Università di Pisa

Dipartimento di Informatica

Technical Report

Bitcoin Protocol Main Threats

Damiano Di Francesco Maesa [1]

[1] *Department of Computer Science, University of Pisa*

Address: Largo B. Pontecorvo 3, 56127 Pisa, Italy. Tel: +39 050 2212700 Fax: +39 050 2212726

Bitcoin Protocol Main Threats

Damiano Di Francesco Maesa *[1]

[1] *Department of Computer Science, University of Pisa*

Abstract

In this paper we explain the basics of Bitcoin protocol and the state of the art of the main attacks to it. We first present an overview of digital currencies, showing what they are and the social need they aim to satisfy. We then focus on the main digital currency up to date, Bitcoin. We treat the basics of the protocol showing what are addresses and transactions and how they are used in a distributed consensus protocol to build the blockchain. After that the main part of this paper presents the state of the art of the three main attacks on the protocol: fraudulent mining techniques, double spending attempts and deanonymization attacks.

Keywords: Blockchain; Bitcoin; Deanonymization; Fraudulent Mining; Double Spending

1 Introduction

Internet changed our society starting the so called "digital revolution". Almost every individual now can directly connect to any other on the globe allowing for a point-to-point information exchange or, particularly, commercial interaction. But still, for most users, a direct point-to-point value exchange (payment) is not possible, and a third party financial intermediary is required. Entrusting a third party with user funds weakens the user control and spending freedom, limiting it to have commercial interactions only with parties recognized by the entity. An example of this was the 2010 Wikileaks blockade, when users were prevented by their financial intermediaries (Bank of America, VISA, MasterCard, PayPal and Western Union) to spend their funds as they pleased. This episode led to the realization that centralized currencies controlled on our behalf by third party financial organizations are not the best way to freely and anonymously exchange value between users in a global distributed community. This need in the new digital society is met by digital currencies.

The only purpose of a currency is to hold value and allow the exchange of that value between users. Originally currencies were born to overcome the limits imposed by barter. Firstly precious commodities (mostly gold) were used to physically represent the value, then we moved to certificates backed by some precious commodities assets, to finally arrive to fiat currencies which are certificates without any backing. Gold was used to directly represent the value owned, gold certificates were backed by gold to indirectly represent the same value.

*damiano.difrancescomaesa@for.unipi.it

The certificate value is indirect because it is not determined by the value of the paper and ink it is printed on, but by the gold amount the certificate represents. Its acceptance is based on the trust of users in both the emitting entity and the intrinsic value of gold it represents. A fiat certificate is not backed by any precious commodity value, so its acceptance is only based on users trust in the emitting entity. This means that we accept fiat currencies just because we expect every one to accept them as well. So we socially agree they represent some value. This is similar to what happened thousands of years before with gold, when people where willing to trade useful things (like food and clothes) for a mostly useless but shiny metal.

Digital currencies make the difference between abstract value (the value the certificate represents) and material value (the value of the material support the certificate is printed/minted on) of a certificate explicit. Considering that the certificate abstract and material values are not correlated, we can as well eliminate the material support obtaining just pure value representation. The value represented is now a free information electronically remembered. This is the same step performed by "electronic money" were value represented by traditional fiat currencies emitted by central entities (central banks) is electronically stored by third parties. Electronic money is not a form of digital currency, because it is just traditional currency digitally processed. That is why a digital currency requires a further step. The currency should be distributed, and exchanges between users should be direct, without the need of third party or other users intervention.

To be distributed a digital currency needs to decentralize two operations to the entire community:

- the minting of new coins;
- the validation of transactions exchanging value.

The problems of performing those two operations are usually solved with cryptography, that is why some digital currencies are also called "cryptocurrencies". The whole idea is to shift the users trust from a human controlled central entity to few reliable cryptographic functions.

It is usually said that with digital currencies every user is its own bank. That is because each user is the only one in charge of controlling and managing its own funds, it can check the validity of every other user transaction and can partake in the minting process. Any user can freely join or leave the community without costs and cannot be banned by others. Only the right owner can access its own funds and so no funds can be frozen or seized. Usually digital currencies are also compared to electronic cash, but to behave like cash additional properties must be fulfilled: non-traceability, non-reversibility, direct exchange. Among those properties we need to point out that non-traceability and point-to-point exchange together give anonymity.

The idea of digital currencies is not new, the paper "Blind signatures for untraceable payments" by David Chaum of 1982 [Chaum 1983] can be considered its manifesto. Chaum was the first to propose the use of cryptography to validate transactions rather than protect them. He also founded a commercial society called DigiCash in 1990 to promote this new cryptography secured currency, but the currency was based on a centralized entity and the project failed. The first true digital currency came out in 2009, Bitcoin[1] [Nakamoto 2008].

[1]In this paper we will follow the established convention to use "Bitcoin" with a capital 'B' to represent the protocol (and so the related technical aspects) and "bitcoin" to represent the currency, the community and all the aspects related to its real world use.

Bitcoin has been without doubt the first digital currency to gain wide mass media coverage and widespread popularity among the broad public of non specialists. Nowadays most of the academic studies published about digital currencies are about Bitcoin, while outside the academic community it is seen as the one and only cryptocurrency (the concept of "digital currency" and "Bitcoin" are often seen as the same). Allowing Bitcoin to gather all the attention of the broad public about digital currencies is a dangerous mistake. Currently bitcoin does not fulfill some of the promises of a true digital currency (most of all scalability, anonymity and decentralization), and so a (maybe inevitable) bitcoin failure could result into an unjustified lack of trust in the concept of digital currency itself.

As we have already explained, designing digital currencies introduces new problems unknown in traditional currencies design. The most notable problem is called "Double spending". Preventing double spending means preventing users from spending the same funds in more than one valid transaction. This problem does not exist in a traditional setting because each user account is controlled by a centralized entity (bank) who has full control over user funds. It is the central entity the one tasked of issuing transactions, not the user (as happens with digital currencies), and so there cannot be double spending attempts by the users (at least not without entity errors). In a distributed scenario each user is tasked with creating its own transactions and so we need to define a method that allows users to reach a consensus on which transaction is to be declared valid in presence of multiple conflicting one (to prevent double spending). Bitcoin double spending solution is to allow every user to know all the previous validated transactions of all users. The entire history is needed because a user cannot recognize a transaction as a double spending attempt if it does not know the original transaction[2]. That means that the entire History of all transactions ever validated is publicly visible to everyone. This leads to a clear threat to users privacy if identities of users involved in transactions are not protected enough.

In this paper we focus on Bitcoin because of its dominant position. It is interesting to verify if the current Bitcoin protocol fulfills its promises as a digital currency, and it is our intention to focus especially on users privacy protection (without ignoring the other interesting research topics). Since the very start [Nakamoto 2008] users anonymity was not Bitcoin main goal. This has brought to a major flaw in the project, which protects user anonymity only with the use of pseudonymity, with the hope of breaking ownership linking between addresses and real world identities. Any cryptocurrency who wants to scale to world acceptance needs strong privacy assurances, which nowadays Bitcoin seems to fail to give. Our work will concentrate on showing some Bitcoin privacy shortcomings.

The rest of this paper is organized as follows. In Section 2 we review the basics of the Bitcoin protocol. We will then focus on three of its main research topics, presenting the state of the art on fraudulent mining techniques in Section 3, double spending attacks in Section 4 and Bitcoin anonymity in Section 5. In Section 6 we draw our conclusions.

2 Bitcoin Protocol

Bitcoin is the first cryptocurrency to have reached worldwide popularity and adoption but it is not the first digital currency and its core ideas were already available ten years before its announcement. The first cryptocurrency attempt is considered to be [Chaum 1983] by

[2]The transaction whose funds are attempted to be spent for the second time.

David Chaum, where the author introduces blind signatures to "allow realization of untraceable payments systems which offer improved auditability and control compared to current systems, while at the same time offering increased personal privacy" [Chaum 1983]. In 1997 the cypherpunk cryptographer Adam Back invented the first proof-of-work called hashcash [Back 2002, Back 1997] as a DoS countermeasure applied to e-mail spamming. In 1998 Wei Dai released b-money [Dai 1998], an anonymous digital currency employing proof-of-works, and at the same time Nick Szabo invented bit-gold [Szabo], which used proof-of-work schemes to simulate gold digging hardness and had a controlled inflation rate.

Bitcoin is a cryptocurrency and that means that it cannot be seen just as a distributed protocol. The bitcoin real world community influences its development as much as its original design principles. Hereafter we will concentrate on the technical topics of the protocol but studies are also available on the more social aspects of bitcoin. Some papers have studied the bitcoin community and ecosystem like [Roio 2013, Holdgaard 2014] and [Christin 2013] where is presented the real world use of bitcoin in the Silk Road dark web marketplace. Even the FBI has produced its own record about bitcoin [FBI 2012], showing concerns about its use as payment for illegal activities. More works have been done studying bitcoin economy [Litecoin 2013, DRAVIS 2014, Iavorschi 2013, WSBI 2014] including financial groups assessments for speculative purposes [Woo et al. 2013, Normand 2014, Sablik 2013]. Works has also been done to try to explain the bitcoin exchange rate, like in [Luther and Olson 2013] where a correlation is found between the first 2013 price spike and Cyprus bailout. Today the biggest obstacle for serious business ventures in entering the bitcoin economy has been the lack of a clear legal definition of cryptocurrencies. Some guidelines have been released by international organizations [BCE 2012, FINCEN 2013, BI 2014, IRS 2014] but still every jurisdiction apply different legal frameworks and tax rules to digital currencies [Elwell et al. 2013, WSBI 2014, Regulation 2014, Marian 2013, BBVA 2013].

In the following we will explain the basics of Bitcoin protocol. Do note that there is no "official" protocol specification available for Bitcoin, so we refer only to the main well defined basic topics of the protocol as originally presented in [Nakamoto 2008], updated when necessary with the information available from the official bitcoin wiki [WIKI], bitcoin improvement proposals [BIP] and official client implementation [BITCOIN CORE].

2.1 Addresses

To enter the bitcoin economy each user only needs to generate a new asymmetric cryptography key pair. The public key is used to obtain an address to send and receive payments, while the private key is needed to prove that the user is the true owner of that address. The private key is used as proof of ownership, because each transaction spending funds contained in the corresponding public key (address) must be signed with the correct private key to be accepted. That is the only ownership proof available, so ownership of an address just means knowing the corresponding private key.

The public key cryptography used by Bitcoin is ECDSA [SEC2 2000], so the private key is a random 256 bits long number and the corresponding public key is 512 bits long. But each user does not advertise its public key as it is. It firstly hashes it with SHA-256 [SHA] to obtain 256 bits on which it applies RIPEM-160 [Preneel 1997, Dobbertin et al. 1996] to obtain 160 bits. Those bits are encoded with the purpose built Base58Check encoding to obtain an ASCII representation which is used as the users public address. The Base58Check encoding is a purpose built encoding containing a check-sum (the first four bytes of the double SHA-256 hash of the public address) and avoiding the use of similar characters symbols. The public address can be advertised and must be know to all those who want to send payments to it.

We observe now that it is not possible to learn the public key of a user from an address because the hash functions used are cryptographic and so irreversible. So the address is used in place of the public key to increase the secret key security because it restrict any attacker who tries to recover the private key from the public key to a limited time window. The public key cannot be kept secret forever, because it is necessary to verify a signature (necessary to make a payment). That is why the public key is attached to any transaction spending funds from the corresponding address. So an attacker has to wait for this transaction in order to learn the public key. The private key is necessary to spend the funds contained in the corresponding address, but the attacker can try to recover it only after a transaction spending those same funds is revealed. This means that the attack can only be attempted during the time that passes from when the transaction is announced to when it is validated (and so the funds are spent and cannot be stolen anymore). The security improvement of this design can be better appreciated considering the example of post-quantum cryptanalysis [Bernstein et al. 2009].

The up to date best know quantum algorithm to break cryptographic hash functions is the Grover search algorithm [Bernstein et al. 2009], which gives a quadratic speedup over classical algorithms. So this algorithm allows to halve difficulty of hash functions and symmetric cryptography breaking in a quantum scenario. A cryptographic hash function requires 2^n computations to be inverted and $2^{n/2}$ computations to find a collision on classical computers, so with a quantum computer we will need $2^{n/2}$ and $2^{n/4}$ computations respectively, so the problem remains exponentially intractable. The Shor quantum algorithm [Proos and Zalka 2003] instead allows to solve in polynomial time the discrete logarithm problem. ECDSA security is based on this problem intractability, so public key elliptic cryptography would not be secure anymore.

As we have seen before, the public key is revealed only when a payment is issued, on the contrary to receive payments only the address is known. Hash security is reduced but not compromised by quantum computers (it would be enough to simply increase the size), so a quantum attacker would not be able to learn the public key from the address. But once it knows the public key it would be able to find the correct private key in polynomial time. So if public key were used instead of addresses the protocol would be compromised. Using addresses allows to restrict the quantum attacker to a limited time frame (dependent on validation time) to try to find the private key. If the validation time is small enough the attack would be fruitless because the attacker would always be looking for private keys protecting funds already spent. So the only effect of a quantum computer would be to force users to use each address (and so each key pair) only once. We note that even if the validation time is significant enough to allow the attacker to find the secret key in time it would anyway be at a disadvantage. Knowing the private key, the attacker could create a different valid transaction sending the victim funds to an attacker controlled address. But this transaction will be in conflict with the victim original transaction and so only one of the two would be validated. Following the protocol rules the first transaction to be relayed on the network is the one with higher probability of been accepted (the exact probability depends on the percentage of validator nodes reached before the other transaction). The time spent from the attacker to find the private key would result in a propagation advantage of the honest transaction which will then have a higher chance of being accepted than the fraudulent one.

The attacker could improve its chances by combining the private key discovery with a sybil attack, to isolate the victim from honest nodes providing an alternative reality to the victim. The attacker would listen to victim transactions without forwarding them to honest peers, and forwarding its fraudulent transaction instead once it is ready. More generally relying on a broken public key algorithm would be pointless per se, so it would be necessary anyway to use a quantum resistant asymmetric cryptographic scheme (like lattice cryptography).

A side effect of ECDSA security is that if a private key is lost it cannot be recovered. So any funds stored or sent in the future to the corresponding address would be lost forever. We also note that creating new ECDSA pairs (and so addresses) is not expensive at all and so each user can create and use different addresses. This leads to the use of pseudonyms, which means that each address is a user alias without any kind of information linking it to the user or other addresses created by the same user. Pseudonymity is the only (weak) anonymity protection in Bitcoin. To improve transactions anonymity is recommended to create a new address for every transaction. This is not computationally expensive but it can lead to an address management problem, if the number of addresses keeps increasing. For users with high daily transaction volumes (such as exchanges and merchants) issuing a new address each time would lead to an overwhelming amount of addresses. To simply discard empty old addresses could be dangerous, because someone could still send payments to them, resulting in forever lost bitcoins.

2.2 Transactions

A transaction is a funds exchange between addresses. Transactions are multi input, multi-output, it means that a transaction may have more than one input (address from which funds are withdrawn) and more than one output (address in which funds are stored). Each transaction completely transfers funds from the inputs to the outputs (no change is left in the input addresses). So to keep the change (difference between input sums and the sum we actually want to pay including fees) a user should include a payer owned address between the outputs. As we already stated, transactions are the only mean to manage funds, so funds can be divided or aggregated only by being spent. That is possible because a transaction involves addresses and not users and every user can have different addresses, so a user can use a transaction to split, merge or move funds between its own addresses. Transactions aren't used only for payments, but also for funds management.

A transaction can also specify a voluntary fee to cover the expenses of the validation process (that we will explain better later). If the sum of input values exceeds the sum of output values[3] then the exceeding value is considered a voluntary fee paid to the validator. Paying a fee is optional but it is considered fair practice and it can shorten the validation time of the transaction.

In a transaction each output can be seen as a couple (bitcoin amount, receiver address). On the contrary each input specifies where to withdraw the funds, so it does not indicate an address but instead the previous transaction (actually its hash) where the funds were stored

[3]If the reverse happens the transaction is obviously invalid.

in the address now used as an input. Funds are represented by a transaction chain showing the passage of those funds (split and merged) between addresses, validated at each step by the previous owner signature. In bitcoin transactions alone specify the entire state of the system. There is no "coin" exchanged between users, the coins are implicitly represented by the flow of value through transactions. We can informally state that a "coin" in bitcoin is its own "history".

2.3 Consensus

The digital signature guaranties that only the rightful owner can spend its funds, but it does not prevent it from spending them more than once in different transactions. This is the so called "double spending problem". Bitcoin solution is to remember the history of all the past transactions to determine the actual owner of every fund at each given time. In case of conflict between different pending transactions it is necessary to reach a distributed consensus on which transaction to add to the official history. The history is maintained in a distributed database of all the valid transactions ever happened. This database is a distributed timestamping service because it fixes transactions in discrete time instants, allowing for a partial chronological ordering. The database is called blockchain because transactions are grouped in blocks linked in a chainand[4]. The linking between blocks is achieved by saving the hash of the header of the previous block in the next block header. To make each block header (and so its hash) dependent from all transactions contained in that block, in the header is included the root of the (implicit) merkle tree [Merkle 1987] built from the block transactions hashes.

2.3.1 Proof of Work

It is necessary to reach a consensus to choose which block (and so which transactions) to add to the chain, because there could be incompatible transactions caused by a double spending attempt. The idea is to let every user (who wants) vote using its computational power. It was chosen the one CPU one vote model instead of the one user one vote to protect the protocol from fake identities (easily deployable in an anonymous P2P network). To create new users is cheap while to dedicate computational resource is not. So the one CPU one vote strategy gives proof of a true commitment. A user participating in the validation process is called miner and the process itself is called mining. Bitcoin uses Prook-of-Work (PoW) [Dwork and Naor 1992] to implement the voting process through computational power.

Finding a proof-of-work means finding the solution of a computationally intense cryptographic puzzle to prove that some amount of effort was spent. The puzzle used should be asymmetric, which means that it should be computationally difficult to find a solution, but, given a solution, should be computationally easy to verify it. It should also allow to adjust the difficulty (of finding the solution, not of verifying it), and it should be dependent from some parameter to allow to have different puzzles with the same difficulty. This is needed to make the solution effort independent of any a priori computation.

In Bitcoin validating a block means finding a hashcash [Back 2002] type proof-of-work with double SHA-256 of the block header. It means finding a Nonce value so that the double SHA-256 of the block header is less that an established Target value. The Target value is automatically updated every 2016 blocks (approximately every two weeks) considering the

[4]It is actually a tree because of chain forks, as we will see later.

computational power of the entire network (estimated by the average time passed to validate a block) in order to keep the average validation rime of a new block around ten minutes.

Solving this type of proof-of-work is equal to a hash partial inversion, but, because the hash function chosen is cryptographic secure, the best known method is a brute force attack. This means that the best method is trying different random nonces until one satisfies. So the average time spent depends only on the computational power (more precisely the hash power) used. This cryptographic puzzle is a PoW because it is computationally expensive to solve but constant to verify (requires only one hash computation), the difficulty is adjustable (changing the target value) and the puzzle is parametric, the parameter being the block header, so it is not possible to work on the puzzle without knowing the block (no a priori advantage possible).

Bitcoin PoW is a probabilistic PoW, that means that finding a solution is computationally expensive on expectation. The randomization is important because otherwise the miner with the biggest computational power will always find the solution first and so would be the only one to produce blocks. The randomized PoW instead allows miners to have a probability to find each block proportional to the hash power dedicated [5].

To encourage mining and repay miners for their computational expenses at each new block is associated a reward collected by the block finder. The reward consists in the sum of all fees of the transactions contained in the block, plus a fixed amount of new coins. The fixed reward starts from 50 BTC and halves over time every 210 000 blocks until it will became zero at the 6 930 000th block, which is expected to be mined during the year 2140. The decreasing coin minting is adopted to reflect a deflationary money supply growth.

2.3.2 Distributed Consensus

Reaching consensus in a distributed system in the presence of byzantine faulty entities is a well studied problem. By byzantine faulty entities we mean we do not make any assumption on the kind of failure the entity can incur in. This allows us to cover any possible kind of situation and attacker, for example both random crashes and active malicious behavior are covered. In [Fischer et al. 1985] was proven that for deterministic asynchronous message reliable networks cannot exist a consensus protocol resilient in the face of even a single entity failure.

In [Miller and LaViola 2014] was proven that Bitcoin does not achieve consensus only with negligible probability when the adversary controls less than half the overall computing power, under the model of anonymous synchronous network. It was also shown that the protocol is scalable, in that the running time and total message bits are all independent of the size of the network, instead depending only on the ratio of faulty processes.

Note that the actual Bitcoin network is not synchronous, but as was shown in [Garay et al. 2014] it is enough that the network synchronizes much faster relative to the PoW solution rate. It means that the assumption holds true as long as the messages propagation time (currently in the order of seconds [Decker and Wattenhofer 2013]) is much smaller than the expected time to solve a PoW (ten minutes on expectation by design). In [Garay et al. 2014] similar results as [Miller and LaViola 2014] are shown, showing a proof that a majority of honest miners will agree on a common prefix of the blockchain, growing in length over time. Even in the

[5] This is true as long as no single miner controls more than half of the total combined hashing power.

light of those formal results we should always remember that the use of monetary rewards introduces new problems in the study of consensus, because this kind of incentive can be seen differently by different entities and as such is difficult to be formally defined.

2.3.3 Other Kinds of Consensus

Bitcoin SHA-256 PoW is not the only possible consensus mechanism used in cryptocurrencies. Many altcoins derived from Bitcoin changed the algorithm used or the concept of PoW itself.

Currently mining is performed on ASICS devices purpose built with performances and power consumption orders of magnitude better than CPUs (and GPUs). This has driven out occasional miners and have transformed mining in a costly business with a high entrance cost, menacing decentralization. So there have been proposals to use PoW algorithms difficult to implement on ASICS devices. The ASIC-resistant algorithms proposed so far are memory intensive, which means that they require efficient access to big memories, feature lacking in traditional ASICS devices. The most famous one is "scrypt" [Percival 2009, Percival 2009b, Percival and Josefsson 2012] used by the most famous altcoin Litecoin [Litecoin a, Litecoin b]. Unfortunately ASIC devices built for scrypt are already available. Also the CPU/GPU mining inefficiency has as useful side effect botnet [Wyke 2012] protection. In [Huang et al. 2014] a survey of the Bitcoin botnet ecosystem of the time is presented. The paper shows how most of the botnets were directly participating in well known honest mining pools and how the increasing difficulty has reduced their profitability.

Another proposal was to try to make the process of solving a PoW to give useful results as a side effect [King 2013, Primecoin a, Primecoin b]. The problem is to find useful problems which are characterizable as PoWs, it means they should allow for proportional difficulty adjustment and no information should be pre-computable.

There have also been proposals to completely substitute the PoW scheme.

The most famous proposal is called Proof-of-Stake (PoS) [Peercoin, King and Nadal 2012, Houy 2014] and consists in proving to own a certain amount of currency. The idea behind is to prove to be involved in the currency. This is mostly done by destroying coin-age. The coin age of a sum of coins is defined as the product of the amount times the time since last transaction involving that sum. So coin age is directly proportional to the amount of coins and to the quantity of time those coins were not used. Then a special transaction (containing a self reward) is created from a user to itself spending those funds and so resetting their coin age, this transaction is a valid PoS (and may be used to validate a block) with probability inversely proportional to its coin age. The PoS is computed on only static data except for the timestamp dependent coin age, so it is independent from the computational power owned.

The main advantage of this approach is that it would be more difficult for an attacker to gain control of large sums rather than to buy specialized hardware. It would also be more expensive for it to perform disruptive attacks to the currency, because it would own a lot of that currency itself and so it would be the first to be damaged by it.

2.4 Forks

Bitcoin is proven to reach an eventual consensus (under the assumptions previously stated), it means that the network will agree on an ever-growing common prefix of the blockchain, but the most recent blocks could be different. When an honest miner finds a new block it announces it immediately but, due to message propagation latency, another miner currently unaware of the new block could find a different new valid block itself. So two different but both valid blocks coexist. In this case we say a "blockchain fork" has occurred. Each miner then votes on which block to deem valid by starting to look for the next block to add on top of the chosen one. It means that the miner vote consists in dedicating its computational power to increase the branch it deems valid. The protocol rule imposes to consider valid only the longest branch, where "longest" means the one with cumulative higher difficulty, and hence more computational power (so more votes) dedicated to its creation. Eventually one branch will grow longer than the other and every honest miner will start looking for new blocks only on top of that branch, *de facto* pruning the loosing one. So a temporary consensus is reached.

3 Fraudulent Mining

In the previous section we have explained how Bitcoin needs the majority of the computational power to be honest to ensure that a consensus is reached. The original idea [Nakamoto 2008] was that the reward system was an incentive for miners to remain honest. This, however, does not hold true in the face of two problems:

- It assumes that the BTC awarded are the only value precious to users, while in reality they alone might not reflect the entire real-world profit.
- It assumes that miners will gain an expected reward proportional to the percentage of computational power controlled, no matter how the miners behave. That is not true in the face of fraudulent mining techniques such as block discarding attacks.

3.1 Real World Gains

Miners are not just interested in the number of bitcoins they can gain, but rather in the real-value of those bitcoins. If a miner was to use known fraudulent techniques to gain more bitcoins while undermining at the same time the public trust in bitcoin, the exchange rate would drop, resulting in a smaller real-world gain for the miner. An example of this reasoning is often used to analyze the likelihood of a 50%+1 attack.

We talk about 50%+1 attack when the majority of the hashing power is controlled by a single entity. This entity would be in total control of the blockchain because it could decide to be the only one to add new blocks. To do so it should just ignore blocks published by other miners. This will lead to frequent forks but the entity branch will always eventually win because will be always backed by the highest hash power. Being the only one to add new blocks, the entity would also be the only one to collect rewards and fees, making mining unprofitable for everyone else, thus driving out miners from the business and hence increasing the entity computational power percentage even more. Controlling the chain the entity could also decide which transactions to include in the blocks, for example demanding high enough fees or just blacklisting some addresses. It could also delete old transactions, voluntary forking the chain before the block containing them (called history revision attack), allowing it to perform double spendings. The main argument against this attack is that being the only one to collect new coins, the controlling entity would also be the one to suffer the most from a bitcoin exchange rate drop. This means that the entity would be encouraged to behave benignly to avoid a drop in public confidence in the protocol that would crash the coins value. Also the miner would be the most exposed in a bitcoin crash due to the huge investment sustained to buy the hardware necessary to reach hash power majority which would became useless and worthless.

Another example of a not-reward driven/compliant strategy would be when the miner is not driven by the value of bitcoin earned, instead it wants to achieve some other real-world gain. For example an attacker would like to perform a 50%+1 attack because it has strong economic interests in a rival cryptocurrency and wants to try to scare users off bitcoin. It also might want to attack bitcoin just to manipulate the exchange rate, to gain an economical advance other than just block rewards. An example of those kind of attacks is the so called "Goldfinger attack" [Kroll et al. 2013]. In this attack the aim of the attacker is to destroy bitcoin, and it is willing to loose money to reach this goal. In this example the base assumptions ruling out a 50%+1 attack (anti-profitability and long run losses) would not hold anymore. This attack is also more effective and difficult to model because it relies on lack of users trust in the system which is a difficult parameter to model. A Goldfinger attacker could try to perform the attack even if it controls a huge percentage of the computational power, but still less than half of it. With the current six confirmation wait to accept transactions an attacker with 40% of the computational power will pull a successful history revision attack on a freshly confirmed transaction with a 50% probability, while an attacker with 30% of the total hash power will succeed in the same attack with a 13% probability. It might not seem much, also considering the enormous cost necessary to control such high percentages, but a smart attacker could announce to control the majority of the hash rate and then use some successful attacks to prove it. If performed right the confusion and disruption created could be enough to drop the exchange rate, thus making mining less profitable and so driving out honest miners. This will increase the attacker percentage, further strengthening its position and the disruption caused until the attack is successful.

3.2 Block Discarding Attack

The reward system was thought [Nakamoto 2008] to be sufficient to encourage miner to behave honestly, under the assumption that each miner gains were only directly proportional to its percentage of the total hash power. Unfortunately this assumption does not hold true in the presence of fraudulent mining techniques, while the contrary holds true instead. Profit driven miners are rationally incentivized to use a fraudulent mining technique.

The underlying idea behind the fraudulent mining techniques under the family of "block discarding attack" [Bahack 2013] is to force the honest miners to waste resources trying to find new block on branches that will later be pruned. To do so, the attacker keeps secret its found blocks and reveals them only to cancel honest miners efforts. The attacker tries to build a secret branch longer than the public branch, leaving the honest miners to work on the public shorter one. This way when the attacker publishes its secret branch it will be longer and thus replace the public honest one. The aim of the attacker is to make the honest miners do useless work, thus wasting their computational resources. This way the total resources dedicated to useful work are lowered and hence the attacker percentage (of block founds) is artificially raised. The simplest example of this technique is called "selfish mining" [Eyal and Sirer 2013]. The proposed technique is the following:

- If the fraudulent miner knows no secret block it starts mining on the public chain exactly as honest miners do.
- When the secret miner finds a new block it does not publish it and instead it keeps it secret. Then it starts mining its next block on top of this secret block. This way it can potentially find a lot of secret blocks and keep building its secret branch.
- When an honest miner finds and publishes a new block, the fraudulent miner can be in four different situations:
  - It has not any secret block. Then it behaves as an honest miner and simply accepts the new block starting mining on top of it.
  - It has only one secret block. Then the rogue miner immediately publishes its secret block to try to win a network propagation race against the honest block. Every miner should mine on top of the first block received if it receives two new different blocks at the same height in the chain. In this case the miner has some probability (network latency dependent) that the next block found is on top of its block, in which case the honest miners work is effectively wasted. If a new block is instead found on top of the honest miner block the fraudulent miner block is discarded and no reward is earned by it. Note that each honest miner will mine on top of the first block received, so, depending on network latency, some honest miners will be helping the fraudulent one, mining on top of its block.
  - It has two blocks kept secret. Then the fraudulent miner immediately publishes its two blocks, creating a fork before the new honest block or adding them to its previous fork. Its branch now will be the longest and so the new honest block will be discarded. Then the rogue miner restarts the process from the beginning.
  - It has three or more blocks kept secret. The rogue miner immediately publishes its older block, creating a fork or adding it on top of its previous fork. Now the fraudulent public branch and the honest one are long the same so the honest miners will work on top of one of those two, but any block they found will be useless because the rogue miner already knows at least two more valid blocks on its fraudulent branch and so it will eventually publish those pruning the honest ones.

The first observation about this technique is that it is counterintuitive. A miner is supposed to immediately publish new blocks to collect its reward before some other miner publishes a new block canceling the first miners efforts. The fraudulent miner strategy is the opposite. It accepts the risk of loosing its certain reward for new blocks hoping to make the honest miners loose more rewards in useless computations. We also note that the rogue miner has a smaller hash power percentage than the honest miners combined, so its secret branches will eventually be surpassed from the honest miners. This is why it never tries to catch up with a longer branch (which would be probabilistically impossible for it), and as soon as an honest branch takes over, it starts to mine on top of it.

If we follow the calculations presented in [Eyal and Sirer 2013], calling α the percentage of computational power controlled by the rogue miner and γ the estimated percentage of miners to accept the rogue block in the event of a new block network race, we find that the relative profit of the fraudulent miner is not α as expected, but

$$\frac{SelfishMinerGain}{SelfishMinerGain + HonestMinersGain} = \frac{\alpha(1-\alpha)^2(4\alpha+\gamma(1-2\alpha))-\alpha^3}{1-\alpha(1+(2-\alpha)\alpha)}$$

This is because the total reward is less than one because some computational power (of both honest and rogue miners) was wasted on useless computations:

$$SelfishMinerGain + HonestMinersGain = \frac{\alpha^3-2\alpha^2-\alpha+1}{2\alpha^3-4\alpha^2+1} < 1$$

$$if \quad 0 < \alpha < 1/2$$

The relative profit could be greater or less then α, depending on α and γ, in particular

$$SelfishMinerGainPercentage > \alpha \quad \textit{iff} \quad \frac{1-\gamma}{3-2\gamma} < \alpha < \frac{1}{2}$$

So the selfish mining strategy is convenient if $\alpha > (1-\gamma)/(3-2\gamma)$. Not only the relative profit is greater than the honestly expected one, but it also grows more than linearly when α increases. Than means that more computational power the rogue miner controls, more unfairly greater its revenues will be. This leads to a scenario in which the rogue miner is incentivized to recruit outside miners to increase its profit, and the outside miners are encouraged to join the rogue mining pool because of the expected higher relative revenues. So as soon as a fraudulent miner passes the profitability threshold (dependent on γ), it will start to attract other miners increasing its revenues until the rogue pool reaches the majority of the hashing power. At this stage the pool controls the chain and it will no longer need to use the selfish mining technique, nor it will accept new members.

We note that the only thing a fraudulent miner could do to increase the effectiveness of the attack is to increase γ. This is theoretically possible if the rogue miner succeeds in being connected directly to all the other miners. In the real network however it is not so simple. Miners usually access the network through one or more gateway nodes and they try to remain as hidden as possible. This is because miners node are the most valuable targets for DDOS attacks. Miners also adopt side channel techniques to communicate between each other to try to reduce the block propagation latency and hence the amount of resources wasted on old branches.

Some doubts about the claims in [Eyal and Sirer 2013] are stated in [Courtois and Bahack 2014]. The authors state that the rogue strategy profits are only studied in the presence of only one rogue pool against an honest majority, no study has been done about more dishonest miners contemporary adopting fraudulent techniques. Also the idea that the rogue miner should accept outside miners lured in by higher profits to grow the rogue pool until it reaches a majority is not so applicable in practice. The first problem is that the expected profit is theoretically higher than the honest pools, but it can still be irrelevant for real world applications. Also setting up a pool leaves the attacker open to attacks itself. First of all the pool members would notice that the blocks are being kept secret and so they could expose the attacker, also big miners could refuse to partake in a rogue pool scared of the negative popularity effect it could have on the trust in bitcoin (and so on the exchange rate). Finally, the new members could perform attacks from the inside of the attacker pool (like the block withholding attack explained later), to achieve even higher gains than the attacker.

3.3 Block Withholding Attack

This kind of fraudulent mining techniques is based on the mining pools share payment method. Bitcoin has seen a huge adoption of miners coalitions called mining pools. A mining pool is a centralized or distributed infrastructure comprising a set of miners working on the same PoW. In centralized pools the pool manager is the one in charge of distributing the revenues among the miners. Miners are not only payed if they find a block, but also for the effort spent n failed attempts. To keep track of the effective effort spent each miner sends to the manager not only valid PoWs with the current difficulty but also solutions with lower difficulties. This proves the manager that the miner is actually working at the POW but was not lucky enough to find a valid solution. The rewards collected by the pool from valid blocks mined are divided among all the pool participants proportionally to the computational power used, as proved by the lower difficulty solutions submitted. It is worth noting that joining a pool does not increase a miners expected profit, which is still proportional to the miners hash power (if it is an honest miner). The expected profit is instead slightly lowered because of the fees taken from the pool operator. The first advantage of joining a pool is a practical one, because the miner does not have to care about transaction listening and block creation, it has just to try to solve PoWs provided by the manager. The major economical advantage of joining a pool is that the expected variance is significantly decreased. The miner is expecting to be paid a little less regularly rather than collecting one big reward once in a while. The block withholding attack [Courtois and Bahack 2014] is based on the idea of collecting the shares participating in the pool without actually increasing the pool computational power. As we have already seen with block discarding attacks the aim of the attack is to fraudulently increase the rogue miner profit lowering the honest miners profits. This allows the rogue miner to gain a percentage of profits larger than its hash power percentage. To perform the attack the miner splits its computational power with two different purposes. The optimal split is to divide exactly in half the hash power. Half of the power is used from the miner to honestly mine for new blocks. The other half is divided among the other honest mining pools. The infiltrated attacker controlled hash power correctly tries to solve PoWs assigned from the honest managers and sends them the solutions with lower difficulty to collect the shares but never reveals (withhold) valid blocks found. Note that the found blocks are useless for the attacker because they are built from honest managers who would be the one to collect the rewards.

4 Double Spending

We have already explained that the Bitcoin consensus protocol is proven to reach consensus at any time only on a common prefix of the blockchain, the most recent blocks could be part of a fork during which no official accepted blocks are defined. During a fork two (or more) transaction histories exist at the same time, so the same funds could be spent twice, once on each history. Only one history will eventually prevail, deleting the conflicting transactions in the pruned branches. But if those transactions were accepted and the real world benefits had already been gained then a double spending attempt was successful. Bitcoin solution is to wait a fixed number of "confirmations" before accepting a transaction in the blockchain as valid. The number of confirmation indicates the minimum number of block the user has to wait to be mined on top of the block containing the transaction. This number is arbitrary set to six by default, but can be increased or decreased depending on the value exchanged by the transaction [Rosenfeld 2014]. Increasing the number of confirmations to wait increases the probability that the transaction will be included in the final valid transactions history, but it also requires the users to wait for more time. In fact on average a new block is validated every ten minutes, so a six confirmation rule already requires the user to wait one hour, on average, for every transaction.

This waiting time is too much for a lot of real world payments scenarios, where users can wait only seconds for a payment. This has brought to the adoption of “fast payments”. A fast payment is validated using rules which do not consider the blockchain. This obviously makes fast payments transactions vulnerable to double spending, because the blockchain is the solution adopted precisely to prevent double spending. The majority of fast payment double spending prevention techniques are based on a listening period on the network to detect double spending attempts [Bamert et al. 2013]. If a transaction reaches first the majority of the miners it will be mined in a block and so no double spending is possible. Because of the Bitcoin fast message propagation [Decker and Wattenhofer 2013] there is only a limited time window during which the transaction is still traveling and has not reached all the miners. This time window is what an attacker should exploit to pull a double spending attempt. The goal of the attacker is to let the majority of the miners to know its fraudulent transaction before the honest one without the victim noticing. Due to the gossip message flooding, if the attacker manages to make the fraudulent transaction to reach the majority of the nodes it will also reach with high probability the honest node, which will become aware of the attack. The attacker can use smarter techniques, like for example using the victim neighbors as a screen to isolate the victim from the network. It can also combine the double spending attack with more powerful techniques like a sybil attack, to feed to the victim a different fake reality. The victim can try to protect itself deploying listener nodes in the network or using a listener service. In the end the attack reduces itself to a stealth race between the honest and fraudulent transaction. This kind of attack is studied in [Karame et al. 2012].

If the attacker is also a miner it can perform a more powerful attack called “Finney attack” [Finney]. The attack is more costly because requires the attacker to invest a lot of computational power in the mining attempt, but it is also much more powerful because it renders any listening countermeasure useless. The base idea of the attack is that the attacker tries to mine a valid block containing the fraudulent transaction. As soon as it succeeds, instead of immediately publish the block as usual it broadcasts the honest transaction instead. The victim cannot know of the secret block and no conflicting transaction is ever published on the network, so it will accept the transaction. As soon as the transaction is finalized by the victim the attacker publishes the secret block, which is valid and so will be accepted by everyone, rendering the honest transaction invalid and so discarded. The double spending was successful.

The main advantage of the attacker is that it can first try to mine the block and only when it manages to mine it, it can launch a double spending attempt, this means that the miner has no hurry to find the block and so also miners with little percentages of the hash power can perform the attack. The main problem for the attacker however is that it has to wait for the victim to finalize the transaction. During the time which passes between the finding of the block and the victim acceptance some other miner could find a valid block which will render the secret block invalid. So the attacker is not only risking to see its double spending attempt failing, but it is also risking to loose any reward granted from the block. Usually the block reward is much higher than any sum in a transaction which is accepted as a fast payment, so the attack would not be economically rational.

5 Deanonymization

While studying bitcoin privacy we should always remember three important properties of the protocol:

- The entire transaction history is publicly available and persistent in the blockchain.
- Anonymity is enforced only through pseudonymity, which means that each user partake in the protocol only through an arbitrarily large number of addresses that carry no real-world information about the user (nor any information between themselves).
- Transactions are broadcasted to the network (mostly) by the creators (gossip), every user has to create is own transactions, so it is also the first one tasked with informing the other peers about that new transaction.

To analyze the effective anonymity provided by the protocol we should first formally define which properties are requested to enforce anonymity. Few works are available on this topic, the most notable being [Androulaki et al. 2013], where two different properties to model Bitcoin privacy are presented:

- Activity (address) unlinkability. It refers to the fact that an adversary should not be able to link two different addresses pertaining to a user of its choice [Androulaki et al. 2013].
- Profile indistinguishability refers to the (in-)ability of an adversary to reconstruct the profiles of all the users that participate in the blockchain. Profiles, here, consist of the set of addresses (address-based profiles) or set of transactions (transaction-based profiles) of Bitcoin users [Androulaki et al. 2013] .

Other efforts on formally defining anonymity properties are mainly present in works also presenting solutions to enforce those same properties, as for example [Miers et al. 2013, Ben-Sasson et al. 2014, Danezis et al. 2013].

We will now analyze the state of the art of the main known attacks on bitcoin privacy.

5.1 Network Listening

The most simple form of network listening is to just deploy one or more listeners in the network recording the message flow. This approach is not a threat to the network and allows for information propagation studies and basic topology findings [Bitnodes, Donet et al. 2014]. The most cited example is [Decker and Wattenhofer 2013] where transactions and blocks propagation times are studied and real network data observation are used to prove how block propagation delay influences blockchain forks.

The goal of network listening based attacks is to link network information with bitcoin addresses [Koshy et al. 2014, Reid and Harrigan 2013]. Obtaining so we have succeeded in connecting some real world information to some addresses. We note that by saying "network information" we do not just mean an IP. IP addresses are not enough information to correctly represent a single user when multiple users are behind the same NAT or employ an anonymization layer like TOR. We need a way to assign to each user a "network signature" to differentiate users. The most effective proposal was presented in [Biryukov et al. 2014] where the authors propose to use the "entry nodes set" as signature of each peer. The entry nodes set is the set of nodes (eight by default) a peer initially establishes its outgoing connections to. This set is not network dependent and sufficiently random to differentiate peers behind the same proxy.

The most studied network deanonymization attack is based on the assumption that the first node in the network to broadcast a new transaction is the creator of that transaction. The creator of a transaction is also the owner of the input addresses of that transaction, because in order to create the transaction it had to sign it with the corresponding private keys.

This assumption is true for the vast majority of transactions, but is not necessary true for some special transactions. An example is given by CoinJoin transactions [Maxwell 2013], which are single transactions involving multiple users. In that kind of transactions the inputs are not owned by a single signer.

In general, automated attacks could always be deceived by active users countermeasures, so any attack should always try to balance false positives with recall. There are in general two kind of attacks, the community attack and the targeted attack. In the community attack, the attacker mostly uses general rules to avoid as much false positives as possible, accepting a low recall rate, acceptable in the presence of a huge amount of data. The aim of the attacker is to break the profile indistinguishability property. In the targeted attack the attacker only considers a relatively small set of addresses and it tries to find at least one linking between one of those addresses and some real-world information. The attacker is trying to break the weaker privacy property of address unlinkability, trying to build an address chain of ownership. An example of attacker for the community point of view could be a researcher trying to gather general information on the community [Bitnodes, Donet et al. 2014, Meiklejohn et al. 2013, Biryukov et al. 2014, Miller et al. 2015], while an example of targeted attacker could be a law enforcement agency who want to find a link between funds involved in illicit activities and a real world entity [Meiklejohn et al. 2013, Ron and Shamir 2013, Reid and Harrigan 2013].

An example of network listening to link IP with transaction creators (and so addresses) is presented in [Koshy et al. 2014]. The deployed listener simply tried to connect to every network address advertised on the network, keeping alive the connection indefinitely [Bitnodes, Donet et al. 2014]. The listening period lasted five months from July the 24th 2012 to January the 2nd 2013, with a median of 2678 active connections per hour. The analysis was based on the standard assumption that the first node to rely a transaction in the network is its creator, but since no refined technique were employed the simple listener could not know if the first node it received the transaction from was also the first to release it on the network or was just propagating it. So the analysis was mainly relying on anomalous relaying behavior, which for example includes transaction being relayed by a single peer (and not by others because, for example, are non standard or double spending attempts) or transactions relayed more than once from the same peer (in the official client a peer sends a transaction again to a neighbor only if the transaction is not confirmed yet and the peer is the sender or a recipient in that transaction). In the end the study allowed to associate a probable IP to

1162 addresses over four millions transactions collected.

A more refined attack was explained in [Biryukov et al. 2014]. The technique is based on noticing that a peer is uniquely identified by its entry node set (as explained before). First of all the attacker tries to find the entry node set for a peer when the peer connects to the network for the first time. When the peer connects it will forward its own address to its eight entry nodes. So the attacker remembers the first nodes from whom it has heard of any new address. Those eight nodes will not be the correct entry node set because of network latency and random message forwarding, but with high probability there will be a lot of nodes in common between this attacker estimated set and the actual entry nodes set. Then the attacker considers the first n (greater than eight) nodes each transaction is first received from and tries to find an estimated entry nodes set correlated to those n nodes (for example with at least three nodes in common), in order to associate the transaction with the peer identified by that set. An attacker can increase the success probability by having a very high number of connections with each peer (in the paper is suggested to have at least fifty connections with each full node), because that increases the probability to be chosen as first node to relay a transaction to. With fifty connections to each peer and three tuple matching as correlation measure, the experimental results in the bitcoin testnet show a success rate of 60% and in the real network the success probability is estimated as 11%. We nevertheless remark that the attack presented is really invasive and disruptive on the real network. Each node keeps by default a maximum of 125 active connections, so an attacker establishing 50 connections to each node would consume almost half of the connection slots available, resulting in a quite heavy DoS on the network. A main advantage of this technique is that it allows for the first time to differentiate nodes behind NAT or firewalls (or in general nodes not accepting incoming connections) introducing the entry set recognition.

In [Biryukov et al. 2014] a technique to exclude TOR (or others similar anonymity proxies) nodes from the bitcoin network is also presented. The basic idea is to ban TOR exit nodes relaying malformed bitcoin data through them to have them blacklisted by honest nodes as DoS countermeasure. This allows an attacker to prevent honest users to connect to the bitcoin network via TOR, but the attack is highly detectable. In a subsequent paper [Biryukov and Pustogarov 2014] the same authors show how using Bitcoin over Tor opens new possibilities for man-in-the-middle and sybil attacks. In [Biryukov and Pustogarov 2014] the authors also present a smart fingerprinting technique to recognize users, based on attacker chosen network addresses injected in the peers network addresses lists.

5.1.1 Topology Discovery

A network listener attack always relies on one or more listeners deployed on the live network, with as many connections to honest peers as possible. The bitcoin network is a pure P2P network with no topology overlay. This means that the only information nodes have about the network topology is the set of their neighbors. Nodes also do not exchange directly any information about their neighbors set, so topology cannot be easily learned. Of course knowing the topology will highly benefit a network listening attack as well as other kinds of attacks, so some research has been directed in topology discovery attempts. The two most promising ideas are presented in [Miller et al. 2015] and [Biryukov et al. 2014].

In [Miller et al. 2015] the authors present a topology discovery technique based on network addresses timestamps. The attacker needs a direct connection to each node it wants to find the set of neighbors because it needs to directly send messages to it (known network addresses list requests). The attacker then asks for as many network addresses known by the target as possible. The underlying idea is that the protocol rules dictate a node to age the timestamp of all the nodes it knows about without being connected to them and keep unchanged (or up to date) the timestamps associated to directly connected nodes. So, by analyzing discrete two hour differences in timestamps, the attacker can obtain the set of neighbors of each node. We now should point out that the technique worked at the time of the paper but is based on protocol rules of the time who may change (and indeed have changed today). This kind of exploit is based on current implementation rules and not on immutable protocol principles. The technique presented is a good example as today is not usable anymore, with version 0.10.1 of the default client the timestamp policy was changed rendering this attack unusable. In the paper [Miller et al. 2015] is also present a more general attack to discover mining pool influential nodes, called by the authors *decloaking*. A mining pool usually interfaces to the network through one or more gateway nodes, nodes tasked with collecting transactions and other miners blocks and relay the newly found pool blocks. The gateway nodes behave as a normal full node and it is important for the pool that they remain anonymous. This is because if a gateway node is discovered it can be targeted with DDoS or Sybil attacks, damaging the pool revenues. The technique presented consists in sending a different marker transaction to each node we are connected to, each transaction must be conflicting with the others so that only one can be accepted in the chain. Then we check the blockchain to see which one was validated. Repeating this experiment we can find the nodes that most often get their marker transactions validated. Those are the nodes closer to mining pool gateways (or maybe are gateways themselves). This simple method was proven effective in the paper where is shown how the top 100 influential nodes found accounted for three quarters of the validated transactions.

In [Biryukov et al. 2014] is present another topology discovery techniques to learn full nodes (or any other type of node accepting incoming connections) degree and neighbors. The underlying idea is to send marker addresses (with a timestamp which allows it to travel only for few hops, hopefully one, under the current relaying rules) to a peer which will then forward some of them to its neighbors. If then the attacker queries the other nodes (neighbor candidates) for the marker addresses (the attacker learns most of the node known addresses and then check how many are markers) it can estimate the probability that they are neighbors of the target node (based on the number of known marker addresses).

5.1.2 Malleability

A side effect of a listening attack is that it also provides the attacker with all the live information circulating on the network which are later forgotten and never recorded in the blockchain. These live data can be very useful to extract more information from the network. An example of this approach is presented in [Decker and Wattenhofer 2014], where the authors use the live data to estimate the actual amount of malleability attacks in the network.

When a transaction is created it is signed and the signature is attached to the transaction but the signature itself cannot be signed. This means that anyone can modify the signature representation (just its representation, not its content) without invalidating it, in this case we say that the transaction was “malleated” [Decker and Wattenhofer 2014, Andrychowicz et al. 2013]. Usually transactions are represented by their hash which is used as the transaction id. If a user malleates a transaction it will change the transaction hash and so its id.

Transaction malleability can be used to perform particular kinds of double spending attempts. We have a double spending attempt because there are two different transactions released on the network at the same time, so there is a conflict, but an important point to remember is that the meaning of the two transactions is the same. Only the signature representation is modified, the addresses and amounts transferred are identical, so whichever of the two gets validated there will be no difference for the addresses balance involved[6]. This double spending attempt becomes dangerous if used against a bad designed software who relies only on transactions ids. Some third party software checks if a transaction issued by them is accepted on the blockchain only looking if the corresponding transaction hash is included in the chain and not the transaction itself. Exploiting this fact, an attack becomes possible. A fraudulent client can ask the id based service for a payment and then malleate the newly issued transaction as soon as it is released on the network by the service. The two transactions will conflict with each other and so only one will be included in the blockchain, starting a validation-race. If the original transaction is included the attack fails. If, instead, the malleated transaction is included in the chain then the attacker can complain with the service that it never received a payment. The service will check the chain for the transaction id it knows without finding it and so will believe the client. Of course if the service tries to publish the transaction again, even modified by increasing the fees, it will be always rejected because it will be seen from the other peers as a double spending attempt. This is because the funds were already spent in the validated malleated transaction. The attack is successful because the attacker has been paid while tricking the service in believing it has not. Eventually the service will emit a new transaction spending new funds to pay the fraudulent user who will so be paid twice (or even more times if it succeeds in malleating also this new transaction). The entire attack is based on the service kept balance history being different from the globally blockchain stored history, so the service does not know of payments already finalized by the network.

A big scandal arose when the biggest bitcoin exchange, Mt.Gox [MtGox 2014], declared bankruptcy bringing a malleability attack as motivation for the users funds loss. In [Decker and Wattenhofer 2014] a single highly connected listener was deployed in the bitcoin network to collect transactions live data, looking for malleability attacks. The results show how the Mt.Gox claims look difficult to believe as only 1.811,58 BTC were involved in malleability attacks from the analysis start until the service lock down (between 1/01/2013 and 7/02/2014). This example clearly explains the huge opportunities provided by live network data analysis.

5.2 Blockchain Deanonymization Attack

The most studied kind of deanonymization attacks is the blockchain analysis attack. The aim of this attack is to break the profile indistinguishability property by aggregating Bitcoin addresses in clusters, each representing a single user, using the historical information stored in the blockchain. The attack is usually strengthened by also using external information to

[6]The addresses and amounts are signed and so cannot be modified without invalidating the signature. The only parts of the transaction not signed, and so modifiable, are the signatures themselves.

link each address cluster with some real world information to deanonymize users. The attack can be divided in three steps:

- In the first step the attacker passively parses the blockchain to obtain a transaction graph. The transaction graph is a graph where each node represents an address used in the blockchain and each oriented edge connecting two nodes represents an atomic transactions involving those two addresses and can be weighted (for example with the transaction timestamp and value). With atomic transaction we mean that we split each multi-input multi-output transaction in more atomic transactions, each connecting one of the inputs with one of the outputs, so that each output has exactly one ingoing atomic transaction.
- In the second step the attacker performs a heuristic rules based clustering applied to the previous graph to obtain a so called "users graph". In this new graph the nodes are now sets of addresses all controlled by same user.
- The final step is the users graph deanonymization using external information to obtain an identities graph. The techniques used to perform the real world identity-user linking are various but always based on external information gathering [Reid and Harrigan 2013]. This can be done crawling the web to search for Bitcoin addresses used in posts or donations signatures. Of course voluntary disclosed information are less reliable and so should be accordingly weighted. A more reliable method is to directly interact with services and well known economic parties to learn their addresses through economic exchanges, as performed in [Meiklejohn et al. 2013]. This step can also be combined with a network listening attack to use the listening results to perform identity linking. Is worth noting that this step can also refine the results obtained at the previous step, further collapsing together user clusters.

The intermediate step is the most important and difficult one of the three. Addresses clustering is the step that breaks address unlinkability and hence should be carefully planned. The clustering is based on heuristic rules i.e. rules based on bitcoin users behavior and common practice observation as well as technical constraints. These rules are time dependent and not general and can lead to a high rate of false positives or negatives if not correctly tuned. Of course it is preferable to decrease false positives at the expense of an increase in false negatives to avoid flawed final results, but false positives are inevitable. Basing ourselves on general rules to model users behavior we inevitably misinterpret some uncommon use cases. This is why in the literature only two heuristic rules are always used and generally accepted.

5.2.1 Multiple Inputs Heuristic

The first heuristic proposed [Ron and Shamir 2013, Meiklejohn et al. 2013, Reid and Harrigan 2013, Androulaki et al. 2013, Ober et al. 2013] and the one considered the safest and most used is called "multiple inputs". The heuristic rule is the following:

"in a multi-inputs transaction all the input addresses belong to the same user"

Every transaction must be signed for every input, it means that to sign the transaction the private key of each input should be known. This implies that the signer knows all the private keys and hence it is the owner of all the input addresses. This first simple heuristic rule has been used with success in the past but it is no longer valid for all transactions. A new countermeasure has been designed to trick the "multi inputs" heuristic.

The first effective countermeasure proposed was CoinJoin [Maxwell 2013] which underlying idea is for a set of users to meet and collectively create a single transaction using their addresses. First several users agree to take part in a new transaction creation through a meet-up server or decentralized consensus. Then they agree on fixed sum input transactions to be joined in a single big transaction. This means that each user uses one input address with a fixed sum and a single new output address to collect its payment to itself. The transaction must be signed by all the users to be valid, so each user can abort the transaction at any time. This prevents thefts but leaves the protocol open to DoS attempts. An attacker can join as many transactions as possible and then always abort refusing to sign them. Another problem is that each information about other participants is shared among all the users creating a transaction together. This means that "internal linkability" is possible and other information, such as IP addresses, might be leaked. An improved protocol providing internal unlinkability called CoinShuffle [Ruffing et al. 2014] has been proposed, but it also suffers from DoS attacks, because it allows to detect a party that aborts but cannot prevent fraudulent parties from aborting. We also remark that the privacy offered by the common transaction is limited to the number of users taking part in that transaction, so we talk about k-anonymity. The anonymity set size can be increased chaining together more joined transactions, to allow the anonymity set to include all users partaking in all the joined transactions in the chain.

5.2.2 Change Address Heuristic

The second most used heuristic rule [Meiklejohn et al. 2013, Reid and Harrigan 2013, Androulaki et al. 2013] is called "change address heuristic". The heuristic rule is the following:

"the change address belongs to the same user of the input addresses"

The rule looks simple but the main problem is to correctly identify which address is actually the change address. This means that the rule yields a lot of false positives and so must be refined to decrease them (at the expense of the recall rate). The refinements are heuristic sub-rules themselves based on users behavior and current wallets software implementation. Some of these sub-rules adopted in [Meiklejohn et al. 2013] are presented here. The address c is a change address in transaction t if and only if:

- t is not a coinbase transaction
- $\mid outputs(t) \mid > 1$
- $inputs(t) \cap outputs(t) = \emptyset$ (no self change)
- there is no address already labeled as change address or belonging to the inputs owner in $outputs(t)$
- it is the first time c appears in the blockchain and there is no other address in $outputs(t)$ satisfying this condition

5.2.3 Countermeasures

We have already seen CoinJoin [Maxwell 2013] and CoinShuffle [Ruffing et al. 2014] proposals to trick the heuristic driven clustering attack, but more general techniques are used to try to protect against blockchain analysis. All of the techniques proposed try to break the ownership tracking among transactions.

The most simple techniques used are called obfuscation techniques [Ron and Shamir 2013] and consist in scrambling funds among user controlled addresses to confuse the attacker. The used techniques rely on well known randomized patterns, the most notable being splits and aggregations, peeling chains, and binary trees. All those techniques succeed only in creating confusion, without effectively breaking the funds ownership and so can be easily detected by a motivated attacker [Ron and Shamir 2013, Meiklejohn et al. 2013, Reid and Harrigan 2013].

A better countermeasure is to use a mixer. A mixer is a third party entity which collects standard-sized payments from customers directed to a set of input addresses and pays back the same clients at random times in the future using funds stored in addresses not linked to any of the input addresses by any transaction. This effectively breaks ownership tracking but leaves the user exposed to thefts from the third party mixer, which needs to be trusted. The mixer also knows the internal linking of users, allowing internal linkability, and users need to pay a fee to use the service. Laundering the funds through a chain of different mixers will increase the anonymity but also the cost and the risk of thefts. A review of mixing services is presented in [Moser 2013] where is shown as the BitLaundry mixing service does not actually break ownership tracking.

We should also note that also benign attacks and positive results are possible through deanonymization attacks. An example is theft tracking. In bitcoin transactions are irreversible and so theft can be publicly reported but cannot be reversed. The stolen funds can anyway be blocked if honest users refuse to accept them as payments. Of course strong evidences need to be presented in order to avoid DoS attacks with fake thefts. The deanonymization techniques allow also to keep track of the stolen funds preventing the thief from laundering them through obfuscation techniques [Meiklejohn et al. 2013].

5.3 SPV

To use the explained protocol a user needs to know the entire blockchain (or its compressed form). For occasional users who use bitcoin only to pay small sums through smart phones, downloading the entire blockchain is impossible. To cope with light weight users the Simplified Payment Verification (SPV) was defined [Nakamoto 2008].

We firstly note that the chain is used to check the validity of transactions received from other users and not to create new transactions, so a SPV user will create and broadcast transactions as everyone else.

When an SPV user needs to check for a transaction validity (containing for example a payment to it) it sends to full nodes a special SPV addresses request containing the addresses the SPV user is interested in. When the full nodes learn new transactions containing the requested addresses they forward the transaction to the user with the merkle tree of the block containing the transaction and the header of all the blocks until a certain depth in the blockchain. The merkle tree allows the user to check that the transaction is really included in that block, while the block headers allow the user to check that the block is part of the longest chain. It is worth noting that the block headers are constant in size and much smaller than the entire block (81 bytes against a variable size up to 1 MB). We also note that the user cannot check the transaction validity because it does not know the other transactions. For example it cannot check that the input used in the transaction are really present in the chain. But the user knows that the block containing that transaction is part of the longest chain and this means that the majority of miners have worked on top of that block and so they deemed the transaction valid.

Of course the fact that the SPV user lets the full nodes know which addresses it is interested in is a clear privacy problem. To avoid direct disclosure the SPV nodes send the full nodes only a bloom filter with a desired false positive rate, this allows the full nodes to check if an address matches with the filter without directly knowing the user addresses. Unfortunately a lot of information is leaked by those bloom filters, resulting in a big privacy issue for SPV users. In [Gervais 2014] the author shows that a lot of addresses can be recovered from one filter of a user with a little number of addresses. They also prove that an attacker can easily discover which different filters are created by the same user and that an attacker can learn a lot of addresses of a user if it knows at least two different filters of the same user. The SPV deanonymization can be combined with the previous techniques to strengthen the obtained results.

6 Conclusions

In this paper we have shown the core rules of Bitcoin protocol. We have also presented the state of the art on three main attacks on Bitcoin. First we showed the main fraudulent mining techniques employable by fraudulent miners to unfairly gain an advantage in the mining race. We then showed attacks aimed at attempting double spending attempts. Finally we presented the main threats to users privacy in Bitcoin.

References

[Androulaki et al. 2013] Elli Androulaki, Ghassan O. Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun, *Evaluating User Privacy in Bitcoin*, in Proceedings of Financial Cryptography 2013, 2013

[Andrychowicz et al. 2013] Marcin Andrychowicz, Stefan Dziembowski, Daniel Malinowski, and Lukasz Mazurek, *How to deal with malleability of BitCoin transactions*, arXiv:1312.3230v1, 2013

[Back 1997] Adam Back, *A partial hash collision based postage scheme*, http://www.hashcash.org/papers/announce.txt, 1997

[Back 2002] Adam Back , *Hashcash - A Denial of Service Counter-Measure* , 2002

[Bahack 2013] Lear Bahack, *Theoretical Bitcoin Attacks with less then Half of the Computational Power*, draft, 2013, http://eprint.iarc.org/2013/868

[Bamert et al. 2013] Bamert T., Decker C., Elsen L., Wattenhofer R. and Welten S. , *Have a snack, pay with Bitcoins*, In Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on (pp. 1-5). IEEE , September 2013

[BBVA 2013] BBVA Research, *Economic Analysis Bitcoin: A Chapter in Digital Currency Adoption*, Economic Watch Global, BBVA research, 2013

[BCE 2012] European Central Bank, *Virtual Currency Schemes*, October 2012

[Ben-Sasson et al. 2014] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, *Zerocash: Decentralized anonymous payments from Bitcoin*, in IEEE Symposium on Security and Privacy, 2014

[Bernstein et al. 2009] Daniel J. Bernstein , Erik Dahmen , Johannes Buchmann , *Post-Quantum Cryptography*, Springer, 2009

[BI 2014] Banca d'Italia, *Rapporto sulla stabilità finanziaria*, numero 1, maggio 2014

[BIP] https://github.com/bitcoin/bips

[Biryukov et al. 2014] Alex Biryukov, Dmitry Khovratovich, and Ivan Pustogarov. *Deanonymisation of clients in Bitcoin P2P network.* In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2014.

[Biryukov and Pustogarov 2014] Alex Biryukov, and Ivan Pustogarov. *Bitcoin over Tor is not a good idea.* arXiv preprint arXiv:1410.6079 (2014).

[BITCOIN CORE] https://bitcoin.org/en/version-history

[Bitnodes] *BITNODES.* https://getaddr.bitnodes.io/

[Chaum 1983] David Chaum, *Blind signatures for untraceable payments*, In Advances in Cryptology: Proceedings of CRYPTO '82, pp.199-203, Plenum Press, 1983

[Christin 2013] Nicolas Christin, *Traveling the Silk Road: a measurement analysis of a large anonymous online marketplace*, In Proceedings of the 22nd international conference on World Wide Web, WWW '13, pp. 213-224, Republic and Canton of Geneva, Svizzera, 2013, International World Wide Web Conference Steering Committee

[Courtois and Bahack 2014] Nicolas T. Courtois, Lear Bahack , *On Subversive Miner Strategies and Block Withholding Attack in Bitcoin Digital Currency*, arXiv:1402.1718v1, 2014

[Dai 1998] Dai, W (1998). "b-money". http://www.weidai.com/bmoney.txt Retrieved 15 September 2015.

[Danezis et al. 2013] G. Danezis, C. Fournet, M. Kohlweiss, and B. Parno, *Pinocchio Coin: building Zerocoin from a succinct pairing-based proof system*, in PETShop, 2013

[Decker and Wattenhofer 2013] Christian Decker, Roger Wattenhofer, *Information Propagation in the Bitcoin Network.* In 13-th IEEE International Conference on Peer-to-Peer Computing, 2013

[Decker and Wattenhofer 2014] Christian Decker, Roger Wattenhofer, *Bitcoin Transaction Malleability and MtGox*, arXiv:1403.6676v1, 2014

[Dobbertin et al. 1996] Hans Dobbertin, Antoon Bosselaers, Bart Preneel, *RIPEMD-160: A Strengthened Version of RIPEMD, Fast Software Encryption*, LNCS 1039, pp. 71-82, Springer-Verlang, 1996

[Donet et al. 2014] Joan Antoni Donet Donet, Cristina Pérez-Solà, and Jordi Herrera-Joancomartì. *The bitcoin P2P network.* In Financial Cryptography and Data Security. Springer Berlin Heidelberg, 2014. p. 87-102.

[DRAVIS 2014] Dravis Group, *Bitcoin, Digital Currency and the Internet of Money*, Dravis Group LLC, San Francisco, USA, 2014

[Dwork and Naor 1992] Dwork and M. Naor. *Pricing via processing or combatting junk mail*, In CRYPTO, 1992

[Elwell et al. 2013] Craig K. Elwell, M. Maureen Murphy, Michael V. Seitzinger, *Bitcoin: Questions, Answers, and Analysis of Legal Issues*, Congressional Research Service, 2013

[Eyal and Sirer 2013] Ittay Eyal, Emin Gün Sirer, *Majority is not Enough: Bitcoin Mining is Vulnerable*, Cornell University, technical report, ArXiv e-prints, novembre 2013

[FBI 2012] FBI, *Bitcoin virtual currency: Unique features present distinct challenges for deterring illicit activity*, tech. rep., Federal Bureau of Investigation, 2012

[FINCEN 2013] Department of the Treasury Financial Crimes Enforcement Network, *Guidance FIN-2013-G001 Issued: March 18, 2013 Subject: Application of FinCEN's Regulations to Persons Administering , Exchanging, or Using Virtual Currencies*, 2013

[Finney] https://bitcointalk.org/index.php?topic=3441.msg48384#msg48384

[Fischer et al. 1985] Michael J Fischer, Nancy A Lynch, and Michael S Paterson, *Impossibility of distributed consensus with one faulty process.* Journal of the ACM (JACM) 32, 2 (1985), 374–382, 1985

[Garay et al. 2014] Juan A. Garay, Aggelos Kiayias, and Nikos Leonardos, *The Bitcoin Backbone Protocol: Analysis and Applications*, IACR: Cryptology ePrint Archive 2014 Issue 765. http://eprint.iacr.org/2014/765, 2014

[Gervais 2014] Gervais A., Capkun S., Karame G. O., and Gruber, D. , *On the privacy provisions of Bloom filters in lightweight Bitcoin clients*, in Proceedings of the 30th Annual Computer Security Applications Conference (pp. 326-335), ACM, December 2014

[Holdgaard 2014] Lars Holdgaard, *An Exploartion of the Bitcoin Ecosystem*, www.Bitcoin-Expert.net, 2014

[Houy 2014] Nicolas Houy, *It will cost you nothing to "kill" a Proof-of-Stake crypto-currency [v.0.1]*, Lyon University, France, 2014

[Huang et al. 2014] Danny Yuxing Huang, Hitesh Dharmdasani, Sarah Meiklejohn, Vacha Dave, Chris Grier, Damon McCoy, Stefan Savage, Alex C. Snoeren, Nicholas Weaver, and Kirill Levchenko, *Botcoin: Monetizing stolen cycles*, In Proceedings of the Network and Distributed System Security Symposium (NDSS), 2014

[Iavorschi 2013] Mihaela Iavorschi, *The Bitcoin Project And The Free Market*, CES Working Papers – Volume V, Issue 4, pp. 529-534, 2013

[IRS 2014] Internal Revenue Service (IRS), *IRS Notice 2014-21*, 2014

[Karame et al. 2012] Ghassan O. Karame, Elli Androulaki, Srdjan Capkun, *Two Bitcoins at the Price of One? Double-Spending Attacks on Fast Payments in Bitcoin*, In Proceedings of the 2012 ACM conference on Computer and communications security, CCS '12, pp. 906-917, New York, USA, 2012

[King 2013] King, Sunny, *Primecoin: Cryptocurrency with prime number proof-of-work*, July 7th (2013).

[King and Nadal 2012] King, Sunny, and Scott Nadal, *Ppcoin: Peer-to-peer crypto-currency with proof-of-stake*, 2012

[Koshy et al. 2014] Philip Koshy, Diana Koshy, and Patrick McDaniel, *An Analysis of Anonymity in Bitcoin Using P2P Network Traffic*, Financial Cryptography and Data Security, 2014

[Kroll et al. 2013] Joshua A. Kroll, Ian C. Davey, and Edward W. Felten, *The Economics of Bitcoin Mining, or Bitcoin in the Presence of Adversaries*, in The Twelfth Workshop on the Economics of Information Security (WEIS 2013), Washington, USA, 2013

[Litecoin a] https://litecoin.info/Main_Page

[Litecoin b] https://en.bitcoin.it/wiki/Litecoin

[Litecoin 2013] *The Litecoin Cryptocurrency Mining Rush And How To Take Advantage Of It Today If We Act Quickly*, Presented by Nimue CryptoMining, LLC.http://www.nimuecrypto.com, 2013

[Luther and Olson 2013] William J. Luther, Josiah Olson, *Bitcoin is Memory*, 2013

[Marian 2013] Omri Marian, *Are Cryptocurrencies Super Tax Havens?*, Michigan Law Review First Impressions, vol. 112:38, pp. 38-48, 2013

[Maxwell 2013] Maxwell, *CoinJoin: Bitcoin privacy for the real worldCoinJoin: Bitcoin privacy for the real world*, bitcointalk.org, August 2013

[Meiklejohn et al. 2013] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage. *A fistful of bitcoins: Characterizing payments among men with no names*. In Proceedings of the 2013 Conference on Internet Measurement Conference, IMC '13, pages 127–140, New York, NY, USA, 2013. ACM.

[Merkle 1987] Ralph C. Merkle, *A Digital Signature Based on a Conventional Encryption Function*, In CRYPTO '87: Proceedings of the 7th Conference on Advances in Cryptology. 369–37, 1987

[Miers et al. 2013] Ian Miers, Christina Garman, Matthew Green, and Aviel D. Rubin, *Zerocoin: Anonymous distributed e-cash from bitcoin*, In Proceedings of the 2013 IEEE Symposium on Security and Privacy, SP '13, pp. 397-411, Washington, USA, 2013, IEEE Computer Society

[Miller and LaViola 2014] Miller Andrew and Joseph J. LaViola Jr. *Anonymous byzantine consensus from moderately-hard puzzles: A model for bitcoin*, 2014

[Miller et al. 2015] Miller A., Litton J., Pachulski A., Gupta N., Levin D., Spring N., and Bhattacharjee B. . *Discovering bitcoin's public topology and influential nodes*. 2015.

[Moser 2013] Malte Möser, *Anonymity of Bitcoin Transactions: An Analysis of Mixing Services*, in Proceedings of Münster Bitcoin Conference (MBC'13), 2013

[MtGox 2014] Crisis Strategy Draft, *MtGox Unredacted*, 2014

[Nakamoto 2008] Satoshi Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System*. 2008

[Normand 2014] John Normand, *The audacity of bitcoin - Risks and opportunities for corporates and investors*, J.P. Morgan Securities, 2014

[Ober et al. 2013] M. Ober, S. Katzenbeisser, and K. Hamacher, *Structure and Anonymity of the Bitcoin Transaction Graph*, Future internet, pp. 237-250, 2013

[Peercoin] http://peercoin.net/

[Percival 2009] Colin Percival , *scrypt: A new key derivation function Doing our best to thwart TLAs armed with ASICs* , Tarsnap, 2009

[Percival 2009b] Colin Percival, *Stronger key derivation via sequential memory-hard functions* In BSDCan '09: The Technical BSD Conference, 2009

[Percival and Josefsson 2012] C. Percival and S. Josefsson, *The scrypt Password-Based Key Derivation Function*, 2012

[Preneel 1997] Bart Preneel, Antoon Bosselaers, Hans Dobbertin, *The Cryptographic Hash Function RIPEMD-160, in RSA Laboratories*, CryptoBytes The technical newsletter of RSA Laboratories, a division of RSA Data Security, Inc., Volume 3, Number 2 — Autumn 1997

[Primecoin a] http://primecoin.org//

[Primecoin b] https://github.com/primecoin/primecoin/wiki/_pages

[Proos and Zalka 2003] Proos, J., and Zalka, C. *Shor's Discrete Logarithm Quantum Algorithm for Elliptic Curves*, Quantum Information and Computation, 3:317–344, 2003.

[Regulation 2014] Global Legal Research Directorate Staff, *Regulation of Bitcoin in Selected Jurisdictions*, The Law Library of Congress, Global Legal Research Center, 2014

[Reid and Harrigan 2013] Fergal Reid and Martin Harrigan, *Security and Privacy in Social Networks, chapter "An Analysis of Anonymity in the Bitcoin System"*. pp. 197–223. Springer, 2013

[Roio 2013] Denis Jaromil Roio, *Bitcoin, the end of the Taboo on Money*, version 1.0, Dyne.org Digital Press, 2013

[SEC2 2000] *Standards For Efficient Cryptography - SEC 2: Recommended Elliptic Curve Domain Parameters*, Certicom Research, Version 1.0, 2000

[Ron and Shamir 2013] Dorit Ron and Adi Shamir. *Quantitative analysis of the full bitcoin transaction graph.* In Financial Cryptography and Data Security, pages 6–24. Springer, 2013.

[Rosenfeld 2014] Rosenfeld Meni, *Analysis of hashrate-based double spending*, arXiv preprint arXiv:1402.2009 (2014).

[Ruffing et al. 2014] T. Ruffing, P. Moreno-Sanchez, and A. Kate, *CoinShuffle: Practical decentralized coin mixing for Bitcoin*, in ESORICS, 2014

[Sablik 2013] Tim Sablik, *Digital Currency New Private Currencies Like Bitcoin Offer Potential and Puzzles*, Econ Focus, Third Quarter, pp. 18-27, 2013

[SHA] NIST, *Description of SHA-256, SHA-384 and SHA-512*, http://csrc.nist.gov/groups/STM/cavp/documents/shs/sha256-384-512.pdf

[Szabo] Szabo, Nick. *Bit Gold*. Unenumerated. Blogspot. Archived from the original on 2011-09-22. Retrieved 5 December 2013

[WIKI] https://en.bitcoin.it/wiki/Main_Page

[Woo et al. 2013] David Woo, Ian Gordon, Vadim Iaralov, *Bitcoin a first assessment*, Bank of America Merrill Lynch, 2013

[WSBI 2014] WSBI, *Virtual currencies: passion, prospects and challenges*, http://www.wsbi-esbg.org/SiteCollectionDocuments/Virtual%20currencies_passion,%20prospects%20and%20challenges .pdf , October 2014

[Wyke 2012] James Wyke, *The ZeroAccess Botnet - Mining and Fraud for Massive Financial Gain* , Technical report, SophosLabs, 2012