2014

Graph-based forensic investigation of Bitcoin transactions

Chen Zhao
Iowa State University

Graph-based forensic investigation of Bitcoin transactions

by

Chen Zhao

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

Master of Science
Major: Computer Engineering

Iowa State University

Ames, Iowa

2014

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

I would like to thank my committee chair, Dr. Yong G. and my committee members, Ahmed E. Kamal and Manimaran Govindarasu for their guidance and support throughout the course of this research.

ABSTRACT

This thesis illustrates forensic research work on Bitcoin, an innovative Internet based global transaction system that attracts ascending popularity during the recent few years. As an open, public and scalable distributed payment system, Bitcoin brings forward significant economic and technological impact to our world. Meanwhile, a new notion of virtual currency, “Bitcoin” comes into existence such that Bitcoin currency can be “mined” from all over world complying with specific algorithms. Mined bit “coins” has practical monetary values that turn the Bitcoin system into a digital currency circulation system. Due to Bitcoin’s decentralized semantics, Bitcoin transaction and currency are not subject to control and censorship from any single authority. Therefore, Bitcoin brings out various security concerns about its application as a long-term reliable system.

The research in the thesis focuses on forensic study on Bitcoin. It covers experimental study on the Bitcoin network as a peer-to-peer system and a graph-based forensic approach against Bitcoin’s transaction data. Major contributions include network data evaluation and transaction history analysis. In case of forensic investigation is needed against criminal incidents such as fraud, false transactions and money theft, which are commonly seen in commonly used digital payment systems, the research provides a guidance of efficient information collection and framework of evidence data processing and extraction.

CHAPTER 1
INTRODUCTION

1.1 Overview – what is cryptographic currency and Bitcoin?

Crypto currency (cryptographic currency) is a brand new concept that came into existence during recent years. It is a new type of digital currency capable of real world transaction activities like any other medium of exchange such as paper money in our everyday life. Unlike other types of digital money people are familiar with such as the PayPal balance and Online banking accounts, crypto currency runs as a decentralized transaction system with no authorized central management. For example, central servers from a specific commercial bank maintain an online banking system. The bank is responsible for all its customers' account statements and transaction authenticity. Servers monitoring transaction behaviors would protect the database from malicious activities and network attacks. However, crypto currency is not subject to any such central servers as all the participants form a peer-to-peer (P2P) system without censorship. The security of such a system actually relies, as the name shows, on cryptographic algorithms instead of any man-made surveillance system.

The idea of crypto currency systems first came out in the year of 2009 [1]. Bitcoin is the earliest crypto currency raised by the well-known Bitcoin whitepaper [2] with the pseudo name Satoshi Nakamoto as its author in 2009. The Bitcoin client software was soon practically implemented so that a P2P payment system was gradually formed as suggested in the whitepaper. Bitcoin has been the most popular and widely deployed crypto currency in the world after several years' practical usage and development since 2009. The have been other crypto currencies with very similar mechanism as Bitcoin after then, such as Litecoin, Namecoin, Altcoin, etc. But still Bitcoin is the most successful one in terms of monetary value, daily transaction amount and total number of users in the world. As a stereotype of crypto currency system, Bitcoin is worthy of intensive study both as a functioning distributed system and a secured transaction network. This is why the thesis is focused on the Bitcoin network data propagation and how this data affects system security.

It is impossible to give accurate measure on how many Bitcoin users there are in the world due to the nature of P2P systems. Early in September 2011 there was estimation claiming that there were only 60,000 users at that point [3]. We believe this estimation is far from the true number because the estimation is not technically reliable, which is to be discussed in detail in the thesis. Moreover, the fact that Bitcoin users have been exponentially increasing during the past few years makes it even more difficult to estimate the user number in the whole system that covers the Internet. But there is other evidence showing the rapid exponential growth of Bitcoin users. As one of the Bitcoin wallet service website, *Blockchain.info* had 250,000 user wallets in June 2013 and this number went up to 1,500,000 in May 2014 [4]. The number of user wallets is the number of Bitcoin users that benefit from the service of this website alone. There are a lot of other Bitcoin wallet service websites nowadays. For the rest of Bitcoin users who use the Bitcoin client they do not need any web wallet service because the software running on their own computers is responsible of fulfilling all the necessary computational processes.

The content organization for the rest of the thesis follows such a guideline that it first introduces technical background related to Bitcoin including cryptographic premises, digital signatures and data structures used. Next it comes to a brief description on how Bitcoin technically works and gives a high level view of the communication protocols between Bitcoin clients. Then it comes to objectives of the thesis consisting of two major parts of work. The first part is a set of in-depth experimental study on Bitcoin transaction propagation and network evaluations. The second part is a systematic approach of forensic investigation on Bitcoin block chain analysis and tracking currency flow. Finally comes the conclusion.

1.2 Cryptography premises

Despite that truth that the Bitcoin system is a cryptographic algorithm based transaction system, Bitcoin does not take care of data encryption. This means that in the Bitcoin protocol nothing is encrypted even though the currently applied Bitcoin client utilizes OpenSSL for data transmission but it is never part of the protocol design. Bitcoin benefits from two major applications of cryptographic, Hashing and Public Key Infrastructure. They are used in both transaction authentication and currency control, which form the basis of availability and security of the system. In the following context we introduce the concepts of crypto hash functions, public key infrastructure and digital signature one after the next.

Hashing

The hash function used in Bitcoin is the well-known hash function SHA-256 from the Secure Hash Algorithm family SHA-2 designed by the U.S. National Security Agency (NSA) in 2001. SHA-2 is also defined as the Secure Hash Standard in Federal Information

Processing Standards Publication (FIPS PUB 180-4) [5] for National Institution of Standards and Technology (NIST) in March 2012.

SHA-256 is a widely deployed crypto hash functions right now in many security systems and encryption applications. SHA-256 function takes any size of string or number as its input, digests the input message and manipulates data to output a fixed size sequence. The SHA-256 is always a 256-bit binary value.

SHA-256 has the basic properties that any other crypto hash functions are supposed to hold. First, SHA-256 calculation should be fast in computation complexity. Second, given a specific 512-bit SHA-256 output, it is computationally infeasible to calculate the input, which means SHA-256 is a strict one-way function. Moreover, it is very difficult to find two different inputs that generate the same hash output. These hash properties are concluded as the following list.

- One-way function
  - Let s be the hash input (image) and function H, it is fast to compute y=H(s)
- Pre-image Resistance
  - Given a hash y=H(s) only, it is very hard to compute image s
- Second Pre-image Resistance
  - Given an image s, it is difficult to find s’ so that H(s)=H(s’)
- Collision Resistance
  - It should be hard to find any two images S_1 and S_2 So that $H(S_1)=H(S_2)$

Example in Figure 1 illustrates the basic logics of SHA-256 function. Detailed SHA-256 algorithm is not covered as it is off the research focus for the thesis.

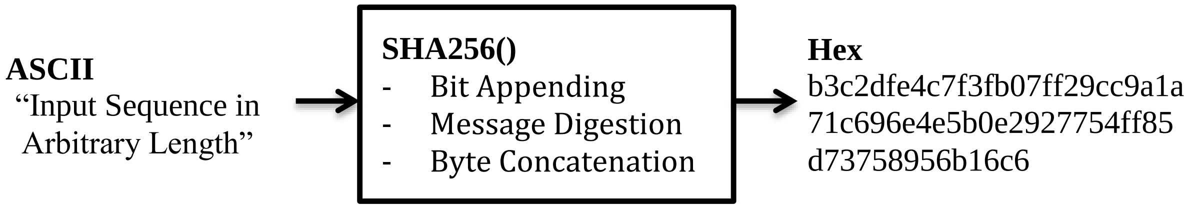


Figure 1-1

There is one other hash function called RIPEMD-160 used in Bitcoin implementation. RIPEMD-160 is a very old hash functions first published in 1996 [6]. It is used in Bitcoin system to generate shorter 160-bit outputs than SHA-256 hashes. RIPEMD-160 is rarely called other than used to generate part of Bitcoin public addresses. Therefore RIPEMD-160 does not account for much importance in the Bitcoin system even though there already exist publicly known vulnerabilities in RIPEMD-160.

Public key infrastructure

Public-key encryption scheme, i.e. asymmetric cryptography is one of the most brilliant contributions in modern cryptography and more or less applied in vast majority of security-featured communication systems. The Bitcoin system adopts asymmetric cryptography for the sole purpose of creating digital signatures rather than data encryption. Instead of nowadays the most typical public key encryption algorithm RSA scheme, Elliptic Curve Digital Signature Algorithm (ECDSA) is used in Bitcoin. ECDSA is a public key algorithm developed for NIST by the Accredited Standards Committee on Financial Services, X9 [7]. It is defined in FIPS PUB 186-3 Digital Signature Standard by

NIST as part of standardized digital signature algorithm. The following section briefly describes how ECDSA is used.

To generate an ECDSA signature, a set of domain parameters has to be predefined, as ECDSA is an elliptic curve based algorithm. Elliptic curve cryptography is about algorithms based on elliptic curves over finite fields and it is secured by intractability of related mathematic problems. Before running the ECDSA algorithm, parameters about the elliptic curve to use should be set as consensus between the signature sender and receiver. These include the elliptic curve's equation that consists of two coefficients, the finite field size, a base point G on the curve and the integer order of n. To sign an ECDSA signature, one needs to generate a secret private key d which is an integer. A public key Q corresponding to d is calculated using Q=d×G which expresses point multiplication operation on an elliptic curve. Here the resulting Q is another point. The signature is generated from the curve and private key d and is a pair of scalars (r, s). To verify the signature, the receiver needs the signature (r, s), the elliptic curve G and the signer's public key Q. Only the owner of private key d is able to generate a satisfactory signature that can be verified using Q. It is infeasible to brute force anyone's private key from Q if d is always kept secret. Figure 1.2 illustrates how ECDSA is used in practice.

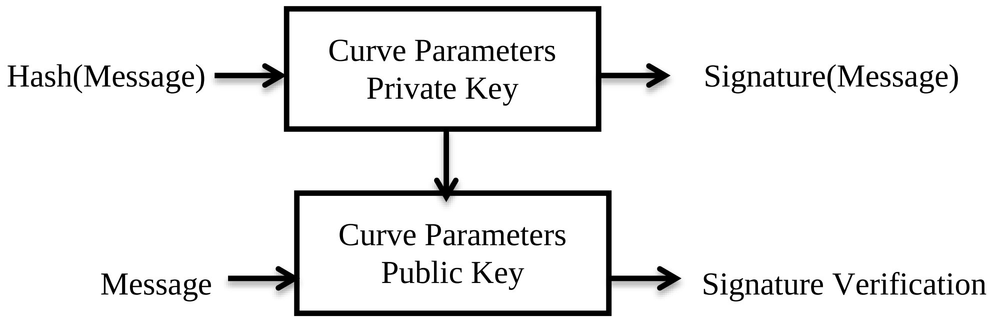


Figure 1-2

The hash function used in Bitcoin is SHA-256 for ECDSA signature generation and the choice of hash function here is not part of ECDSA standard. Mathematical details inside ECDSA signature generating and verification algorithm are not covered in the thesis.

1.3 How Bitcoin works

At present, the Bitcoin system consists of millions of Bitcoin client hosts all around the Internet. These users clients form the vast majority of the Bitcoin network and are treated as equivalent. As a purely decentralized network, the Bitcoin infrastructure has essential difference from other most commonly used P2P networks such as P2P based content delivery network (eDonkey, Bit torrent, etc.) for file sharing and P2P streaming protocols (P2P TV, P2P radio, etc.) for audio and video communication. This section gives a description on how Bitcoin as a distributed system works in network communication aspect that might differ from most online Bitcoin documentations.

From the user side the Bitcoin is extremely simple to use. Downloading the Bitcoin client software is all that is needed. However, to realize a practical and universal transaction system, the Bitcoin needs to satisfy specific properties and well tackles several major problems.

- Scalability

  The Bitcoin network allows for perfect scalability. It accommodates any number of user clients at any time. It is always free for new nodes to join the network and those in the network to exit whenever and as frequently as they want without affecting any transaction processes.

- Integrity, Malleability and Authenticity

As the basic functional requirement for any valid transaction system, a user transaction has to be verified of its authenticity once it is made. It is never possible for anyone authorized to issue transactions for others into the Bitcoin network. Also transaction forgery and malicious modification against legal transactions are avoided. All Bitcoin transaction records are guarded against Malleability.

- Immunity against Replay Attacks

In traditional digital transaction systems there are always carefully designed countermeasures against replay attacks. For example, the same network request of paying 100$ from one's PayPal account cannot be sent twice to make another payment of 100$ from that account, in case the request is intercepted by an attacker. In Bitcoin, no central security server is available to trace replay attacks but such attacks are well prevented by protocol logics. We always use the term "Double Spending" for acts similar to replay attacks in Bitcoin context.

- System Availability and Performance

Every time we mention availability of a P2P network, we mostly refer to its performance and its resistance to possible DoS attacks. The Bitcoin system right now has pretty high network response and full availability that is satisfactory in capacity for the current user basis. Bitcoin client software has been updated all the time and is already carefully coded to guard against almost all the possible DoS attacks people are now thinking about, even though new types of DoS attacks will attempt to bring down the system availability in future when new updates will be required for the program.

The technique

The essential idea of Bitcoin is built upon the concept of the block chain. The block chain can be understood as the unique database that consists of all the transaction records from the birth of Bitcoin up till now. To participate into transaction activities, all the nodes are mandatorily required to synchronize with the network, which means that everyone needs to download and maintain the latest version of transaction record database, the "block chain", to get themselves notified on what has been going on. For new coming users who do not have any transaction data or those who do not hold the latest transaction data due to absence from the network for a long time, they have to get the block chain copy from other peers who already have it. When each new transaction is generated, this transaction has to be broadcast onto the global Bitcoin network to inform as many others as possible so that the transaction is confirmed and accepted by the network if it is valid. Failure completing this job makes this transaction expire.

To more clearly illustrate the technique, let us look at a simplified and idealized scenario. The Bitcoin network is analogous to a large group people involved in real-time transactions. Every time one issues a valid transaction, he/she has to "yell out" that payment for the whole group to hear about it. Suppose most group members will hear and record this transaction to update their block chain. For those who did not hear this payment they periodically ask people nearby to quickly query transaction updates. The purpose of broadcasting is to make sure everyone has a complete history record from the beginning so that they are able to verify if it is legitimate for someone to issue certain transaction.

1.4 Incentives of Bitcoin

The rise of Bitcoin and its continuing popularity come with various reasons. The major advantage is that Bitcoin is far more than private money and mere medium of exchange. Bitcoin is a self-supplying financial system with much more flexibility than traditional online payment systems.

For end-users primary reasons of using Bitcoin in everything life include both security and cost considerations. The most attractive point of Bitcoin is that its money is issued by coin miners around the world and is not subject to central authorized management. The money of Bitcoin is more like gold than any government issued paper currency due to its distributed nature so that a person's coins can never be forfeit or voided unless one's Bitcoin wallet data is no longer kept secure. The second reason is Bitcoin's decent transaction anonymity. Bitcoin has privacy-oriented nature and transaction history alone does not reveal user identity despite that all the transactions are broadcast onto the Internet. Pure Bitcoin activities never bond real-world financial authentication such as bank accounts. Other than the above, the average transaction fee of Bitcoin is lower than traditional online approaches like bank transfer or PayPal because it is an adjustable tip paid to Bitcoin miners around the world. How transaction fee is handled will be technically illustrated in later chapters.

Merchants can also benefit from accepting Bitcoin for goods. First Bitcoin transactions are irreversible after commitment, compared with credit card or PayPal payments that can be charged back even after a long time of completing a transaction. Next making Bitcoin payment is a user friendly and easy process without exchange rate between currencies of nations, because Bitcoin is globally circulated and unified. This potentially increases selling and net profit.

Lastly, Bitcoin currency attracts real capital as Bitcoin potentially makes fortune if properly invested. As Bitcoin price is mostly determined by the rate of coin mining and its popularity, its value fluctuates at times. Other than direct coin mining, Bitcoin selling is an essential approach for profits for many Bitcoin investors.

CHAPTER 2
A HIGH-LEVEL VIEW OF BITCOIN SYSTEM AND PROTOCOLS THEREOF

This chapter covers a high level view on the Bitcoin transaction protocol without discussing too much low level details, whereas two major parts are introduced, the block chain data structure and transaction data processing. The majority of the chapter attempts to briefly summarize the fundamental design pattern first brought forward by the famous Bitcoin whitepaper [8] and illustrate how such design cryptographically achieves most of the functional requirements specified in Section 1.3. Then this chapter also talks about how privacy is protected through anonymity of Bitcoin, or actually more strictly speaking, pseudo-anonymity.

The simplified broadcasting scheme is almost impossible to realize in typical distributed systems and even if were it would not address any security-related problems either. Therefore special data structures and crypto logics have to be adopted.

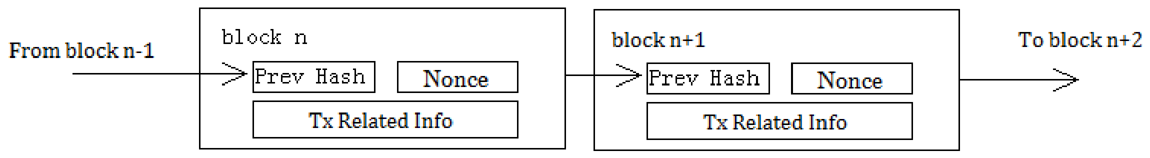


Figure 2-1 Block Chain Conceptual Diagram

Figure 2-1 is a diagram about a block chain episode. The block chain is a long tree-based data structure that consists of large sequence of blocks. The figure shows two consecutive blocks on the chain with sequence number n and n+1. A real Bitcoin block contains tons of details about the block origination, transaction information and lots of other trivial data fields. For the interest of high-level logics, we focus on the several fields drawn in Figure 2-1 at this moment.

The “Prev Hash” field in every block is a hash value of the previous block. Here the hash refers to twice SHA-256 applied on the previous block. The doubly applied output is stored as a fixed-size field in the current as the case in Figure 2-1. The “Tx Related Info” is a description on transactions in the current block but it is NOT simply a plain record of transactions.

2.1 Bitcoin mining and proof-of-work

Bitcoin mining is perhaps the most popular term in Bitcoin technology and economics community. Bitcoin mining is the process of generating a block on the basis of the current block chain. Suppose the current block chain has length of n, meaning that the latest block is block n. Suppose someone intends to generate a new block based upon block n, he/she takes block n and calculates the hash (always double SHA-256) of block n. The hash is stored as the “Prev Hash” field in the block to be mined (block n+1). To mine is to determine the nonce value in block n+1, the miner has to brute force attempt as many possible nonce values as possible until such a nonce is fetched that makes the hash of block n+1 below specific threshold. Once the miner succeeds, he/she has to broadcast the newly generated block n+1. The rest of the network verifies the satisfactory hash value, accepts it and update the block chain. As SHA-256 is a well secure hash function, it is difficult to tell what the nonce value it is to hash the block below the threshold. Therefore mining is always an extremely computation intensive brute force cracking process to reverse the double SHA-256 algorithm. There is no shortcut unless SHA-256 is no longer secure. This ordered block mining logic is in Bitcoin terminology called “Proof-of-Work”. Next we answer questions how to put such logic into practical network.

- What is the motivation of Bitcoin mining

Mining blocks is mining Bitcoin, thus mining money. As part of Bitcoin essential protocol, a miner gets specific amount of "coins" as reward for each block he/she mines. The reward is implemented in the way that a miner is allowed to write the reward for himself/herself to claim as transaction information in the block mined. This means every block has the identification information on who the miner is and we usually call it "block owner".

- Controlled Currency Supply

The threshold that restricts the block hash outputs is called "difficulty" of Bitcoin mining. It is not a constant. Bitcoin protocol requires that mining difficulty increase after every 2016 blocks. We call the block hash threshold "target" in Bitcoin context. To make the difficulty increase, the target is lowered so as to force miners to calculate smaller outputs as time passes. However, in rare cases the difficulty can be reduced and the change depends on the time it takes to find the recent 2016 block. There are currently more than 300,000 blocks in the block chain and the average time interval between any two consecutive blocks generation is 9.35 minutes [9] based on Blockchain Stats. As each block keeps a time stamp on its generation, anyone who possesses the current block chain should be able to compute it. It is a global indication on how much effort miners in this world investing in mining. Therefore 2016 blocks roughly takes 2 weeks to spawn, with random deviation. If the time to get the last 2016 blocks is over 2 weeks, which is mostly the practical case right now, the difficulty is raised; otherwise reduced, which is rarely seen up till today.

Apart from the difficulty change, the mining reward gets halved every 210,000 blocks. The Bitcoin protocol initially rewards each miner 50 BTC (unit for amount of one "coin") and it is now 25 BTC per block, as it is in the second 210,000-block phase. That is, from the 420,001th block on, mining reward goes to 12.5 BTC and so on. Obviously this geometric decreasing pattern makes the reward zero at the end so that total number of possible Bitcoin currency amount converges at 21 million BTC [10]. This value can either be referenced from online documents or very quickly verified using geometric sequence summing equations.

Bitcoin, from a non-technical point of view, is not only a transaction agency, but also a sophisticated economic system. The purpose of both difficulty increasing and reward reduction comes from the demand of controlled currency supply, which is fundamental requirement for any economic system to healthily function in the long run. Without controlled supply, a currency suffers rapid inflation that eventually brings down the ecology of coin circulation.

- How to deal with chain forks

"Chain fork" is a severe problem and probably the most challenging security concern in the Bitcoin system because it directly enables the possibility of double spending attacks. Chain forks result from either in-purpose attacks or natural reasons

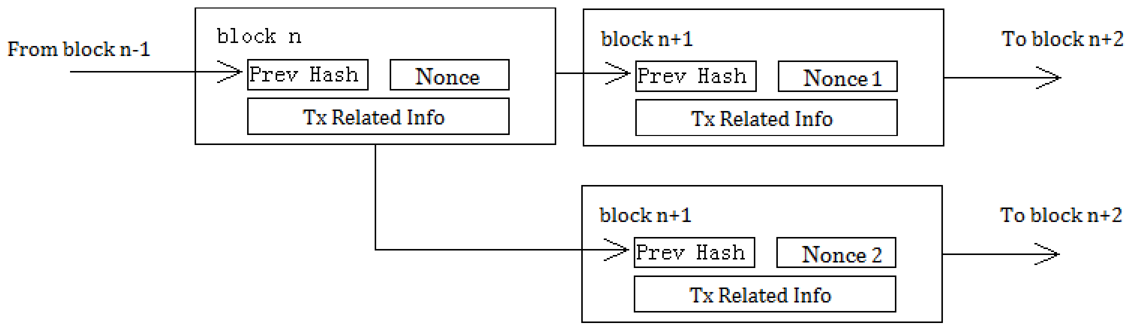


Figure 2-2 Chain Fork Example

such presence of long network latency. A chain fork is defined as when a block has two or more valid successor blocks with valid nonces. The multiple successors can hash to complete different outputs but all are below the specific current "target".

Figure 2-2 is the forking case extending from Figure 2-1. Note that in this diagram both Nonce 1 and Nonce 2 are valid and both blocks should be accepted into the network according to the protocol.

There is previous work of dedicated analysis on how chain forks quantitatively emerge depending on how much network latency there is [11]. Observation in the work showed that 169 out of 10000 blocks between block height 180,000 and 190,000 are forked; their theoretical model calculates the forking probability of a block at 1.78% close to the observed value of 1.69%. This is not a negligible probability so the block chain is required to well accommodate forking. The proof-of-work protocol further demands that only the longest fork of chain survive. If there are multiple miners mining from different block chains, the growing rate of each block chains purely depends on the computing power devoted on each chain. The author of the original Bitcoin whitepaper pointed out that one chain will soon wins out against the others and once other mines find it out, they will all switch to the longest and fastest growing chain, in absence of double spending attack, because mining the longest chain ensures the highest rate of reward and best efficient usage of computing power for honest miners.

The motivation of complying with the longest chain does not always apply with double spending attackers. The poof-of-work is designed with natural expectation on attackers deliberately forking the block chain. As the most fundamental stereotype of double spending, an attacker mines on alternative chain forks other than the longest one that contains transaction he made. If he manages to win against the valid chain by generating a longer fork chain than the current, he reverts the payment he sent before.

Typically such brute force forking requires immense computing power beyond imagination and practice. Satoshi Nakamoto proposes that as long as the majority of miners all over the Internet are honest miners, attackers can never succeed in competing with the rest of the world unless he were to be able to dominate the world's computing resource more than 50%, which is the very reason such double spending attack is termed 51% attacks in Bitcoin context.

There is a lot of previous research focused on feasibility of 51% attacks while all are about purely theoretical analysis model. The paper studying hash-rate based attacks (equivalent to brute force attack) [12] creates a negative binomial model to describe the generation of each new block with bias to favor either the attacker or the honesty network. Theory reveals that reverting more than 10 blocks on the valid chains is close impossible even by assuming the attacker possesses an improbable 10% of the network's computing power.

Real Bitcoin network has major deviation in the sense of block generating processes for two reasons. First SHA-256 does not give truly uniformly random outputs so the probability model is hard to precisely depict. Second computing power on the network is an abstract concept that cannot be statistically categorized. For example, a million simultaneous miners do not make equivalent contribution as one million times a single miner makes. So it is not practically accurate to say an attacker holding 51% computing resource ensures double spending success. But 51% has been used in terminology and we abide by the convention.

- Whether chain forks affect user transactions

It is a pretty common user-concerning question whether user transactions behavior is restricted in the presence of chain forks. The answer is definite not. A user transaction will not be a problem so long as this transaction is written into block on all the currently existing chains. The next section covers transaction processes and further illustrates this point.

2.2 Transaction generation and verification

The block chain manipulates everything about currency generation and takes on the functionality of transaction database shared by all nodes running the Bitcoin client. The second portion of Bitcoin protocol's capstone design is transaction circulation scheme. Despite the intuition people might have from what the name "Bitcoin" implies, there is no reified representation on "coins" in the block chain. It also means Bitcoin never records user's account balance as a typical online banking system would do. Currency circulation is purely controlled by transaction histories. The Bitcoin software verifies user balance by scanning relevant user transaction records available on the block chain and calculating the net transaction output. Below is explanation how transactions are logically linked with each other.

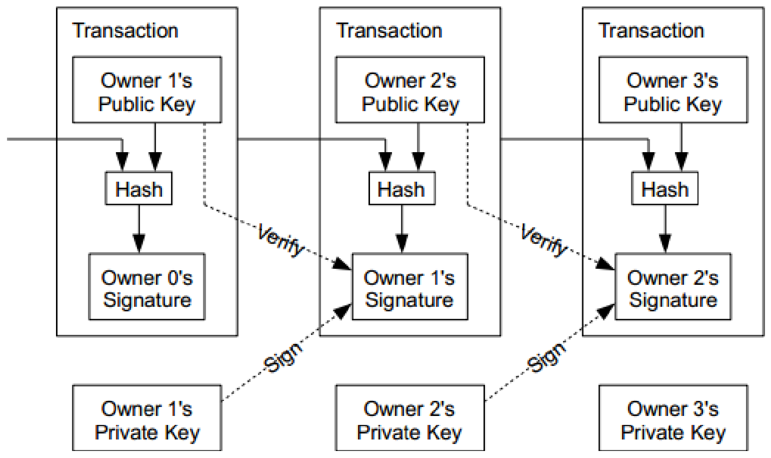


Figure 2-3 Transaction Logic from Satoshi's Whitepaper

Figure 2-3 is the same figure from the original whitepaper that designs transaction generation and verification process using digital signature. Every time one spends Bitcoin, he/she has to take previous transactions which are payment from others as the input, include the public key belonging to the payee and signs this transaction with his/her private key. The payee is able to quickly verify if that payment is made to the correct receiver with the payer's public key. In Figure 2-3, Owner 2 is paying Bitcoin to Owner 3 so he takes the transaction paid to him from Owner 1, includes Owner 3's public key and signs the transaction. Owner 3 checks if the transaction has the correct signature and if it does it means Owner 2 has made the payment. Owner 3 can use this payment as input paid to others later on.

Figure 2-3 is merely a conceptual diagram and practical Bitcoin transaction implementation differs a lot and is far more complicated that what is covered above. As two most functional features of currency, payment has to be dividable and change has to be supported like real money. Real-life Bitcoin realizes payment division by allowing multiple transaction inputs; and change by multiple transaction outputs in a single transaction.

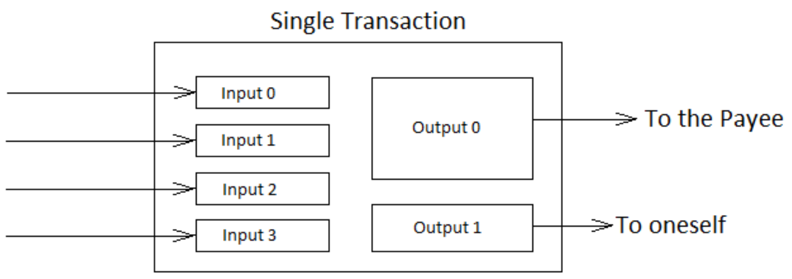


Figure 2-4 Simple Transaction Structure

Figure 2-4 is what Bitcoin typically does to craft a specific transaction. A user may combine multiple smaller transactions to send a larger amount of money. In case no satisfactory combination can be constructed from previous transactions, for example, in case sum of the 4 inputs exceeds the desire amount of payment, the user sets up multiple outputs. Here output 0 is signed as a payment to the payee and output 1 is signed as payment to himself. Consequently output 1 is the change to the user during this single transaction.

The digital signature based transaction scheme is resistant to transaction forgery and money theft, ensuring transaction integrity and authentication. Also the ECDSA signature used in Bitcoin is has nice cryptographic property against malleability.

2.3 Transaction confirmation and genesis block

Section 2.1 illustrates that the Bitcoin protocol maintains the block chain to allow for large-scale data synchronization and high scalability for Bitcoin as a distributed system. Section 2.2 summarizes the transaction circulation logic raised in the original Bitcoin whitepaper and shows that Bitcoin as a transaction system has its integrity and authentication scheme achieved by digital signature. Low level details into the block chain and transactions are covered in the next chapter dipping into system performance and hazard mitigation. This section covers two problems derived from the Bitcoin protocol discussed so far, the concept of confirmation and genesis block.

Transaction Confirmation

Transaction Confirmation is a precisely defined term in Bitcoin.

Definition *Bitcoin transaction is said to be confirmed if a transaction has been written onto the major block chain.*

It is mentioned in Section 1.3 that every single transaction is broadcast for the network to accept. A Bitcoin client user issues a transaction and waits for the transaction to be confirmed, or to appear in the block chain. It is not specified by the protocol how the transaction is written onto the block chain. Right now, Bitcoin miners around the world handle transaction confirmation as part of their jobs. Every time miners succeed finding a new block, they include the transactions they hear about from the network into the new block and broadcast this block. The first transaction in newly mined block is always the transaction that rewards the miner, as mentioned in Section 1.3. This transaction is called generation transaction and it simply stands for creation of currency instead of payment received from others.

Definition *Bitcoin transaction confirmation number of a transaction is the inclusive count of blocks starting from the block this transaction is in.*

The purpose to introduce the definition of confirmation number is for practical transaction safety. Section 2.1 demonstrates the occurrence of chain forks and how it relates to double spending attack. The fact that a transaction is confirmed does not guarantee that the payee safely receives the payment. A confirmed transaction can be reverted by blocks on alternative chain forks. The common practice against such hazard right now is to wait for certain confirmation number on a transaction. Meni Rosenfeld's [13] negative binomial model shows that reverting a block is probabilistically impractical when the block has 5 or more blocks appended after it. Even with brute force attack, it is highly unlikely to revert 6 blocks on the main chain against the honest network.

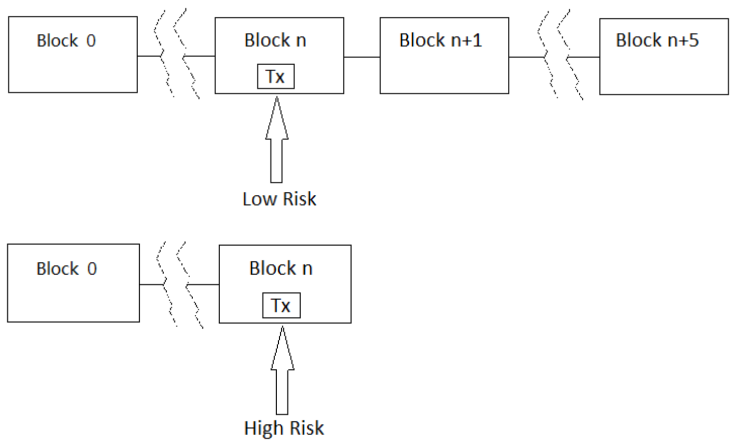


Figure 2-5 Chain Fork vs. Risk

Figure 2-5 describes two different cases for the same transaction. The risk in gets block n much lower due to the existence of block n+1 to n+5 because the attacker has to regenerate 6 alternative block n to block n+5 to revert the main chain.

In case of naturally occurring chain forks, the payee should be aware of transaction confirmation on both chain forks because it is ready to tell which fork

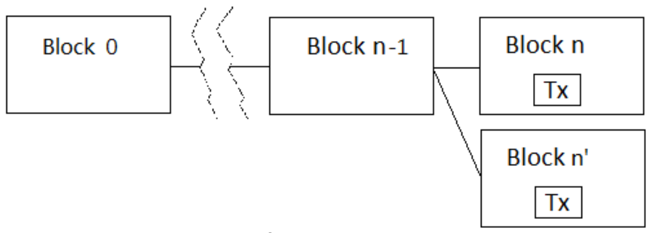


Figure 2-6 Confirmations on Forks

eventually wins out in a short period of time. As in Figure 2-5, the transaction Tx is expected to get confirmed in both Block n and Block n’. In this way the payee would get the payment regardless of which of the two forks wins out in the end. Otherwise, Tx should not be considered a valid payment.

Genesis Block

Genesis block is the first block (Block 0) of the block chain. There is only one genesis block on a block chain thus for each specific crypto currency. The genesis block is set up solely to start up Bitcoin and mining. It has different data format than all the subsequent blocks. The generation transaction in the genesis block is not spendable. The Unix time stamp field in this block has the value 1231006505 standing for the date **Sat, 03 Jan 2009 18:15:05 GMT** which is the time when Bitcoin first came out and practically started to function.

CHAPTER 3
OBJECTIVES

The objectives of the thesis are experiment findings and answers to a set of different questions that can be helpful to Bitcoin forensic techniques and related research areas.

3.1 Practical motivation

It has been several years since Bitcoin first came out in 2009. Although it is well functioning as a valid and reliable transaction system there have still been several issues about Bitcoin as money. The first problem is money theft and fraudulent transactions. Due to Bitcoin's irreversible nature, refunding is not possible after a transaction without the payee's approval. In case of money theft or loss it is most probable that the owner does not any knowledge of what happened to the coin, because users are only identified by anonymous Bitcoin addresses. Distinguishing raw transactions is often difficult. The most influencing Bitcoin theft incident just occurred this year where the once largest Bitcoin market provider Mt. Gox sustained a huge loss and filed bankruptcy. Conspiracy of the case is still undergoing police investigation and no conclusive answers are given yet.

The second problem is illegal activities involving Bitcoin. Bitcoin is an ideal currency for black market transactions of illegitimate goods like drugs. It is not feasible to answer what a specific Bitcoin transaction is used for and who made that transaction.

As a most well-known Bitcoin drug market, Silk Road [14] was first launched in 2011 and soon gained nearly one million of registered users accounts, which was far below the number of actual users. Approximately 10 million Bitcoins (BTC) circulated through Silk Road before it was seized by FBI in September, 2013 but it was re-launched two months later [15].

To help with forensic work against illegal Bitcoin actions, the thesis researches a framework of evidence retrieving and analysis, in case criminal incidents occur and demand that investigators need to know what are transaction relations between different Bitcoin users or user groups and where the coins ended up in.

3.2 Research contents

The thesis first carries out empirical study on Bitcoin transaction synchronization problems. A series of experiments were set up to discover impacting factors that affect how transactions are delivered and synchronized onto the Bitcoin network. Throughout these real network based experiments, the following questions are answered.

- In what way are Bitcoin transactions relayed by various nodes and propagated through the network?
- How does the Bitcoin network ensure its efficiency and accuracy in broadcasting information to all its participants?
- Whether the Bitcoin system is subject to particular Network topology, so that not all participants have a fair chance to hear transaction information and have their transactions confirmed?
- What role does transaction fee play in practical Bitcoin usage? Does transaction fee always help a transaction get confirmed sooner?

The second area of research is focused on forensics. Bitcoin as a real-world currency circulation system has been involved into various criminal cases during the last few years. One much demanding technique to investigate Bitcoin incidents is to track the currency flow. Due to anonymity property of Bitcoin addresses, it is difficult to determine where a specific sum of money goes. This thesis creates a data model to rearrange Bitcoin addresses and utilize a graph based algorithm to track money flow. Briefly, it answers the following questions.

- Given the block chain data, how do we extract transactions and determine what addresses should belong to the same Bitcoin entity?
- How to turn the block chain data into graphs that reflect single transactions between different Bitcoin entities?
- What algorithm can be used to traverse the graph so that we discover where a money flow goes

The next two chapters of the thesis covers each of the above research areas and all these questions will be discussed in detail or by experiment outcomes.

CHAPTER 4
GUIDANCE ON BITCOIN EVIDENCE COLLECTION

Bitcoin has been evolving into a vast peer-to-peer network system with magnificent amount of traffic going on and rapidly increasing computing resource invested. As an ever fast growing distributed system the Bitcoin network behavior is actually far more complicated than it was assumed to be. This chapter studies network topology and show empirical results about how information propagation in Bitcoin system can possibly be affected subject to current protocols and many programming level details. The significance of empirical study comes from the technical aspect that how information is delivered over the network in many ways affects end-user security and the whole system's resilience to malicious activities, as Bitcoin does not purely consists of end-user clients and not all participants are considered equivalent despite Bitcoin's completely decentralized semantics when it was initially devised in 2009. Experiment outcomes in this chapter reveal some insights that information synchronization is subject to specific Bitcoin nodes on the network. Characteristics found can provide some guidance to efficient data extraction in need of detailed forensic investigation.

4.1 Principles of experiments

Bitcoin is a large-scale transaction system based on peer-to-peer (p2p) network and cryptographic algorithm. Current Bitcoin system is an open and practically working economy system that redefines the flow of digital currency and involves increasing amount of real world transactions every day. Apart from basic functional requirements on any typical transaction system such as identity authentication and transaction integrity, Bitcoin network working as a decentralized distributed system is responsible for processing all transaction related computations under various constraints. Due to the nature of Bitcoin protocol that was initially devised in 2009 [16] it is required for every participant in the network first to get their database well synchronized to the network in order to carry out any subsequence activities. This imposes performance requirement that achievable synchronization and efficient multicasting is expected from the network. This paper illustrates sets of experiments observing how different nodes in the Bitcoin system get organized and in what way information propagation works. Experimental Analysis based upon the current Bitcoin client would be sequentially described followed by discussion that leads to possible security concern. We also make hypothesis from those empirical results so as to consider the feasibility of using Bitcoin as a reliable transaction system and a scalable distributed system in both short run and long run sense.

The major motivation to focus the study particularly on network synchronization comes from Bitcoin's security reliance on data consistency at each node. There is previous analysis by Christian Decker and Roger Wattenhofer [17] depicting a model that showed network inconsistency caused by network failure or longer latency increases potential vulnerability to malicious activities. One other reason is that information propagation is pretty sensitive to protocols adopted by every single participant in p2p systems. Ideally such a synchronized system exhibits perfect behavior whenever a new message comes out from arbitrary source node. The new message is expected to reach every other node in participation in a negligible time interval in analogy to global broadcasting. Practically, complicated Internet environment makes the whole system far from perfect consistency. Therefore it is a worthy-of-discussion topic to study how Bitcoin actually sustains current

Internet environment and whether experimental results in this paper indicate any existing or future tendency in terms of end user financial security.

The experimental study consists mostly of observations and discussion. Different sets of experiments will be illustrated throughout the context. It intends to show major findings regarding two aspects. First, the Bitcoin network topology tends to rely more on specific groups of nodes subject to node discovery methods. Second, it illustrates that transaction confirmation is a complicated process that involves some randomness. For end users, transaction fees do not always pay them off.

4.2 Experimental setup

Experiment study is partitioned into two major sets. The first set focuses on Bitcoin client network communication behavior and attempts to explain what it implies about some features to network topology. Two problems are primarily concerned in this observation. How do Bitcoin clients find each other? How perfect data synchronization is achieved? Bitcoin clients are right now programmed with two ways of node discovery. The client is hardcoded with a list of fixed hostnames of specific DNS servers called "DNS seeds". The list can be subject to change when a newer version of Bitcoin software comes out but stays hardcode for the same version. Querying these DNS seeds replies a list of publicly routable IP addresses referring to other nodes running the Bitcoin client.

All Bitcoin communication between nodes is via Bitcoin messages that can be considered as the application protocol in Bitcoin system. The other node discovery technique is through node address broadcasting messages. It is the process of requesting IPs of other nodes from the nodes one is already connected with. Sending a "getaddr" message to your connected peer nodes requests peers to reply with an "addr" message containing a list of other nodes' addresses. The addresses fetched by a client are stored in a local file "peer.dat" for future reference to select random IPs from when handling "getaddr" requests from others. There is one important detail of using "addr" for public IP broadcasting. Every 24 hours, a client broadcasts its current public IP address through "addr" message to its connected nodes, to notify the network of existence of itself [18].

Data transmission in Bitcoin is similar to gossip based approach but still implemented in messages. Transactions and blocks in Bitcoin are broadcast in the same way on the network. Both of the two types of data transmission starts with an "inv" message used to advertise what new block/transaction one has heard. Peers may ignore "inv" if it has the advertised transaction/block or if not it replies with "getdata"/"getblock" message to request that transaction/block.

There are default connection limits regulated by each standard client. Each Bitcoin node requests a maximum of 8 outbound connections to others and a maximum number of 125 total connections including inbound and outbound connections. For most clients that run behind NAT their connection number will never exceed 8 given the NAT router is not specially configured to forward Bitcoin application port number, as these clients cannot accept any inbound TCP connection necessary for relaying Bitcoin messages. Such connection limits reveal some critical features regarding the network topology. As each message can be multicast only to a limited number of nodes, information propagation becomes sensitive to latency. Due to lack of central coordinator, it is a theoretically correct assumption that some nodes might experience difficulty keeping them informed of most up-to-date block and transaction information in time. However, the first set of experiments show that such an issue rarely occurs in real environment.

The second set of experiments seeks for patterns in which transactions are confirmed. Bitcoin transaction is instantly generated and issued, but transaction confirmation is not. A user always has to wait for one transaction to be confirmed by some miner before this transaction can be considered valid and successful. The best optimal estimation of time taken to confirm a transaction is simply the time interval from transaction generation to the next block coming out. Up to the recent block 327,000 on the block chain, the average time interval between two subsequent blocks is roughly 11.52 minutes [19]. It is reasonable to expect to see a transaction appear on the block chain within the 11.52 minutes however much longer confirmation time is still commonly seen because for monetary purpose most miners confirm transactions in order based on transaction fee, which is a key factor that determines how fast a transaction is confirmed.

Transaction fee is a concept as part of Bitcoin core protocol and it enhances economic motivation for more miners to contribute to currency generation and transaction confirmation. A payer indicates transaction fee amount in a transaction as extra fee paid to whoever helps confirm this single transaction. The miner permanently claims the amount of fee in the transaction he confirms and this fee can also be 0 for some transactions. A common practice of recent miners is to first sort unconfirmed transactions by the fee indicated then confirms those with larger amount of fee with higher priority. For most end users, paying a higher transaction fee raises their expectation to have the transaction confirmed sooner. However, confirmation time vs. transaction fees does not abide by simple monotonic pattern, as there are factors that affect how a transaction is relayed before it reaches specific miner.

The experimental observation also helps find out an approximate pattern how fast transactions get confirmed with different ranges of transaction fee and model the trade-off between the fee amount and confirmation promptness, from the real block chain statistics within the recent 20 months.

4.3 Experiments and analysis

4.3.1 Network Investigation

Experiment set A consists of two experiments mainly on network environment study.

1) Experiment 1 – DNS seeds

DNS seed servers in Bitcoin function as small IP databases and are maintained by volunteers. Those seed servers are no more than other ordinary DNS servers on the Internet and they only provide IPs of Bitcoin users but never run anything about Bitcoin themselves. The IPs provided by seed servers have significant impact on how Bitcoin clients discover other nodes, especially for the first time a Bitcoin client runs because it has not yet kept any peer node's IP in history. The client has to query one or more DNS seeds for the first list of IPs to get engaged into the Bitcoin system.

DNS seeds are always hardcoded in Bitcoin client program. But the list of DNS hosts do not always remain fixed, neither do the IP addresses answered from them. During the investigation period of this paper, DNS seeds hardcoded in the clients within 3 months increased from 4 to 7 servers as shown by the standard client "bitcoind" source code [20].

Experiment 1 did exhaustive search by repeating DNS requests for a long time to the DNS seeds trying to get a maximum number of possible unique resolved IPs. We launched a customized program to handle DNS requests and collected all unique IPs. The process kept running for a long time until the total number of unique IPs from those servers hardly increase. The whole list of IPs is then sorted by relative frequencies (proportion to the total address count) in the list. The experiment was repeated every 2 months for 3 times. The following 3 datasets display the results from trial in each time.

Table 4-1 #IPs in DNS Seed Lists

# Data Set	***1***	***2***	***3***
Total # Unique IP	6187	10191	5928
Highest Frequency	0.0079	0.0062	0.0068

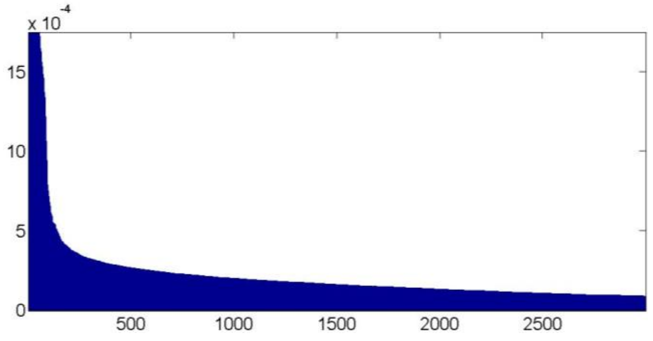


Figure 4-1 Data Set 1 - Address Distribution 1

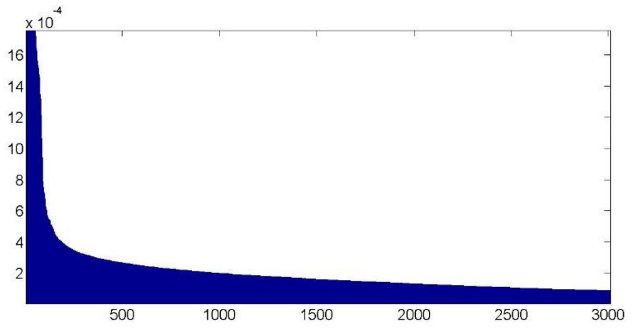


Figure 4-2 Data Set 2 - Address Distribution 2

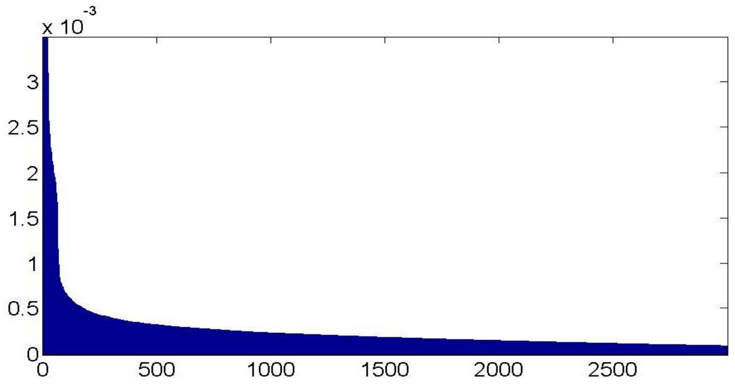


Figure 4-3 Data Set 3 - Address Distribution 3

The above three data sets refer to different IP lists resolved by the DNS seed servers. Even if the IP list from DNS seeds vary a lot, the distribution is relatively stable and far from uniform. This also implies the fact that some nodes in the DNS seed IP list are queried with far higher probability than the rest.

Hypothesis may be inferred that during specific period of time during which IP lists from DNS seeds remain relatively stable, there are a small set of nodes that are much more active than others on Bitcoin network. This node set can result in a more compact core than the rest of network topology.

2) Experiment 2 – Peer Discovery

The node address distribution outcome in Experiment 1 is not yet a direct reflection on the probability which node to connect to during the runtime. The standard client adopts a complicated technique to randomize as much as possible the choice of peers regardless of some preference to previously connected IPs that worked before. Nodes that were previously connected to are saved in the file "peer.dat" for future reference.

Experiment 2 consists of 3 steps to model the practical situation described above. For the first part, we deleted "peer.dat" so as to make Bitcoin software rebuild an empty profile. The client process was killed and restarted every 10 minutes for more than 100 times. We modified the client source code so that every time new addresses are written to "peer.dat" those addresses are also dumped into separate files for us to collect.

Table 4-2 Addresses in Local Cache

# Attempt	***1***	***2***	***3***	***4***
# IPs from seed	141	252	141	23
# total IPs	480	930	570	75
Seed nodes/total ratio	0.294	0.271	0.247	0.306

Table 4-2 shows that a pretty stable proportion (around 30%) of IP addresses from DNS seeds are cached at local. That is to say these addresses are what have been cached into peer.dat.

The second step makes statistics on the other information source of “peer.dat”, the “addr” messages from other nodes. Addresses relayed by “addr” are periodically dumped into this file. Within a 24-hour period, we collected all the “addr” messages that were heard and examined each single unique IP. The main reason to stay for 24 hours is to allow for every online node to broadcast its own IP for at least once, according to the rule mentioned in Section 4.2.

Table 4-3 Addresses in “addr” Message

Total # unique addresses	*Addresses from DNS seeds*	*Ratio seed nodes/ total*
20005	1249	0.0624

In the last step, we simply saved a list of all nodes that were actually connected during the past 24 hours.

Table 4-4 Addresses Actually Connected

Total # unique nodes connected	*Addresses from DNS seeds*	*Ratio seed nodes/ total*
154	13	0.0844

Table 4-4 is a real use case description. It shows that 8.4% of connected nodes come from DNS seeds given a client program stays online for a long time.

4.3.2 Transaction synchronization

Experiment set B includes 4 experiments studying how transactions get propagated and confirmed. Experiment 1 and 2 both focus on the frequency of nodes from seed servers and how active they are. Starting from Experiment 3, we investigate their property and real functionality.

1) Experiment 3 – Transaction Arrival Test

Experiment 3 relies on the well-known third-party Bitcoin wallet service website blockchain.info which supports various API requests about detailed block chain data. The experiment first started the Bitcoin client and only connected to one node IP of choice. This was to force the connected node to be the only information source for our client to synchronize with the network. Within 2 hours period, we recorded every single unconfirmed transaction heard from this node, attached with a time stamp on the moment we received it.

Next we queried blockchain.info transaction database with each single transaction we collected for their arrival time on the server. blockchain.info also keeps a time stamp when a transaction reaches their server. Our initial assumption on this experiment is that due to low connectivity our client would have most transactions arrival lagged in time, compared to the blockchain.info server which maintains hundreds of connections all the time that far exceeds the connection limits for end user clients.

Experiment 3 was in parallel to Experiment 1 and also depends on IP list from DNS seeds. It was first done at the same time Data Set in Figure 1-1 and was repeated after two months when seed IP list changed to Figure 4-2.

Table 4-5 Average Latency 1

Test Node	***1***	***2***	***3***	***4***
Total # Transactions Collected	5354	5194	5906	7600
Frequency in DNS Seed List	0.0079	0.0046	0.0019	0.0003
Average Latency with Server	-7.9s	1.46s	-17.4s	-5s

Table 4-5 illustrates the average latency calculated over all the collected transactions from 4 different nodes picked up from DNS seed list of different frequencies starting from the highest 0.0079 as shown in Table 4-1. The results show that given only one node of our choice, we are mostly not lagging the web wallet server in terms of synchronization. The average latency in the table is a relative value by local time stamp minus the time stamp on the server. Only Node 2 here shows a tiny average delay of 1.46s and the other negative values tell us that we were receiving transactions faster.

The result goes against our initial assumption and proves that single connectivity to specific node still guarantees a functional information source. However, Table 4-5 does not explain why latency differences between the four tested nodes occur in such a way. Nodes 1 to 4 are ranked with decreasing order of frequency and we hope to see higher frequency node is more active than others.

To further investigate such property, we moved on to the second time evaluation based on the seed node list pattern shown in Figure 4-2. We picked 6 nodes in decreasing frequency and made the following statistics.

Table 4-6 Average Latency 2

Test Node	***1***	***2***	***3***	***4***	***5***	***6***
Total # Transactions Collected	4626	4852	4051	4077	1615	6021
Frequency in DNS Seed List	0.0062	0.0057	0.0055	0.0003	0.0001	5.4e-6
Average Latency with Server	-10s	-1.13s	-323s	2.25s	-4s	650s

Table 4-6 shows a much clearer pattern what performance each tested node has in relaying transactions. Nodes 1-3 are the three most active nodes from the seed list at the moment and they all proved to provide us in-time information delivery somewhat better than the blockchain.info server, with Node 3 as the most active one that leads an average of more than 5 minutes.

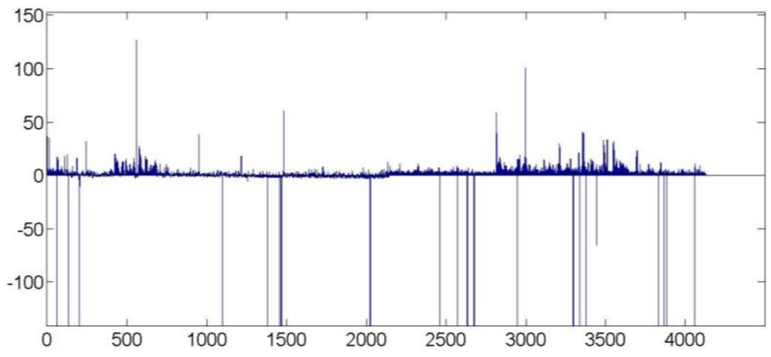


Figure 4-4 Tx Arrival Latency for Node 3

Transaction relaying process on Node 3 is elaborated in Figure 4-4 which shows each transaction arrival status in time order. For transactions arriving to our client later than the blockchain.info server, the relative latency is no longer than 140s that is within 3 minutes. There are many more transactions we received ahead of time for more than a few thousand seconds some of which are several hours. These facts give us sense that Node 3 is a pretty active Bitcoin node and has very nice performance of data relay.

Node 6 works in contrary to Node 3 case. Since we are lagging for more than 10 minutes in average transaction arrival time, we consider this node as much less active as Nodes 1 to 3.

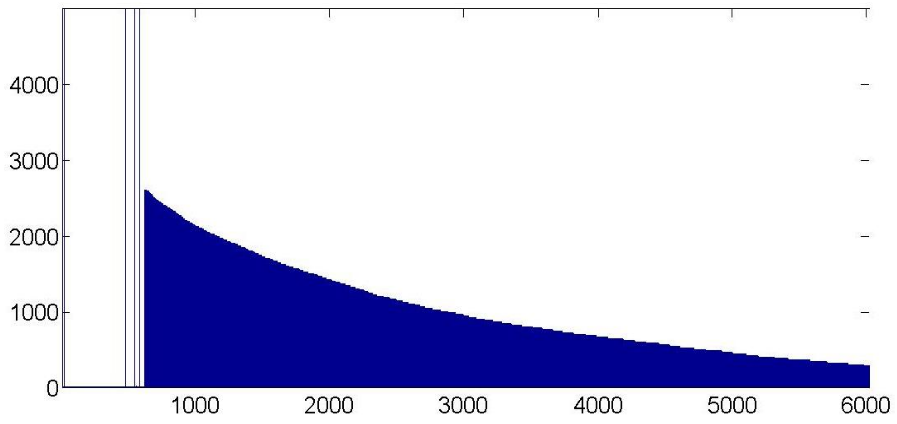


Figure 4-5 Tx Arrival Latency for Node 6

Figure 4-5 shows detailed transaction arrival pattern tested on Node 6. Anomaly exists in Table 4-5. Node 5 only relayed 1615 transactions far below the average number during the same length of time over the network. This reveals that connection to Node 5 was interrupted during the experimental process.

Not all nodes given by DNS seeds are guaranteed active all the time such as Node 5 here. Our observation shows high frequency nodes in the DNS seed feedback tend to be more active and stable with better efficiency of information propagation.

2) Experiment 4 – Transaction Loop Back

Experiment 3 evaluates efficiency on different sample nodes provided as seed nodes. Experiment 4 attempts to find out how a typical Bitcoin transaction travels around different paths on real network. But Bitcoin transactions are never recorded with source or relay paths, capturing how a transaction was routed is difficult. Experiment 4 does the following tests trying to detect difference nodes inside the seed list and others.

We launched two processes of Bitcoin clients on the same host; say Client 1 and Client 2. We connected Client 1 to only one peer node to Test Node 1 mentioned in Experiment 3; and Client 2 only to Test Node 2. Then we sent multiple Bitcoin payments at Client 1. Since both test nodes are found to be highly active from previous experiments, we assume transaction propagation between them is within very little number of hops. We kept track of the time interval from sending transaction from Client 1 to receiving it at Client 2. In this way the transaction goes back in a loop to the same destination as origin.

Table 4-7 Loop-around Time 1

Test Case	Seed Node to Seed Node
# Transactions	100
Average Intervals	7.07s

T

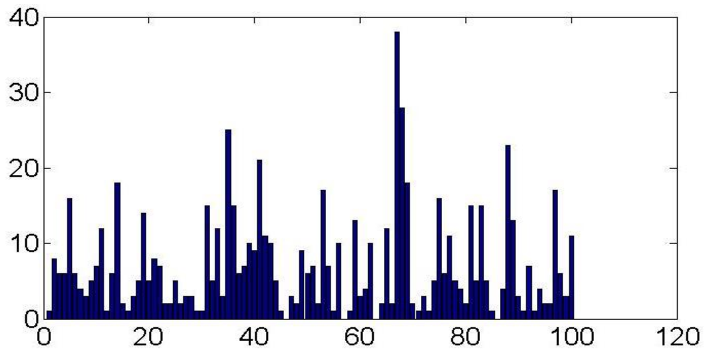


Figure 4-6 Per-transaction Loop-around Time

Figure 4-6 shows all the 100 transactions loop-around time and in average we need to wait 7.07 seconds for a transaction we signed to come back. This also approximates the time it takes from Test Node 1 to 2.

For a second trial in Experiment 4, we replaced the two test nodes with nodes outside the DNS seed list at the moment.

The two nodes are chosen among peers that were previously used in the program history. Initial expectation is that we want to get somewhat longer average loop-around delay due to less activity of these nodes compared to those shown in Table 4-7.

Table 4-8 Loop-around Time 2

Test Case	Nodes Outside Seed List
# Transactions	76
Average Intervals	6.28s

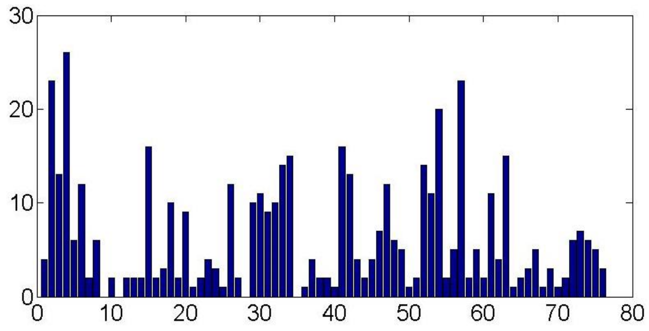


Figure 4-7 Per-transaction Loop-around Time 2

The outcome from Table 4-8 did not comply well with our expectation and the pattern in Figure 4-7 did not exhibit essential difference from Figure 4-6. The average loop-around interval is roughly the same with 7.07s in the first trial case.

The major reason of deviation from expectation comes from two restrictions. First, only a low number of actual transactions were issued so the average interval was still not of adequate statistical significance. Second, the nodes of choice are still of nice relaying performance because they were picked from the program history of nodes that had been used before.

3) Experiment 5 – Transaction Fee Distribution

Experiment 5 does in-depth study on the transaction fees over Bitcoin system within the recent two years. The study covers block 200,000 to block 305,000 corresponding to time starting from 10:45:59, 09-22-2012 to 21:58:26, 2014-06-09. There are more than 33,000,000 transactions during this time period and this experiment uniformly picked 10,944,186 out of those as samples to analyze how much transaction fee is included in each of them. Also we care about whether there is a steady trend in Bitcoin users' preference of paying transaction fees. For example, we hope to find conclusion if most

Bitcoin users are used to paying a fixed amount of transaction all the time or instead this amount gets changed by time. Also we hope to know if there have been more people willing to reward miners fee or not since Sept 2012.

The Bitcoin client is implemented with an important rule on transaction fee, called "minimum transaction fee". This rule has significant impact on the economy of Bitcoin circulation. The "minimum transaction fee" is the minimum amount that is imposed on every Bitcoin transaction unless the transaction is 0. Any transaction fee below this amount will be treated the same as 0 in the transaction relay process.

Bitcoin client does not always forward transactions of 0 transaction fee unless a transaction satisfies a priority threshold. The priority [21] of a transaction is calculated over a transaction's input age, payment amount and its data size in bytes. There is no way to forge transactions with high priority so that no attacker is able to send large traffic of free payments to lay down other Bitcoin nodes, which is called "penny flood". Transaction fee is calculated in the base unit in Bitcoin named "satoshi" after the pseudonym of the initial whitepaper. One Bitcoin (BTC) is equivalent to 10^8 satoshi and current "minimum transaction fee" is set at 10000 satoshi. Right now Bitcoin client automatic appends a minimum transaction fee on every payment the user sends in order for transactions to be smoothly relayed.

During the transaction analysis process, we excluded all generation transactions. Generation transactions are the first transaction in each block signed by miners as mining reward so it makes no sense to define transaction fee for generation transactions.

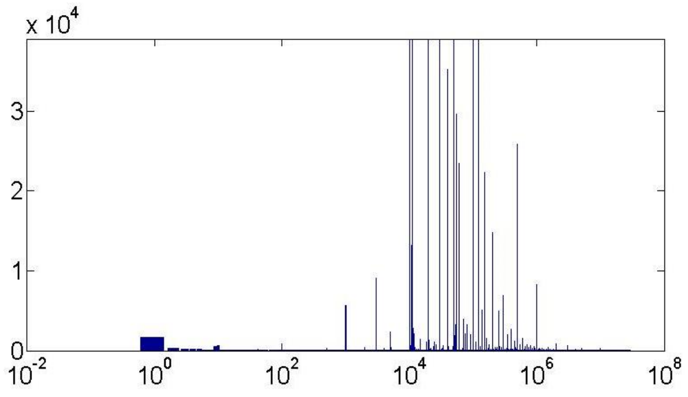
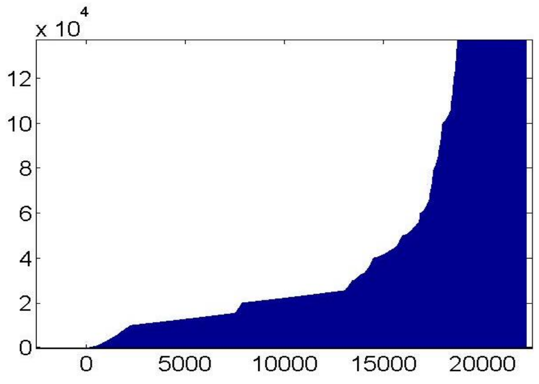


Figure 4-8 # Transactions with Transaction Fees

A global statistics on the 100944186 collected transactions is displayed in Figure 4-8 with logarithmic scale of transaction fees in satoshi on x axis and number of transactions with corresponding fee on y axis. The transaction fees range from a maximum value of 980000000 (9.8 BTC) to 0.



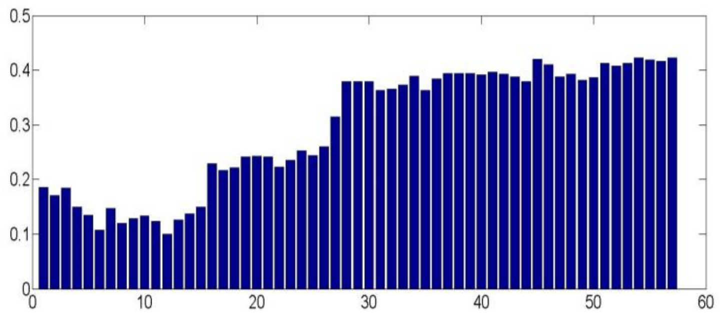


Figure 4-9 Distinct Transaction Fee Sorte

Figure 4-10 Percentages of Transaction Fees Under 10000

Due to a large sum of transactions, we rearranged all the transactions fees, collected a total of 22148 distinct fee amounts and sorted the sequence in ascending order. Figure 4-9 illustrates the sorted results with y axis representing one unit of minimum transaction fee

(104 satoshi) and x axis only as the index in the sorted list. A majority (about 13500 out of 22148) of fee amounts are distributed under 30000 satoshi.

From the arranged transaction fee data set, we are more concerned transactions with a fee less than 10000 satoshi which is below the minimum fee threshold. To make a better observation we split the data set in time order into 56 slices with each consisting of 167,000 transactions that account for about half a month real time. Figure 4-10 displays in each column the proportion of transactions with miner fee below 10000 satoshi in each half a month period starting from Sept 22, 2012. The increasing trend verifies that current protocol of relaying transactions based on priority filtering is working. As coin age is determined by the age of block an input transaction is in, there will be more transactions with older inputs as time goes by because unspent transactions become “mature” as the block chain grows.

4) Experiment 6 – Tx Fee vs. Tx Confirmation Time

For end users, paying a transaction fee is for the sole purpose of having their payments confirmed in a shorter time. In Experiment 6 we investigate what role transaction fees are playing to improve end user experience. However it is not possible to calculate transaction confirmation time from the block chain data because there is no information what time a transaction was generated. There is a public stat chart available on blockchain.info website [22] showing long-term average confirmation time on a macro scale without any per transaction based information.

In this experiment we compute the approximate transaction confirmation time by assuming that all transactions arrived at the blockchain.info server in a very short time.

The confirmation time is estimated as the interval from the arrival to its confirmation into a block, whereas confirmation time is guaranteed to be exact as all blocks have timestamps. Experiment 3 showed that it is not a true case for all transactions as some of them would take hours to arrive at the server. This would cause the estimated guaranteed confirmation time to be negative in extreme cases.

We sampled 56,000 transactions from block 200,000 to block 305,000 and used blockchain.info database as reference. Although larger amount of attempt can improve data precision we limited the sample size at 56,000 to avoid abusing API provided by blockchain.info.

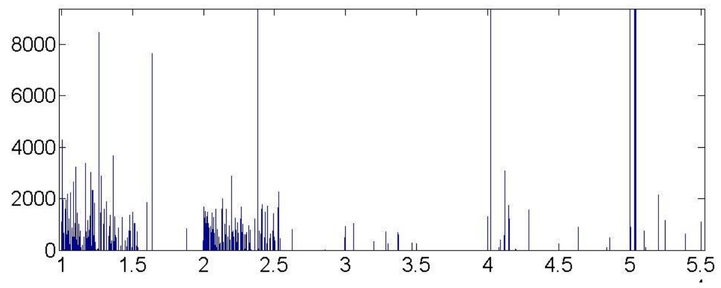


Figure 4-11 Confirm. Time vs. Transaction Fee

Figure 4-11 is the global statistics on the 56,000 transactions with y axis in every 10000 satoshi as unit. We filtered out all negative confirmation times that are trivial. There is no clear pattern to extract from the diagram and transaction confirmation time vs. transaction fees are not going in a monotonic way. Figure 4-12 is ascending order arrangement of transaction fees from Figure 4-11 to make x axis more compact.

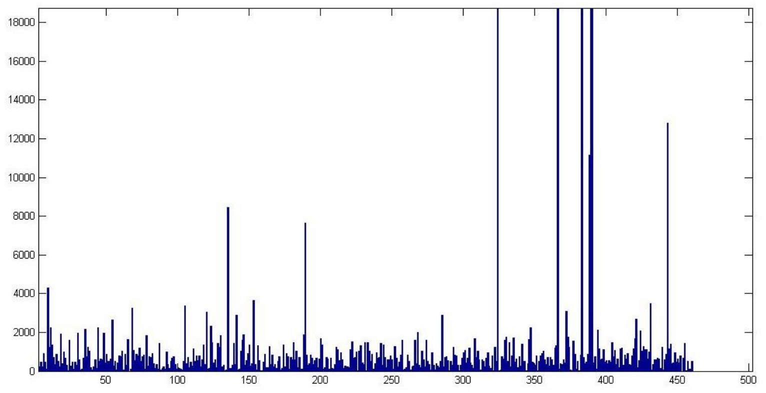


Figure 4-12 Confirm. Time Arranged by Increasing Fee

By results from Figure 4-9 a majority of transactions are ranged above and round 10000 satoshi. So if we try to detail part of the data set of Figure 6-1, we have the following table.

Table 4-9 Confirm. Time for Tx above Fee 10000

Fees in satoshi	10000	10001	10004	10009	10010
Confir m Time (s)	1083	723	205	457	241
	10019	10021	10022	10055	20000
	900	446	90	4290	667

Table 4-9 is the average estimated confirmation time for transactions starting from the minimum amount to twice the amount, as a local view into Figure 4-9. The table shows the effectiveness of raising transaction fee to amounts higher than the minimum value. However, higher transaction fees do not always reward with sooner confirmation. It is not correct to say paying a large sum fee to the miner guarantees privilege than others even when error in the sampling space is considered. We would suggest an end user pay a default amount of minimum transaction fee in everyday use of Bitcoin as the minimum amount suffice to make confirmation within acceptable time in most cases.

In this experiment we would like to go one step further to magnify the significance of "minimum transaction fee". We collected all the transactions with exactly 10000 satoshi fee, the default value by the standard client and most web wallets.

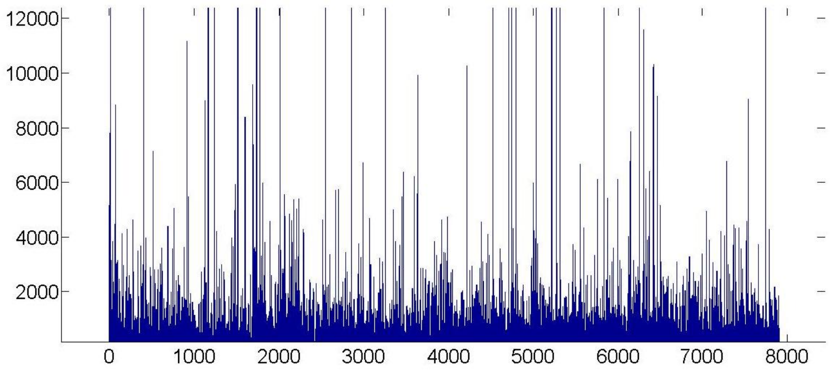


Figure 4-13 Avg. Confirm. Time for Fee 10000

Figure 4-13 lists out all transactions with the default 10000 satoshi fee by time order still with all negative estimation values filtered out. Out of fluctuation the confirmation times do not show any monotonic pattern. This means that for end users, their waiting time for transaction confirmation has not changed much at this moment compared to Sept 2012. This verifies our above suggestion.

4.4 Summary on experiments

This chapter presented two sets of altogether 6 experiments to reveal some parameters that affect end user experience from network environment investigation and transaction synchronization evaluation. We conclude from experiments in Set A to the proposition that DNS seeds tend to choose more active Bitcoin nodes and those of the highest frequency tend to stay on top of the address list during some fixed time. The way that DNS seed servers organize IP addresses can affect the network's topology in which some group of nodes get far more inbound connections than others. Despite the existence of nodes that seem more essential than others address list on DNS seed servers are altered a lot by time. Given such servers are not manipulated by malicious colluding mining pools, the effect of DNS seeds can be consider positive to Bitcoin network because it improves the efficiency of node discovery for end users.

Experiment set 2 is mostly focused on the way transactions get relayed. Via study over recent blocks within two years, we found that there has not been much change to Bitcoin user habit in terms of transaction fee amount. We suggest that all Bitcoin users attach a minimum of 10000 satoshi fee in their payment as refusing the minimum amount can significantly increase their waiting time for confirmation. But it is not worthwhile to pay too much more "tip" to miners because experiment results show that it does not necessarily pay off.

CHAPTER 5

LARGE GRAPH BASED FORENSIC ANALYSIS OF BTC TRANSACTIONS

In this chapter, about two years of Bitcoin transaction data is collected. We develop a graph-based method to analyze the money flow and identity clustering properties of these transactions. The approach is scalable for processing large Bitcoin graphs and focuses on transactions relevant to criminal cases such as Mt Gox. We seek for answers to forensic questions regarding specified time interval and/or a selected set of user identities. Some interesting results include the answer whether Bitcoin system achieves true anonymity, how money flows among selected users and community structures of Bitcoin users. The developed graph-based forensic investigation approaches are expected to help law enforcement in their investigations that deal with Bitcoin transactions.

As stated in Chapter 1, Bitcoin as digital currency logically exists in different forms than paper money or balance from online banking. No user balance is recorded on the block chain. Every transaction is a record on a sum of money paid from one source to a destination. Bitcoin is identified and authenticated based on Bitcoin addresses, which are hashes generated from users' public/private key pairs [23] and cannot be manipulated unless one possesses the correct private key. A Bitcoin address can be randomly generated based on a randomly picked key and is the only unique way of identification in the system. Due to the random property of Bitcoin addresses and extremely low probability of key collision due to the enormous ECDSA key space [24], the number of Bitcoin addresses grows rapidly because one may generate hundreds of different addresses for different transactions without address reusing. This address generation scheme results in pretty decent anonymity to Bitcoin transactions and makes it difficult to track Bitcoin transactions relating to real world entities in such an almost perfect anonymous system.

This chapter focuses on the transaction aspect of the Bitcoin system in an attempt to construct a low-cost transaction data analysis model, and atop it, develops analytical approaches to support forensic investigation of bitcoin transactions and relationship of bitcoin users. It reveals that different Bitcoin addresses belonging to the same entity can be effectively clustered to reduce the forensic efforts in tracking criminal bitcoin users, and graph analysis on bitcoin transactions help discover transaction behaviors and money flows among bitcoin users. It will show data analysis results in study about Mt Gox case. It is expected that this work will provide an easy and fast approach in the investigation of various criminal cases that involve the use of Bitcoins, such as money laundry or fraudulent transactions.

5.1 Transaction data collection and the platform

All transaction data used in this paper are confirmed Bitcoin transactions. We collected them from the block chain maintained in Bitcoin system, starting from block 210,000 to block 314,700 corresponding to the block creation time from 15:24:38, 2012-11-28 to 10:45:14, 2014-08-09. During this 20-month time period, a total number of 34,839,029 Bitcoin transactions have been successfully released and globally confirmed. In these transactions, a total 35,770,360 distinct Bitcoin addresses were involved. All the results from this paper are based upon the transactions and addresses from this data set.

Analytical study in this chapter is based on the block chain data downloaded using the compiled full Bitcoin client source Bitcoind in C++ [25]. This data set involved genuine with its integrity ensured through the crypto schemes used in Bitcoin system. Transaction and block data were queried with *getrawtransaction* and *getblock* command line RPC tool provided by *Bitcoind* itself. Alternatively it can be retrieved by directly revoking the *getrawtransaction* and *getblock* methods from the source. Both methods are globally visible in file *rpcrawtransaction.cpp* and *rpcblockchain.cpp.* JSON format of single transactions were parsed from the above methods and written into the file as shown in Figure 5-1 and Figure 5-2.

```
{
"inputs":
[
{"value": 2.5, "address": "13vjvKHDeFhtwRGseVkLF1eXA7ZrJA2Uo9"},
{"value": 0.0092427, "address": "1EyH7htSWjSyuXgofuE9gKX5Sr4yEyDcD2"}
],
"blocktime": 1389490523,
"outputs":
[
{"value": 2.5, "address": "1Q5ztGyLj7KxL2rs7bmVcKYDfTYmpQs5bo"},
{"value": 0.0090427, "address": "1G7HKqqTUCrTpMDEYV5SdzcarFPhKKHb12"}
]
}
```

Figure 5-1 One Single Transaction

```
{
"inputs":
[{"address": "coinbase"}],
"blocktime": 1354133545,
"blockhash": "0000000000000498e426effa08fc54070cd7ca70e80206f8763e7ceaaab734bc",
"outputs": [{"value": 25.0512, "address":"1811f7UUQAkAejj11dU5cVtKUSTfoSVzdm"}]
}
```

Figure 5-2 One Coinbase Transaction

Major content in Bitcoin transactions is displayed in JSON notation in Figure 5-1. Figure 5-2 is an example of Bitcoin generation in which there is no incoming address. It indicates creation of a new block with the hash value shown. For this example, 25.0512 BTC (as 1 coin unit) was mined and it belongs to the output address in the output array. This single transaction is also stored in the block with the hash value shown.

5.2 Graph-based Bitcoin transaction analysis

The transaction investigation model consists of three major procedures. In the first step we collected transaction data set and parsed them into transaction input-output based format as shown in Figure 5-1. Based on the each transaction, we attempt to classify all addresses involved in the data set into groups in which we are confident enough to determine that these addresses belong to the same Bitcoin entity. Such an entity can be the same person, groups of Bitcoin users that share Bitcoin wallets (private keys) or colluding Bitcoin miners that cooperate in mining activities.

The second step is to convert the transaction data into address based graph notation. The objective is that the transaction flow is clearly displayed between groups of addresses with each group representing addresses belonging to the same entity. We designed the graph in such a pattern that each node in the graph stand for the same group of addresses. Any transaction spent from addresses in the group is notated as an outward edge and any transaction paid to a group is the incoming edge to this group node.

The last step seeks for answers to several questions. How efficient we are able to track currency flow with the help of address graph from the second step? How does graph behavior help us discover transaction activities like money laundry and large-scale Bitcoin theft/fraud? How do we investigate Bitcoin entities engaged in conspiracy if those activities exist in the transaction data record? The rest of this section covers all detailed algorithm and experiment processes regarding the problems.

5.2.1 Address Clustering

We attempt to adopt a fast algorithm of checking and grouping all the addresses into distinct sets as precise as possible so that we know what entity group an address belongs to given an arbitrary Bitcoin address. There exists previous work [26] that characterizes

Bitcoin transactions into different usage categories. During the work it was proposed as a heuristic that different Bitcoin addresses of inputs from the same transaction are guaranteed to be possessed by the same entity. The reason is that due to the nature of Bitcoin transaction signature scheme, one cannot spend transactions that are not paid to him. Therefore multiple input addresses in the same transaction are strong indication that whoever issued this transaction actually owns, or at least share private keys to all these addresses.

In case of coin base transactions in Section II there is only one output address in the transaction output (Fig. 2.3) so this address alone get all the mining reward. But multiple addresses are also allowed in Bitcoin transactions including coin base transactions. Bitcoin mining intrinsically supports collaboration so that mining pools formed by multiple nodes participate intensive mining effort and addresses in the pool simply share the reward. For these reasons, the output addresses in the same coin base transaction are always pool related. We tell from coin bases the message of cooperation. For purpose of the problem definition in this paper, we also treat addresses cooperative in the same mining pool as an atomic entity for more clear reference.

Another hint for address grouping is support for change described in Fig. 2.1 where a payer sends the change amount to his own address which we call change address in this paper. Generally the change address is a one-time generated address for the sole purpose of receiving one's own money according to implementation by the current standard Bitcoin client software [27] the Bitcoin-Qt. By this property, we can be a bit aggressive to assume there are many one-time change addresses. One necessary condition for an output address to be classified as a change address is that it is not used as output one more time in the data set. Along with some other conditions summarized by previous work [28] we are able to collect change addresses with low false positives. To summarize, addresses with the following patterns are classified to the same set.

1. All input addresses in the same transaction must be in the same set.

2. All output addresses in the same coin base transaction must be in the same set.

3. An output address that is only used as output for once is classified to the same set of its transaction's input addresses if the following conditions are met.

  - This output address is not the only address in this very transaction's output. (A payment is not valid if it only sends the change.)
  - This output address does not exist in this very transaction's input list. (It is to avoid self-loop change though it is a rare case.)
  - None of the rest of output addresses in this very transaction's output list is used as output for only once. (While a change address probably exists in this case, we are unable to tell which one in the output list is used for change).

The actual classification takes effort in identifying and rearranging nearly 36 million addresses with each being a discrete and distinct string in 26-34 bytes length [29]. The problem is about iterating all the addresses in each transaction and writing them into proper sets so that addresses always belong to the same entity if they are in the same set. Moreover, this process involves set union operations because every time an address set that intersects with two or more existing sets, we have to union all those sets into an atomic new set. The process comes to a worst O(N^2) address access time complexity and unacceptably intensive redundant memory consumption if we directly apply with hash tables. Instead of the above intuitive process we adopted a much more efficient algorithm to perfectly reach the same grouping objective described above.

We used an open-source database developed by Google [30] as a light weighted on-disk key-value mapping tool called LevelDB. LevelDB is a formal database toolset featured in the standard Bitcoin client Bitcoin-Qt for quick queries into the huge block chain data on disk.

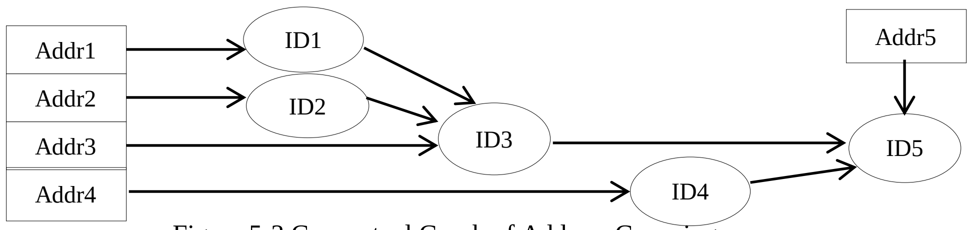


Figure 5-3 Conceptual Graph of Address Grouping

Figure 5-3 illustrates intuition of clustering different address sets. Every time we have a new address we map it to a new node. Suppose we have the following transaction input lists.

Table 5-1 Operation on Sample Transaction Inputs

Transaction ID	Tx1	Tx2	Tx3	Tx4	Tx5
Input Addr. List	Addr1	Addr2	Addr1,Addr2,Addr3	Addr4	Addr1,Addr4,Addr5
Database Operation	(Addr1,ID1)	(Addr2,ID2)	(Addr3, ID3) (ID1, ID3) (ID2, ID3)	(Addr4, ID4)	(Addr5, ID5) (ID4, ID5) (ID3, ID5)

We start from Tx1 with only 1 transaction input address. Mapping Tx1 and Tx2 is straightforward as Addr1 and Addr2 are from different transactions. However, Tx3 intersects with both Tx1 and Tx2 so it tells the fact that Addr1, Addr2 and Addr3 are in the same set. Instead of modifying the Addr1 and Addr2, we simply search for the node ID they are mapped to. Then we map all the nodes we found to the node new with ID3. In this way, we join all the sets without sequential accessing addresses in each set.

For Tx4 with no intersection with any other address sets in our current record, we simply map it to a new ID4. Similarly Tx5 intersects with several previous addresses so we update the database with operation shown in Table 5-1. After iterating from Tx1 to Tx5, all the 5 addresses are finally mapped to ID5. In this approach, given any sets of addresses we are able to correctly union all those with intersection, regardless of the order of iteration. (For instance, Tx5 to Tx1.)

However, in case of our 36 million addresses the above algorithm is still far from efficient as the directed path in the Figure 5-3 tree structure can be long enough as to make a single address search at expense of traversing through thousands of node IDs. We improve the algorithm by simply remapping a graph node to the root every time we need to traverse a path.

Table 5-2 Improved Operation on Sample Transaction Inputs

Transaction ID	Tx1	Tx2	Tx3	Tx4	Tx5
Input Addr. List	Addr1	Addr2	Addr1,Addr2,Addr3	Addr4	Addr1,Addr4,Addr5
Database Operation	(Addr1,ID1)	(Addr2,ID2)	(Addr3, ID3) (ID1, ID3) (ID2, ID3)	(Addr4, ID4)	(Addr5, ID5) (ID4, ID5) (Addr1, ID3) (ID3, ID5)

Table 5-2 shows improvement in operation at Tx5. As Tx5 has Addr1 for which we need to search from Addr1 to ID3 to get its root node ID. Once we find that Addr1 is actually mapped to ID3 we update this information for future reference. Similar shortcuts can also be applied to intermediate nodes such as ID2, if traversal through ID2 is needed at any later point. To summarize the algorithm, we do the following.

1. Grab the next Bitcoin transaction and extract its input addresses (or outputs in case of coinbase)

2. Examine the extracted address set. For addresses not yet in the database, we map all them onto a new ID. Ignore all existing addresses but traverse to the root node ID that address points to and memorized it in a set. During each traverse, optimize a path with shortcut if possible.

3. Examine the root node ID set in step 2. If the set is empty we have finished the work. Otherwise map the root node IDs in the set onto the new node ID in step 2. In case no new addresses are found in step 2, create a new node and do the same.

4. Go on and grab the next Bitcoin transaction. Repeat 1-3.

The above steps accomplishes set mapping among transaction input addresses and coinbase output addresses. We need one last step to scan for one-time change addresses by applying rules mentioned above. We conclude the address set classification to the following outcomes.

- Among all the 35,770,360 addresses we processed, 35,587,286 addresses are used as outputs. We identified 398,954 one-time change addresses that qualified our heuristic.
- A total number of 13,062,822 distinct address sets were generated.

5.2.2 Address graph observations

Up till this point, we have sufficient knowledge of what unique set a Bitcoin address belongs to. We are now ready to interpret raw Bitcoin transactions into a currency flow model between Bitcoin address sets. Our objective is to sketch a graph that reflects monetary relationship between sets of addresses so that tracing the incoming and outgoing payments about specific Bitcoin entity is possible. We define our directed graph as follows.

1. Each vertex in our graph stands for a set of address, which represents a discrete Bitcoin entity that possibly possesses multiple addresses.

2. Each edge is a payment of specific Bitcoin. An outgoing edge from a vertex is payment from the current address set the vertex stands for to another set. Similarly an incoming edge is payment received by this address set. Each edge in our directed graph is weighted with amount of payment.

Furthermore, we expect to study the transaction behavior within fixed time intervals so we divided Bitcoin transaction data from block #210,000 to block #314700 into multiple discrete samples with each consisting of 20 subsequent blocks, so that we have a total of 5236 graph data files. Each graph data file was converted into a graph defined above. We came to the following observations by quantitatively analyzing each graph.

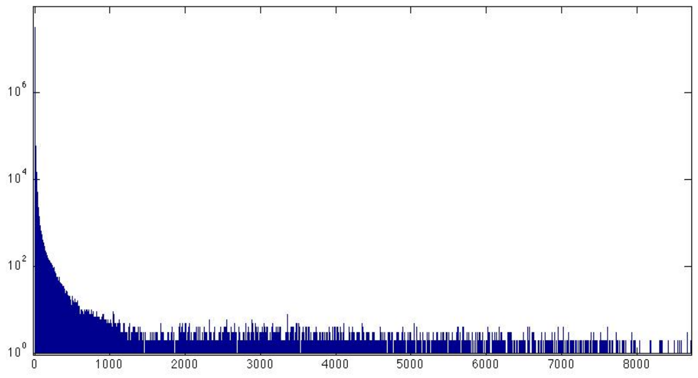


Figure 5-4 In-degree Distribution

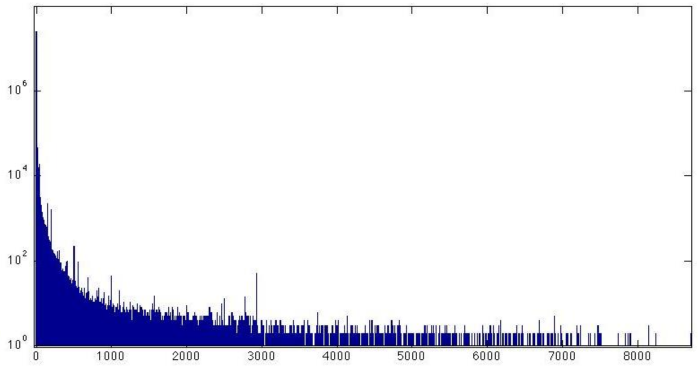


Figure 5-5 Out-degree Distribution

Figure 5-4 is a count of nodes with all numbers of incoming edges, calculated over all the 5236 graphs. A majority of nodes are with less than 100 incoming edges as shown by the logarithmic y-axis. Figure 5-5 is a count of nodes with all numbers of outgoing edges, calculated over all the 5236 graphs. It has similar distribution with the incoming edges shown in Figure 5-4 still with y-axis as logarithmic scale counting the number of nodes with specific number of degree. Other than the degree counts, we found that among the 94880896 edges over all of the graphs, 89406338 are payments below 10 BTC, accounting for a 94.2% proportion.

5.2.3 Money flow analysis and algorithm

As we concentrate on the graph edges, the major problem we consider is what happened to a specific sum of money. Whenever theft or fraud occurs, the coin goes two ways. First they are actually spent which is equivalent to being split and sent to multiple other entities. Second the stolen coin can be sent to multiple addresses through different transactions before they are finally collected to a criminal's Bitcoin wallet. This process is in some sense similar to money laundry if such a pattern exists in our graph. To quantitatively capture the money flow we decide to breadth-first search to determine the most probable direction the stolen coins goes in.

Our algorithm starts with a set of nodes from the graph within fixed time periods. With these nodes as the starting level, we BFS search the graph and memorize the nodes' number and total net output at each level until there is no more new node to find.

```
BFS_Graph(G):
    Initialize curLevel={set of starting nodes}, nextLevel=new set()
    Initialize countList=new list[], netOutputList=new list[]
    while curLevel is not empty:
        add #node in curLevel into countList
        for each node in curLevel:
            add all unvisited neighbors of the node to nextLevel
            mark neighbors as visited
            netOutput= netOutput +sum(out edges)-sum(in edges)
        add netOutput into netOutputList
        curLevel=nextLevel
        nextLevel=new set()
```

BFS algorithm on address graphs for money flow

The output of this algorithm is a list of node counts and a list of net coin output at each BFS level. We define the net coin output by calculating difference between total value of output edges and incoming edges.

We forward track the Bitcoin so that given a set of starting nodes we search for the pattern of coin transfer to elsewhere in split sums or go through money laundry process. Due to potential incompleteness of our address classification database, money laundry can also show up as a large amount of coins ending up at different graph nodes than the start. In this case we expect two graph patterns:

1. Starting nodes are involved in cycles consisting of large edges directed to them, with approximately the same total values as the total values of outgoing edges from them.
2. No valid graph cycles are found but coins converge into smaller number of large value edges that are gathered by a set of other nodes.

5.2.4 Case study: Mt Gox case

We have built a concrete data model that describes detailed Bitcoin flow pattern. We now come to reach our objective to find specific patterns that help with forensic investigation. We concentrate our application on a case study about the Mt. Gox incident.

Mt. Gox used to be the largest Bitcoin exchanger and web wallet service provider in the world. On February 7, 2014 Mt. Gox announced technical difficulty blamed for hacking attacks followed by a press release on February 10, 2014 claiming to have lost a total amount of more than 850,000 BTC worth a half billion dollar according to the market price of Bitcoin at the moment. The Mt. Gox incident, which leads to its official filing for bankruptcy, has been undergoing investigation for months and no clear explanation has been given. There is study about the Bitcoin transaction malleability [31] that was once believed to lead the huge loss from Mt. Gox. In their analysis, malleability attacks did exist to falsely modify transactions to enable twice amount of currency withdrawal from Mt. Gox. However, only a very small fraction of the alleged 850,000 Bitcoin could be tracked from their malleability detection data. It is still not clear how the rest of the coins were lost. According to a police investigation report [32] there were signs that the company's offices had been physically infiltrated and at least one former employee pilfered electronic data.

For whatever reason it was, Mt.Gox sustained huge loss of coins from the incident, which we believe must have been reflected on the block chain data and is also in our graph data. With high probability, the theft involves large sums of currency flows during short time, which we expect to find from our graphs. By observation described above, we have 94.2% of edges in our graph below 10BTC. To reduce the dataset for computing feasibility, we pruned all our graph files of edges of small values and kept edges of 10 BTC and above only. This simplifies the graph sets and brings efficiency to discover large flows.

Based on the Mt. Gox incident timeline, technical difficult was experienced and withdrawal service was suspended on February 7. As no theft case regarding the incident could have occurred after withdrawal stopped, we restrict our time window within a few days earlier before that. We started at graphs edges with timestamps at 19:49:42, 2014-02-03 till the time ending at 08:36:47, 2014-02-07.

We first extracted all graph nodes and edges within the specific time frame and integrated them into a new graph that describes global transaction patterns during the incident.

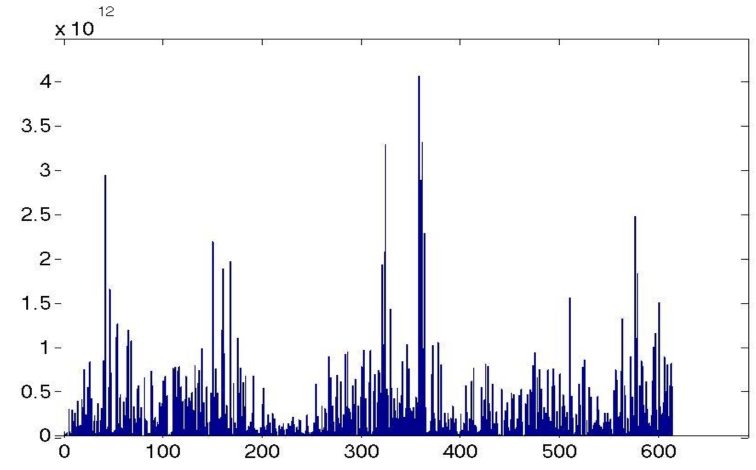


Figure 5-6 Sum of Transaction Amount by Time

Figure 5-6 is a time based sum-of-edge-value chart that displays the sum of values of all edges, equivalent to total transaction amount at each time stamp. Sum of edge values helps us know what time and in which groups of edges a burst of Bitcoin flow occurs. A burst of Bitcoin flow indicates large-scale currency transfer. During calculation of edge sums, we remove any edge that starts and ends on the same node so that we make sure that the transfer is between different entities.

The maximum peak of burst is shown in Figure 5-6 on x-axis value 359 to 363, corresponding to the UTC timestamp from 23:57:01, 2014-02-05 to 00:28:17, 2014-02-06. However, the total output values of these edges only account for about 100,000 BTC still far from being able to put Mt. Gox down to bankruptcy. Although we are not able to make perfect explanation for the claimed loss of 850,000 BTC from our dataset, we still reasonably found some behavior that is related to theft with high probability. There must be theft activities elsewhere but within the above time frame we believe it is the latest possible peak we are able to determine. As far as this case alone is concerned, it is acceptable to give about 1-day time before Mt. Gox eventually suspended cash withdrawal and made formal announcement.

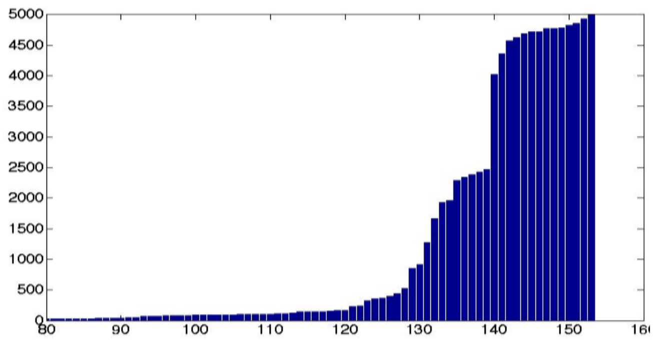


Figure 5-7 Suspicious Nodes with Largest Values

We first picked up all nodes connected to edges that contribute to the peak discussed in Figure 5-6 and we identified 153 distinct nodes involved. Figure 5-7 displays sorted values of the 153 nodes we spotted during the sensitive time period. The y-axis value is the sum of values from all edges at that time. We focus on the last 14 nodes that received the highest payment in total. That is, the nodes with values higher than 4000 BTC. Starting with the 14 nodes, we adopted the BFS algorithm against global graph context after specified time point. At the starting level, we initialize our traversal list to contain the above 14 nodes. This process was completed after 1461 levels before no more could be found.

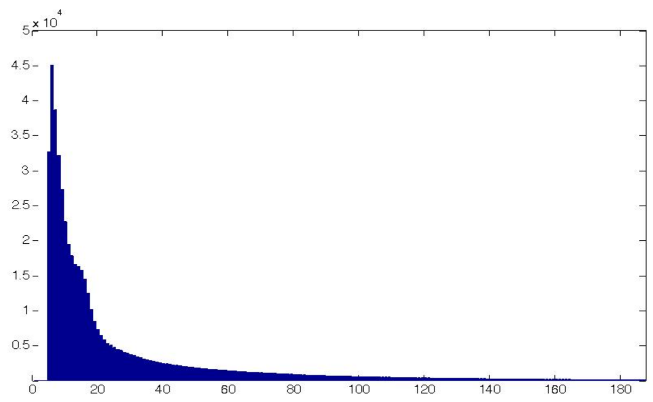


Figure 5-8 Node Count at Each BFS Level

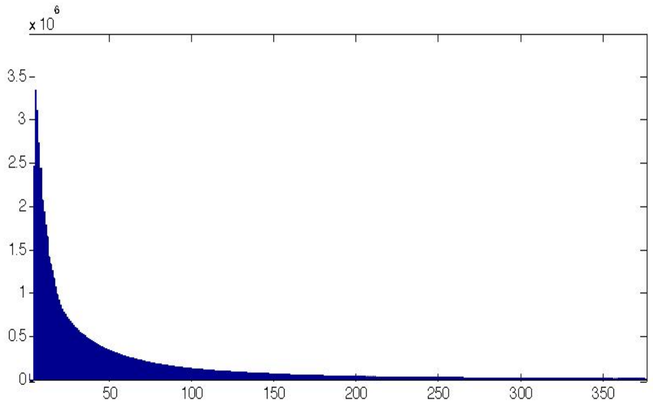


Figure 5-9 Values of All Nodes within Each Level

Figure 5-8 displays node counts during the BFS process. The node count drastically declines after the first 20 levels and finally converges to 1 single node. However, this figure is not telling enough information about the destination the transaction burst from 23:57:01, 2014-02-05 to 00:28:17, 2014-02-06 was actually transferred to, because we need the coin output into consideration. The net output of nodes at each level is plotted as Figure 5-9.

Table 5-3 Initial Stages of BFS

# Nodes	12	14	17	12	32621
Net Output	70510	-31205.998	-49545	2458376	3335101

The initial 14 nodes transfer 70510 BTC to neighbors. We are searching for some level such that nodes all the 70510 BTC are spent. Negative output shown in Table 5-3 is indication that nodes in the current level receive more income from other sources. When the BFS comes to the level where 12 nodes has much larger output than all previous income, it reflects that the initial 70510 we are tracking is split into the following 32621 nodes. From the decreasing output pattern in Figure 5-9, we are not able to find enough evidence of money laundry for which we expect to see large sum of coins converge to a small set of node or go back to the starting nodes in a reasonable length of time period.

5.3 Summary

Findings from the experiments are concluded as follows.

- Regarding the large coin transfer we focused on, there is no significant evidence showing money laundry activities for which we expect either edge-convergent pattern or graph cycles consisting of edges of equivalent values.
- The experiment results show high probability that a great portion of suspected Bitcoin has been spent and split by quite a few transactions paid to different entities instead of being kept to specific addresses of conspiracy.

CHAPTER 6
FURTHER DISCUSSION

Experiments illustrated in Chapter IV hopefully not only reveal current user end environment parameters but also provide some insights into how Bitcoin system as a large P2P network can be partially modeled from a local scope with limited resource so that similar methodology possibly gets adapted to designing more sophisticated empirical approaches for future study. Other than the major discovery that the Bitcoin network resource is shared with different weights among all nodes and specific groups of nodes have higher priority of receiving messages and having transactions confirmed, we also found that transaction fee is not a significant indication on suspicious activities. Because higher transaction fee does not statistically guarantee a faster confirmation, it does not offer much help in real forensic investigation.

Approaches suggested in Chapter V have two major restrictions. First, address classification is transitive so that future transactions potentially affect the group database by having multiple intersections with existing address sets. The latest block transaction data available in our data set is 347,000 but we expect that future transactions can be stronger hints for existing address sets to be joined. Moreover, we encourage application of anti-anonymity approaches above our classification method because it can significantly reduce the number of sets, which we conclude to 13,062,822.

The second major restriction lies in our address graph experiment. As the dataset did not cover the complete block chain data, this experiment was not able to determine the exact amount of coin value a specific vertex in the graph holds at each time stamp. By tracing the relative in-flow and out-flow of currency, our BFS outcome only reveals the flow of suspected currency without listing an accurate list of addresses of conspiracy.

CHAPTER 7
SUMMARY AND FUTURE WORK

Two major contributions are expected from this thesis. The first contribution lies in detailed practical study on information propagation in the Bitcoin network. Experimental verification is set up and discussed step by step to reveal specific facts regarding certain network environment parameters that impact message relay, node discovery and transaction confirmation. Although restrictions may apply to the experimental outcomes considering the P2P nature of Bitcoin with lots of probabilistic variants subject to network topology and long term time scale. But some results are acquired that can differ from the theoretical Bitcoin network model which is based on ideal multicasting mechanism and much simplified transaction verification schemes than what is currently programmed in standard Bitcoin applications.

The second contribution refers to the forensic investigation about Bitcoin transaction records. The investigation is based on experiments of classifying Bitcoin addresses and reorganizing data structures of the block chain so that different Bitcoin user entities can to some extent be revealed. This brings one step closer to forensic demands to track where large sums of currency flow to during specific time frame. On top of the classification model, graph based approach is applied to a case study on the recent Mt. Gox incident, a large Bitcoin theft case leading to bankruptcy of the once biggest Bitcoin market exchanger in the world.

Future work derived from this research include a sophisticated graph analysis through the whole block chain; an integration of anti-anonymity Bitcoin address identification database and consistent up-to-date copy of classified Bitcoin addresses; and quantitative analysis how powerful Bitcoin mining tools can be in future to manipulate the growth of the block chain.

Bitcoin is now generally considered as a promising decentralized global transaction system with open, public and scalable advantages. However, security concern has been a persisting topic since the genesis of Bitcoin. This thesis focuses its two study areas onto network topology and currency flow investigation. These aspects correspond to controversies whether Bitcoin will remain a well decentralized network because miners impose great significance on the system functionality; and whether it is possible to trace the destination of Bitcoin involved in large theft cases. Experiments in these researched hopeful help provide reference for future security related Bitcoin evaluations.

REFERENCES

[1] Bitcoin Community. *http://en.wikipedia.org/wiki/Bitcoin*

[2] Satoshi Nakamoto. (2009) Bitcoin: A Peer-to-Peer Electronic Cash System.

[3] Bitcoin's wiki documentation. (May 2014)

https://en.bitcoin.it/wiki/Introduction#How_many_people_use_Bitcoin.3F

[4] Blockchain.info. (May 2014) Bitcoin's most popular Bitcoin wallet and block explorer.

https://blockchain.info/charts/my-wallet-n-users

[5] National Institute of Standards and Technology. (March 2012) FIPS PUB 180-4.

FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION. Secure Hash

Standard.

[6] Bart Preneel, Hans Dobbertin, Antoon Bosselaers. (1997) The Cryptographic Hash

Function RIPEMD-160. CryptoBytes 3(2), pp. 9-14.

[7] National Institute of Standards and Technology. (June 2009) FIPS PUB 186-3. FEDERAL

INFORMATION PROCESSING STANDARDS PUBLICATION. Digital Signature Standard.

[8] Blockchain. (May 2014) Bitcoin currency statistics. *https://blockchain.info/stats*

[9] Blockchain. (May 2014) Bitcoin currency statistics. https://blockchain.info/stats

[10] Bitcoin Wiki Documentation. (May2014) Controlled Supply.

https://en.bitcoin.it/wiki/Controlled_Currency_Supply

[11] Christian Decker, Roger Wattenhofery. Microsoft Research. (2013) Information

Propagation in the Bitcoin Network. 13-th IEEE International Conference on Peer-to-

Peer Computing.

[12][13] Meni Rosenfeld. (December 2012) Analysis of hashrate-based double-spending.

[14]Burns, Matt. "FBI Seizes Deep Web Black Market Silk Road, Arrests Owner". TechCrunch. Retrieved 8 February 2014.

[15]Love, Dylan (2013-11-06). "Silk Road 2.0". Business Insider. Retrieved 2013-11-07.

[16] Satoshi Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System. 2009.

[17] Christian Decker, Roger Wattenhoferl, Information Propagation in the Bitcoin Network, ETH Zurich, Switzerland 2013 IEEE

[18] Bitcoin community: Protocol specification. https://en.bitcoin.it/wiki/ Protocol_specification, retrieved Sep. 2013

[19] Currency Stats. Time Between Blocks. https://blockchain.info/stats

[20] GitHub, Bitcoin Core integration/staging tree, https://github.com/bitcoin/bitcoin/tree/master

[21] Technical info, Transaction fees. Bitcoin Wiki. https://en.bitcoin.it/wiki/Transaction_fees

[22] Blockchain. Average Transaction Confirmation Time. https://blockchain.info/charts/avg-confirmation-time

[23] Technical background of Bitcoin addresses.
https://en.bitcoin.it/wiki/Technical_background_of_Bitcoin_addresses

[24] Digital Signature Standard.FIPS PUB 186-4.July 2013

[25] Bitcoin Core integration/staging tree. https://github.com/bitcoin/bitcoin

[26] Sarah Meiklejohn. Marjori Pomarole. Grant Jordan. Kirill Levchenko. Damon McCoy. Geoffrey M. Voelker. Stefan Savage. A Fistful of Bitcoins: Characterizing Payments Among Men with No Names. 2013

[27] Bitcoin Core integration/staging tree. https://github.com/bitcoin/bitcoin

[28] Dorit Ron and Adi Shamir. Quantitative Analysis of the Full Bitcoin Transaction Graph.2012

[29] Address. https://en.bitcoin.it/wiki/Address

[30] Leveldb Documentation.

http://leveldb.googlecode.com/svn/tags/1.17/doc/index.html

Thread-safe Python bindings for LevelDB.

https://code.google.com/p/py-leveldb/

[31] Christian Decker. Roger Wattenhofer. Bitcoin Transaction Malleability and MtGox. Mar,2014

[32]Takashi Mochizuki And Eleanor Warnock. Mt. Gox Head Believes No More Bitcoins Will Be Found. Mark Karpelès Describes Sleepless Nights in First Interview Since Exchange's Demise. The Wall Street Journal. June 29, 2014