the millionaires' problem in which two millionaires want to know who is richer between them but they do not want to reveal their amount of wealth. Andrychowicz, Dziembowski, Malinowski, and Mazurek (2014) studied the usage of secure multiparty computations (MPC) into Bitcoin environment in which a trusted party is not needed in term of betting some bitcoins and how to determine the winner without the possibility of any party cheating the protocol.

2.3.7. Zero Knowledge Proof

Zero knowledge proof is a method of proving the possession of an information without revealing the information (Franco, 2015). Two parties involved in the scheme are prover and verifier. The prover holds the answer of a problem while the verifier asks questions regarding to the problem to the prover and verifies whether the answers provided by the prover. The idea of zero knowledge proof in Bitcoin system is implemented by using hash functions in term of binding transactions by using a password without revealing the password yet (Bitcoin Wiki, 2011e). While the verifier does not have the information about the password, the prover cannot change the value of the hash which has been informed to the verifier.

2.4. Bitcoin Transaction

Bitcoin transactions are series of linked information, in which a transaction has at least one input and one output. An input of a transaction refers to an output of other previous valid transaction. The transaction is basically a mathematical puzzle, whereas the input part is the answer of the referred transaction output, and the output is the puzzle that needs to be solved by any future transaction which will spend the output (Franco, 2015). A Bitcoin transaction consists of a set of transaction inputs called TxIn and a set of transaction outputs called TxOut. TxIn answers the mathematical puzzle of a referenced transaction while TxOut questions. All of these transactions create a chain as expressed by the transactions reference scheme. The transaction referencing scheme and the place where TxIn and TxOut reside in the transaction are explained in Figure 3.

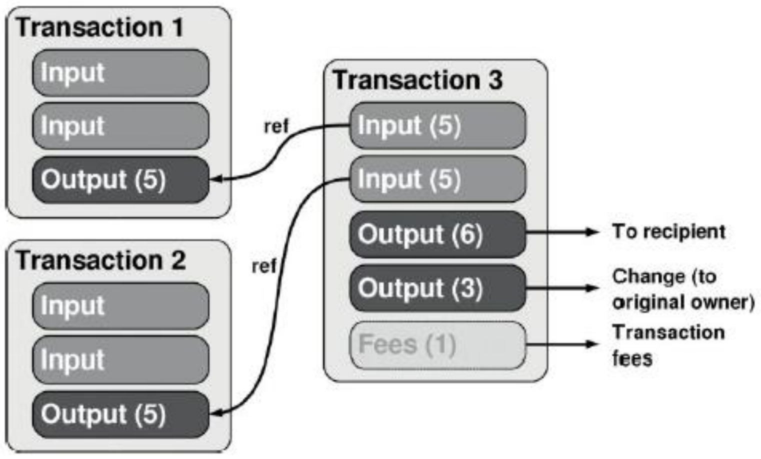


Figure 3. Bitcoin Transaction (Franco, 2015)

A regular bitcoin transaction consists of at least 1 input and 1 output. It also determines the amount of bitcoins which are spent by the input and sent to the output. The difference of input and output will become the transaction fee which will be collected by the miner who includes this transaction into Bitcoin blockchain. The process of Bitcoin transaction is different than traditional financial transaction. In Bitcoin transaction, all of the amount of bitcoins held by an address as an input are sent to the system, but if there is a difference between the total amounts of bitcoins held by the address and the amount of bitcoins transacted, the difference is sent back to the owner via a change address.

2.4.1. Bitcoin Address

Bitcoin address is generated from the Elliptic Curve public key scheme (Bitcoin Wiki, 2011b). The process can be described in Figure 4.

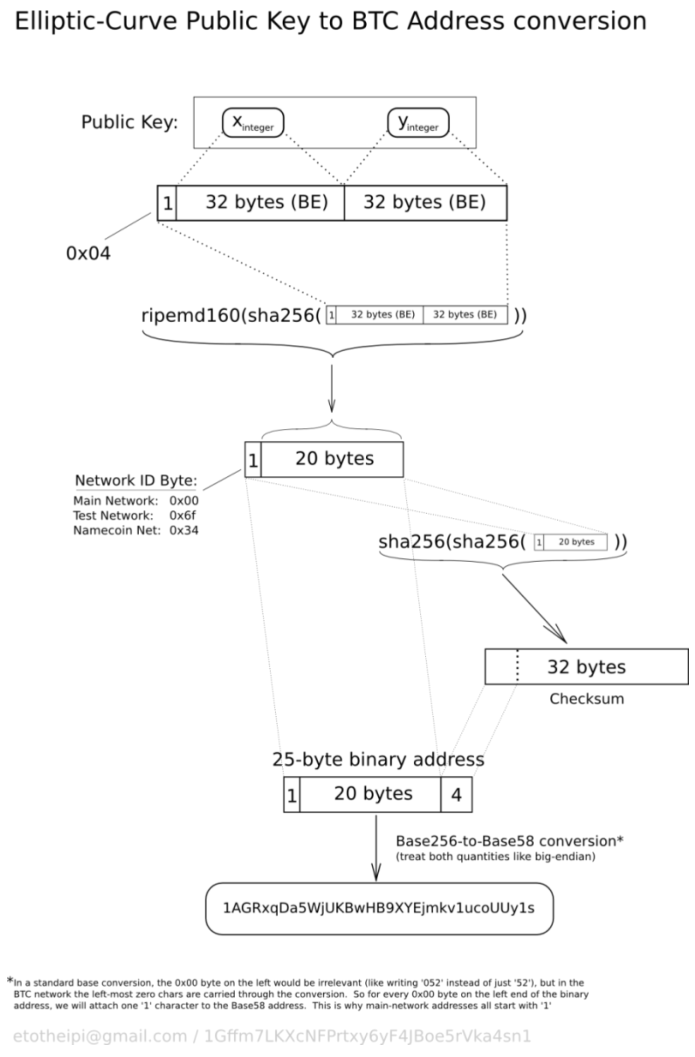

Elliptic-Curve Public Key to BTC Address conversion
Public Key:
$X_{integer}$
$Y_{integer}$
1
32 bytes (BE)
32 bytes (BE)
0x04
ripemd160(sha256(1 32 bytes (BE) 32 bytes (BE)))
1
20 bytes
Network ID Byte:
Main Network: 0x00
Test Network: 0x6f
Namecoin Net: 0x34
sha256(sha256(1 20 bytes))
32 bytes
Checksum
25-byte binary address
1
20 bytes
4
Base256-to-Base58 conversion*
(treat both quantities like big-endian)
1AGRxqDa5WjUKBwHB9XYEjmkv1ucoUUy1s
*In a standard base conversion, the 0x00 byte on the left would be irrelevant (like writing '052' instead of just '52'), but in the BTC network the left-most zero chars are carried through the conversion. So for every 0x00 byte on the left end of the binary address, we will attach one '1' character to the Base58 address. This is why main-network addresses all start with '1'
etotheipi@gmail.com / 1Gffm7LKXcNFPrtxy6yF4JBoe5rVka4sn1

Figure 4. Bitcoin Address Generation (Bitcoin Wiki, 2011b)

There are steps to generate Bitcoin address as explained below.

a. Generate an EC public key pair (private and public key).

b. Take the public key and construct 65 bytes by using following structure.

   1) First part consists of 1 byte 0x04. This is a flag to mark that the point in elliptic curve is uncompressed. To create a compressed point, then 0x02 or 0x03 is used. If a compressed mode is used then only X value is used while Y value will be calculated from X value.

   2) Second part consists of 32 bytes of X coordinate of elliptic curve point.

   3) Third part consists of 32 bytes of Y coordinate of elliptic curve point.

c. Generate hash value from SHA-256 operation of the data constructed in step (2).

d. Generate hash value from RIPEMD160 operation of hash value from step (3). The result is 20 bytes data.

e. Concatenate version byte with step (4) operation result. There are some version bytes which can be used, 0x00 is for Bitcoin main network, 0x6f is Bitcoin test network, while 0x34 is for Namecoin network.

f. Generate hash value from double SHA-256 operation of the data in step (5). First 4 bytes of the result will be used as checksum.

g. Concatenate the data in step (5) with the first4 bytes from step (6).

h. Create Base58Check version of data from step (7).

Bitcoin address is used to receive Bitcoin payment which works similar to a bank account number. The sender creates an output in the transaction which can only be spent to whoever controls the private key that corresponds to the address. Therefore, although the address is known to everybody, only the address owner can spend the fund held by the address. By using cryptographic technique, whenever the address owner wants to spend the bitcoins, the owner supplies a valid digital signature. Bitcoin system then examines whether the owner has the correct digital signature associated with the transaction.

2.4.1.1. Deterministic Wallet

A deterministic wallet is a wallet which can generate new addresses in advance and therefore can be backed up once (Maxwell, 2011). It works by generating new address from a single key, for example from a password (Franco, 2015). There are 2 types of deterministic wallet: type-1 and type-2. Type-1 deterministic wallet creates a new addresses from a password concatenated with a counter. While the private keys are generated by hashing the combination of the password and a counter number, the private keys can be recovered at any time and they can be safely deleted. The scheme is explained in Figure 5.

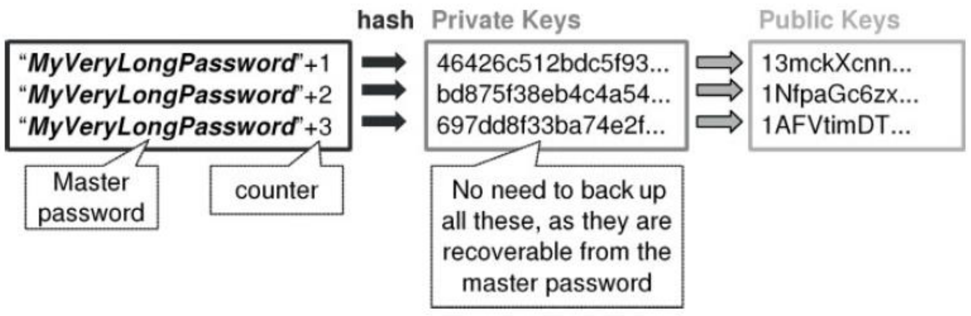


Figure 5. Type-1 Deterministic Wallet (Franco, 2015)

The generation of private keys and public keys of Type-1 Deterministic wallet is described in the following equations.

$$priv = H(pw|n)$$

$$B = priv \cdot A \bmod p$$

where *priv* is the private key, *pw* is the password, *n* is the counter, and *H* is a hash function. *B* is a public key generated by using the private key *priv*, *A*, and *p*.

Type-2 Deterministic Wallet can be explained as follows. The generation of public key B is described in the following equations (Franco, 2015).

$$priv = mpk + H(pw|n)$$

$$B = priv \cdot A \, mod \, p$$

$$B = mpk \cdot A + H(pw|n) \cdot A \, mod \, p$$

$$B = B_{mpk} + H(pw|n) \cdot A \, mod \, p$$

where *mpk* is parent private key and B_{mpk} is the parent public key. This type separates parent private key *mpk* and password *pw*, therefore increases the security and the usability of the scheme. By using this scheme, a user can safely provide a receive-only wallet with her password *pw* and parent public key B_{mpk} and knowing these 2 information will still secure the fund because the private key *mpk* is secured.

2.4.1.2. Hierarchical Deterministic Wallet

Hierarchical Deterministic wallet (HD wallet) is a type of deterministic wallet with capability of deriving addresses from a parent (public or private) key (Wuille, 2012). The deriving process is constructed like a hierarchical tree. The public keys and the private keys can be derived separately. Each public key can derive up to 2^{31} child public keys (Franco, 2015). The child keys can be generated by a parent key but the child keys cannot be used to regenerate the parent key.

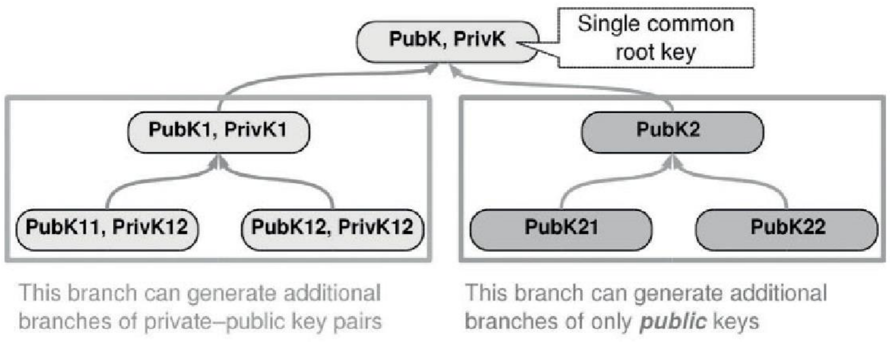

PubK, PrivK
Single common root key
PubK1, PrivK1
PubK11, PrivK12
PubK12, PrivK12
PubK2
PubK21
PubK22
This branch can generate additional branches of private–public key pairs
This branch can generate additional branches of only *public* keys

Figure 6. Key Generation in HD Wallet (Franco, 2015)

2.4.2. Scripts

Bitcoin transaction consists of inputs and outputs. These inputs and outputs are actually constructed by Bitcoin Script. The script in Bitcoin is similar to a programming language called Forth (Bitcoin Wiki, 2013).

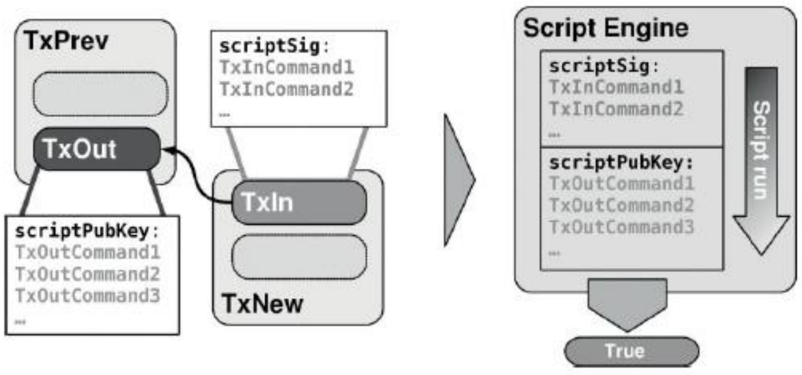


Figure 7. Bitcoin Script (Franco, 2015)

In the common Bitcoin transaction, <*scriptSig*> is used as the TxIn, while <*scriptPubKey*> is used as the the TxOut. The scripts are evaluated from top to bottom and evaluated as stack with LIFO (Last In First Out) method. The script in Bitcoin is not Turing-complete; no iteration is allowed to avoid the network being attacked by complex computation. The calculation types are limited as defined in Bitcoin OpCodes manual (Bitcoin Wiki, 2013). The OpCodes are represented in hexadecimal value (Bitcoin Wiki, 2013).

Bitcoin commands are represented in hexadecimal values (Bitcoin Wiki, 2013). An example of raw transaction is shown below.

```
010000000015fcb28422e3ef3d36f0ea0ed33f965c117c457cce02bc3272902fd8229
fcd4990000000000ffffffff01905f0100000000001976a914937fe2ee82229d282e
dec2606c70e755875334c088ac00000000
```

The raw transaction can be decomposed and explained as follows.

01000000	Protocol version
01	Number of inputs (1)
5fcb28422e3ef3d36f0ea0ed33f965 c117c457cce02bc3272902fd8229fc d499	Input transaction id (little endian of 99d4fc2982fd022927c32be0cc57c417c165f933eda 00e6fd3f33e2e4228cb5f)
00000000	Input transaction index (little endian)
00	Scriptsig length (for unsigned transaction)
ffffffff	Sequence number (hexadecimal value of 4294967295)
01	Number of outputs (1)
905f010000000000	Value of the input in satoshi (little endian of 90000 in hexadecimal)
19	Number of bytes to be pushed to the stack (hexadecimal representation of 25)
76	Hexadecimal representation of OP_DUP Bitcoin OpCode
a9	Hexadecimal representation of OP_HASH160 Bitcoin OpCode

14	Number of bytes to push to the stack (the payload)
937fe2ee82229d282edec2606c70e755875334c0	The payload (big endian)
88	Hexadecimal representation of OP_EQUALVERIFY Bitcoin OpCode
ac	Hexadecimal representation of OP_CHECKSIG Bitcoin OpCode
00000000	Locktime value

A common structure of bitcoin transaction script is shown in Figure 8.

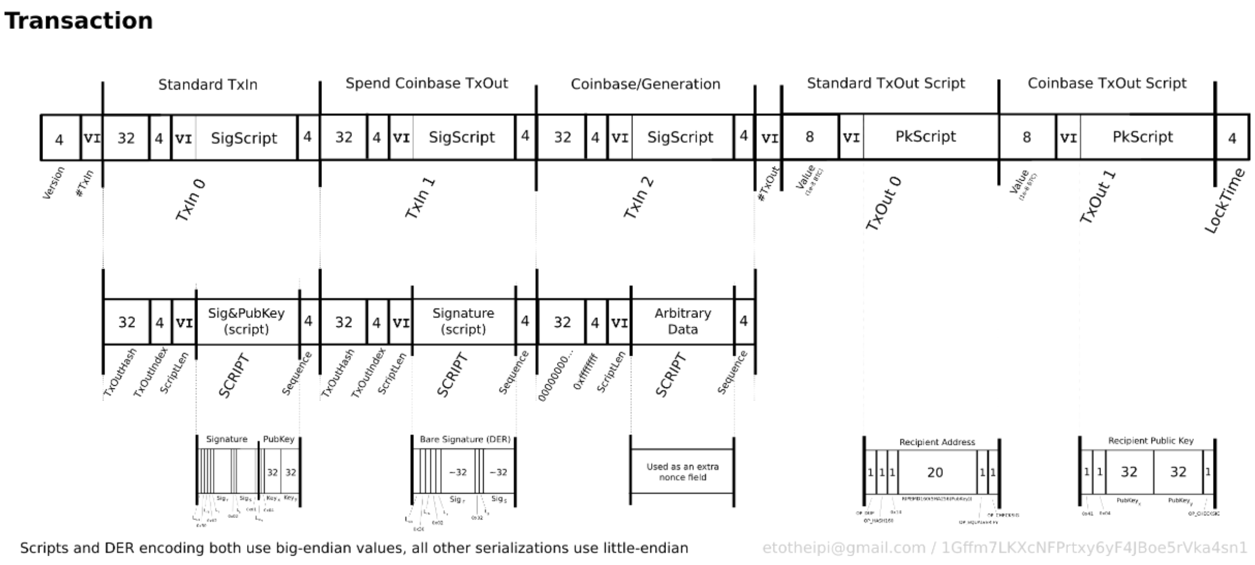


Figure 8. Bitcoin Transaction Script Construction

2.4.3. Locktime

Locktime, or also called as nLockTime, is a feature in Bitcoin system which can be used to determine the earliest time the transaction can be confirmed in the system by using 4 bytes data (Harding, 2015b). The Locktime can be used in 2 ways. If it is less than 500 million, then it means block height, but if it is bigger or equal to 500 million, then it means Unix time format which is calculated as seconds after 1 January 1970 00:00 UTC (Harding, 2015a). Block height can be used as timestamp because Bitcoin system creates a new block every roughly 10 minutes, it means if the locktime is 10 blocks higher than the current block height, then it is locked roughly for 100 minutes. The locktime in Bitcoin transaction raw data is represented as low endian. Locktime of 00000000 means there is no locktime and can be confirmed immediately.

2.4.4. Sequence Number

Sequence number is 4 bytes information in Bitcoin raw transaction which can be used to setup the transaction version (Harding, 2015c). If the sequence number is less than FFFFFFFF (or 4,294,967,295 in decimal), then the transaction is not considered as final and can still be changed. To change the transaction, the next version of the transaction must have higher sequence number than its predecessor.

2.4.5. CheckLockTimeVerify

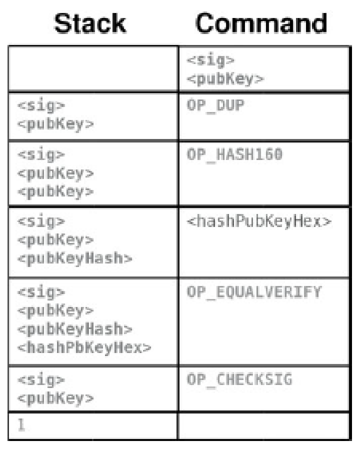
CheckLockTimeVerify (CLTV) is a feature proposed by Todd (2014) to lock a transaction until a certain time. By using CLTV, the transaction can be immediately included in the blockchain but it freezes and cannot be redeemed until a certain time. The CLTV is a redefinition of OP_NOP2 command. The information of CLTV uses a low endian data type. CLTV also works with Locktime. If the Locktime is used, then sequence number must be set to less than maximum value (the maximum value of sequence number is FFFFFFFF) and the value defined in CLTV must be the same or lower than Locktime.

2.4.6. Pay To Address

One of the methods available in Bitcoin transaction is Pay to Address. It is the most common transaction type in Bitcoin system, in which the payer pays the payee by using Bitcoin addresses. It means that the payee can spend the fund by providing her public key. As the transaction works by evaluating a script, the pay to address scheme provides a way for the payer to pay the payee by referencing the payee’s Bitcoin address in the transaction script. The <scriptPubKey> script is structured as follows:

OP_DUP OP_HASH160 <*hashPubKeyHex*> OP_EQUALVERIFY OP_CHECKSIG

The script above is placed in the TxOut, where it will verify any TxIn which wants to spend the fund. The TxIn required to spend this kind of transaction is <*sig*> and <*pubKey*>. The <*sig*> is the digital signature of previous transaction which is referenced by the information inside the digital signature, and it is created by the one who wants to spend the fund, while the <*pubKey*> is the ECC public key of the same owner.



Stack	Command
	<sig> <pubKey>
<sig> <pubKey>	OP_DUP
<sig> <pubKey> <pubKey>	OP_HASH160
<sig> <pubKey> <pubKeyHash>	<hashPubKeyHex>
<sig> <pubKey> <pubKeyHash> <hashPbKeyHex>	OP_EQUALVERIFY
<sig> <pubKey>	OP_CHECKSIG
1	

Figure 9. Pay-to-Address Script Evaluation (Franco, 2015)

The evaluation works as follows.

a. The <*sig*> and <*pubKey*> are pushed into the stack in LIFO method.
b. First command is OP_DUP which duplicates the last entry of the stack, in this case <pubKey>, then we now have <*sig*>, <*pubKey*>, and <*pubKey*> in the stack.
c. OP_HASH calculates the RIPEMD160 hash value of the last entry of the stack. After this command is executed, the stack has <*sig*>, <*pubKey*>, and <*pubKeyHash*>.
d. A value of <*hashPubKeyHex*> is pushed into the stack; it changes the entries of the stack to <*sig*>, <*pubKey*>, <*pubKeyHash*>, <*hashPubKeyHex*>.
e. OP_EQUALVERIFY verifies the last two entries of the stack, which are <*pubKey*> and <*pubKeyHash*>. If the two entries match, then both of them are removed from the stack.
f. OP_CHECKSIG evaluates the validity of the digital signature in <*sig*> by using the <*pubKey*> provided.

At the end of evaluation, the stack returns TRUE if the values match or else it returns FALSE.

2.4.7. Pay To Public Key

Pay to Public Key transaction is similar to Pay to Address. The difference with these 2 methods is that the Pay to Public Key only has the last step in Pay To Address. The <*scriptPubKey*> script as the TxOut is structured as follows

<*pubKey*> OP_CHECKSIG

The <*scriptSig*> contains <*sig*> which is the digital signature of the one who wants to spend the fund. The evaluation works as follows.

a. The <sig> data from the <scriptSig> part is pushed into the stack.
b. The <pubKey> data from the <scriptPubKey> part is pushed into the stack.
c. OP_CHECKSIG evaluates the validity of digital signature <sig> by using <pubKey>.

Although the operation looks simpler compared to Pay to Address scheme, Pay To Public Key scheme results in larger size in transaction data, and as a consequence it requires bigger transaction fee. It is also said that the scheme is prone to quantum computer attack (Franco, 2015).

2.4.8. Pay To Script Hash

Pay To Script Hash (P2SH) is another method of Bitcoin payment (Bitcoin Wiki, 2012b). P2SH is a standard under BIP 16 which describes the detail of P2SH (Andresen, 2012). It enables Bitcoin users to construct a script as a requirement before redeeming the fund. P2SH is added into Bitcoin system mainly in order to support multisignature without the need of describing the multisignature requirements in the ScriptPubKey. By using P2SH, the payer only needs to supply the hash of the script and therefore will make the transaction fee cheaper for the payer. P2SH transaction is marked to have the following format in its ScriptPubKey:

OP_HASH160 <20-byte-hash-value> OP_EQUAL

The 20 byte hash value is the hash value of the script supplied by the redeemer. Besides the script, the redeemer also need to supply the inputs which unlocks the script. The inputs are evaluated to determine whether these inputs result in TRUE under the given script.

2.4.9. Multisignature

Multisignature is a type of digital signature which requires multiple participants to sign a single document (Bellare & Neven, 2007). In a certain case, it is useful to add more security feature by dividing the authorization right to several participants. Multisignature is used in the Bitcoin system in which the user creates a transaction requiring multiple signatures to validate (Andresen, 2011).

The multisignature scheme in Bitcoin is denoted as m-of-n multisignature. The value m is the minimum number of signatures required to validate the transaction, while the value n is the total number of possible signatures which can be used to validate for the transaction. Multisignature feature enables the escrow scheme to be constructed within Bitcoin system and thus may increase the security of the transaction (Bitcoin Wiki, 2012a).

The escrow itself is useful to protect the buyer or the seller in a transaction. By constructing a 2-of-3 multisignature transaction, in case there is a dispute between the buyer and the seller, then the escrow takes part in determining who gets the money. If the seller has sent the product but the buyer does not wish to pay, then the escrow signs the transaction as such the seller can still get paid. If the buyer does not receive the product and the seller does not wish to refund the money, the escrow helps the buyer to refund the payment. The 2-of-3 multisignature means that the fund can be redeemed by at least 2 out of 3 possible signatures.

The <*scriptPubKey*> of Multisignature in Bitcoin is constructed as follows.

m <pubkey>...<pubkey> n OP_CHECKMULTISIG

While the <*scriptSig*> required to redeem this transaction will be

OP_0 ...signatures...

The OP_0 operation is appended because of a bug that still exists in OP_CHECKMULTISIG code implementation where it pushes one too many items into the stack and therefore an empty array should first be added into the stack by OP_0 command to avoid the problem (Andresen, 2011).

2.4.10. Hash Locked Transaction

Hash locked transaction was proposed by Tiernan (2014) to become one of transaction scheme in Bitcoin system. It enables anyone who can provide the value of a preimage hash value can spend the fund. In the scheme, the value is considered as a password to unlock the fund. Let x is the password, then Hash(x) is included in <*scriptPubKey*>, and whoever wants to spend the fund has to include x in <*scriptSig*> part of the transaction. The <scriptPubKey> of the transaction takes a form of

OP_HASH160 <20-byte-password-hash> OP_EQUAL_VERIFY <Standard Script>

The hash locked transaction can be combined with any standard Bitcoin transaction. For example, one must include her signature together with the password to create stronger transaction security. It means the <*scriptSig*> takes a form of

...signatures... <password>

Hash Locked Transaction is also used in chained transactions where a transaction can unlock another transactions. Therefore, the method is suitable for exchanging coins as in the case discussed by Tiernan (2013).

2.4.11. Transaction Signature

In the Bitcoin system, the ownership of an address and any fund that corresponds to the address is proved by using a digital signature. The digital signature is appended into the transaction to become a transaction signature. To spend an output, a valid digital signature must be supplied by the user and validated by the system. The steps required in transaction signature are explained as follows.

a. Create a copy of the original transaction.
b. Replace the content of <*scriptSig*> with the <*scriptPubKey*> of the previous transaction referenced by this transaction. The <*scriptSig*> itself will be created after the signing process because it contains the signature of the transaction.
c. Calculate the hash value and sign it by using a private key.

A Transaction signature has several types of hash which also determines the characteristics of the transaction.

a. SIGHASH_ALL, the hash includes all of the outputs. In this type of transaction, the outputs of the transaction cannot be changed.
b. SIGHASH_NONE, none of the outputs are included in the hash.
c. SIGHASH_SINGLE, only one output is included in the hash.

There is also a special flag of SIGHASH_ANYONECANPAY which could be combined with aforementioned types of SIGHASH. SIGHASH_ANYONECANPAY means that any input can be included in the transaction.

2.4.12. Contract

A contract in Bitcoin system is used to form an agreed transaction (Bitcoin Wiki, 2012a) which explores the features in Bitcoin transactions. There are many forms of contract in Bitcoin which can be done thanks to the flexibility of scripting language in Bitcoin. Basically, the features in Bitcoin that can be used in a contract can be:

a. SIGHASH
b. Multisignature
c. Locktime

There are many forms of contract. By using multisignature, a user can propose an escrow transaction. By using SIGHASH, a user can create a crowdfunding transaction. By using locktime, a user can setup a future payment.

2.4.13. Atomic Transaction

According to Tiernan (2013), atomic transaction is a type of transaction in which the participants can cancel the transaction at any stage without anyone losing money. If the transaction occurs, every participant gets what the participant wants, or if the transaction is cancelled, then no participant gets the payment nor suffers loss. The standard transactions cannot be used to construct atomic transaction; it needs a non-standard transaction or P2SH format.

2.4.14. Bitcoin Days Destroyed

Bitcoin days are the age of the bitcoins since their last spend transactions. Bitcoin days destroyed is a method of measuring the volume of Bitcoin transactions in a specific time (Bitcoin Wiki, 2011a). Bitcoin days destroyed is calculated by multiplying the number of bitcoin and the number of days of the last spend transactions. High Bitcoin days destroyed indicates that old and large amount of bitcoins are spent in short period of time.

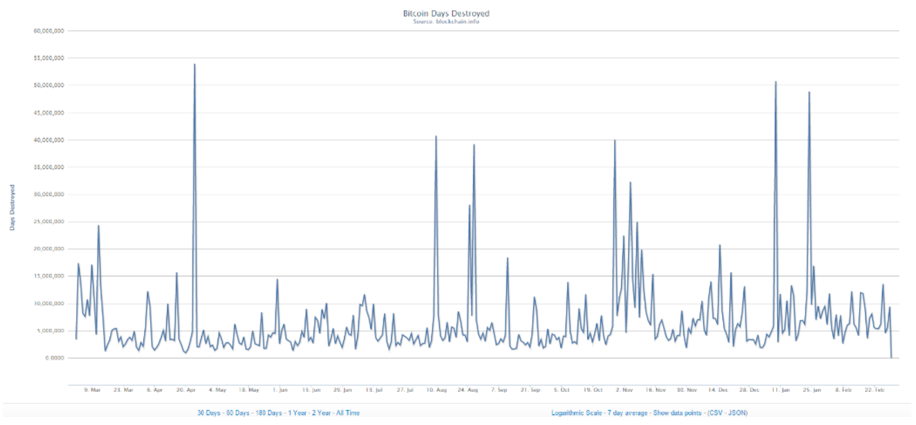


Figure 10. A Chart Showing Bitcoin Days Destroyed.

2.5. Privacy Issue in Bitcoin

Privacy is a big issue in the Bitcoin system. Although bitcoin originally offers anonymous transactions, some techniques have been developed to reveal the connections between public keys, the transaction patterns, and the real identity of the users.

2.5.1. Privacy Problems

Bitcoin is designed to have a privacy model in which the transactions and addresses do not have direct relationship to the real identity of the users. The Bitcoin privacy model and its comparison to traditional privacy model can be described in Figure 11. Basically, everyone gets involved into Bitcoin system without first registering into any centralized party, because there is no such thing as a centralized organization in Bitcoin system which controls the users and the transactions made within the system. Although the transactions are revealed and can be seen by everyone, the identities involved in the transactions are remain hidden.

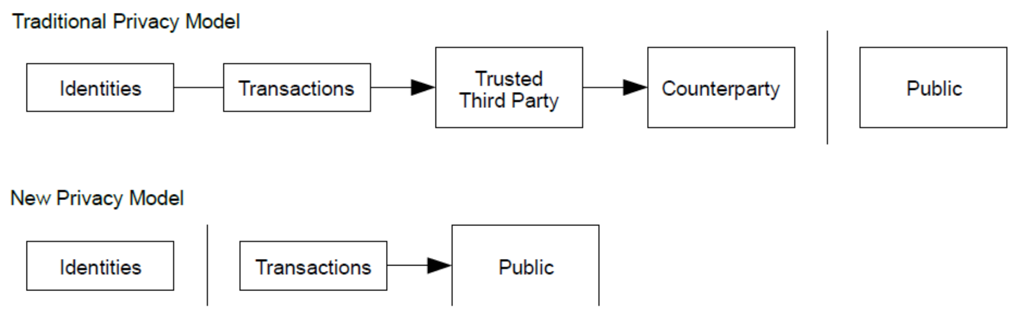


Figure 11. Bitcoin Privacy Model (Nakamoto, 2008)

However, the original Bitcoin privacy model does not guarantee the users to be fully anonymous. There are rules and properties of Bitcoin which can be used to create connections between the transactions and the real identity of the users.

2.5.1.1. Know Your Customer Policy

The governments become more aware of money-laundering scheme by using digital currencies such as Liberty Reserve (US Department of Justice, 2013), thus enforce financial institution to employ Know Your Customer (KYC) Principle. By implementing KYC Policy, somebody cannot create bank account without identity card. The same principle is enforced to other financial institution related to Bitcoin system, such as bitcoin exchange services, which will enable users to buy and sell bitcoins and convert their fiat currency to bitcoin or vice versa (Moser, Böhme, & Breuker, 2013).

The Bitcoin exchange services under United States law must register themselves as Money Services Businesses and therefore must follow regulations as for any traditional financial services (Greenberg, 2013). Several Bitcoin exchange services such as Mt. Gox (already closed), CoinJar, and

Bitcoin.co.id require users to scan their user ID and validate these IDs manually. As the result of KYC Policy, the bitcoin exchange services have records of relationship between bitcoin addresses of the users and their real identity.

Because every transactions in Bitcoin system can be seen by everyone, the bitcoins bought from bitcoin exchange services can be tracked. Thus, if a transaction is suspected to be involved in unlawful activities, the identity of the users involved can be identified if they buy their bitcoins directly from exchange services.

2.5.1.2. Greenlist

Another effort of complying Bitcoin transaction with US regulatory in financial transaction is done by linking the Bitcoin address and the real identity of the individual in a concept called greenlist (Buterin, 2013). The idea of greenlist is proposed by a company called CoinValidation[1] which will hold a database of Bitcoin addresses which have been verified to be owned by lawful users, by then it can be used for law enforcement agencies to track down the users buying illicit goods by using their bitcoins (Hill, 2013). Even President Obama signed an initiative of National Strategy for Trusted Identities in Cyberspace (Cohen, 2013).

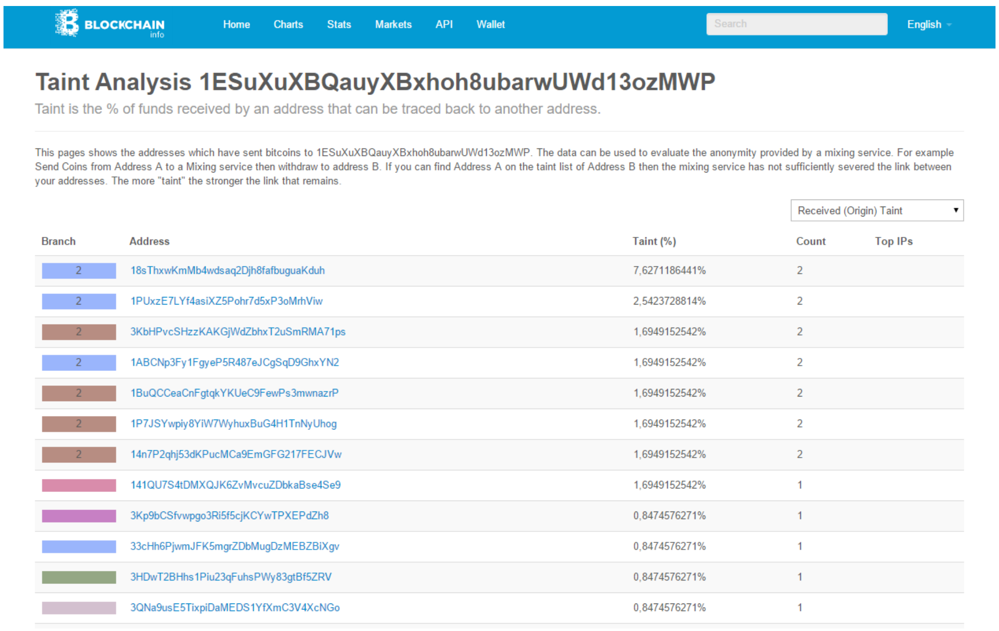
The greenlist idea has already received rejections as expressed by community in Bitcointalk (Back, 2013) and in Reddit (djsjjd, 2014) as it annihilates the idea of fungibility, in which any bitcoin must have the same value if the amount of bitcoins are the same no matter where they come from: freshly minted bitcoins must have the same value as the ones owned by a drug dealer. The greenlist scheme may create different valuation in bitcoins by regarding the bitcoins held by registered addresses to have higher value compared to the ones held by addresses not registered in greenlist. It may also become an “anchor” or a starting point for any agencies which have access to the database to track down the identity of anyone which transacted with those addresses in the database (Buterin, 2013).

2.5.1.3. Taint Analysis

In Bitcoin system, taint is defined as a correlation between Bitcoin addresses (Maxwell, 2013c). The correlation comes from the past transactions (received or spent). Taint analysis determines the closeness between one address to another. As the Bitcoin system can be considered as an open ledger, the taint analysis can be queried from the Bitcoin network based on the transaction history of the address. In term of anonymity, it is expected for the addresses to be not closely related each other, which can also be determined as taint proof.

[1] http://coinvalidation.com

Taint analysis can be used to track the flow of money within Bitcoin system. The correlations between addresses in taint analysis are formed because one address pays to other addresses directly or indirectly, or one address receives money from other addresses directly or indirectly.



Taint Analysis 1ESuXuXBQauyXBxhoh8ubarwUWd13ozMWP

Taint is the % of funds received by an address that can be traced back to another address.

This pages shows the addresses which have sent bitcoins to 1ESuXuXBQauyXBxhoh8ubarwUWd13ozMWP. The data can be used to evaluate the anonymity provided by a mixing service. For example Send Coins from Address A to a Mixing service then withdraw to address B. If you can find Address A on the taint list of Address B then the mixing service has not sufficiently severed the link between your addresses. The more "taint" the stronger the link that remains.

Branch	Address	Taint (%)	Count	Top IPs
2	18sThxwKmMb4wdsaq2Djh8fafbuguaKduh	7,6271186441%	2	
2	1PUxzE7LYf4asiXZ5Pohr7d5xP3oMrhViw	2,5423728814%	2	
2	3KbHPvcSHzzKAKGjWdZbhxT2uSmRMA71ps	1,6949152542%	2	
2	1ABCNp3Fy1FgyeP5R487eJCgSqD9GhxYN2	1,6949152542%	2	
2	1BuQCCeaCnFgtqkYKUeC9FewPs3mwnazrP	1,6949152542%	2	
2	1P7JSYwpiy8YiW7WyhuxBuG4H1TnNyUhog	1,6949152542%	2	
2	14n7P2qhj53dKPucMCa9EmGFG217FECJVw	1,6949152542%	2	
	141QU7S4tDMXQJK6ZvMvcuZDbkaBse4Se9	1,6949152542%	1	
	3Kp9bCSfvwpgo3Ri5f5cjKCYwTPXEPdZh8	0,8474576271%	1	
	33cHh6PjwmJFK5mgrZDbMugDzMEBZBiXgv	0,8474576271%	1	
	3HDwT2BHhs1Piu23qFuhsPWy83gtBf5ZRV	0,8474576271%	1	
	3QNa9usE5TixpiDaMEDS1YfXmC3V4XcNGo	0,8474576271%	1	

Figure 12. Taint Analysis of a Bitcoin Address

2.5.2. Existing Solutions on Bitcoin Anonymity

As all of the information regarding the transactions are available to be seen by everyone, there is a need for anonymizing certain transactions. This need is provided by several anonymizing methods. Each of the solutions has different approaches, properties, strengths, and weaknesses. Several solutions will be discussed below.

2.5.2.1. CoinJoin

Maxwell (2013a) introduced an approach to add another level of privacy for Bitcoin users called CoinJoin as a refinement of his previous idea in Maxwell (2013c). The idea came from a fact that some users may have different security perspective while creating transactions which enables an analysis over the relationship between those users. By using the proposed CoinJoin, the coins in some transactions are combined. The most popular implementation of CoinJoin was provided by P. Martin and Taaki (2013) in a system with the same name.

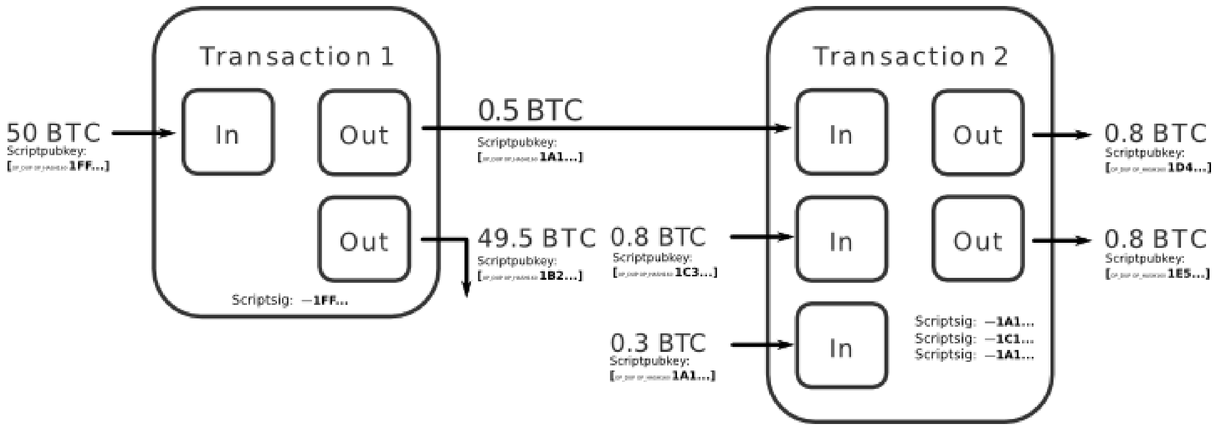


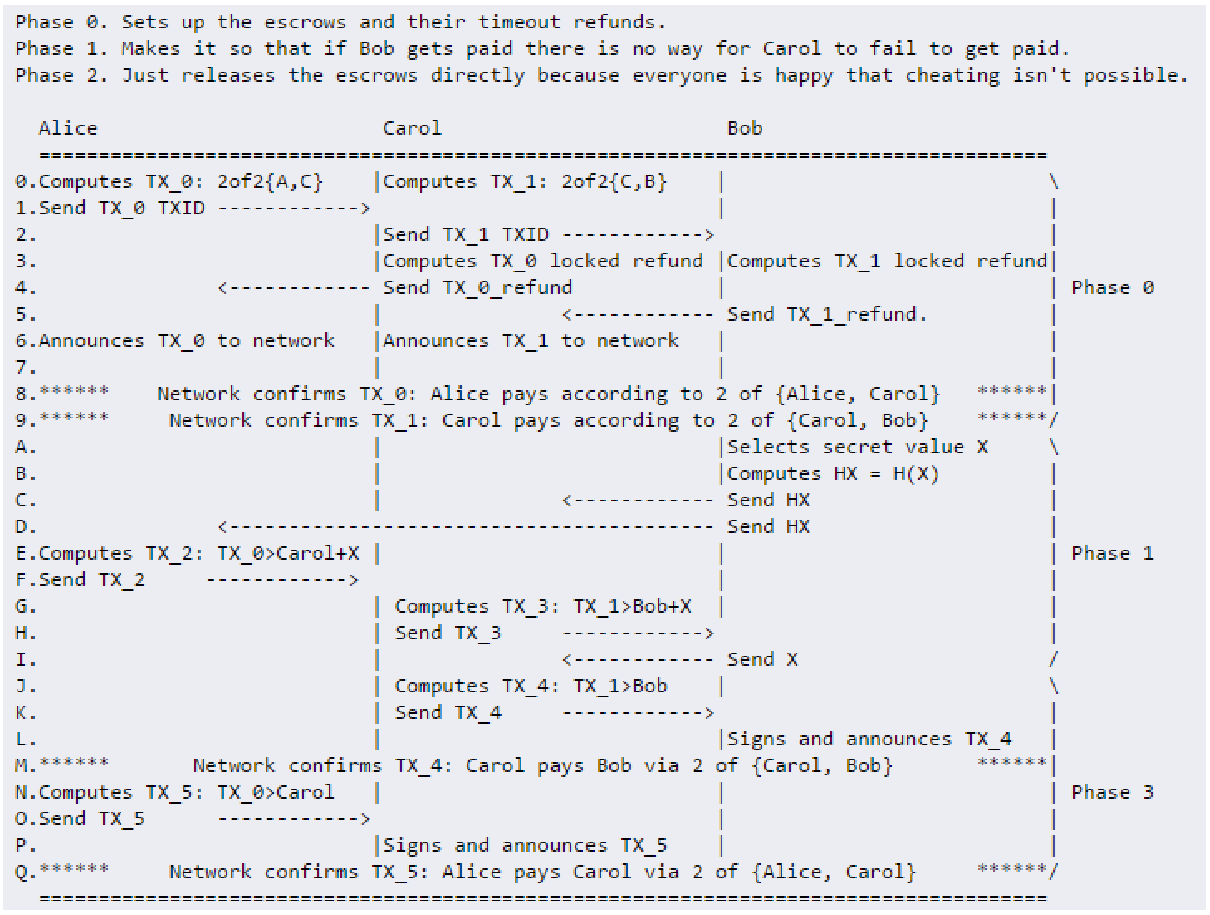
Figure 13. CoinJoin Transaction (Maxwell, 2013a)

Figure 13 explains how CoinJoin operates and how it compares to normal Bitcoin transaction. Transaction 1 is an example of how normal Bitcoin transaction is done. An address 1FF holding 50 BTC wants to send 0.5 BTC to another address, 1A1 and sets a change address to 1FF. In the end of the transaction, the address 1FF holds 49.5 BTC.

In transaction 2, there are multiple input addresses and multiple output addresses. Although this scheme is considered as a normal Bitcoin transaction, the property of this kind of transaction can be used to add identity protection for the users involved in the transaction. Suppose the owner of the address 1A1 wants to send 0.8 BTC to 1E5 and does not want any adversaries to spot his transaction, he combines his transaction with another different transaction with the same size, for example a transaction from address 1C3 to 1D4. By looking at the transaction, any adversary cannot perfectly guess which address receives bitcoins from address 1A1, because it can be either 1D4 or 1E5.

2.5.2.2. CoinSwap

The idea of CoinSwap was proposed by Maxwell (2013b) in Bitcointalk forum. The operation of CoinSwap results in hiding the relationship of the transaction between the payer and the payee. CoinSwap enables those parties to create reliable transactions by guaranteeing that each party involved cannot steal the bitcoin. In CoinSwap, a third party is needed to be a gateway between the payer and the payee. It also involves several transaction methods to accommodate this solution. The transaction methods involved are 2-of-2 escrow and hash-locked transaction.



```
Phase 0. Sets up the escrows and their timeout refunds.
Phase 1. Makes it so that if Bob gets paid there is no way for Carol to fail to get paid.
Phase 2. Just releases the escrows directly because everyone is happy that cheating isn't possible.

  Alice                        Carol                        Bob
  ======================================================================================
0.Computes TX_0: 2of2{A,C}    |Computes TX_1: 2of2{C,B}    |                           \
1.Send TX_0 TXID ------------>                             |                           |
2.                            |Send TX_1 TXID ------------>                            |
3.                            |Computes TX_0 locked refund |Computes TX_1 locked refund|
4.              <------------ Send TX_0_refund             |                           | Phase 0
5.                            |             <------------ Send TX_1_refund.            |
6.Announces TX_0 to network   |Announces TX_1 to network   |                           |
7.                            |                            |                           |
8.******    Network confirms TX_0: Alice pays according to 2 of {Alice, Carol}    ******|
9.******     Network confirms TX_1: Carol pays according to 2 of {Carol, Bob}     ******/
A.                            |                            |Selects secret value X     \
B.                            |                            |Computes HX = H(X)         |
C.                            |             <------------ Send HX                      |
D.              <------------------------------------------ Send HX                    |
E.Computes TX_2: TX_0>Carol+X |                            |                           | Phase 1
F.Send TX_2     ------------>                              |                           |
G.                            | Computes TX_3: TX_1>Bob+X  |                           |
H.                            | Send TX_3     ------------>                            |
I.                            |             <------------ Send X                       /
J.                            | Computes TX_4: TX_1>Bob    |                           \
K.                            | Send TX_4     ------------>                            |
L.                            |                            |Signs and announces TX_4   |
M.******       Network confirms TX_4: Carol pays Bob via 2 of {Carol, Bob}        ******|
N.Computes TX_5: TX_0>Carol   |                            |                           | Phase 3
O.Send TX_5     ------------>                              |                           |
P.                            |Signs and announces TX_5    |                           |
Q.******     Network confirms TX_5: Alice pays Carol via 2 of {Alice, Carol}      ******/
  ======================================================================================
```

Figure 14. CoinSwap Protocol Diagram (Maxwell, 2013b).

The CoinSwap protocol can be explained as follows. Let Alice serves as the payer, Bob as the payee, and Carol as the middle party. The protocol is divided into 3 phase: phase 0, phase 1, and phase 2. Phase 0 contains escrow transactions setup between the three parties. In phase 0, Alice sets up an escrow transaction TX_0 which is 2-of-2 escrow of Alice and Carol by using Alice's bitcoin. It can be read as "Alice will pay certain amount of bitcoin to Carol as long as Alice and Carol agree on the transaction". Then, Carol also sets up an escrow transaction TX_1 which is 2-of-2 escrow of Carol and Bob by using Carol's bitcoin. It can be read as "Carol will pay certain amount of bitcoin to Bob as long as Carol and Bob agree on the transaction". To secure this step, Carol creates TX_0_refund with a timeout which enables Alice to claim the bitcoin she pays to Carol if the scheme is cancelled. Similar to Carol, Bob also creates TX_1_refund with a timeout and sends it to Carol after being signed by Bob. This TX_0_refund and TX_1_refund are not broadcasted to the network by neither Alice nor Carol. Then Alice and Carol broadcast TX_0 and TX_1 to the Bitcoin network.

Phase 1 prepares transaction guarantees for the parties involved. The guarantee works as if Bob redeems his coins from Carol, Carol also redeems her payment from Alice. This scheme is supported by a hash-locked transaction scheme. In the scheme, Bob selects a random secret value of X then computes HX - the hash value of X - which is expressed as HX = H(X). Bob sends the HX to Alice and

Carol. Alice creates a new hash-locked transaction TX_2 which spends TX_0 by using Carol's signature and a value of X. Alice sends the transaction TX_2 to Carol. Similar to Alice, Carol creates a new hash-locked transaction TX_3 which spends TX_1 by using Bob's signature and a value of X. Carol then sends the transaction TX_3 to Bob. In this phase, if Bob cheats and redeem his coins by using TX_3, then Carol redeems TX_2 because the value X is exposed by Bob's TX_3. Or if Alice does not proceed and create TX_5, Carol gets paid by redeeming TX_2 because the value X is already revealed by Bob as revealing X is required before creating TX_4. But if all of the parties do not redeem any coin at this phase and behave honestly, TX_2 and TX_3 are not published in the network. Bob reveals the value of X to Carol.

Phase 2 is the last phase of the protocol where Carol creates a new standard transaction TX_4 which spends TX_1 by using Bob's signature. Bob then redeems the transaction TX_4 by signing it and broadcasting it to the Bitcoin network. Alice creates transaction TX_5 which spends TX_0 by using Carol's signature. Carol then redeems the transaction TX_5 by signing it and broadcasting it to the Bitcoin network. In summary, the network gets information of TX_0 and TX_1 as escrow transactions, and TX_4 together with TX_5 as spending transactions. When the phases are completed, information of TX_0_refund, TX_1_refund, TX_2, and TX_3 can be deleted need not to be sent to the network.

The CoinSwap is an example of Zero Knowledge Contingent Payment (Bitcoin Wiki, 2011e) which includes following features:

a. Hash locked transaction
b. Time-locked transaction as the guarantee.
c. 2-of-2 escrow transaction

In order to secure the transaction, the steps must be carefully followed in the exact sequence to provide a secure and guaranteed transaction without the need of trusted party.

2.5.2.3. Mixing Services

Another method to handle the privacy issue of Bitcoin transaction is by using mixing services such as OnionBC[2], Bitcoin Fog[3], BitLaundry[4], and Send Shared[5] (Moser et al., 2013). Those services have different method of laundering or mixing the bitcoins of their users, although the method can be classified into two groups. In the first group, the services require the users to send their bitcoins into virtual wallets controlled by the services, and then the users can withdraw their bitcoins to any address. In this group, the services replace users' original bitcoins with another bitcoins and therefore the new bitcoins hold no relationship to the original bitcoins. If the users want to pay some bitcoins to any

[2] http://6fgd4togcynxyclb.onion
[3] http://bitcoinfog.com
[4] http://app.bitlaundry.com
[5] https://blockchain.info/de/wallet/send-shared

merchant, they can set the withdrawal address to the merchant's address. Services fall into this group are OnionBC, Bitcoin Fog, and BitLaundry. In second group, the services combine several transactions of the users into one transaction. This services's method is similar to CoinJoin. An example of this group is Send Shared. As the users use the service, they must pay certain amount of bitcoins to the provider. The fee varies from 0.5% to 3%. Although the services can create problem in transaction analysis, the users pose some risk as the users cannot control their bitcoins once they send their bitcoins to the virtual wallets controlled by the service providers. If the service providers steal the users' bitcoins, the users do not have any means to get their bitcoins back because in Bitcoin system, every transaction is irreversible.

2.5.2.4. MixCoin

The idea of MixCoin as proposed by Bonneau et al. (2014) is to create an accountability mechanism for mixing services. The implementation of MixCoin does not require any change to current Bitcoin transaction, thus can be easily implemented by Bitcoin users. In MixCoin protocol, two parties are involved. First party is the one who wants his/her coin mixed, and the second party is the mixing services. To be able to add an accountability feature of this service, the user holds a proof that the mixing service promises to do the transaction. If the service steals the user's bitcoins, then the user publicly exposes the cheating of the provider with the proof which destroys the mixing service's reputation, thus prevents the service of having new customers.

The protocol of Mixcoin can be explained in the Figure 15 which involves few steps. First, a user A creates a request to a service provider M to do a bitcoin transaction. If M agrees, M signs the information of the transaction requested by A by using M's private key. This signed data is the proof held by A which can be verified by anyone by using M's public key, and second step is done. Next step, the user A pays agreed amount of bitcoins, including the bitcoins involved in the transaction and the transaction fee paid to M. In the next step, if M behaves honestly, then the proof held by A is deleted, but if M steals the bitcoins, A publicizes the signed data by M.

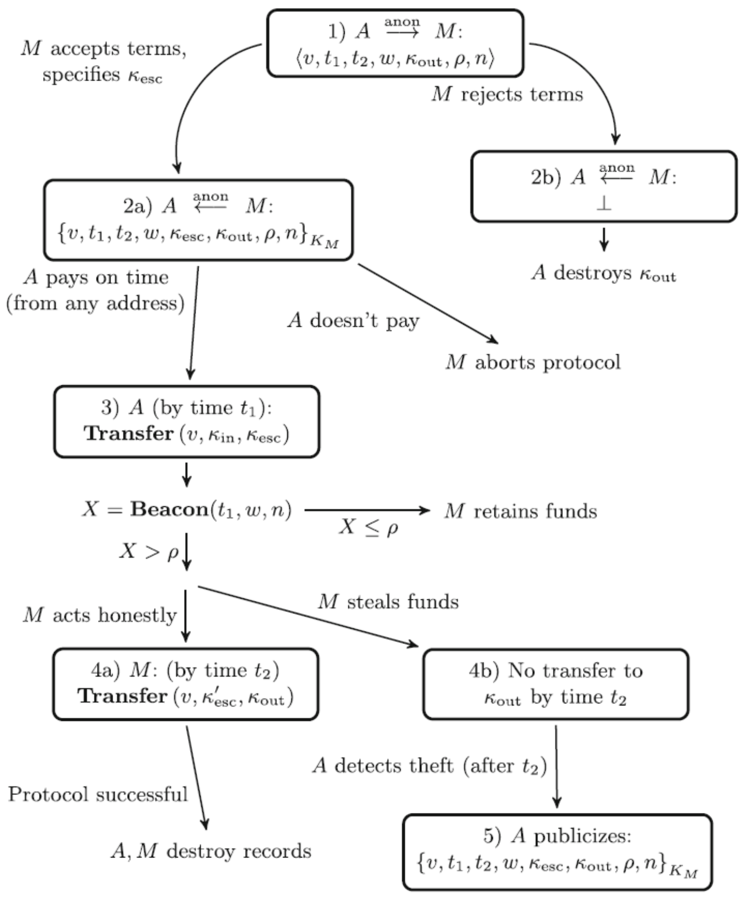

1) $A \overset{\text{anon}}{\longrightarrow} M$: $\langle v, t_1, t_2, w, \kappa_{\text{out}}, \rho, n \rangle$
M accepts terms, specifies κ_{esc}
M rejects terms
2a) $A \overset{\text{anon}}{\longleftarrow} M$: $\{v, t_1, t_2, w, \kappa_{\text{esc}}, \kappa_{\text{out}}, \rho, n\}_{K_M}$
2b) $A \overset{\text{anon}}{\longleftarrow} M$: $\perp$
A destroys κ_{out}
A pays on time (from any address)
A doesn't pay
M aborts protocol
3) A (by time t_1): **Transfer** $(v, \kappa_{\text{in}}, \kappa_{\text{esc}})$
$X = \textbf{Beacon}(t_1, w, n)$
$X \leq \rho$
M retains funds
$X > \rho$
M acts honestly
M steals funds
4a) M: (by time t_2) **Transfer** $(v, \kappa'_{\text{esc}}, \kappa_{\text{out}})$
4b) No transfer to κ_{out} by time t_2
A detects theft (after t_2)
Protocol successful
A, M destroy records
5) A publicizes: $\{v, t_1, t_2, w, \kappa_{\text{esc}}, \kappa_{\text{out}}, \rho, n\}_{K_M}$

Figure 15. Mixcoin Protocol (Bonneau et al., 2014)

2.5.2.5. Networks of Transactions

Coutu (2013) investigated the concept of using of networks of transactions in an attempt to anonymize the Bitcoin transaction. Rather than using single transaction to mix the coins of users, the networks of transaction will add more anonymity while an adversary could not define the complex structure of the network, therefore the output of the transactions can be more anonymous.

2.5.2.6. Zerocash

A new concept called Zerocoin was proposed (Miers, Garman, Green, & Rubin, 2013). Zerocoin was developed based on zero knowledge mechanism. It supports a fully anonymous transactions without a single authority nor trusted party. The main part of this approach is to allow users to create their own coins with assumption that they have sufficient amount of bitcoin represented in the new coins they create. The newly created coins and the original bitcoins are bounded by using digital commitment scheme which will prevent double spending of bitcoins they originally hold. Although this approach seems to be promising, it needs a major change into current Bitcoin protocol and the requirements of running such protocol will require larger storage and memory than the current Bitcoin system.

As an improvement of Zerocoin, Zerocash was introduced (Ben Sasson et al., 2014). Zerocash is equipped with a scheme called decentralized anonymous payment. It eradicates the information of the coin receivers as in Zerocoin, thus offer a higher level of anonymity. Zerocash transaction allows its users to privately pay each other and hides information related to the transaction such as the source coins, destination, and the amount of transacted coins. However, similar to Zerocoin, the Zerocash scheme cannot be implemented in the current Bitcoin system because it requires modification in the current Bitcoin protocol.

2.5.2.7. Merge Avoidance

Merge avoidance as proposed by Hearn (2013) identifies the privacy issue of Bitcoin users and offers a way of increasing the privacy by a certain method. The idea of merge avoidance is that a certain amount of bitcoins can be an artefact to search within Bitcoin transactions; therefore, only by splitting the amount of bitcoins received into smaller amount of bitcoins in several transactions can increase the privacy of the parties involved in the transaction. Merge avoidance is not actually a complete solution. Rather, it complements other solutions which may add the anonymity of the transaction.

2.5.3. Related Studies on Anonymity of Bitcoin

Researchers have conducted studies in the area of privacy issue and anonymity of Bitcoin. The importance of these studies are determined by the fact that every information related to Bitcoin transactions are published and become records that everybody can verify. Although it seems bad in term of privacy, it was a design decision by the Bitcoin creator, Satoshi Nakamoto, in order to eliminate the need of a central authority to run the system. Although the Bitcoin system itself holds a privacy model, in practice there are methods developed to investigate the information about the users of Bitcoin system. In a research , Meiklejohn et al. (2013) showed how the unlinkability feature of Bitcoin system was tested. While the system itself does not hold any direct relation with the real identity of the users, the transactions inside the system can be analyzed to uncover some information about the users. The research explored the idea that Bitcoin wallets can use more than one address as the input of the transaction. There is a possibility of those input addresses are controlled by the same user. Also, if the transaction needs to have a change address because the amount of bitcoins inside the input addresses are larger than the bitcoins paid to output addresses, the difference will be returned to an address owned by the payer. It means that the change address is also owned by the same user of the input addresses. By using this 2 properties of Bitcoin transaction, they managed to identify not less than 500,000 addresses controlled by Mt. Gox and more than 250,000 addresses controlled by Silk Road.

While Meiklejohn et al. (2013) focused on addresses owned by at least 2 big online vendors, Ron and Shamir (2013) were able to provide a general characteristics of Bitcoin transactions by using quantitative analysis over a transaction graph. They downloaded all Bitcoin transactions and calculated the distribution of bitcoins over Bitcoin addresses and the owners of those addresses. They managed to determine addresses owned by some big vendors such as Mt. Gox, Instawallet, and Deepbit. They also found that there were lots of small transactions descended from a big transaction in November 2010. Although it may not be a problem for users with small amount of bitcoins, users holding large amount of bitcoins may be worried of their privacy revealed by someone monitoring their transactions; this may be a big issues related to a large scale payment using Bitcoin system as everybody can see the striking transactions in the distributed blockchain.

An investigation done by Androulaki, Karame, Roeschlin, Scherer, and Capkun (2013) provides an insight on how a proper analysis revealed users' identity within Bitcoin transaction. Because the addresses of the vendors accepting bitcoins are exposed to public in order to inform their customers to pay bitcoins to them, the research suggested that geographic clustering can be done by tracing involved addresses. To reduce the possibility of such analysis, they suggested for the vendors to provide new addresses to the users when paying bitcoins to the vendors. Although the Bitcoin system is supposed to be an online system, the nature of its usage by offline vendors, there is no doubt that those vendors' customers have visited the vendors' business place.

A paper published by Reid and Harrigan (2013) described de-anonymization attempt of Bitcoin users by investigating the patterns of the transaction and some properties in Bitcoin transaction. The researchers were aware that some information can be gathered to create links between Bitcoin addresses and real identity of the users, e.g. from Twitter, blogs, forums in which those users mention their Bitcoin address to receive donations. The paper described an experiment regarding to relationships between Wikileaks donation address and an alleged theft of large amount of bitcoins. The research managed to present the transactions related to the event in a network graph. Although the thief tried to disperse the stolen coins into thousands of addresses, the bitcoins can still be traced. Therefore, combination of network analysis and external information about the addresses can be used to measure and determine the relationship between Bitcoin addresses and the real identity of the owners.

Another issue regarding to Bitcoin privacy was presented by Kaminsky (2011). He investigated the anonymity of Bitcoin by using the property of Bitcoin system which employs peer-to-peer (P2P) network and a static port of 8333/TCP. While P2P relays the Bitcoin transaction from one node to another nodes, one can trace back to the first node publishing the transaction, thus that node can be identified to be one closest to the user. Furthermore, Koshy, Koshy, and McDaniel (2014) developed a specialized Bitcoin client to capture the relayed information in Bitcoin P2P system to create a link between the owner of Bitcoin address and IP address. While Biryukov, Khovratovich, and Pustogarov (2014) discovered a way to disclose the information flow inside Tor Network by acquiring the relay nodes and collect information that go through these relay nodes.

Möser (2013) investigated some mixing services by creating experiments using those services, measuring and comparing the result between the transactions created during the experiments. The author used a method called taint analysis (Piuk, 2012) which is freely available from blockchain.info website. The taint analysis is used to measure the connectivity between addresses linked by Bitcoin transactions. The taint analysis uses the property of Bitcoin transaction which determines that although the fund can be transferred within addresses, the connection between those addresses is kept intact into the blockchain. The method explores the closeness between one address to another addresses from past transactions.

2.5.4. Properties of Bitcoin Privacy

To summarize, some properties of Bitcoin can be used to reveal some degree of Bitcoin user anonymity by using them into an analysis, as can be described below.

a. Bitcoin address exposure

   Although it is normal to expose the Bitcoin address to somebody else to receive payments or donations, the adversary can directly create relationship between the addresses and the real identity of the users. The transactions made from and to these addresses are in a risk of privacy issue in which everybody can investigate the information by accessing the blockchain. Not only the owners suffer from privacy issue, but also the users they transact with also suffers the same issue when the geolocation analysis is done with respect to these transactions.

b. Multiple input addresses

   A wallet can contain multiple Bitcoin addresses as it manages these addresses on behalf of the user. If the user tries to create a transaction which requires more than the fund held by an address, the wallet automatically creates multiple input addresses into the transaction. Therefore, a clustering analysis works by associating these addresses to be owned by the same user.

c. Change address

   As a common practice of Bitcoin transactions in which all amount of bitcoins in an address are involved in the transaction, change address receives the difference between total fund held by the address and the total fund paid. Because of this nature, change addresses can be identified to be owned by the same user of the input address. This can lead to a clustering analysis.

d. Amount of bitcoins transacted

   The amount of bitcoins can be a unique artefact to search in Bitcoin transaction data. Some specific amount of bitcoins involved may be analyzed and determined to be owned by a user, e.g. a user tries to transfer her own fund from an address she controls to another new address she created in which these two addresses hold the same amount of fund.

e. Guaranteed Transaction

The transactions of securing the privacy must abide an important property of which those transaction must be guaranteed that no party has the ability to cheat which result in the other party suffers the loss of the fund.

f. Record Keeping

The attempt of protecting the anonymity of the Bitcoin users must not hold the record of the attempt, e.g. a mixing service keeps records of users' input and output addresses. This can raise a problem when the mixing service is subpoenaed and forced to disclose the information in the database.

g. Compatibility with Bitcoin Protocol

It is obvious that the effort of hiding the information about Bitcoin users must be compatible with the current Bitcoin protocol. If not, then the method is not useful towards the current system. The solution must be able to be implemented by using features provided by the current system.

2.5.5. Comparing Existing Solution

Existing privacy solutions can be compared in respect to properties of Bitcoin transaction they tried to protect. These properties are related to the anonymity of Bitcoin transactions.

Table 2. Existing privacy solutions comparison

No	Characteristics	Proposed Protocol	CoinJoin (Maxwell, 2013a)	CoinSwap (Maxwell, 2013b)	Mixing Services (Moser et al., 2013)	MixCoin (Bonneau et al., 2014)	Networks of Transactions (Coutu, 2013)	Zerocash (Ben Sasson et al., 2014)
1	Atomic transaction[6]	V	V	X	X	X	X	V
2	No participant holds all information	V	X	X	X	X	X	V
3	Compatible with current Bitcoin protocol	V	V	V	V	V	V	X
4	Hides payer's address from the payee	V	X	V	X	X	X	V
5	Taint proof[7]	V	X	V	V	V	X	V
6	Cheating security	V	V	V	X	X	V	V

From the Table 2, it can be concluded that the proposed protocol must be able to fulfill all the required characteristics of an anonymizing protocol. Zerocash in its protocol requires the Bitcoin transaction to be flagged as a Zerocash transaction and therefore requires modification to Bitcoin core

[6] The concept of atomic transaction is discussed in section 2.4.13.
[7] Taint analysis and taint proof is discussed in section 2.5.1.3.

system. Moreover, to create a payment, a payer needs to know the public key of the payee, despite the transaction will be encrypted and no observer will know which coin is spent.

In CoinJoin, all participants have the full information of the transaction created in the protocol because they need to sign the transaction. Despite they may not be able to determine the identity of other participants, they can still enumerate the input addresses and the output addresses. The addresses may also be connected each other because they are used in the same transaction and therefore it is not taint proof.

CoinSwap is not atomic because it requires approval from the receiver to create refund transaction. If the receiver does not want to sign the refund transaction, the fund owned by the sender cannot be claimed. Moreover, if one of the participants decides to reveal the secret value, then the chained transactions can be easily linked each other as they have the same secret value. CoinSwap only utilizes a single third party and therefore creates a single point of failure in case of the third party decides to reveal the information about the created transaction.

Mixing services are commonly used by bitcoin users to obfuscate the origin of the bitcoin owned. But there are problems related to those services. The services are usually run by unknown parties in which if these parties decide to steal the bitcoins they are supposed to mix, the users are left with no means to claim their fund back. The information of the mix is also kept by the service providers and therefore it can harm the privacy of the users if the information is leaked to public.

MixCoin solution basically adds a degree of security by holding a proof that an order has been placed into a mixing service and in case of the mixing service does not behave as expected, the user reveals the verifiable proof that an order has not been executed as instructed by the user and therefore destroys the reputation of the service. But other than destroying the reputation of the service, the user cannot get the stolen fund back because the transaction itself is not secured by any mean. Also there is still an issue related to the service in which the service keeps the record that this transaction occurs in their system.

As a new concept of mixing bitcoins, Networks of Transactions can be seen as an expansion of CoinJoin solution. Instead of using a single transaction to mix the transactions, Networks of Transactions protocol allows the users to combine the bitcoins into multiple joined transactions. This seems to be convenient, but there are difficulties regarding to this. As joining transactions may require all users to sign the transactions, users can find it inconvenience to sign the transactions multiple times and therefore it consumes more time and effort to finish a complete set of the circuit. Moreover, although the transactions can hardly be determined precisely by an adversary because of its multiple input and multiple output, the adversary could become one of the users inside of the mix and therefore may have the knowledge of the circuits. The original address may also be traced by using taint analysis because the original address is always involved in the transaction circuit.

Combining similar transactions into a single big transactions containing multiple inputs and multiple outputs may not be a good solution in protecting the user privacy. Let the outputs are also controlled by the same user, that user may use the mixed coins in a way which can expose any information related to the user's whereabouts or the identity. For example, if the fund mixed is large enough that it has a change after being used to pay to others, if that change money is used to buy something in a restaurant, then the user's location is exposed, then the later transaction history is exposed, which also destroys the attempt of mixing the coins.

Most of the solutions except mixing services and MixCoin have a mechanism of preventing the participants from cheating. Zerocash has a cryptographic mechanism to prove that the participants are honest. In CoinJoin, each of the participant can check the validity of the transaction prior to signing the transaction. In CoinSwap, the transactions are guaranteed by the hash-locked-transaction and 2-of-2 multisignature mechanisms. The proposed protocol must also be able to handle the cheating possibility of the participants involved in the transaction.

Chapter 3 - Security Model

3.1. Overview

The security model described in the chapter will be used to evaluate the proposed protocol. The protocol is measured against the security model determined. There are 2 parts of the security model. The first is anonymity model and the second is cheating model. The proposed protocol must be able to fulfil both of the models.

3.2. Anonymity Model

The concept of unlinkability and anonymity to measure the privacy are proposed. Unlinkability is the inability to relate different items (Pfitzmann & Hansen, 2010). The items involved in the same transaction sequence must not have a specific attribute to distinguish them from any other similar items.

Anonymity is the inability to identify a particular subject in a set of subjects (Pfitzmann & Hansen, 2010). It is assumed there are N number of transactions created by N number of different payers employing the same protocol in the same configuration of middlemen within a time period. A transaction sent to Bob from Alice is chosen uniformly random from N transactions within that time period. Bob then tries to identify Alice by cooperating with one of the middlemen. The proposed scheme has the anonymity characteristic if the probability (P) of Bob guessing Alice's address is determined in the following equation.

$$P = \frac{1}{N}$$

3.3. Cheating Model

The cheating model of the protocol is defined as follows. In the cheating scenario, one or more participants try to cheat by not paying anything or paying less amount of money to others despite getting a payment from others. With the assumption that at least 1 of the sender or the escrow within each group is honest and assuming that the receiver is always honest, the proposed scheme is secure if the probability of any participant tries to cheat is negligible.

3.4. The Attacker

An attacker is defined as someone that is curious about the information of the transaction. The attacker tries to recover the information of the initial payer or steals the fund if it is possible. In a transaction of Alice paying to Bob, Bob also acts as the attacker by trying to uncover the hidden information of Alice in the protocol. It is also assumed that Bob is able to cooperate with one of the middlemen in the protocol in order to get the address of Alice.

Chapter 4 - The Protocol

4.1. Overview

The basic idea of the protocol is by replacing the fund paid by the original payer with other fund owned by the middlemen. Before the protocol is commenced, there should exist multiple sources of fund which are assumed not to have any relationship between each other in term of the transaction history in the Bitcoin distributed ledger. The participants transfer their fund to other participants constructing a transaction circuit. The starting point of the circuit is the original payer, while the other end of the circuit is the payee.

This section describes the detail of the protocol proposed to achieve the goal of the research. It also explains about the participants, the methods, and the transaction circuits which may be used in the protocol. Several unsuccessful protocols are also described including their evaluations which leads to a final protocol as the product of the research.

4.2. Participants

The participants of the protocol are defined as those who are actively involved in creating the transactions. The participants can be divided according to their roles. The payer is the one who starts the protocol in order to pay some amount of money, while the payee is the one who receives money from the payer. Middlemen are those who are helping the payer and the payee to construct the transaction circuit. Intermediary payers are those who pay money to others but are not the initial payer, while intermediary payees are those who receive money from others but are not the final destination of the money.

The middlemen can be anyone who are willing to help constructing the protocol in exchange of some amount of fee. For the protocol to work, it is assumed that there exists some middlemen who are willing to help users to anonymize their bitcoin in exchange of a small portion of profit.

4.3. Communication Channels

The proposed protocol of anonymizing the Bitcoin transaction without any trusted system cannot be created without a communication channel. In this paper, it is assumed there exists an anonymous communication channel e.g. Tor (Dingledine, Mathewson, & Syverson, 2004) which can be used by multiple users to exchange information without revealing any information about their identity. The communication records cannot be linked with the transactions created within the proposed protocol. It is also assumed that the participants use a secure communication channel to send raw transactions and signed transactions between participants.

After the participants agree to form the transactions, a secure anonymous communication channel must be setup between them which can only be accessed by the participants. Let it be called general channel. Another separate communication channels must also be setup for each group in the protocol. Let the latter channels be called group channels. Although the participants may become members of multiple groups, by default the participants are assumed not to share information gained with any member of another groups.

These communication channels are used to communicate and exchange several information required to construct the protocol, e.g. the public keys, the amount of fund transferred and the CLTV information. This information is shared among the participants by using rules specified in the protocol.

4.4. Identification

To identify themselves, the participants uses pseudonyms or fictitious names in the communication channels utilized in the protocol. Due to the nature of the proposed protocol of not requiring a trusted scheme, the identity of the participants do not need to be confirmed nor validated. For the payer to identify the destination of the payment which pays to the payee, a method can be used. Instead of exposing a Bitcoin address to the payer, the payee provides a hierarchical deterministic public key to the payer.

4.5. Bitcoin Address Generation

The hierarchical deterministic wallet scheme is utilized in Bitcoin address generation. In the protocol where the payer does not directly pay to the payee, the payees' addresses are generated by the payers. After generating the address, the payers publish the paths or indexes used to generate the addresses to each group channel in which the addresses will be used. By then, the members of the group channel can each verify whether the address generation is correct as it is claimed to be. In the same time, knowing the path, the payees which hold the parent private keys can generate the child private key which corresponds to the addresses using the same paths as in address generation process. It is also important to keep the paths away from other group.

The addresses generated in the protocol are used as the destination of the payment. Of course the source of the fund cannot be pre-determined. Even so, the protocol "cleanses" the transaction records of the payer's address by replacing them with transaction records of other participants.

4.6. Verification

Each group member has an important role of verifying the information shared within the group. The verification includes address verification and transaction verification which will be discussed below.

4.6.1. Bitcoin Address Verification

Bitcoin address verification is about verifying Bitcoin address created by by specific group members such as intermediary payers. Knowing that the address generation employs the HD wallet technique, the generated addresses can be verified against the parent public keys shared by each participant in the communication channels and the paths used by the intermediary payers to generate the addresses. If the group members are able generate the same addresses by using the same parent public keys and the same paths, then the addresses are verified.

4.6.2. Transaction Verification

Each group member also has a role in verifying the transactions created within the protocol. Because the initial payer does not have all information about the transactions in the protocol, as it is intended to be, then the verification can be done partially. Each group member verifies whether the intermediary payers have sufficient fund to commence the transactions. It is also important to verify the behaviour of each member to alert other members in different groups in case of malicious members try to steal the fund by not paying to payers.

4.7. Transaction Circuit

There are several methods investigated in the research which may be potential solutions to achieve the goal of anonymizing Bitcoin transaction with determined characteristics. This section discusses about each potential solution and evaluate each solution in terms of the security models defined in chapter 3.

4.7.1. Notation

In this paper, a form of notation is used to represent Bitcoin transactions used in the proposed protocol. Let Alice (A) sends money to Bob (B), then the transaction (TX) will be called as TX_AB. If the transaction happens in the first phase of the protocol, then the transaction will be written as TX_AB1, while if it is in the second phase of the protocol then it will be called as TX_AB2. Other participants are also determined to be represented with names such as Carol, Darth, Eve, Frank, and George. The participants could also be represented by the first letter of their names.

4.7.2. Unsuccessful Attempts of Creating The Transaction Circuit

4.7.2.1. Two Middlemen, 2-of-2 Multisignature, and CheckLockTimeVerify

The protocol utilizes P2SH scheme which enables user to create a custom ScriptPubKey if needed. Bitcoin atomic transaction (Tiernan, 2013) can also be introduced in the protocol. A detailed explanation about the script used in the scheme will be provided in Appendix 1. The protocol utilized multiple middlemen to allow information splitting. This is an enhancement of the CoinSwap protocol in which the CoinSwap protocol has a weakness of a single middleman knows the entire information of the protocol.

The protocol also uses CLTV. CLTV is used to lock the fund in the Bitcoin system until certain amount of time. Therefore, instead of not publishing the transaction nor setting up partial transactions, utilizing CLTV in P2SH script creates security for both parties. In the protocol, for the receiver to be able to redeem the transaction, the receiver needs to successfully supply a secret key X and a valid signature. The fund is locked until a certain time as defined in CLTV part of the script. It gives guarantee for the receiver of getting paid. It also gives guarantee to the sender in case the receiver fails to supply the correct secret key X that corresponds to a hash value H(X), then the sender can get the fund back.

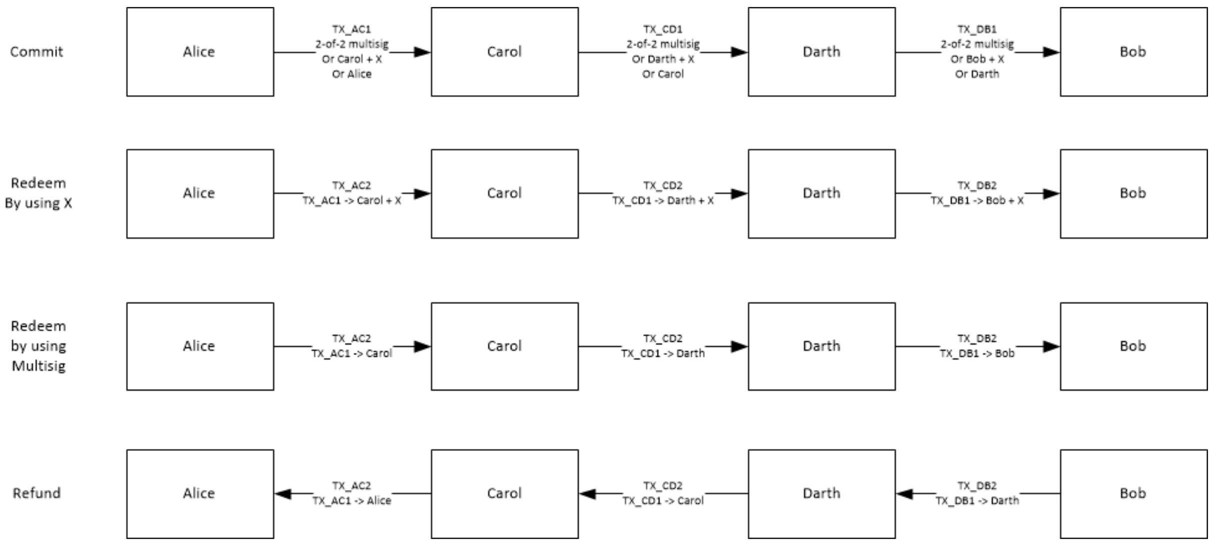


Figure 16. Two Middlemen and 2-of-2 Multisignature Scheme

4.7.2.1.1. The Procedure

The transaction circuit can be described as follows.

- Phase 1: Setup the commit escrow transactions. Bob chooses a random value X, calculate the hash value H(X) and sends it to Alice.
    - Alice creates a P2SH transaction TX_AC1 which can be redeemed by 2-of-2 multisignature of Alice and Carol or by Carol's signature and a secret key X or by Alice's signature after certain amount of time defined in CLTV. TX_AC1 is then published to the network. The TX_AC1 has a CLTV of C_AC1.
    - Carol creates a P2SH transaction TX_CD1 which can be redeemed by 2-of-2 multisignature of Carol and Darth or by Darth's signature and a secret key X or by Carol's signature after certain amount of time defined in CLTV. TX_CD1 is then published to the network. The TX_CD1 has a CLTV of C_CD1 < C_AC1.

    - Darth creates a P2SH transaction TX_DB1 which can be redeemed by 2-of-2 multisignature of Darth and Bob or by Bob’s signature and a secret key X or by Darth’s signature after certain amount of time defined in CLTV. TX_DB1 is then published to the network. The TX_DB1 has a CLTV of C_DB1 < C_CD1.
- Phase 2: Redeem the fund by using a secret key X. This phase is executed if a participant does not want to continue the protocol and each participant can still get paid.
    - Bob creates a signature to redeem the transaction TX_DB1 and thus provides the secret key X to the network.
    - Darth knows that the TX_DB1 has been redeemed by Bob and learns the secret key X. Darth creates a signature and redeems the transaction TX_CD1.
    - Carol knows that the TX_CD1 has been redeemed by Darth and learns the secret key X. Carol creates a signature and redeems the transaction TX_AC1.
- Phase 3: Redeem the fund by using 2-of-2 multisignature. If the participants decide to continue the protocol to the next step, then each pair constructs the multisignature redeem transaction.
    - Darth creates TX_DB2, signs it, and sends it to Bob. Bob signs the transaction and sends TX_DB2 to the network.
    - Carol creates TX_CD2, signs it, and sends it to Darth. Darth signs the transaction and sends TX_CD2 to the network.
    - Alice creates TX_AC2, signs it, and sends it to Carol. Carol signs the transaction and sends TX_ AC2to the network.
- Phase 4: If the protocol failed and the fund is not redeemed by the receivers after CLTV time is expired, then the senders can get refunded.
    - Alice creates TX_AC2 which redeems TX_AC1 by using Alice’s signature and sends TX_AC2 to the network.
    - Carol creates TX_CD2 which redeems TX_CD1 by using Carol’s signature and sends TX_CD2 to the network.
    - Darth creates TX_DB2 which redeems TX_DB1 by using Darth’s signature and sends TX_DB2 to the network.

4.7.2.1.2. Evaluation

By employing the scheme, no middleman holds all of the transaction information and therefore no participant is able to reconstruct the chained transaction without the help of other participants to get more information about the protocol. It also implies that in the same situation, using the protocol, Bob himself will not be able to discover Alice’s address.

The problem with the protocol lies in the usage of the same hash value H(X). Each of the participant knows the value of H(X) and in the end of the protocol, each participant also knows the value of X. If Bob cooperates with Carol by asking any transaction linked with the value of H(X) or X, then Carol is able to determine which transaction was commenced by Alice. There is also a problem in the failsafe of H(X). If Bob decides not to follow the protocol until the end and reveals the secret value X, then all of the transactions TX_DB3, TX_CD3, and TX_AC3 have the same secret value X and therefore can be identified as chained transactions. Any participant can also decide not to follow the complete protocol by immediately redeem the fund by revealing the secret value X which reveals the chained transactions. Therefore, based on the evaluation conducted, the transaction circuit has failed in anonymity requirement.

The protocol employs several protections to prevent any cheating participant. The transaction protections are in the form of the hash-locked transaction and the 2-of-2 multisignature. The hash-locked transaction is used to make sure that the middlemen are paid if Bob is paid. The 2-of-2 multisignature is used to make sure that the sender agrees to construct transaction before paying the money to the receiver.

4.7.2.2. Two Middlemen and Multiparty Computation

To minimize the risk of exposing the relationships between the transactions, a multiparty computation (MPC) is employed as suggested by Steinfeld (2016). MPC can be used to securely construct independent uniform random values to be used as secret keys to be used in the protocol.

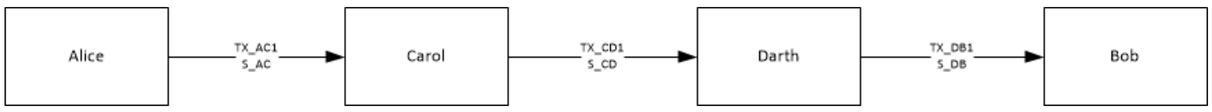


Figure 17. Two Middlemen and Multiparty Computation

4.7.2.2.1. The Procedure

The protocol works as follows.

- Constructing a secret value S_AC which unlocks TX_AC1.
    - Alice, Carol, Darth, and Bob choose random values r_{1A}, r_{1C}, r_{1D}, and r_{1B}.
    - Alice constructs $r_{1A} = r_{1AA} + r_{1AC} + r_{1AD} + r_{1AB}$ and sends r_{1AC} to Carol, r_{1AD} to Darth, and r_{1AB} to Bob.
    - Carol constructs $r_{1C} = r_{1CA} + r_{1CC} + r_{1CD} + r_{1CB}$ and sends r_{1CA} to Alice, r_{1CD} to Darth, and r_{1CB} to Bob.
    - Darth constructs $r_{1D} = r_{1DA} + r_{1DC} + r_{1DD} + r_{1DB}$ and sends r_{1DA} to Alice, r_{1DC} to Carol, and r_{1DB} to Bob.

  - Bob constructs $r_{1B} = r_{1BA} + r_{1BC} + r_{1BD} + r_{1BB}$ and sends r_{1BA} to Alice, r_{1BC} to Carol, and r_{1BD} to Darth.
  - Alice constructs a shared secret $Z_{1A} = r_{1AA} + r_{1CA} + r_{1DA} + r_{1BA}$ and sends Z_{1A} to Carol.
  - Carol constructs a shared secret $Z_{1C} = r_{1AC} + r_{1CC} + r_{1DC} + r_{1BC}$ and sends Z_{1C} to Alice.
  - Darth constructs a shared secret $Z_{1D} = r_{1AD} + r_{1CD} + r_{1DD} + r_{1BD}$ and sends Z_{1D} to Alice and Carol.
  - Bob constructs a shared secret $Z_{1B} = r_{1AB} + r_{1CB} + r_{1DB} + r_{1BB}$ and sends Z_{1B} to Alice and Carol.
  - Alice and Carol can reveal the secret value S_AC = $Z_{1A} + Z_{1C} + Z_{1D} + Z_{1B}$.
- Constructing a secret value S_CD which unlocks TX_CD1.
  - Alice, Carol, Darth, and Bob choose random values r_{2A}, r_{2C}, r_{2D}, and r_{2B}.
  - Alice constructs $r_{2A} = r_{2AA} + r_{2AC} + r_{2AD} + r_{2AB}$ and sends r_{2AC} to Carol, r_{2AD} to Darth, and r_{2AB} to Bob.
  - Carol constructs $r_{2C} = r_{2CA} + r_{2CC} + r_{2CD} + r_{2CB}$ and sends r_{2CA} to Alice, r_{2CD} to Darth, and r_{2CB} to Bob.
  - Darth constructs $r_{2D} = r_{2DA} + r_{2DC} + r_{2DD} + r_{2DB}$ and sends r_{2DA} to Alice, r_{2DC} to Carol, and r_{2DB} to Bob.
  - Bob constructs $r_{2B} = r_{2BA} + r_{2BC} + r_{2BD} + r_{2BB}$ and sends r_{2BA} to Alice, r_{2BC} to Carol, and r_{2BD} to Darth.
  - Alice constructs a shared secret $Z_{2A} = r_{2AA} + r_{2CA} + r_{2DA} + r_{2BA}$ and sends Z_{2A} to Carol and Darth.
  - Carol constructs a shared secret $Z_{2C} = r_{2AC} + r_{2CC} + r_{2DC} + r_{2BC}$ and sends Z_{1C} to Darth.
  - Darth constructs a shared secret $Z_{2D} = r_{2AD} + r_{2CD} + r_{2DD} + r_{2BD}$ and sends Z_{2D} to Carol.
  - Bob constructs a shared secret $Z_{2B} = r_{2AB} + r_{2CB} + r_{2DB} + r_{2BB}$ and sends Z_{2B} to Carol and Darth.
  - Carol and Darth can reveal the secret value S_CD = $Z_{2A} + Z_{2C} + Z_{2D} + Z_{2B}$.
- Constructing a secret value S_DB which unlocks TX_DB1.
  - Alice, Carol, Darth, and Bob choose random values r_{3A}, r_{3C}, r_{3D}, and r_{3B}.
  - Alice constructs $r_{3A} = r_{3AA} + r_{3AC} + r_{3AD} + r_{3AB}$ and sends r_{3AC} to Carol, r_{3AD} to Darth, and r_{3AB} to Bob.
  - Carol constructs $r_{2C} = r_{3CA} + r_{3CC} + r_{3CD} + r_{3CB}$ and sends r_{3CA} to Alice, r_{3CD} to Darth, and r_{3CB} to Bob.
  - Darth constructs $r_{3D} = r_{3DA} + r_{3DC} + r_{3DD} + r_{3DB}$ and sends r_{3DA} to Alice, r_{3DC} to Carol, and r_{3DB} to Bob.
  - Bob constructs $r_{3B} = r_{3BA} + r_{3BC} + r_{3BD} + r_{3BB}$ and sends r_{3BA} to Alice, r_{3BC} to Carol, and r_{3BD} to Darth.

- Alice constructs a shared secret $Z_{3A} = r_{3AA} + r_{3CA} + r_{3DA} + r_{3BA}$ and sends Z_{3A} to Darth and Bob.
- Carol constructs a shared secret $Z_{3C} = r_{3AC} + r_{3CC} + r_{3DC} + r_{3BC}$ and sends Z_{3C} to Darth and Bob.
- Darth constructs a shared secret $Z_{3D} = r_{3D} + r_{3CD} + r_{3DD} + r_{3BD}$ and sends Z_{3D} to Bob.
- Bob constructs a shared secret $Z_{3B} = r_{3AB} + r_{3CB} + r_{3DB} + r_{3BB}$ and sends Z_{3B} to Darth.
- Darth and Bob can reveal the secret value S_DB = $Z_{3A} + Z_{3C} + Z_{3D} + Z_{3B}$.

4.7.2.2.2. Evaluation

By employing the MPC protocol, each transaction uses different secret key and therefore the secret key similarity cannot be used to correlate the chained transactions. Also, the participants can check the correctness of the hash value provided. This scenario minimizes the risk of any participant misbehaves by providing the wrong hash value.

The problem within this protocol is in a case where Bob decides to cancel the protocol while all of the participants have learned about the secret key, these participants would be able to claim the fund despite Bob cancels the deal. In this case, Alice suffers loss. The participants redeem the transactions by using the secret values learned during the MPC protocol. As the conclusion, the protocol may not be able to satisfy the atomic characteristic in which the sender gets refund if the protocol is cancelled before it finishes.

4.7.3. The Final Protocol: Five Middlemen and 2-of-3 Multisignature

In order to handle one or more malicious participant, a change must be made. Instead of using 2 middlemen, 5 middlemen can be employed to construct a 2-of-3 multisignature scheme. Let Alice acts as the payer, Bob as the payee, while Carol, Darth, Eve, Frank, and George as the middlemen as shown in Figure 18.

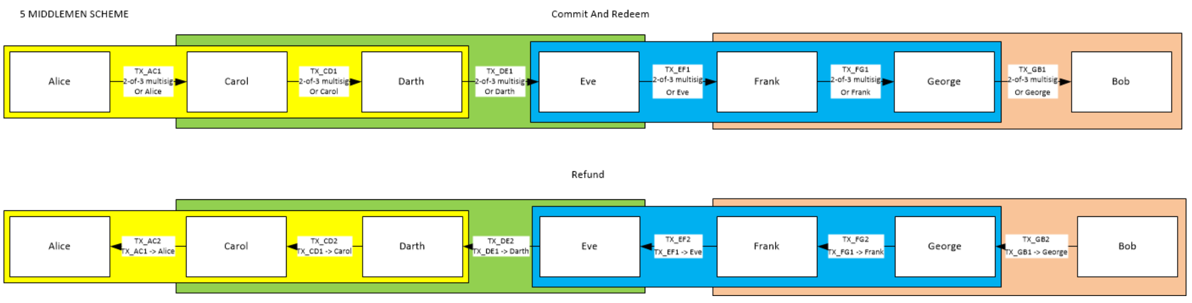


Figure 18. Five Middlemen and 2-of-3 Multisignature

As can be deducted from the image above, in order to securely construct a 2-of-3 multisignature scheme, the minimum number of middlemen is 5, because using 5 middlemen or less exposes the whole transactions to one of the participants, thus will not achieve the goal of dividing the information as such no participant has the complete information about the whole transactions. The participants are grouped into 4 groups, each consists of 3 members:

- Group 1: Alice, Carol, Darth
- Group 2: Carol, Darth, Eve
- Group 3: Eve, Frank, George
- Group 4: Frank, George, Bob.

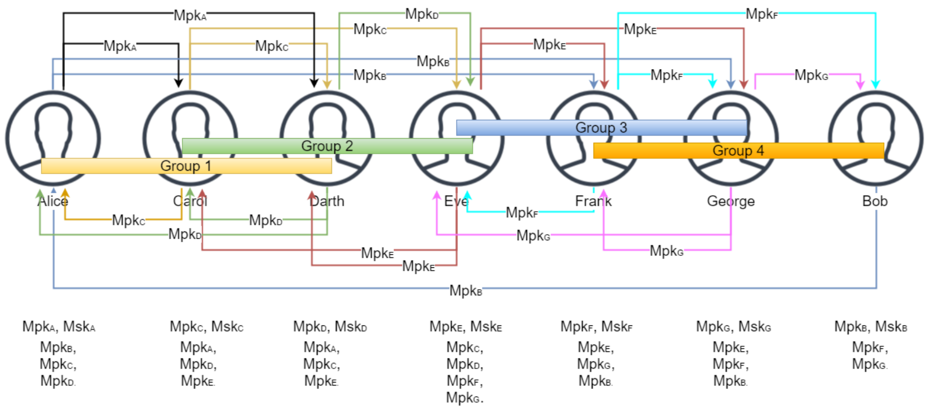
Each group constructs 2-of-3 multisignature over Pay to Script Hash (P2SH) scheme. By employing the 2-of-3 multisignature, if 1 of 3 members in a group cheats, then the payee of the group can still get paid. This creates a form of an escrow mechanism. The protocol also employs CheckLockTimeVerify (CLTV) to lock the fund from the payer and therefore ensures the payees that the payers already have a sufficient fund before constructing the Bitcoin transactions. In order to make sure that the payers get refund in case of the transaction is cancelled, the form of atomic transactions are used in the P2SH script. LockTime is also used to ensure that the transactions are done in the correct sequence. The script for the scheme will be discussed in Appendix 2.

The protocol consists of several phases which can be described as below.

- Phase 0: Preparation
  - Instead of providing his address, Bob creates a new deterministic public key pair which consists of a parent public key and a parent private key. Bob then sends the public key to Alice in a secure channel.
  - Alice sets up an anonymous communication channel with all of the participants including Bob.
  - All participants except Bob create a new deterministic public key pair. Each of the participant publishes the public key only to the members of the group and relate the public key to the session.
  - Alice sends Bob's public key to members of Group 4 (Frank, George, and Bob) with the transaction order of paying Bob certain amount of money.
  - The sender of each transaction creates the transaction along with new addresses for the receiver and the escrow by using the deterministic public keys.
  - TX_AC is defined as a transaction between Alice and Carol, while Darth acts as an escrow. Alice creates new addresses for Carol and Darth and then informs the random value used in the address generation to the group.

- TX_CD is defined as a transaction between Carol and Darth, while Alice acts as an escrow. Carol creates new addresses for Darth and Alice and then informs the random value used in the address generation to the group.
- TX_DE is defined as a transaction between Darth and Eve, while Carol acts as an escrow. Darth creates new addresses for Eve and Carol and then informs the random value used in the address generation to the group.
- TX_EF is defined as a transaction between Eve and Frank, while George acts as an escrow. Eve creates new addresses for Frank and George and then informs the random value used in the address generation to the group.
- TX_FG is defined as a transaction between Frank and George, while Eve acts as an escrow. Frank creates new addresses for George and Eve and then informs the random value used in the address generation to the group.
- TX_GB is defined as a transaction between George and Bob, while Frank acts as an escrow. George creates new addresses for Bob and Frank and then informs the random value used in the address generation to the group.
- The group members can check the address generation and create the private keys which correspond to the addresses generated. The random values are shared within the group but they need to be kept secret from other groups.

The processes within phase 0 can be illustrated in Figure 19.



Alice
Carol
Darth
Eve
Frank
George
Bob
Group 1
Group 2
Group 3
Group 4
Mpk_A
Mpk_B
Mpk_C
Mpk_D
Mpk_E
Mpk_F
Mpk_G
Mpk_A, Msk_A Mpk_B, Mpk_C, Mpk_D.
Mpk_C, Msk_C Mpk_A, Mpk_D, Mpk_E.
Mpk_D, Msk_D Mpk_A, Mpk_C, Mpk_E.
Mpk_E, Msk_E Mpk_C, Mpk_D, Mpk_F, Mpk_G.
Mpk_F, Msk_F Mpk_E, Mpk_G, Mpk_B.
Mpk_G, Msk_G Mpk_E, Mpk_F, Mpk_B.
Mpk_B, Msk_B Mpk_F, Mpk_G.

Figure 19. Key Sharing Diagram.

- Phase 1: Setup the commit 2-of-3 multisignature escrow transaction.
    - Alice creates a P2SH transaction TX_AC1 which can be redeemed by 2-of-3 multisignature of Alice, Carol, and Darth or by Alice’s signature after certain amount of time defined in CLTV. TX_AC1 is then published to the network. The TX_AC1 has a CLTV of C_AC1.

    - Carol creates a P2SH transaction TX_CD1 which can be redeemed by 2-of-3 multisignature of Alice, Carol, and Darth or by Carol’s signature after certain amount of time defined in CLTV. TX_CD1 is then published to the network. The TX_CD1 has a CLTV of C_CD1 < C_AC1.
    - Darth creates a P2SH transaction TX_DE1 which can be redeemed by 2-of-3 multisignature of Carol, Darth, and Eve or by Darth’s signature after certain amount of time defined in CLTV. TX_DE1 is then published to the network. The TX_DE1 has a CLTV of C_DE1 < C_CD1.
    - Eve creates a P2SH transaction TX_EF1 which can be redeemed by 2-of-3 multisignature of Eve, Frank, and Bob or by Eve’s signature after certain amount of time defined in CLTV. TX_EF1 is then published to the network. The TX_EF1 has a CLTV of C_EF1 < C_DE1.
    - Frank creates a P2SH transaction TX_FG1 which can be redeemed by 2-of-3 multisignature of Eve, Frank, and George or by Frank’s signature after certain amount of time defined in CLTV. TX_FG1 is then published to the network. The TX_FG1 has a CLTV of C_FG1 < C_EF1.
    - George creates a P2SH transaction TX_GB1 which can be redeemed by 2-of-3 multisignature of Frank, George, and Bob or by George’s signature after certain amount of time defined in CLTV. TX_GB1 is then published to the network. The TX_GB1 has a CLTV of C_GB1 < C_FG1.
- Phase 2: Redeem the transactions by using 2-of-3 multisignature.
    - Bob creates TX_GB2 which redeems TX_GB1, signs it, and sends it to George. George signs the transaction and sends TX_GB2 to the network. If George does not want to sign the transaction, then Bob asks Frank to sign the transaction TX_GB2.
    - George creates TX_FG2 which redeems TX_FG1, signs it, and sends it to Frank. Frank signs the transaction and sends TX_FG2 to the network. If Frank does not want to sign the transaction, then George asks Eve to sign the transaction TX_FG2.
    - Frank creates TX_EF2 which redeems TX_EF1, signs it, and sends it to Eve. Eve signs the transaction and sends TX_EF2 to the network. If Eve does not want to sign the transaction, then Frank asks Bob to sign the transaction TX_EF2.
    - Eve creates TX_DE2 which redeems TX_DE1, signs it, and sends it to Darth. Darth signs the transaction and sends TX_DE2 to the network. If Darth does not want to sign the transaction, then Eve asks Carol to sign the transaction TX_DE2.
    - Darth creates TX_CD2 which redeems TX_CD1, signs it, and sends it to Carol. Carol signs the transaction and sends TX_CD2 to the network. If Carol does not want to sign the transaction, then Darth asks Alice to sign the transaction TX_CD2.

    - Carol creates TX_AC2 which redeems TX_AC1, signs it, and sends it to Alice. Alice signs the transaction and sends TX_AC2 to the network. If Alice does not want to sign the transaction, then Carol asks Darth to sign the transaction TX_AC2.
- Phase 3: If the transaction is cancelled and the fund is not redeemed by the receivers after CLTV time is expired, then the senders can get their money back.
    - Alice creates TX_AC2 which redeems TX_AC1 by using Alice's signature and sends TX_AC2 to the network.
    - Carol creates TX_CD2 which redeems TX_CD1 by using Carol's signature and sends TX_CD2 to the network.
    - Darth creates TX_DE2 which redeems TX_DE1 by using Darth's signature and sends TX_DE2 to the network.
    - Eve creates TX_EF2 which redeems TX_EF1 by using Eve's signature and sends TX_EF2 to the network.
    - Frank creates TX_FG2 which redeems TX_FG1 by using Frank's signature and sends TX_FG2 to the network.
    - George creates TX_GB2 which redeems TX_GB1 by using George's signature and sends TX_GB2 to the network.

In a normal situation where the participants decide to follow the protocol until the end, those participants do phase 0, 1, and 2 without continuing to phase 3. But if the protocol is broken and the participants do not wish to continue, then if the current phase is 1, then the participants continue to phase 3 without doing the phase 2.

4.8. Fund Splitting

It is a good idea to split the fund sent from the payer to the payee in multiple transactions, especially when the amount of fund is quite unique or large. As suggested in an idea called Merge Avoidance (Hearn, 2013), it is important to create more common amount of money and not too specific as it may be used as an artefact to link the transactions and may reveal certain degree of users' identity. The problem with fund splitting is that it will require more transaction fee which will increase according to the number of transactions created.

Yang (2012) suggested that the amount of fund is in the power of 2, e.g. $\frac{1}{4}$ BTC, $\frac{1}{2}$ BTC, 1 BTC, 2 BTC, and so on. It is suggested that the convention is followed by all users using the protocol which in turn constructs uniformity between the transactions employing the same protocol.

4.9. Protocol Fee

To motivate the middlemen of providing such service, time, and resource to support the protocol used by the users, those middlemen deserve a certain amount of protocol fee. To accommodate all 6 middlemen, each of the middlemen could take a random fee between 0 to 1% of the transacted fund. Assuming the middlemen take the maximum protocol fee possible, the highest protocol fee to be paid by the user will be 6% of total fund sent to the payer. This protocol fee should be able to cover the transaction fee paid to the Bitcoin miners for the transactions to be confirmed in the blockchain.

4.10. Constant Transference

Constant transference is implemented to hide large amount of bitcoins and to avoid spikes in Bitcoin Days Destroyed monitoring. Constant transference means that the bitcoins are constantly moved to different addresses every short period of time. Bitcoin Days Destroyed could be a starting point of analysis if large amount of bitcoins which have been stored for quite long time are immediately transferred without planning. Combined with fund splitting, the constant transference can make it harder for the analyst or adversaries to track the transactions.

Despite provides the ability to destroy the bitcoin age, Constant transference has the similar problem with fund splitting. The problem is about the transaction fee which must be paid for each transactions.

Chapter 5 – Implementation

5.1. Overview

This chapter describes a limited small scale test implementation of the final protocol (described in Section 4.7.3) into the real Bitcoin transaction network. This chapter also discusses several topics such as the tools for the implementation, limitations that might exist during the implementation, and the transactions created by using the protocol.

5.2. Tools

Several tools utilized to implement and evaluate the protocol is mentioned below. These tools are run in a Windows environment with a stable Internet connection.

5.2.1. BX

BX (Bitcoin Explorer) is a command-line tool created by Eric Voskuil as a major update of an existing software called SX. SX and later BX were developed over libbitcoin library and obelisk servers to run queries to Bitcoin network (Voskuil, 2014). BX has features that can be used to construct the Bitcoin transaction script, evaluate, and send the script to the network.

BX has local functions as well as Internet-based functions. Several local functions including creating and signing transaction script, while Internet-based functions including gathering information about the Bitcoin network: block height, transaction history, unspent transaction output, and several other functions. For the later functions, BX relies on Bitcoin Server (BS) which is an upgraded version of Libbitcoin/Obelisk server to get the information for BX (Voskuil, 2015). BX itself can be considered as a client application of BS.

5.2.2. BITA

BITA (Bitcoin Anonymizer) is a graphical user interface (GUI) and automation tool written to operate BX in a simpler way. BITA is used to implement and evaluate the protocol into real Bitcoin transactions. There are several tasks handled by BITA:

- Creating and managing Bitcoin key pairs.
- Creating and signing Bitcoin transactions.
- Validating and sending the transactions into Bitcoin network.

BITA is not a production tool but more like an evaluation tool. BITA solely depends on BX; if the BX could not operate, then BITA could not produce proper results as well. BITA translates the operation executed by the user through its user interface and creates the commands which is then sent to BX. The result from BX is translated by BITA and displayed by using a graphical interface.

5.3. Limitations

There are several limitations which affect the implementation of the protocol. First, as the number of transactions in Bitcoin network increases, new blocks created in the network almost reach the maximum size of a block which is 1 MB. Therefore, transaction delays tend to increase. It means, it requires more time for a transaction to be confirmed in the Bitcoin system. This delay affects the time required for the protocol to finish all of the phases. To minimize the impact of the delay, several adjustment will be made, such as increasing the transaction fee paid to the miner and expanding the CLTV time.

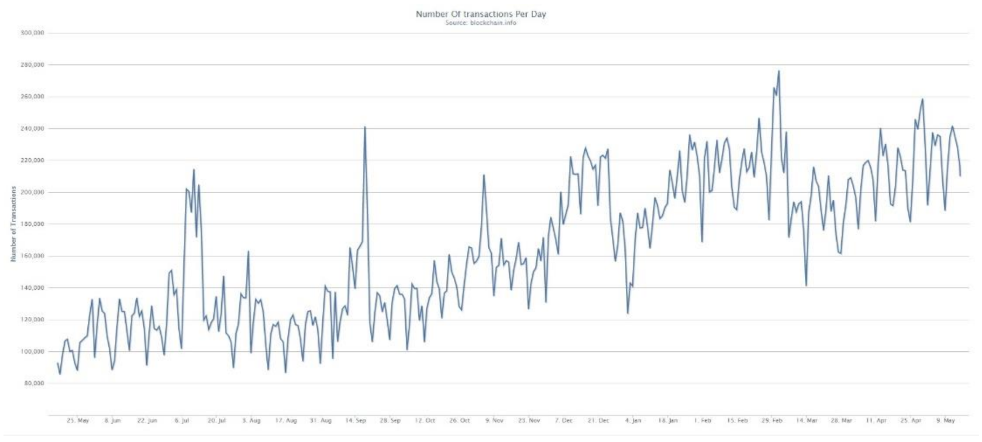


Figure 20. Number of Transactions per Day, data per 15 May 2016. Source: blockchain.info.

Second, the protocol is supposed to be run by multiple participants each with different coins, therefore it is not possible to mimic this property due to resource limitation in the research. The coins used in the protocol implementation of the research comes from a same source and therefore the taint-proof property could not be examined. The transactions are sent from a simulation application run in the same machine, the same IP address, and the same tool which might connects to the same Bitcoin node. This condition is obviously not ideal to construct a real situation. Because the transactions will be sent from a simulation application, it implies that the implementation could not describe the process of the participants communicating the information required by the protocol. The limitations stated above implies that the implementation only reproduces the transactions themselves without fully implementing the phases of the protocol.

5.4. Transactions

In order to examine the proposed protocol in different scenarios, several experiments are deployed by employing the protocol. The scenarios including all honest participants, several cheating participants, and all refund scenario which will refund the money back to the payer.

5.4.1. All Honest Participants

This scenario means that every participants behave honestly and wants to follow the protocol. The first step would be creating Bitcoin addresses for each of the participants. Let Alice wants to send 0.0001 BTC to Bob by using the protocol. Alice, Carol, Darth, Eve, Frank, and George each must have their own bitcoin to be used in the transaction. This bitcoin is prepared in their "payer address". Then, due to each transaction requires a certain amount of fee, these 6 participants need to prepare at least 0.0005 BTC to create all of the transactions. The list of the transactions in the protocol can be seen in Table 3.

Table 3. The list of the TXID for All Honest Participant Scenario

No	TX	Phase	Transaction ID
1	TX_AC	Commit	92a32124479decf535c949f401a311ba71281c935dc643c838a725b2b2617c53
2		Redeem	d08303e4459bc0f39c35e873ce1564cd36375a1e4598f05db7ddd21577e13270
3	TX_CD	Commit	c077d1ae003b264ec582ea6be5c9310f196d278e37b0d71948183cce90167383
4		Redeem	1047ebed7b7fd9ee21e9e4f32fb10665f84fca128f394993dbc66c22c0a2dedf
5	TX_DE	Commit	d396b8759db2553fa976e3f0e45dc7de35ae13a56aee38e9b935e434b16661f6
6		Redeem	4cb7cb8d9d333e21c0edacc4f70cee82b50fa277a8d3463e210ea35e58b1e777
7	TX_EF	Commit	9742bef8e9e1a040a6bd58da5af8d155ac576a6326f0d5beb033ace32890ba34
8		Redeem	272e899d22b68d252b6f15884f28390810a5d5caf1cb823c8dcc6881af0063d9
9	TX_FG	Commit	e1c04f47674b4d1a6c1585e54593a592702ddc0aca5e85a2d1758de2c2826221
10		Redeem	90686782a8ccb2cd4784e97ca9b62b1e8c2b5dafbccd9c27a1df5f4080daa98e
11	TX_GB	Commit	de4b26d91f1065c9e90b27eea432a540fb1279c6e1f859849a19b531fb83432b
12		Redeem	88ff95465465097b9edf1068e3e297f44a5058ae890d21964ca92c10e5ba7f50

The transactions in the commit phase were confirmed within a same block, which is block 411861. For the redeem phase, the timeline data can be seen in Table 4.

Table 4. Transaction Timeline for All Honest Participant Scenario

Transaction	CLTV Block	LockTime Block	Confirmed Block	Transaction Delay
(1)	(2)	(3)	(4)	(5) = (4) – (3)
TX_AC	411880	411877	411878	1
TX_CD	411877	411874	411875	1
TX_DE	411874	411871	411872	1
TX_EF	411871	411868	411869	1
TX_FG	411868	411865	411867	2
TX_GB	411865	411862	411863	1

The LockTime is the earliest time that the transaction can be confirmed in the blockchain, while the CLTV is the latest time for the confirmation before it is available for the payer to be redeemed. In this implementation, the CLTV is 3 blocks away from each transaction. It means, the transaction delay cannot exceed 3 blocks. Table 4 shows that the transaction delay is between 1 to 2 blocks which is still safe and the protocol can be executed without any time overlap. The transaction confirmation before the CLTV means that the payer could not refund the money and therefore it creates a guarantee to the payee that the fund can still be redeemed.

5.4.2. Several Cheating Participants

The protocol is supposed to handle several cheating participants, as long as each of the groups satisfies honest majority requirement. In this scenario, Alice, Eve, and Bob do not cooperate; even so, the protocol could still proceed. Assuming that Alice wants to send 0.0001 BTC to Bob, but Alice actually hesitates to pay for it. Bob as the receiver does not want to help the protocol, while Eve does not want to cooperate as well. The list of the transactions involved is shown below.

Table 5. The list of the TXID for Several Cheating Participant Scenario

No	TX	Phase	Transaction ID
1	TX_AC	Commit	2eceea1bd7803306a58e7747b6fba8236f93f4687140aa43a8c6188374b877df
2		Redeem	f6a5b94524284d8353555ea592b969e7d6394210288a467358e37dc1d4dcca56
3	TX_CD	Commit	14b7d6b669c169fbc82b923c84a316e730d1be5868644c18d88efeff4ed6a2a7
4		Redeem	e5ed032d98f556196c761a95d87b7fa9081ba9488eecc2b526dd3e2a2c481b50
5	TX_DE	Commit	dd25f047888112e1309c6d91b7d4da24b9e3dc0c43b4d0a5c2646b1e1c1fe999
6		Redeem	0461f166056fb2966b474b972fbca487df9d747ca12d1ab8f8031610fe139aa8
7	TX_EF	Commit	4580178f62b002ffb39f65b141e9fa34ebbb8983a44e2ed0a185e76cdfc6bab0
8		Redeem	089d06e498bbabbbfcf7a8888daababb49f32c67c06f2e41fcb003b104fb33bc
9	TX_FG	Commit	03dfc278b82fa540ed9680f87307b1af208613ce60b7cba5a7b020cbdd93ce41
10		Redeem	e1b782006b9d8a1d661a517acdaa32447553a1485dc69b51d0d777f9c3162203
11	TX_GB	Commit	052f12f6c8ed0de9f3dd04ba175c10200bdc0ca43569946af36227555af289dd
12		Redeem	832abbd0baf84a203104bb635297e71014ed3e9924a3e52b6a1cb54156d6598a

The commit phase transactions are confirmed in the same block of 411960, but the redeem phase has a timeline as shown in Table 6.

Table 6. Transaction Timeline for Several Cheating Participant Scenario

Transaction	CLTV Block	LockTime Block	Confirmed Block	Transaction Delay
(1)	(2)	(3)	(4)	(5) = (4) – (3)
TX_AC	411978	411975	411976	1
TX_CD	411975	411972	411973	1
TX_DE	411972	411969	411970	1
TX_EF	411969	411966	411967	1
TX_FG	411966	411963	411965	2
TX_GB	411963	411960	411962	2

5.4.3. Refund

This scenario describes a case in which the participants started the protocol but later they do not want to proceed and then tries to refund their money they have deposited in the commit phase.

Table 7. The list of the TXID for Refund Scenario

No	TX	Phase	Transaction ID
1	TX_AC	Commit	64b20fd996dd7eda4f1217300ccab038717d2a0a94e2961b7becbbe168e3e525
2		Redeem	177202e0808b632d67d564e563f558b5e5735c77b70cdce68a6d76fe00faeb30
3	TX_CD	Commit	1f6c57e57559c2d30d1b92c141cd93dc48adff0a4b016889a5dbed59674fc132
4		Redeem	026f414fa889889a576ddf26a3e81ab0c3a1512c92d61ff1b9c3fc9393549c3a
5	TX_DE	Commit	86e0597840bf0a75359af9c1f3b8311f80caff68b31617e37f8042e211b08dcb
6		Redeem	9f34b2f92876a8986e17fe8bcb4a9182d7af12503d8672cf121643dd269ede7c
7	TX_EF	Commit	34694c267066533371a7770288f3907c66fee0b279945ec2a26a3980cd66bd07
8		Redeem	b6ce481cedd3e274b188631dffb2ee33b7885b81a7147187f834a37308eb7e33
9	TX_FG	Commit	b5fa6604c31cef39553a79bd09bcd43d90fc37f6351789a4460d849bd4735f8e
10		Redeem	bab148d1e1d01c26009ecfcc0b2f3842020fcefcedaa269b10bc2425b767b540
11	TX_GB	Commit	6153fc60d8029922d02f824d651b97104cb46f479f2de3e469e958d89baeabb8
12		Redeem	f18055b0608aae4dd18e0d5acc93571d31197ad5b489ae1d6d27fb7b71ba1a26

The commit phase transactions are confirmed in the same block, but the redeem phase has a timeline as shown in Table 8. There is a slight different compared to 2 previous scenarios in which this scenario requires the LockTime value to be any value larger than the CLTV value and does not bound by any upper limit, but to ensure that the payer get refunded, the payer needs to send the refund transaction as soon as possible.

Table 8. Transaction Timeline for Refund Scenario

Transaction	CLTV Block	LockTime Block	Confirmed Block	Transaction Delay
(1)	(2)	(3)	(4)	(5) = (4) – (3)
TX_AC	413469	413470	413471	1
TX_CD	413466	413467	413468	1
TX_DE	413463	413464	413465	1
TX_EF	413460	423461	413463	2
TX_FG	413457	413458	413459	1
TX_GB	413454	413455	413456	1

Chapter 6 – Evaluation

6.1. Overview

As it is already discussed in chapter 4.7.3, the transaction circuit that consists of five middlemen and 2-of-3 multisignature is evaluated for the fitness of the requirements that have already been determined. It is assumed that Bob as the payee also wants to know the original address of Alice, thus tries to gather any information he could get to try guessing Alice’s address. In this scenario, Bob acts as the attacker for the protocol.

Aside from the information gained by himself, Bob also wants to cooperate with any of the middlemen. In any anonymizing protocol, it is obvious that if all of the middlemen cooperate with each other, Alice’s address will be easily recovered. In this evaluation, the attacker is only limited to cooperate with 1 of the middlemen.

6.2. Anonymity Evaluation

The information gained by each participant is shown in the Table 9 below. Because the transactions within the Bitcoin system is publicly available, we also assume that everyone has the ability to access that information.

Table 9. Information Gained by Each Participant.

Participant	Knowledge of Transaction	Knowledge of Deterministic Public Key	Group Membership
Alice	TX_AC,TX_CD	Alice, Carol, Darth, Bob	1
Carol	TX_AC, TX_CD, TX_DE	Alice, Carol, Darth, Eve	1,2
Darth	TX_AC, TX_CD, TX_DE	Alice, Carol, Darth, Eve	1, 2
Eve	TX_CD, TX_DE, TX_EF, TX_FG	Carol, Darth, Eve, Frank, George	2, 3
Frank	TX_EF,TX_FB, TX_GB	Eve, Frank, George, Bob	3,4
George	TX_EF, TX_FB, TX_GB	Eve, Frank, George, Bob	3,4
Bob	TX_FG,TX_GB	Frank, George, Bob	4

If Bob cooperates with Frank or George, then they can construct the transaction circuit from TX_EF through TX_GB by linking the public keys of the session and the transactions created. But since Bob, Frank, and George do not have any information about Alice (either Alice’s public key or Alice’s Bitcoin source address), they cannot determine the transactions coming from Alice’s Bitcoin address.

If Bob cooperates with Eve, then they can construct the transaction circuit from TX_CD through TX_GB by linking the public keys owned by Frank and George with the transactions created between them. But since Bob and Eve do not have any information about Alice, they cannot determine which transactions coming from Alice’s Bitcoin address.

If Bob cooperates with Carol or Darth, then they may not be able to determine which transactions are related to Bob's public key, since Carol and Darth never get any information about Bob's public key from anyone, it may not be possible to associate the transactions with Bob's public key. Therefore, Carol or Darth must guess the transaction coming from Alice.

In order to reveal the transaction sent from Alice to Bob, Bob must cooperate with at least 2 of the middlemen. By using the methods explained above, Bob then can construct the linked transactions which lead to the original transaction sent by Alice.

The same arguments would also apply to the unlinkability characteristic of the protocol. Bob cannot tell 2 different transactions coming from Alice assuming Bob receives multiple transactions from multiple senders each has the same amount of money.

6.3. Cheating Evaluation

The protocol utilizes 2-of-3 multisignature scheme and timing to mitigate the cheating risk. By using 2-of-3 multisignature scheme, if a payer refuses to sign the redeem transaction, then the payee can ask the escrow party to sign the transaction and the redeem transaction is valid. Despite the middlemen have the chance to cheat, they stake their reputations if they do not behave honestly.

The similar way goes to a case in which a middleman tries to pay less money to the payee, then the payee rejects the transaction and asks the escrow to sign the transaction on behalf of the payer. The middlemen can check whether they have set the correct amount of money by using the information provided within the transactions and the information provided by Alice in the beginning of the protocol.

The protocol also uses a specialized P2SH script which can be used to construct atomic transactions which can be cancelled at any stage. If the transaction is cancelled, all participants can redeem their own fund. The timing scheme is implemented by the CLTV and Locktime, and therefore the cheating scheme can be easier to detect.

In case of one or more participants do not behave honestly, then there is a limitation of this protocol. Assuming that that all of the payees behave honestly, then the rules of cheating are as follows.

- If George cheats then Frank must be honest for Bob to get paid. George acts as an escrow and sign the transaction TX_GB2 to pay Bob.
- If Frank cheats then Eve must be honest for George to get paid. Eve acts as an escrow and sign the transaction TX_FG2 to pay Bob.
- If Eve cheats then Bob must be honest for Frank to get paid. Bob acts as an escrow and sign the transaction TX_EF2 to pay Frank.
- If Darth cheats then Carol must be honest for Eve to get paid. Carol acts as an escrow and sign the transaction TX_DE2 to pay Eve.

- If Carol cheats then Alice must be honest for Darth to get paid. Alice acts as an escrow and sign the transaction TX_CD2 to pay Darth.

The cheat (C) and honest (H) possibilities for 5 middlemen scheme can be represented in the Table 10.

Table 10. Cheat (C) and Honest (H) Possibilities of the protocol with cheating participants.

	Alice	Carol	Darth	Eve	Frank	George	Bob
Case 1	C	H	H	C	H	H	C
Case 2	H	C	H	H	C	H	H
Case 3	H	H	C	H	C	H	H
Case 4	H	H	C	H	H	C	H
Case 5	H	H	C	H	H	H	C

As shown in the Table 10, no more than 1 malicious participant is allowed in a group in order for the participants to get paid once they pay to other participants. If multiple participants in the same group are cheating, then the cheating attempt will succeed and result in financial loss by other participants.

In case of any middleman tries to forge the transactions by faking the digital signature of Bitcoin, then the security relies on the unforgeability of the 256 bit private key of ECDSA. Several attacks against ECDSA is discussed by Johnson, Menezes, and Vanstone (2001), and also by Brown (2000). The security of ECDSA itself is beyond the scope of the research.

6.4. Performance and Correctness Evaluation

The implementation in chapter 5.4 shows that the protocol can be implemented in real Bitcoin transactions. Each transaction is expected to be confirmed within the maximum of 3 blocks for the protocol to be confirmed. This configuration works despite the current Bitcoin network is almost reach its maximum capacity which introduces a longer delay than expected.

In the all honest participants scenario, the commit phase is confirmed in block 411861 while the last transaction in redeem phase is confirmed in block 411878. Therefore, it took 17 blocks to finish the protocol. Despite a block is supposed to be generated in 10 minutes, this is not always the case. Block 411861 was generated in 15 May 2016 at 03:41[8] while block 411878 was generated in 15 May 2016 at 07:22[9]. Therefore, although in theory it is supposed to create 17 blocks in 170 minutes, in reality it needed 221 minutes to create those 17 blocks.

[8] https://chain.so/block/BTC/411861
[9] https://chain.so/block/BTC/411878

Similar result was generated in several cheating participants scenario in which the commit phase is confirmed in block 411960 and the last transaction in redeem phase is confirmed in block 411976. It means it took 16 blocks to finish the whole transactions in the protocol. In reality, it only took 83 minutes to create the whole blocks.

The refund scenario will only be executed if the participants do not wish to proceed to the redeem phase and decide to cancel the protocol. Therefore, the redeem transactions will not be generated. Instead, each of the paying participants construct refund transactions to get their money back. In this scenario, it is not necessary to have any upper time limit, because the participants can refund their money at any time.

The information in Table 8 shows the refund scenario timeline. Despite the confirmation time seems to be instant (in 1 or 2 blocks after sending the transaction to the network), there is a possibility of time delay. Unfortunately, the delay time is beyond the Bitcoin users' control. The prioritization is done by the Bitcoin miners by sorting the transaction fee per kilobyte (Bitcoin Wiki, 2011d). Therefore, there is actually no guarantee that the transactions will be included in the next block assuming that other transactions could pay higher transaction fee.

In general, the timeline setting will have a big impact towards the success or the protocol. The success level of the protocol does not determined by the speed or how fast the protocol can be finished. In contrast, the longer the timeline setting is, the better the outcome will be. The reason is that the number of transactions in the time period can become larger, hence it increases the anonymity of the transactions within the protocol. If the timeline setting is too short, there is a chance that the protocol will break due the transaction delay caused by a heavy traffic of Bitcoin network.

Chapter 7 – Conclusion

7.1. Contribution

The research provides a protocol as a solution of anonymizing the Bitcoin transaction. The protocol is proposed after studying the characteristics of Bitcoin transaction and existing solutions of anonymizing Bitcoin transactions. Revisiting the research goal, the proposed protocol fulfils the requirements as specified.

- **Create an anonymizing protocol which is compatible with the current Bitcoin network.**

  The proposed protocol is fully compatible with the current Bitcoin network and therefore does not require any modification to the Bitcoin protocol. The proposed protocol employs several techniques such as P2SH, multisignature, CLTV, and LockTime. The P2SH is used to create a custom payment script. The multisignature is used to construct an escrow scheme. The CLTV and the LockTime is used to create a transaction window time in which the transaction should be sent and confirmed by the Bitcoin system.

- **The anonymizing protocol must protect the information of the payer. The payee cannot identify which Bitcoin addresses belong to the payer.**

  The mechanism in the protocol provides an anonymity of the payer. The anonymity requires the payee not to be able to identify the payer's original transactions among several similar transactions during a certain time period.

- **No participant should be able to get information on the transactions within the protocol.**

  In order to fulfil the requirement, the protocol employs several middlemen. Those middlemen do not have a full information of the whole protocol. The information is split among the participants, therefore each of the middlemen only have a part of the information. This characteristic enables the protocol to be effective against any 2 participants cooperating together to discover the information about the original payer.

- **It must be infeasible for any adversary outside of the protocol to link the transactions created by the protocol.**

  Despite each transaction employs a similar script, the transactions themselves are not related each other. This is achieved by replacing the original fund with other fund owned by the middlemen. The source of money is assumed not to be related each other and therefore the transaction holds no relationship between each other.

- **No participant can steal the money.**

  The protocol employs an escrow mechanism in constructing a guarantee that the payee will always be paid if any of the payer or the escrow is honest. Therefore, despite a participant tries to cheat by not providing a correct signature for the payment, the escrow mechanism can cover the signature requirement for the Bitcoin transaction to be confirmed as a valid transaction.

7.2. Challenges and Limitations

Despite the ability of the protocol to handle securely up to 2 malicious participants, there are still potential security concerns regarding the protocol in some application scenarios. If the number of transactions utilizing the same protocol is small during a certain period of time, then the effort of analysing the transactions can be smaller, thus can make it easier for the curious payee to learn the information of the payer. The custom P2SH script can be utilized to distinguish the transactions and mark them as part of anonymizer protocol.

The proposed protocol could still protect the anonymity of the payer although the payee cooperates with at most 1 participants trying to recover the information of the payer. But without further expansion, the protocol could not protect the information of the payer if more than 2 participants cooperate by combining any information they have trying to recover the payer's information.

7.3. Conclusion

Bitcoin is an open system in which anyone can trace every transaction created in the system beacause it employs a distributed ledger to replace a trusted central authority. In such system, the privacy of Bitcoin users may need to be protected in terms of protecting them from potential of harmful monitoring activities. The proposed protocol may be suitable to become an alternative to solve the problem. The protocol exchanges the user's bitcoin with another bitcoin which may not have any relationship to the original bitcoin.

7.4. Future Research

The future works could be used to investigate the use of a more common Bitcoin transaction script towards the protocol which will minimize the characteristics of the proposed protocol. Another direction would be combining several middlemen in every transaction to obfuscate the linked transaction furthermore.

Considering that there are several limitations in this research, the future works could also be used to eliminate the limitations to get a broader view and larger perspective of the protocol. Multiple Bitcoin sources each represents a middleman of the protocol could characterize the protocol in a more detail way, because that is what the middlemen are supposed to do. Multiple concurrent transactions implementing the protocol could also be done to further investigate the effectiveness of the protocol in a large scale.

References

Andresen, G. (2011). M-of-N Standard Transactions. Retrieved from https://github.com/bitcoin/bips/blob/master/bip-0011.mediawiki

Andresen, G. (2012). Pay to Script Hash. Retrieved from https://github.com/bitcoin/bips/blob/master/bip-0016.mediawiki

Androulaki, E., Karame, G. O., Roeschlin, M., Scherer, T., & Capkun, S. (2013). Evaluating user privacy in bitcoin *Financial Cryptography and Data Security* (pp. 34-51): Springer.

Andrychowicz, M., Dziembowski, S., Malinowski, D., & Mazurek, L. (2014). *Secure multiparty computations on bitcoin.* Paper presented at the Security and Privacy (SP), 2014 IEEE Symposium on.

Back, A. (2013). Coin Validation misunderstands fungibility and could destroy bitcoin. Retrieved from https://bitcointalk.org/index.php?topic=333882.0

Bartlett, J. (2014). *The Dark Net*: The Random House.

Bellare, M., & Neven, G. (2007). Identity-based multi-signatures from RSA *Topics in Cryptology–CT-RSA 2007* (pp. 145-162): Springer.

Ben Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., & Virza, M. (2014). *Zerocash: Decentralized anonymous payments from Bitcoin.* Paper presented at the Security and Privacy (SP), 2014 IEEE Symposium on.

Biryukov, A., Khovratovich, D., & Pustogarov, I. (2014). *Deanonymisation of clients in Bitcoin P2P network.* Paper presented at the 2014 ACM SIGSAC Conference on Computer and Communications Security.

Bitcoin Wiki. (2011a, 22 June 2012). Bitcoin Days Destroyed. Retrieved from https://en.bitcoin.it/wiki/Bitcoin_Days_Destroyed

Bitcoin Wiki. (2011b, August 17, 2015). Technical background of version 1 Bitcoin addresses. Retrieved from https://en.bitcoin.it/wiki/Technical_background_of_version_1_Bitcoin_addresses

Bitcoin Wiki. (2011c). Testnet. Retrieved from https://en.bitcoin.it/wiki/Testnet

Bitcoin Wiki. (2011d, 11 December 2015). Transaction fees. Retrieved from https://en.bitcoin.it/wiki/Transaction_fees

Bitcoin Wiki. (2011e, May 7, 2014). Zero Knowledge Contingent Payment. Retrieved from https://en.bitcoin.it/wiki/Zero_Knowledge_Contingent_Payment

Bitcoin Wiki. (2012a, July 8, 2015). Contract. Retrieved from https://en.bitcoin.it/wiki/Contract

Bitcoin Wiki. (2012b, 27 May 2015). Pay to Script Hash. Retrieved from https://en.bitcoin.it/wiki/Pay_to_script_hash

Bitcoin Wiki. (2013, September 25, 2015). Script. Retrieved from https://en.bitcoin.it/wiki/Script

Bitcoin Wiki. (2014, June 30, 2014). RIPEMD-160. Retrieved from https://en.bitcoin.it/wiki/RIPEMD-160

Bohr, J., & Bashir, M. (2014, 23-24 July 2014). *Who Uses Bitcoin? An exploration of the Bitcoin community.* Paper presented at the Privacy, Security and Trust (PST), 2014 Twelfth Annual International Conference on.

Bonneau, J., Narayanan, A., Miller, A., Clark, J., Kroll, J. A., & Felten, E. W. (2014). Mixcoin: Anonymity for Bitcoin with accountable mixes *Financial Cryptography and Data Security* (pp. 486-504): Springer.

Brown, D. R. L. (2000). *The exact security of ECDSA.* Paper presented at the Advances in Elliptic Curve Cryptography.

Brown, D. R. L. (2010). *Standards for Efficient Cryptography*. Retrieved from http://www.secg.org/sec2-v2.pdf

Bryans, D. (2014). Bitcoin and money laundering: mining for an effective solution. *Ind. LJ, 89*, 441.

Buterin, V. (2013). Why The Bitcoin Greenlist is Structurally Dangerous to the Bitcoin Ecosystem. Retrieved from https://bitcoinmagazine.com/articles/why-the-bitcoin-greenlist-is-structurally-dangerous-to-the-bitcoin-ecosystem-1384492133

Chaum, D. (1983). *Blind signatures for untraceable payments.* Paper presented at the Advances in cryptology.

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

Cohen, B. (2013). Obama Initiative Spawns Identity Based Bitcoin Greenlist. Retrieved from https://bitcoinmagazine.com/articles/obama-initiative-spawns-identity-based-bitcoin-greenlist-1384277984

Coutu, O. (2013). Decentralized Mixers for Bitcoin. Retrieved from https://www.youtube.com/watch?v=6hc8qaR_Fok

Dai, W. (1998). B-money. Retrieved from http://www.weidai.com/bmoney.txt

Diffie, W., & Hellman, M. E. (1976). *Multiuser cryptographic techniques.* Paper presented at the Proceedings of the June 7-10, 1976, national computer conference and exposition.

Dingledine, R., Mathewson, N., & Syverson, P. (2004). *Tor: The second-generation onion router*. Retrieved from

djsjjd. (2014). Sanitizing Bitcoin: This Company Wants To Track 'Clean' Bitcoin Accounts. Retrieved from https://www.reddit.com/r/Bitcoin/comments/1qj7sw/sanitizing_bitcoin_this_company_wants_to_track/cddie7r

ElGamal, T. (1985). *A public key cryptosystem and a signature scheme based on discrete logarithms.* Paper presented at the Advances in cryptology.

Franco, P. (2015). *Understanding Bitcoin: Cryptography, engineering, and economics.*: John Wiley & Sons Ltd.

Greenberg, A. (2013). Dark Wallet Aims To Be The Anarchist's Bitcoin App Of Choice. Retrieved from http://www.forbes.com/sites/andygreenberg/2013/10/31/darkwallet-aims-to-be-the-anarchists-bitcoin-app-of-choice/

Harding, D. A. (2015a). Bitcoin Developer Guide. Retrieved from https://bitcoin.org/en/developer-guide

Harding, D. A. (2015b). Locktime, nLockTime. Retrieved from https://bitcoin.org/en/glossary/locktime

Harding, D. A. (2015c). Sequence Number (Transactions). Retrieved from https://bitcoin.org/en/glossary/sequence-number

Hearn, M. (2013). Merge avoidance A note on privacy-enhancing techniques in the Bitcoin protocol. Retrieved from https://medium.com/@octskyward/merge-avoidance-7f95a386692f

Hill, K. (2013). Sanitizing Bitcoin: This Company Wants To Track 'Clean' Bitcoin Accounts. Retrieved from http://www.forbes.com/sites/kashmirhill/2013/11/13/sanitizing-bitcoin-coin-validation/

Johnson, D., Menezes, A., & Vanstone, S. (2001). The elliptic curve digital signature algorithm (ECDSA). *International Journal of Information Security, 1*(1), 36-63.

Kaminsky, D. (2011). Black Ops of TCP/IP. Retrieved from http://dankaminsky.com/2011/08/05/bo2k11/

Knowles, I. (2015). Re: Is this BIP65 sample script standard? Retrieved from https://bitcointalk.org/index.php?topic=1300723.msg13368835#msg13368835

Koshy, P., Koshy, D., & McDaniel, P. (2014). *An analysis of anonymity in bitcoin using p2p network traffic*: Springer.

Martin, J. (2014). Lost on the Silk Road: Online drug distribution and the 'cryptomarket'. *Criminology and Criminal Justice, 14*(3), 351-367.

Martin, P., & Taaki, A. (2013). Anonymous Bitcoin Transactions. Retrieved from https://sx.dyne.org/anontx/

Maxwell, G. (2011). Deterministic Wallets. Retrieved from https://bitcointalk.org/index.php?topic=19137.0

Maxwell, G. (2013a). CoinJoin: bitcoin privacy for the real world. Retrieved from https://bitcointalk.org/index.php?topic=279249.0

Maxwell, G. (2013b). CoinSwap: Transaction graph disjoint trustless trading. Retrieved from https://bitcointalk.org/index.php?topic=321228.0

Maxwell, G. (2013c). I taint rich. Retrieved from https://bitcointalk.org/index.php?topic=139581.0

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. *USENIX ;login:*.

Merkle, R. C. (1980). *Protocols for public key cryptosystems.* Paper presented at the 1980 Symposium on Security and Privacy, IEEE Computer Society.

Miers, I., Garman, C., Green, M., & Rubin, A. D. (2013). *Zerocoin: Anonymous distributed e-cash from bitcoin.* Paper presented at the Security and Privacy (SP), 2013 IEEE Symposium on.

Möser, M. (2013). *Anonymity of Bitcoin transactions.* Paper presented at the Münster Bitcoin Conference.

Moser, M., Böhme, R., & Breuker, D. (2013). *An inquiry into money laundering tools in the Bitcoin ecosystem.* Paper presented at the eCrime Researchers Summit (eCRS), 2013.

Nakamoto, S. (2008). *Bitcoin: A peer-to-peer electronic cash system*. Retrieved from http://bitcoin.org/bitcoin.pdf

Pfitzmann, A., & Hansen, M. (2010). A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management.

Piuk. (2012). What is taint? Retrieved from https://bitcointalk.org/index.php?topic=92416.msg1018943#msg1018943

Reid, F., & Harrigan, M. (2013). *An analysis of anonymity in the bitcoin system*: Springer.

Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM, 21*(2), 120-126.

Ron, D., & Shamir, A. (2013). Quantitative analysis of the full bitcoin transaction graph *Financial Cryptography and Data Security* (pp. 6-24): Springer.

Shamir, A. (1979). How to share a secret. *Communications of the ACM, 22*(11), 612-613.

Smyth, L. (2014). THE POLITICS OF BITCOIN. Retrieved from http://simulacrum.cc/2014/03/07/the-politics-of-bitcoin/

Steinfeld, R. (2016, 7 January 2016) *Secure Multiparty Computation/Interviewer: D. A. Wijaya*.

Stinson, D. R. (1995). *Cryptography Theory and Practice*: CRC Press.

Tiernan, N. (2013, May 7, 2013). Alt Chains and Atomic Transfers. Retrieved from https://bitcointalk.org/index.php?topic=193281.msg2224949#msg2224949

Tiernan, N. (2014, April 26, 2014). Hash Locked Transaction. Retrieved from https://github.com/TierNolan/bips/blob/bip4x/bip-0045.mediawiki

Todd, P. (2014). OP_CHECKLOCKTIMEVERIFY. Retrieved from https://github.com/bitcoin/bips/blob/master/bip-0065.mediawiki

US Department of Justice. (2013). Manhattan U.S. Attorney Announces Charges Against Liberty Reserve. Retrieved from http://www.justice.gov/usao-sdny/pr/manhattan-us-attorney-announces-charges-against-liberty-reserve-one-world-s-largest

Voskuil, E. (2014, 15 February 2015). About Bitcoin Explorer. Retrieved from https://github.com/libbitcoin/libbitcoin-explorer/wiki

Voskuil, E. (2015). About Bitcoin Server. Retrieved from https://github.com/libbitcoin/libbitcoin-server/wiki

Wijaya, D. A. (2015). Re: Is this BIP65 sample script standard? Retrieved from https://bitcointalk.org/index.php?topic=1300723.msg13408621#msg13408621

Wuille, P. (2012). Hierarchical Deterministic Wallets. Retrieved from https://github.com/bitcoin/bips/blob/master/bip-0032.mediawiki

xHire. (2015). Atomic protocol #1. Retrieved from http://www.coincer.org/2015/01/27/atomic-protocol-1/

Yang, E. Z. (2012). Secure multiparty Bitcoin anonymization. Retrieved from http://blog.ezyang.com/2012/07/secure-multiparty-bitcoin-anonymization/

Yao, A. C.-C. (1982). *Protocols for secure computations.* Paper presented at the FOCS.

Appendix 1 - Two Middlemen, 2-of-2 Multisignature, and CLTV

A script introduced by xHire (2015) which employs the concept of atomic transaction is shown in the script below. The Bitcoin Opcodes used in the script will be provided in Appendix 3.

```
OP_IF
    2 <PUBKEY PAYER> <PUBKEY PAYEE> 2 OP_CHECKMULTISIG
OP_ELSE
    <PUBKEY PAYEE> OP_CHECKSIGVERIFY OP_HASH160 <H(X)> OP_EQUAL
OP_ENDIF
```

The script above enables user to redeem the commit transaction in 2 ways. The first one is using 2-of-2 multisignature in redeem transaction while the second one is using a signature and a secret key. By using the script above, the payer and the payee can stop at any time without any party suffering any money loss. To redeem by using multisignature scheme, the user must construct the following ScriptSig.

```
0 <SIGNATURE PAYER> <SIGNATURE PAYEE> 1 <P2SH SCRIPT>
```

If the user wants to redeem the transaction by using a secret key X which hash match the value of H(X), then the ScriptSig is constructed in following form.

```
<X> <SIGNATURE PAYEE> 0 <P2SH SCRIPT>
```

Knowles (2015) suggested the usage of CheckLockTimeVerify (CLTV) in the atomic transaction as well as modifying the script as follows.

```
OP_DUP OP_SHA256 <HASH(X)> OP_EQUAL

OP_IF

   OP_DROP

   <PUBKEY PAYEE> OP_CHECKSIG

OP_ELSE

   <CLTV> OP_NOP2

   OP_DROP

   <PUBKEY PAYER> OP_CHECKSIG

OP_ENDIF
```

The construction above can be slightly modified to increase the security by first checking the validity of the public key resulting in the format below (Knowles, 2015).

```
OP_DUP OP_SHA256 <HASH> OP_EQUAL

OP_IF

 OP_DROP

 OP_DUP OP_HASH160 <PUBKEYHASH PAYEE> OP_EQUALVERIFY OP_CHECKSIG

OP_ELSE

 <CLTV> OP_NOP2 OP_DROP

 OP_DUP OP_HASH160 <PUBKEYHASH PAYER> OP_EQUALVERIFY OP_CHECKSIG

OP_ENDIF
```

To redeem the transaction by using the secret key, then the user can construct the redeem transaction in the following format.

```
<SIGNATURE PAYEE> <PUBLIC KEY PAYEE> <X> <P2SH SCRIPT>
```

While the payer can get refunded by constructing the redeem transaction in the following format.

```
<SIGNATURE PAYER> <PUBLIC KEY PAYER> <P2SH SCRIPT>
```

In order to extend the functionality of the script to support 2-of-2 multisignature, then the script can be constructed as follows as suggested by Wijaya (2015).

```
OP_IF

 OP_IF

 OP_DUP OP_SHA256 <HASH> OP_EQUAL

 OP_DROP

 OP_DUP OP_HASH160 <PUBKEYHASH PAYEE> OP_EQUALVERIFY OP_CHECKSIG

OP_ELSE

 <CLTV> OP_NOP2 OP_DROP

 OP_DUP OP_HASH160 <PUBKEYHASH PAYER> OP_EQUALVERIFY OP_CHECKSIG

OP_ENDIF

OP_ELSE
```

```
     2 <PUBKEY PAYER> <PUBKEY PAYEE> 2 OP_CHECKMULTISIG

    OP_ENDIF
```

The script can be redeemed by 3 different ways. The first one, if the payee wants to redeem the transaction, then the construction is as follows.

```
    <SIGNATURE PAYEE> <PUBLIC KEY PAYEE> <X> 1 1 <P2SH SCRIPT>
```

Secondly the 2-of-2 multisignature is preferred, then the construction is as follows.

```
    0 <SIGNATURE PAYER> <SIGNATURE PAYEE> 0 <P2SH SCRIPT>
```

The last option, if the payer wants to get refunded, then a transaction can be created by using the following construction.

```
    <SIGNATURE PAYER> <PUBLIC KEY PAYER> 0 1 <P2SH SCRIPT>
```

Appendix 2 - Five Middlemen and 2-of-3 Multisignature

The script provided below is used in the proposed protocol. This script is constructed based on 2-of-3 multisignature which comprises an escrow scheme. The CLTV is also used to lock the transaction until a future time. The explanation for each OpCodes is provided in Appendix 3.

```
OP_IF

2 <PUBKEY PAYER> <PUBKEY PAYEE> <PUBKEY ESCROW> 3 OP_CHECKMULTISIG

OP_ELSE

 <CLTV> OP_NOP2 OP_DROP

 OP_DUP OP_HASH160 <PUBKEYHASH PAYER> OP_EQUALVERIFY OP_CHECKSIG

OP_ENDIF
```

There are 3 possible methods of redeeming the script above. Firstly, by constructing a redeem script between the payer and the payee below.

```
0 <SIGNATURE PAYER> <SIGNATURE PAYEE> 1 <P2SH SCRIPT>
```

Secondly, if the payer cheats then an escrow can take over the role by constructing the redeem script in the following scheme.

```
0 <SIGNATURE PAYEE> <SIGNATURE ESCROW> 1 <P2SH SCRIPT>
```

Lastly, in case of the payer wants to refund the money because the transaction is can-celled, then the payer constructs the redeem script as follows.

```
<SIGNATURE PAYER> <PUBLIC KEY PAYER> 0 <P2SH SCRIPT>
```

Appendix 3 – Bitcoin Operation Codes

The Bitcoin Operation Codes (OpCodes) used in the proposed solutions are listed and explained as follows (Bitcoin Wiki, 2013).

No	Word	Opcode	Hex	Input	Output	Description
1	OP_0, OP_FALSE	0	0x00			Empty array is pushed to the stack
2	OP_PUSHDATA1	76	0x4c			The next byte is the number of bytes of data to be pushed to the stack
3	OP_1, OP_TRUE	81	0x51		1	Value 1 is pushed to the stack
4	OP_IF	97	0x63	1		If the top stack value is 1 then the script block is executed
5	OP_ELSE	103	0x67	0		If the top stack value is 0 then the script block is executed
6	OP_ENDIF	104	0x68			Ends the if-else block
7	OP_VERIFY	105	0x69	True / False		If the top stack is not true, then the transaction is invalid.
8	OP_DROP	117	0x75	x	nothing	Removes the top stack value.
9	OP_DUP	118	0x76	x	x x	Duplicates the top stack value.
10	OP_EQUAL	135	0x87	x y	True / False	If x = y then true else false.
11	OP_EQUALVERIFY	136	0x88	x y		Runs OP_EQUAL followed by OP_VERIFY.
12	OP_SHA256	168	0xa8	X	H(x)	Calculates SHA256 hash value.
13	OP_CHECKSIG	172	0xac	Signature Pubkey	True / False	Checks the signature based on the public key supplied and the transaction.
14	OP_CHECKSIGVERIFY	173	0xad	Signature Pubkey		Runs OP_CHECKSIG followed by OP_VERIFY.
15	OP_CHECKMULTISIG	174	0xae	0 Signature .. Signature	True / False	Checks multisignature validity against supplied script.
16	OP_CHECKLOCKTIMEVERIFY (OP_NOP2)	177	0xb1	Block length		Checks the block length against the current block length.

The Bitcoin OpCodes must be constructed into a script. The script is then evaluated within the Bitcoin system and it must produce a “True” for a given input. Only by producing a “True” output, then the fund can be spent. If the script cannot produce “True”, then the fund could not be spent.