Bitcoin a Peer-to-Peer payment solution

[Security Considerations]

Jean-Guillaume Dumas
University Joseph Fourier
Grenoble
Jean-Guillaume
.Dumas@imag.fr

Pascal Sygnet
Grenoble INP Ensimag
Pascal.Sygnet
@Phelma.Grenoble-inp.fr

Vincent Xuereb
Grenoble INP Ensimag
Vincent.Xuereb
@Phelma.Grenoble-inp.fr

1. INTRODUCTION

This paper is the conclusion of a second year project in a French engineering school Ensimag for more information see `www.ensimag.grenoble-inp.fr/` or `www.ensiwiki.ensimag.fr /index.php/4MMPCRYPTO_2013`. We will investigate a successful electronic currency, the Bitcoin using various papers[3][4][5][6]. In this article look into three different security issues:

- the first one is the heart of this system : forgery of a transaction;
- the second is different implementations introducing various flaws;
- the third is inequality of distribution in the network

2. A PEER-TO-PEER PAYMENT SYSTEM: BITCOIN

2.1 Traditional Banking

Electronic currencies have been a field of interest long before the internet was widely use around the world [3]. To be a good way of payment it should respect some specifications like anonymity of transactions, security of payment and the absence of trust needed between the involved parties.
A traditional electronic cash system (via internet or any kind of network) is based on a central authority **the mint**. This mint (or bank) is aware of all transactions, of the balance of each and every account in his own network and is responsible for security and anonymity of transactions.
To ensure privacy banks keep information only between the involved parties. The main advantage is simplicity of protecting a transaction. The only required information to spend money is a single identifier as banks have access to both balance of accounts and time of transactions. The entire intelligence (verification and issuing the keys) is transferred into the mint so users only need to remember their own key (& id).

A real problem of this system is that it relies on a single central authority. If it was to collapse every history of transactions and every amount of money would be lost without any chance of recovery. That is why people started to investigate different solutions like BitCoin (based on a peer-to-peer[1] network). Here is a summary of the specifications of five of the most famous electronic trading systems.

Table 1: Five Electronic trading systems

	Mint	Public Trans.	Anonymity	PtP
Bank	Yes	No	Yes	No
Ripple	No	Yes	depends	Yes
KARMA	distributed	No	Yes	Yes
PPay	user = Mint for every coin he generates	No	Yes	Yes
Bitcoin	No	Yes	Yes	Yes

In the rest of this article we will focus on one particular currency, the Bitcoin.

2.2 BitCoin specifications

Bitcoin is a peer-to-peer electronic currency system first described by S. Nakamoto in 2008 [4]. It is based on digital signature to prove ownership of a coin and an history of transactions publicly available to avoid double-spending. This history is shared using a peer-to-peer network and users agree on it using a proof-of-work system. This network has become very popular raising interest for its safety and its use [1][2]. In the following parts we will look into the specifications of Bitcoin.

2.2.1 Nodes

Each user in the system is a node and is identified by one (or numerous for anonymity reasons) Bitcoin address (an **ECDSA** public key hashed with **SHA-256** and then **RIPEMD-160**).

2.2.2 A coin

Keep in mind that only transactions are recorded by the network. On this regard a Bitcoin (or simply coin) is simply a chain of transactions, the first one being the creation of the coin.
A transaction is a chain of digital signatures. In order to

[1] or PtP

pay each owner signs a hash (using an **ECDSA** private key) of the previous transaction and the public key of the next owner, and puts this at the end of the coin to transfer. Each transaction is a transfer of propriety to a Bitcoin address from the current owner to the next one. **Everyone** can verify the chain of ownership of a coin by checking the signatures (using the public keys) as demonstrate in the figure bellow.

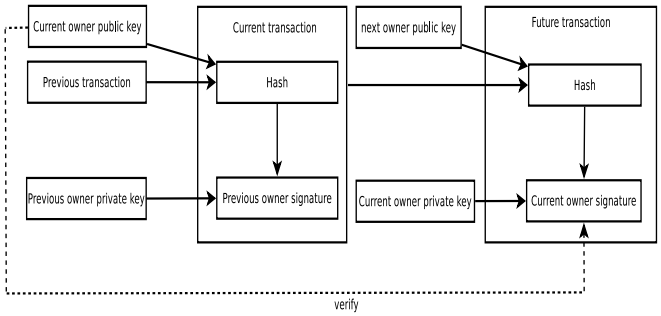

Figure 1: A "coin" in the Bitcoin system

2.2.3 *Double-spending and proof-of-work*

To avoid double spending the system keep a list of all valid transactions in a Blockchain. A block keeps track of transactions using a Merkle tree :

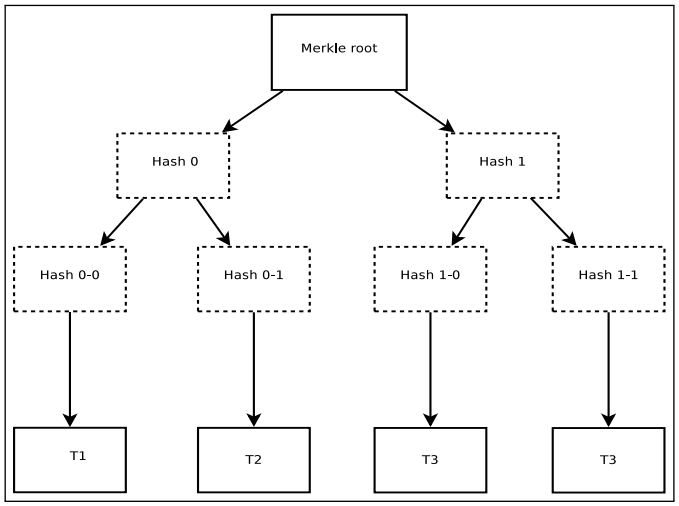

Figure 2: A Merkle tree with 4 transactions

Blocks are added in the blockchain using a **proof-of-work** system. The difficulty to add a block to the chain is what protects the bitcoin network from having forged transaction in the blockchain (thus considered to be valid).
Below is a schema of a block header which is used to chain blocks, a block header is composed of several fields :

- a timesamp used to monitor the rate at which block are created;
- a previous block header chains the block to the previous one;
- a Merkle tree root is a summary of transaction the block contains;
- a nonce, an integer with no other utility than to make the hash of the block header change when we change it.

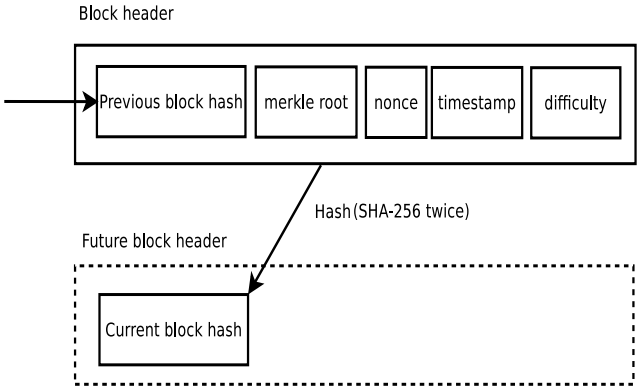

Figure 3: Part of the proof-of-work

When a user wants to add a new block in the blockchain the hash of the his block header must be lower than a specified target. The target is imposed by the network (the difficulty field in the block header is a representation of the target). The target is adjusted by the network every two weeks to control the rate of block production. Depending on the total hash rate the target is set to limit block production every ten minutes.
The hash is computed using SHA-256 twice, therefore there is no known other way to get a hash bellow the target than to try. At each attempt the nonce is incremented and a new hash is produce and tested. When a valid block header is found it means a good nonce of the block have been found which proves that **computation work** has been done.
Only the longest chain is valid, so the chain which is produce faster is the valid one (if they start at the same length). If some attacker want to erase an earlier transaction he would likely have to start with a smaller chain and generate blocks faster than the rest of the network in order to catch up.
The history of transaction is the history written by those who detain the majority of the computing power[2] of the network.
We will estimate in this article the current difficulty to forge a transaction, that is to modify the blockchain.

2.2.4 *Mining*

The larger the computing power of the user is, the more secure the system is for an outside attacker. However the more homogeneously distributed this computing power is, the more secure the network is against an insider.
The network encourages users to participate in the blockchain by giving them **new** Bitcoins for every valid block they add. Currently for one block found the remuneration is 25 Bitcoins, at the time the article was written, it represents 1800 euros.
This reward is halved every 210,000 blocks so the rate of

[2]by that and for the rest of this article we intend hash rate

Figure 4: Total hash power available at a given time, source : `bitcoin.sipa.be/`

new coin generation is controlled by the network. The discoverer of the new block can also be paid by a system of transactions fees.

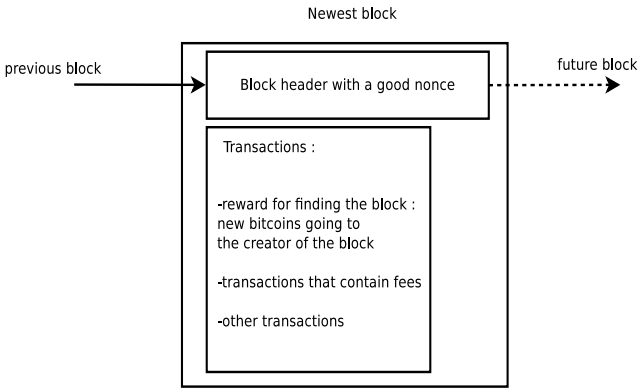

Figure 5: At each block created, new Bitcoins are generated

Giving time of calculation to gain bitcoin is called **mining**. Thanks to this, the power of the network is growing.
The later one mines, the less bitcoin he is going to earn because his chance to find a block will get smaller. There is possibility to earn money by participating in the network and the faster you participate the better.

3. FORGING A PROOF-OF-WORK

3.1 Test of complexity

Using the libbitcoin library[3] we have implemented a simple database allowing us to redo the proof-of-work[4] from the beginning in order to check the effective complexity of a brute force attack.
To be accepted in the blockcahin a block header mush have a hash smaller than a specified target, so we measure the time needed to compute a hash of a block header in order to determine whether it is possible to forge a block. This hash is the result of SHA-256 performed twice on the block header. The time of execution has been measured using the PAPI library[5].
Below is the actual code used by our experiment, there is one loop to calculate hashes and incremente the nonce (we can decide how many time the loop is executed).

```
1  // before is the creation of the bock header blk and its
       initialization , we use his nonce to determine the
       number of iteration the loop in calcul_hash that will be
       executed.
2  long_long start_usec, end_usec;
3
4  hash_digest calcul_hash(block_header blk)
5  {
6
7    hash_digest hash;
8    hash_digest target = calc_target(blk); // function which
         return the target using the bits of difficulty stored in
         the block header blk.
9
10   hash=hash_block_header((const libbitcoin::block_type&)
         blk); // computation of the hash (SHA256 twice)
11   while(hash>target)
12   {
13     blk.nonce=blk.nonce + 1;
14     hash=hash_block_header((const libbitcoin::block_type
           &)blk);
15   }
16
17 start_usec = PAPI_get_virt_usec();
18
19 calcul_hash(block_header)
20
21 end_usec = PAPI_get_virt_usec();
22 // start_usec - end_usec is what will interest us in the
       following.
```

On the figure 6 you can see the time per iterations of the loop computing hashes.

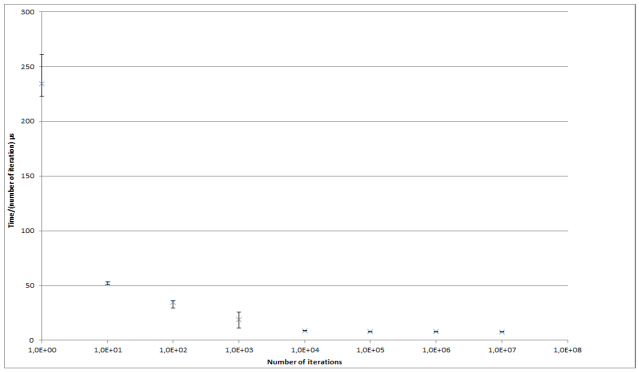

Figure 6: Time per Iteration

As you can see on the graph, there are some variations in the duration of computation for small iterations, but when the number of iteration is large enough ($> 10^4$) the time of computation is stable and the variability is smaller than in the beginning.

[3]available for public use here `www.libbitcoin.dyne.org/` this library is not very well documented but the example files are very complete
[4]i.e. blockchain
[5]you can find this open source library here : icl.cs.utk.edu/papi/

It is what we can expect because of the instructions before the "while" loop take times regardless of the number of iterations. We will consider in the following the average of the tree last values to minimize the impact of instructions taking place before the loop (required for proper initialization). Our hash rate is : 10^{+8} hash/second. It has been measured with the CPU AMD Phenom 965 (each experiment was repeated 10 times for better accuracy).
To determine the average time needed to find a good hash we suppose the value of the hash we found in an iteration follows a uniform distribution on $[2^{256} - 1; 0]$.

$$D = \frac{Number\ of\ hash}{Number\ of\ hash < target}$$

is the average number of hash we need to compute before finding one bellow the target. The average time is then :

$$D/hash\ rate$$

in our case if we take the maximum target (the minimum difficulty which is the target of the first blocks : 0x00000000FFFF0000000000000000000000000000000000000000000000000000) we have :

$$Average\ time = \frac{1}{10^{+8}} * \frac{2^{256} - 1}{2^{208+16}}\ seconds$$

where 2^{208+16} is the target. According to this calculations, it takes on average 9 hours to find a block with the minimum difficulty. It would have required a computer only 54 times faster to generate a block in 10 minutes.
We can conclude it seems possible to have enough CPU power to overtake the network at his launch. The figure bellow display the evolution of the average number of hash to compute before finding one acceptable to the network[6].

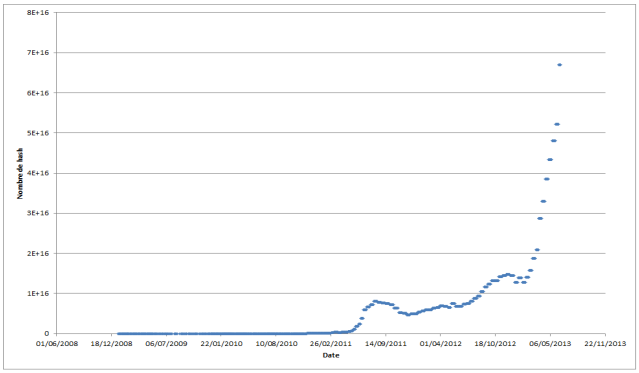

Figure 7: Evolution of the hash power in the network (data from `blockexplorer.com/q/nethash`)

If we now consider the current target : 0x00000000000011337000000000000000000000000000000000000000000000 it would take $5 * 10^{11}$ s to find a new block, which shows the limitation of a CPU to calculate hashes and the current difficulty to forge block in comparison with the debut of the network. We can get an idea of the power of computation involved in mining Bitcoin :

$$Hash\ rate = \frac{Average\ time * Current\ target}{Number\ of\ hashes} = 10^5 Ghash/sec$$

[6]public data, accessible in the block chain for example

it is roughly 10^9 times the hash rate of the CPU used in the experimentation.

If we try to estimate the gain/costs ratio to mine with our system of experimentation we need to estimate the gain and the costs.

$$Gain = \frac{Hash\ rate\ user}{hash\ rate\ network} * Reward\ per\ block$$

$$= 25 * 10^{-9} coins/(10min)$$

$$= 1,8 * 10^{-6} euros/(10min)$$

$$Cost = Electric\ power\ needed * Power\ cost$$

$$= 2.5 * 600 * 10^{-6}\ euros/(10min)$$

We minor the consumption of the machine by 100 W (average consumption of our CPU) and we are using EDF[7] prices for electric power.

$$Profitability = \frac{Gain}{Cost}$$

$$= 1.2 * 10^{-3}$$

The machine used is not optimized for the calculation of hashes (high consumption for few hashes). I order to reach cost effectiveness (Profitability > 1) we will need to use machines a lot more efficient (at least 10^3 more). CPU are not design to calculate hashes. Other device can be used :

- GPU with its highly parallel architecture is more adapted but use a lot of power;
- FPGA are used by some;
- ASIC[8] is the most efficient way to compute hashes.

3.2 An other way to forge a transaction

As we saw before in this paper if you use the Bitcoin protocol properly (and if the network is majority control by honest nodes) you should be safe. But in the daily use of the system there are dangerous behaviors that require investigations.

3.2.1 Web based wallets

A very large amount of users (up to 25% according to different sources[9]) is using web based wallets. Instead of performing your own verifications and your own transactions you trust a provider to do this for you. By doing this you defeat the main purpose of the Bitcoin as trust in someone else is required. Of course it is easier to use as you only have to remember a single password to access your Bitcoins. But the main advantage is that you can access it from everywhere and with any device capable of browsing the web.

[7]a french provider of electricity
[8]application-specific integrated circuit
[9]`elecrum.org` documentation and BitMinter `bitminter.com/` website

3.2.2 *"Light wallet"*

We choose the term light wallet to expose a second way of using the Bitcoin system without great security. Those light wallets (the most famous example is Electrum [10]) use deterministic keys (incrementation of a single "origin key" to build any necessary key) which can be store outline in order to regain access to all addresses if need be. This system is supposedly safe as you keep your secret key on your computer but to avoid computation time you use a distant server who provide the CPU time needed (for transaction verification). Another time, by its centralization, it defeat the purpose of Bitcoin.

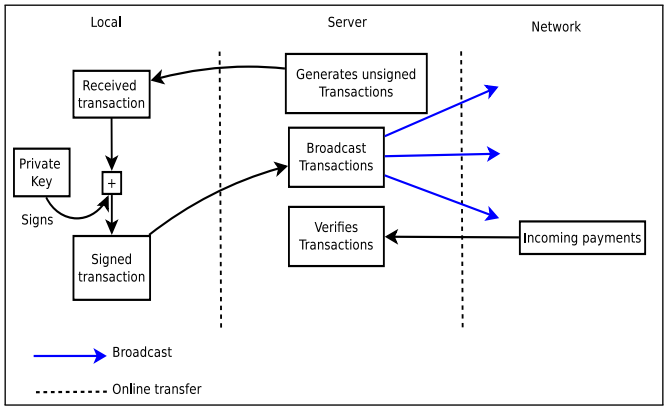

Figure 8: summary of light wallet operations

3.2.3 *Peer validation (simple verification)*

In order to gain time and to save CPU power one node receiving a payment can in theory (and very often in practice) decide not to rule a full verification of ownership on the longest proof-of-work. Instead the node can broadcast the transaction and wait for other nodes to broadcast a new longest proof-of-work, and then check if the transaction was accepted in the later proof-of-work.

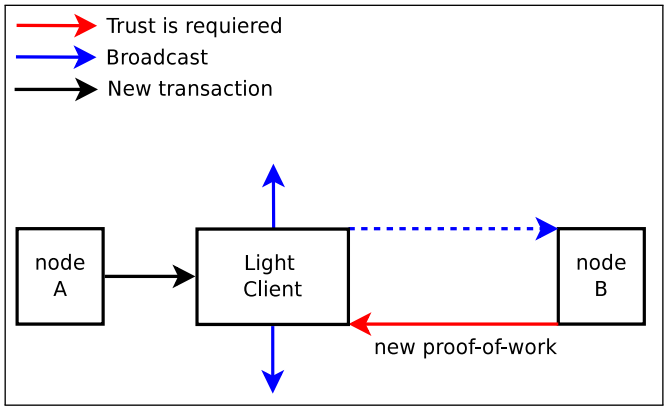

Figure 9: Principle of simple verification

This is unsafe because it defeat in fact the purpose of one CPU one vote exposing a node if an attacker is controlling enough nodes accessed by the victim.

[10] cf `www.electrum.org`

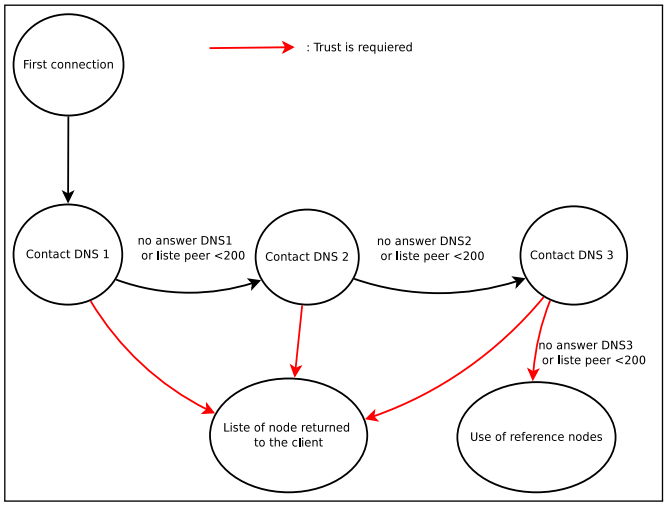

Figure 10: State machine : first connection of a node

We chose to investigate how a new node (and an existing one) is discovering and using its peers to check if he has the longest proof-of-work. As described on the Bitcoin wiki [11] a new node connecting to the network tries to contact four DNS servers[12] in order to get the address of some (at least 100) active nodes. Then it puts them into a list with a time of validity.
Once every 24 hours a node broadcast its presence to the entire network. Consequently if a node is connected firstly with honest nodes he will have access to a large amount of safe peers. On the other hand, the node is at the mercy of the attacker. The DNS are a the heart of the network safety, another limitation to a truly distributed system.

4. THE MINING ISSUE

There are two methods of mining. The first one is doing it by yourself and hoping to compute a good block header. The gains for one good header is huge (1800 euros per header currently) but the probability to find one is extremely low if you do not have an enormous computing power. The second one is to gather in group in order enhance the frequency of hits and to share the gains, this is called mining pool. The sole purpose of mining pools is to smooth the gain rate.

As we can see more than 50 per cent of the hashing power available is in the hands of tree large entities.
BTCguild and 50BTC are mining pools powered by independent users. They are mining from the mining client of his choice (for 50BTC) or the client of the pool (open source client for BTCguild). The pools take up to 3 per cent of generated bitcoins.
On the the other hand, ASICMiner, currently the biggest miner is a company that own and sell ASIC chip. It is private and very discrete as it does not try to recruit new

[11] `www.en.bitcoin.it/wiki/Satoshi_Client_Node_Discovery#DNS_Addresses` we checked if this was corresponding to an actual implementation and it was the case for example in **Bitcoin-qt** (see net.cpp)
[12] first `bitseed.xf2.org` down at the moment of the experiments and then `dnsseed.bluematt.me` this order was the same while we tested.

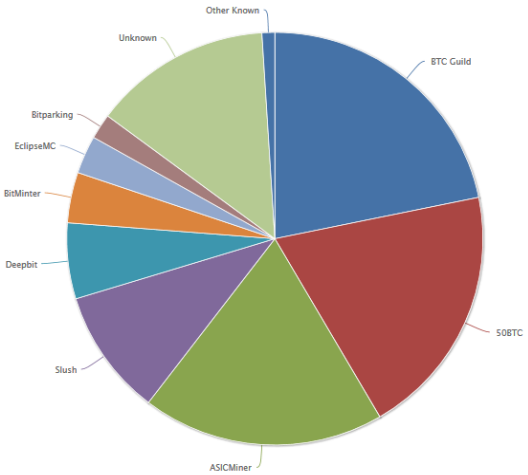

Figure 11: Mining repartition

members (they sold shares at the beginning but not anymore). We were able to locate their headquarter in China. We are concerned by this situation because a big chunk of the power is already detained by a private company that is very opaque.
Such a concentration is worrying but not yet dangerous as they do not own the majority of hashing power.

5. CONCLUSION

In this article we have presented the main principles of the Bitcoin system of payment. We have evaluate whether it is possible to forge a transaction and give a lower bound of the cost of mining on our machine as well as pointing out weakness of Bitoin both in clients (Light wallet and dns queering) and in the mining process.
Currently it would take a hash rate one billion time the hash rate our machine achieve to be able to produce hashes every 10 minutes. It would require an enormous amount of hashing power to threaten the integrity of the network.
We saw that node discovery was a source of weakness especially in light wallets. It is a vector of centralization, if someone was to compromise DNS' integrity, every new node would be vulnerable. An other issue regarding centralization is the fact that the majority of mining power is in the hands of very few entities.
Nonetheless the network is in constant evolution[13] and a strong community of developer is working on improvements[2][14].

6. REFERENCES

[1] M. Babaioff, S. Dobzinski, S. Oren, and A. Zohar. On bitcoin and red balloons. *research.microsoft.com/pubs/156072/bitcoin.pdf*, 2011.

[2] S. Barber, X. Boyen, E. Shi, and E. Uzun. Bitter to better: How to make bitcoin a better currency. *crypto.stanford.edu/ xb/fc12/bitcoin.pdf*, May 2012.

[3] D. Chum, A. Fiat, and M. Naor. Untraceable electronic cash. *crypt'88*, 1988.

[4] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *bitcoin.org/bitcoin.pdf*, October 2008.

[5] F. Reid and M. Harrigan. An analysis of anonymity in the bitcoin system. *arxiv.org/abs/1107.4524*, May 2012.

[6] D. Ron and A. Shamir. Quantitative analysis of the full bitcoin transaction graph. *eprint.iacr.org/2012/584*, October 2012.

[13] `en.bitcoin.it/wiki/Bitcoin_Improvement_Proposals`
[14] the number of forums and the reactivity of its member is astonishing. For example see `bitcointalks.org`