Marsalek A., Kollmann C., Zefferer T. (2018) A Security Analysis of FirstCoin. In: Janczewski L., Kutyłowski M. (eds) ICT Systems Security and Privacy Protection. SEC 2018. IFIP Advances in Information and Communication Technology, vol 529. Springer, Cham
The final publication is available at Springer via http://doi.org/10.1007/978-3-319-99828-2_10

A Security Analysis of FirstCoin

Alexander Marsalek[1], Christian Kollmann[2], and Thomas Zefferer[2]

[1] Graz University of Technology, IAIK, Austria
Alexander.Marsalek@iaik.tugraz.at
[2] A-SIT Plus GmbH
Christian.Kollmann@a-sit.at, Thomas.Zefferer@a-sit.at

Abstract. Supported by the current hype on Bitcoin, the number of available cryptocurrencies has steadily increased over the past years. Currently, relevant portals list more than 1.500 cryptocurrencies. Many of them slightly deviate from approved and tested technical concepts and realize security-related functionality in different ways. While the security of major cryptocurrencies has already been studied in more detail, security properties of less popular cryptocurrencies that deviate from approved technical concepts often remain unclear. This is a problem, as users run the risk of losing invested money in case the respective cryptocurrency is unable to provide sufficient security. In this paper, we underpin this statement by means of a detailed analysis of the cryptocurrency FirstCoin. We identify and discuss vulnerabilities of FirstCoin, which lead to a low network hash rate and allow for 51 % attacks. We propose a double-spending attack that exploits these vulnerabilities and demonstrate the proposed attack's feasibility by running it in an isolated evaluation environment. This way, we show FirstCoin to be insecure and provide a real-world example that underpins the general problem of cryptocurrencies deviating from approved security concepts and relying on weak security designs.

Keywords: Blockchain, double spending, proof-of-work, FirstCoin, cryptocurrency, 51% attack

1 Introduction

In 2017, cryptocurrencies have experienced an impressive increase in popularity and market capitalization [1]–[4]. In the wake of Bitcoin, which has reached a market capitalization of more than 336 billion USD in 2017, an increasing number of alternative cryptocurrencies have been introduced. The website Cryptocurrency Market Capitalizations [5] currently lists more than 1.500 different cryptocurrencies. While none of them is yet as successful as Bitcoin in terms of market capitalization, all currencies aim to employ the current hype around cryptocurrencies and seek to attract potential investors. Although all currencies have been subject to fluctuating market values during the past months, their popularity remains high.

On a technical level, most cryptocurrencies rely on the same basic concepts as Bitcoin [6]. However, other cryptocurrencies still cannot be regarded as simple Bitcoin clones, as they often rely on a slightly modified underlying design

and implement various technical aspects differently. For instance, several cryptocurrencies adapt Bitcoin's proof-of-work based consensus mechanism [7] by varying certain parameters of the consensus algorithm or even follow a completely different approach to achieve consensus between participating entities. These deviations from the original Bitcoin algorithm have yielded a technically heterogeneous ecosystem of cryptocurrencies. This heterogeneity bears a risk: while the security of popular currencies such as Bitcoin [8], Ethereum [9] or Ripple [10] have been subject to detailed analyses [11], the security of less popular currencies often remains uninvestigated and hence undetermined. In particular, it remains unclear if and to what extent deviations from the original Bitcoin concept affect a cryptocurrency's overall security. In the worst case, only slight deviations can already cause flaws that undermine the currency's security and pose serious risks to people investing money in this currency.

In this paper, we underpin this statement with hard facts. We show by means of a real-world example that bad design decisions and inadequate deviations from well-tried security concepts can make a cryptocurrency prone to rather simple attacks. While our research interests are not limited to certain cryptocurrencies, we focus on one specific currency in this paper, i.e. the cryptocurrency FirstCoin[3]. FirstCoin has been chosen for multiple reasons. First, this currency has shown a rather unusual development of both, price and market capitalization during the past months. Second, the currency's proposed consensus mechanism and its implementation seem rather unorthodox. Finally, the published source code of FirstCoin does not include the necessary mining functionality, which supports the hypothesis that FirstCoin at least partly applies the disproved concept of security by obscurity.

For the listed reasons, we analyze FirstCoin in more detail. Applied analyses include a detailed evaluation of FirstCoin's source code, reconstruction of the not-published mining functionality, as well as the successful mounting of a double-spending attack in a protected environment. This way, we confirm the hypothesis that double-spending attacks are feasible if mining power is not sufficiently distributed amongst participants. Furthermore, our results confirm the common sense that the concept of security by obscurity does not necessarily yield secure solutions. Overall, our work shows that the cryptocurrency FirstCoin is insecure and prone to attacks. This supports the above-made statement that cryptocurrencies deviating from approved concepts of established cryptocurrencies must not be assumed to be secure. This is a relevant finding also for people planning to invest money in one of the many available cryptocurrencies out there.

The remainder of this paper is structured as follows. We first provide general background information on cryptocurrencies in Section 2 and discuss related work on the security of cryptocurrencies in Section 3. In Section 4, we introduce FirstCoin and analyze its most important features and properties. Based on the properties of FirstCoin, we then identify potential vulnerabilities and propose an attack vector in Section 5. To prevent any damage to the FirstCoin network, we have tested and evaluated the proposed attack vector in a special evaluation

[3] Available at http://www.firstcoinproject.com/ and https://github.com/firstcoinofficial

environment. Details on the conducted evaluation are presented and discussed in Section 6. Finally, conclusions are drawn in Section 7.

2 Background

Bitcoin was the first project to build a cryptocurrency based upon a distributed ledger modeled by a blockchain. Bitcoin uses the blockchain as a public record of all transactions ever made, whereas a transaction transfers a certain amount of coins from a sender to a receiver. Transactions are grouped together into blocks. Blocks, in turn, are linked with each other, forming a chain of blocks. This makes any ex-post modification on blocks or their contents detectable. Blocks are created by so-called miners, which perform costly computations to solve cryptographic puzzles defined by the Bitcoin software. The difficulty of the puzzle is regularly adapted to reach an average block creation interval of ten minutes. The first miner able to solve the puzzle is allowed to create a new block and broadcast it on the network. Along with transactions, each created block contains a link to the previous block and a proof of work. In case that more than one block tries to build upon the existing chain of blocks concurrently, i.e., forking the blockchain, the chain with the higher proof-of-work is selected as the valid chain. As a reward for spending its computation power, the successful miner receives the reward for the block, i.e., newly mined coins, and the fees associated with each transaction in the block. With that compensation, all participants of the Bitcoin network have an incentive to be honest and follow the protocol.

The consensus algorithm of Bitcoin guarantees several properties of the blockchain, all without relying on a central trusted authority, e.g., the correct chaining to the previous block, that sufficient work was put into the creation of the block, valid cryptographic signatures of all transactions, and most important that no double-spending has occurred. In a double-spending attack, an attacker sends the same coins to different recipients, to e.g. wrongfully buy goods from a vendor. To mount such an attack, the attacker first creates a valid transaction, spending its coins, to make a purchase. As soon as the vendor sends the good to the buyer, e.g. upon receiving and verifying the transaction, the attacker creates a second transaction, spending the same coins again, but sending them to a different recipient. To mitigate this attack, the vendor is advised to wait until the respective transaction is included in the public blockchain and has received enough confirmations. A confirmation is merely a block in the chain building upon the block that includes the transaction. To have a high certainty to actually be in possession of the received coins, recipients usually wait until six confirmation blocks are appended to the blockchain. This confidence stems from the fact that rewriting the history of the blockchain requires quite large computation power. The attacker would not only need to recalculate the existing blocks (leaving out the first transaction to the vendor), but also would need to keep pace with the benign nodes extending the valid blockchain. A double spending attack is only guaranteed to succeed if the attacker controls more than half of the network's computing power, as we shall see later.

3 Related Work

The possibility of double spending attacks on blockchain-based currencies was known from the beginning. In the work introducing Bitcoin, Nakamoto [12] calculated its success probability for the Bitcoin network. In 2014, Rosenfeld [13] showed that an entity controlling more than 50 % of the computing power in an proof-of-work-based blockchain network can always (successfully) perform a double-spending attack. Even with less than 50 % of the computing power, the attack will succeed with a certain probability, mostly dependent on the number of confirmation blocks required by the vendor. Pinzón *et al.* [14] build on the work of Rosenfeld and include time-based information in their attack model to account for attackers secretly mining blocks in advance. Gervais *et al.* [8] construct a framework to evaluate security features of several proof-of-work-based blockchains. With that framework, the authors can give recommendations for vendors on how many confirmation blocks to wait on to achieve the desired security level. Karame *et al.* [15] analyze double-spending attacks in Bitcoin on zero-confirmation transactions, i.e. transactions only published to the network, but not included in any block at all. The authors deduce that such attacks are easy to mount and do not require significant work by the attacker. Carlsten *et al.* [16] analyze the consequences of the block reward halving every 210,000 blocks (about every four years) and the limited supply of tokens. Carlsten *et al.* argue that Bitcoin would become unstable and insecure, as soon as the reward for mining a block falls below a threshold. Then, the reward for investing computing power to append blocks to the blockchain would be the sum of the transaction fees in the new block only. The authors propose that new cryptocurrencies should set a fixed block reward, to take an effective countermeasure against the looming instability of the underlying blockchain.

So far, double-spending attacks have been discussed in related work mainly from a theoretical perspective. Practical double-spending attacks published so far target transactions before they are added to the blockchain and obtain a sufficient level of confirmation. The attack presented in this paper is way more powerful, as it enables double-spending attacks on already confirmed transactions by rewriting the public history of the blockchain. To the best of our knowledge, this is the first scientific publication of such an attack.

4 FirstCoin

FirstCoin is a proof-of-work based peer-to-peer cryptocurrency. It shares many similarities with Litecoin [17], like the PoW algorithm *scrypt*. Litecoin, in turn, resembles Bitcoin in its core principles—sans its PoW algorithm[4] (Bitcoin uses *SHA-256*). However, FirstCoin neither forked Litecoin's blockchain[5] nor its source code repository. Still, there are various similarities in the source code and in

[4] Litecoin also aims at a different block interval and limits the supply to 84 million coins amongst other differences. [5] FirstCoin's blockchain starts from a different first block (called the genesis block) as Litecoin and is therefore completely disjunct.

the provided build instructions, which support the assumption that FirstCoin is technically closely related to Litecoin. Despite these similarities, there are some considerable differences between the two cryptocurrencies. In contrast to Litecoin, FirstCoin aims at a block interval of 60 seconds, premined 109,999,999 coins, limited the number of coins to 110 million, and reduced the block reward to one Satoshi[6]. The minimum difficulty (difficulty target) of the puzzle to be solved, is updated every 3.5 days to reach an average block interval of 60 seconds.

As most cryptocurrencies, FirstCoin uses a hard-coded genesis block[7] as the root of trust. Furthermore, the source code defines a domain name service seed node as well as eight additional checkpoints (hard-coded block data) between the genesis block and block 52. All these features are also used in Litecoin.

A key difference between Litecoin and FirstCoin is the missing mining code. While the help message of the FirstCoin daemon states "-gen Generate coins (default: 0)", the parameter is actually ignored in the source code. Furthermore, the JavaScript Object Notation Remote Procedure Call (JSON-RPC) methods needed for mining are not available. The fact that no white paper or other documentation describing FirstCoin's mining process is available either, supports the hypothesis FirstCoin aims to increase security by obscurity.

As additional security-increasing measures, the FirstCoin network enforces additional rules, like a maximum block size of one megabyte and several time-related constraints. The timestamp of the block has to be greater than the time of the last checkpoint and greater than the median time of the last eleven block times, but must no be greater than the median time of the connected nodes plus two hours. These block-timestamp rules are obviously inherited from Bitcoin [18], [19]. The network also enforces that every block smaller than 100,000,001, except block 2, pays one Satoshi plus the fees of all included transactions as block reward. Block 2 is an exception and is allowed to pay out 109,999,999 coins (the premined coins). Other blocks are only allowed to pay out the fees as block reward. These rules enforce a limited coin supply and a defined, albeit very small, block reward. This way, miners cannot reward themselves with arbitrary high rewards.

In summary, FirstCoin appears to be a modified version of Litecoin. From a security perspective, some deviations applied by FirstCoin seem questionable at first glance. We elaborate on that observation in the next section and propose an attack that exploits FirstCoin's questionable security properties.

5 Proposed Attack

Due to the missing mining code and the low block reward, it is reasonable to assume that only a few miners participate in the FirstCoin network. Mining for FirstCoin seems unattractive, as a miner has to implement the missing mining code and then receives only a low reward of one Satoshi for each accepted block. This assumption is supported by concrete figures. Figure 1 shows the network hash rate of FirstCoin in red color and of Litecoin in blue color. Both plots start

[6] One Satoshi is a one hundred millionth of a single FirstCoin (0.00000001 FRST), the smallest representable unit in FirstCoin. [7] The first block is called genesis block.

from their respective genesis block and end in March 2018. We have derived the hash rate using the *getnetworkhashps* API call and estimated the network hashes per second based on the last 120 blocks. Additionally, Figure 1 shows the hashes per second of a Nvidia GTX 1070 graphic card and an Antminer L3 unit. Both plots start at the release date of the respective product. As shown in Figure 1, the

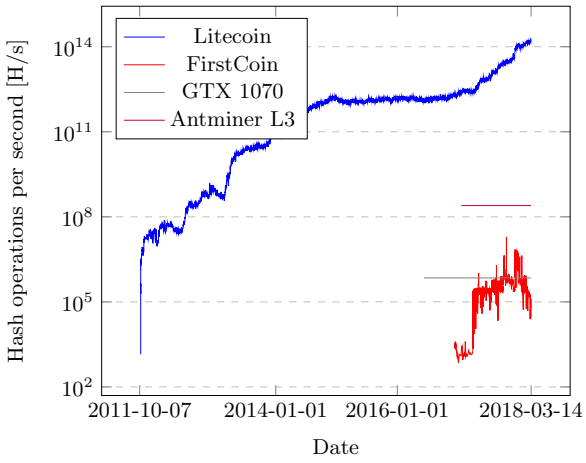


Fig. 1: Comparison of the hash rate of FirstCoin and Litecoin network with a logarithmically scaled y-axis. Additionally, the hash rate of a single GTX 1070 GPU and an Antminer L3 is shown, starting from their release dates.

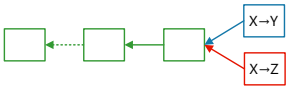
hash rate of Litecoin is currently about 800 million times bigger than the hash rate of FirstCoin. Because of this big difference Figure 1 uses a logarithmically scaled y-axis. From Figure 1 it becomes apparent that FirstCoin's hash rate is rather low and does not pose a serious computational challenge. Thus, we assume that it is feasible to mount 51 % attacks against FirstCoin with a single consumer graphics card, provided that we manage to create a working FirstCoin miner. We propose a double-spending attack on FirstCoin comprising the following steps:

1. Implement the missing methods in the source code to create a working miner for FirstCoin.
2. Buy or mine a sufficient number of FirstCoin coins. Send them to Address X.
3. Create and publish Transaction TX, which sends the coin(s) from Address X to Address Y. Wait until TX is included in a Block B and Block B is added to Chain C.
4. Create the double-spending Transaction TX', which sends the same coin(s) to Address Z. Start mining the conflicting Block B' which includes TX' instead of TX. Block B' must reference a block before Block B in Chain C, otherwise the network will later refuse to accept the forked chain C'. Block B' must be kept secret.

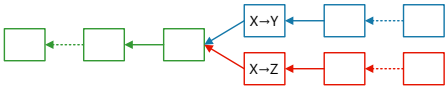
5. The network will continue to mine blocks based on Block B. Wait until the Block B has enough confirmations, so that TX is considered accepted. Keep mining on the secret Chain C'.
6. As soon as C' has a higher cumulated proof-of-work than the benign Chain C, publish C'.
7. As C' has a higher proof-of-work, the network will accept it as the main chain. From now on, the network will mine blocks based on C'. Hence, address Y will not receive any payment.

For a better understanding, the listed steps are visualized in Figure 2.

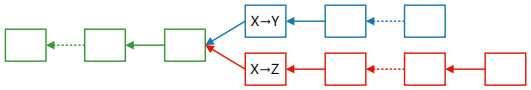
(a) The valid blockchain C in green color, with a newly created Block B including Transaction TX, paying coins to Address Y.



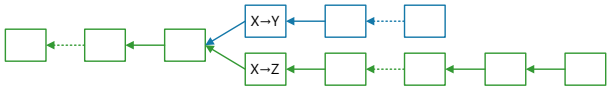
(b) The blockchain C' with a conflicting, not published, Block B' in red color. The block contains Transaction TX', which double spends the coins sent in Transaction TX.



(c) The honest network adds blocks to Chain C including Block B. Meanwhile, blocks are secretly mined for Chain C', including Block B'.



(d) After some time, Transaction TX has enough confirmations and the owner of Address Y assumes to be in possession of the coins. After that, mining is continued until the modified Chain C' has a higher cumulated proof-of-work than all other chains.



(e) After publishing C', the network evaluates and accepts C' as long as it is the chain with the most proof-of-work and every block and transaction included is valid. As all rules have been followed, the new chain C' is accepted. Thus, Transaction TX' has become part of the chain and Transaction TX (and its block and all subsequent blocks) have been dismissed. From now on, the honest network will append blocks to C'.

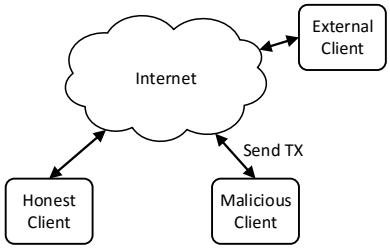
Fig. 2: Visualization of a double spending attack

6 Evaluation

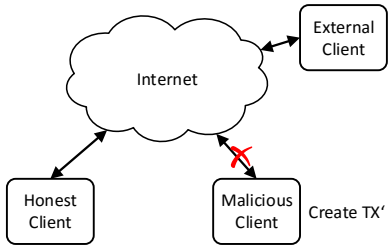
In this section, we describe how we evaluated our proposed attack. We aimed to evaluate the feasibility of the attack without harming the network or diminish users' trust into FirstCoin. Therefore, we conducted our attack on an air-gapped network. The entire evaluation setup is introduced in the following.

6.1 Evaluation Setup

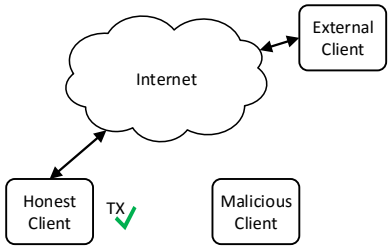
Our setup consists of two virtual machines, one running the original unmodified FirstCoin daemon and a second one running our modified version, which includes the complemented mining code. We call these machines *Honest VM* and *Malicious VM*, respectively. Figure 3a shows the first evaluation phase. Both VMs are connected to the Internet and synchronize their local blockchain with the remaining network, visualized as *External Client*. After both daemons are



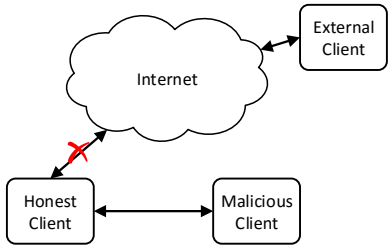
(a) The initial phase of the attack. Both VMs are connected to the Internet and synchronize the FirstCoin blockchain. The malicious VM sends Transaction TX: A→B to the network.



(b) We disconnect the *Malicious VM* from the Internet, create the conflicting Transaction TX': A→C, and start secretly mining it into a block.



(c) We continue secretly mining and wait for the network to accept Transaction TX as valid. After that, we continue mining until our chain has a higher combined proof-of-work than the benign chain.



(d) As soon as our chain has a higher proof-of-work, we stop mining, disconnect the *Honest VM* form the Internet, connect it to our *Malicious VM*, and monitor which chain will be accepted by the clients.

Fig. 3: Visualization of the evaluation setup and the attack steps.

synchronized, we create Transaction *TX* on the *Malicious VM*, transferring coins from *Address X* to a second *Address Y*, running on the *Honest VM*. All addresses used are under our control. The *Address Y* simulates the victim's wallet (e.g. a merchant's or exchange's). Then, we wait until *TX* is included in a valid block by the honest network.

Subsequently we disconnect the *Malicious VM* from the Internet and create a second Transaction *TX'*, transferring the same coins as *TX* to a different *Address Z* under our control. This step is visualized in Figure 3b. *TX'* spends the coins a second time, hence the name *double-spending* attack. Next, the *Malicious VM* forks the blockchain by creating Block B' including TX'. Block B' is designed as a competitor to Block B, which includes *TX*. Thus the network will only accept one of these blocks and dismiss the other one. Next, we use a standard *scrypt* miner-software and start secretly mining the second chain.

The honest network will mine blocks based on Block B. We continue to secretly mine on our chain and wait until Transaction TX has enough confirmations that the victim accepts it as confirmed. The default client requires six confirmations (Figure 3c). We continue secretly mining until our chain has a higher combined proof-of-work than all other chains. The probability that this situation occurs depends on the ratio of our hash rate to the networks' hash rate [13]. As we have a higher hash rate than the remaining network, it is guaranteed that this situation occurs at some point.

As soon as our chain has a higher proof-of-work than the other chain, we stop mining on the *Malicious VM*. We subsequently disconnect the *Honest VM* form the Internet and connect it to our *Malicious VM* via a private local area network to simulate publishing our chain to the network. If the client on the *Honest VM* accepts our chain, it is very likely that the whole network would accept it in a real attack, since it runs an unmodified FirstCoin daemon. After a short time, both clients will agree on the one valid chain with the most combined proof-of-work (Figure 3d). As our chain holds a higher combined proof-of-work than the forked chain, our chain will be accepted.

6.2 Implementation

A key aspect for a successful attack was the reconstruction of the missing mining functionality. In order to gain a working FirstCoin miner, we implemented the missing RPC functions required for mining and used a default, publicly available *scrypt* miner. As we assumed a close technical relation between FirstCoin and Litecoin, we cloned both git repositories and started comparing key files of FirstCoin with Litecoin using the *git diff* operation. This step was repeated for every commit in a reasonable time-frame. The resulting diff files were sorted based on their file size. Several files had the same size, so we chose a file (commit) from the smallest ones and used it as the reference. From this commit we took all source code necessary to implement the RPC methods *getblocktemplate* and *submitblock*. *getblocktemplate* returns all information necessary to mine a block on top of the current best chain. *submitblock* is called by the miner after it finds a block to submit it to the FirstCoin daemon. The daemon subsequently validates

the block and sends it to the network or rejects it depending on the validation result. After implementing these two methods and deactivating a check that verified if the daemon is connected to the network, we were able to mine blocks for the FirstCoin blockchain on an isolated machine.

Next, we implemented a double-spending proxy that manipulates the *getblocktemplate* responses of the FirstCoin daemon to create a blockchain fork containing our double-spending Transaction TX'. The proxy is visualized in Figure 4. It

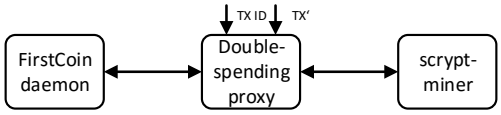


Fig. 4: A double-spending proxy is put between the FirstCoin daemon and the miner. It takes as input the transaction ID to double spend and the double-spending Transaction TX'. Based on these inputs, it creates manipulated *getblocktemplate* information and sends them to the miner when requested.

takes as input the transaction ID of TX and the raw Transaction TX'[8]. Based on the transaction ID the proxy searches for the block containing the corresponding transaction. Then the proxy analyzes and stores all information for later use. When the miner starts and asks for *getblocktemplate* information, the proxy loads the previously stored information and creates a double-spending block based on it. Compared to the original block the proxy changes the transactions only. It removes all existing transactions and adds only TX'. Listing 1.1 shows the values of an *getblocktemplate* response without any transactions.


```
{
  "result": {
    "version": 2,
    "previousblockhash": "4
        b9a4b3e875a522cdf2a2c0e70da520a5711c8c2c35aab0253e7758ca8b40d7e
        ",
    "transactions": [],
    "coinbaseaux": {
      "flags": "062f503253482f"
    },
    "coinbasevalue": 1,
    "target": "0000064
        e640000000000000000000000000000000000000000000000000000000",
    "mintime": 1521719865,
    "mutable": ["time", "transactions", "prevblock"],
    "nonce range": "00000000ffffffff",
    "sigoplimit": 20000,
    "sizelimit": 1000000,
    "curtime": 1521720490,
    "bits": "1e064e64",
    "height": 490914
```


[8] Instead of TX' we could also give the proxy access to our wallet. This would allow the proxy to create TX' on its own. However, for this to work we would have to verify if all necessary RPC calls are implemented and work correctly. Creating TX' with a second wallet that is taken offline before creating TX appeared to be more elegant.

```
  },
  "error": null,
  "id": 0
}
```

Listing 1.1: Response of a getblocktemplate request

After the miner submits a valid block, the proxy increments the stored block height and calculates and stores the hash value of the block. For subsequent *getblocktemplate* calls the proxy always returns the most current previous block hash value and block height. Also the *mintime* and *curtime* fields are adapted to the current requirements. Finally, all block-templates except the first one contain an empty transaction list.

6.3 Results

After carrying out the previously described attack steps and connecting the *Honest VM* and the *Malicious VM*, the two machines started to exchange their blockchains. The honest network was able to find eight blocks instead of the minimally required six blocks, while the *Malicious VM* managed to find 12 blocks. After the *Honest VM* had received the ninth block, it began to reorganize its chain as shown in Listing 1.2. The log has been shortened to improve readability.

```
REORGANIZE: Disconnect 8 blocks; 8eba95a63c9b5214...4d381800d4682efb..
REORGANIZE: Connect 9 blocks; ..3d654d31a6855094...05ca1eb30fa2847d
Committing 371 changed transactions to coin database...
SetBestChain: new best=3d654d31a6855094...05ca1eb30fa2847d height=472191
ProcessBlock: ACCEPTED
```

Listing 1.2: Firstcoin daemon log showing the reorganization of the blockchain

The remaining three blocks were not necessary for the double spending attack, but provided some reserve. The log is shown in Listing 1.3, demonstrating the feasibility of a double spending attack with average consumer hardware. Note that the used CPU miner[9] was only able to calculate about 230,000 hash operations per second while a modern GPU like a GTX 1070 achieves 700,000 hash operations per second. Hence, this attack could be carried out even more efficiently when using improved hardware.

```
received block c13de8e5241787c6...e98425613d148090
Committing 1 changed transactions to coin database...
SetBestChain: new best=c13de8e5241787c6...e98425613d148090 height=472192
 ↪  log2_work=43.792578 tx=580648 date=02-26 19:44:40 progress=0.999999
ProcessBlock: ACCEPTED
received block 1ff6d3b5e0bf3511...a8ccda5de81b2011
Committing 1 changed transactions to coin database...
SetBestChain: new best=1ff6d3b5e0bf3511...a8ccda5de81b2011 height=472193
 ↪  log2_work=43.792579 tx=580649 date=02-26 19:44:55 progress=0.999999
ProcessBlock: ACCEPTED
received block 27d10e64dcf4e534...c0835111ba84f8e0
Committing 1 changed transactions to coin database...
SetBestChain: new best=27d10e64dcf4e534...c0835111ba84f8e0 height=472194
 ↪  log2_work=43.79258 tx=580650 date=02-26 19:47:55 progress=1.000003
ProcessBlock: ACCEPTED
```

Listing 1.3: Firstcoin daemon log showing the received three reserve blocks.

[9] We did not manage to get a modern graphic card.

6.4 Discussion

As shown in Section 6.3, the proposed attack on FirstCoin was successful. We showed that 51 %-attacks can be mounted even with consumer hardware. 51 %-attacks can be used to double-spend coins, disrupt the network or rewrite the blockchains' history till the last checkpoint. This attack was possible due to the combination of the proof-of-work consensus algorithm and the low hash rate of the network. We do not see a quick way to fix this issue and prevent the demonstrated attack. However, the attack could be complicated by publishing the mining code and providing miners an incentive to mine, e.g. by increasing the mining reward. Adding a recent checkpoint in the daemon's code would prevent rewriting the prior blockchain. Alternatively, in case a centralized cryptocurrency is desired, shifting to a proof-of-authority consensus algorithm, where only authorized accounts are allowed to approve transactions and blocks, might lead to viable results.

As our attack was executed in an air-gapped network, no double-spending transaction or forked chain was submitted to the public network and its blockchain. Thus, the double-spending attack was not publicly visible, to prevent diminished user trust or to harm the network. Still, considering the realistic evaluation environment, we strongly believe that the attack would also be successful on the real FirstCoin network.

7 Conclusions

In this paper we have presented a successful double-spending attack on FirstCoin, which proves this cryptocurrency to be insecure. Our analyses have revealed two basic weaknesses that undermine FirstCoin's security. First, FirstCoin relies on a proof-of-work based consensus mechanism but does not sufficiently reward miners. This leads to a very low hash rate and enables attackers to easily achieve more than 50 percent of the overall mining power. Second, FirstCoin intentionally keeps parts of its source code undisclosed and hence implicitly relies on the concept of security by obscurity. This concept is well known to be inappropriate for making systems sustainably secure. FirstCoin is hence another prime example demonstrating that security by obscurity is never a good choice.

Although this paper has focused on one particular cryptocurrency, the lessons learned apply in principle to other currencies as well. The key finding is that intentional deviations from approved security-related crypto-currency concepts can cause serious vulnerabilities and undermine the overall security of a cryptocurrency. FirstCoin is one example for that. However, considering the impressive number of available currencies—each implementing certain technical details slightly different—it must not be assumed that FirstCoin is the only problematic currency out there. Applying the findings on FirstCoin to other cryptocurrencies is hence regarded as important future work.

During the past months, the market value of FirstCoin has dropped significantly compared to other cryptocurrencies. It is unclear whether this is already a result of FirstCoin's weak security. In general, it can be expected that in the

long term the free market will automatically sort out cryptocurrencies that do not meet relevant security requirements. Still, this does not eliminate the risk of short-term financial damage when using cryptocurrencies with insufficient security. Following a security-centric approach when designing a cryptocurrency is hence crucial for the success of a cryptocurrency as well as for end-users of that currency.

8 Responsible Disclosure

We informed FirstCoin in February 2018 about the identified issue in order to allow for mitigation. We would like to thank the FirstCoin team for their quick response.

References

[1] BitcoinExchangeGuide.com, *2017 – the year cryptocurrency became more than bitcoin*, Dec. 23, 2017. [Online]. Available: https://bitcoinexchangeguide.com/bitcoin-cryptocurrency-2017-review/ (visited on 12/23/2017).

[2] I. Damti, *2017 will be remembered as the year of bitcoin*, Oct. 25, 2017. [Online]. Available: https://www.forbes.com/sites/outofasia/2017/10/25/bitcoins-ipo-moment-has-arrived/ (visited on 10/25/2017).

[3] Bonpay, *Looking back: 2017 – the year of cryptocurrency*, Dec. 29, 2017. [Online]. Available: https://medium.com/@bonpay/looking-back-2017-the-year-of-cryptocurrency-e9aa00414a2f (visited on 12/29/2017).

[4] A. Robertson, *2017 is the year cryptocurrency joined the global financial system*, Nov. 29, 2017. [Online]. Available: https://www.theverge.com/2017/11/29/16711304/bitcoin-price-10000-cryptocurrency-regulation-finance (visited on 11/29/2017).

[5] CoinMarketCap, *Cryptocurrency market capitalizations*, Mar. 15, 2018. [Online]. Available: https://coinmarketcap.com/ (visited on 03/15/2018).

[6] A. Hern, *Bitcoin and cryptocurrencies – what digital money really means for our future*, Jan. 29, 2018. [Online]. Available: https://www.theguardian.com/technology/2018/jan/29/cryptocurrencies-bitcoin-blockchain-what-they-really-mean-for-our-future (visited on 01/29/2018).

[7] bitcoinwiki, *Comparison of cryptocurrencies*. [Online]. Available: https://en.bitcoin.it/wiki/Comparison_of_cryptocurrencies (visited on 01/12/2018).

[8] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the Security and Performance of Proof of Work Blockchains," *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security - CCS'16*, pp. 3–16, 2016, ISSN: 15437221. DOI: 10.1145/2976749.2978341.

[9] M. Bartoletti and L. Pompianu, "An empirical analysis of smart contracts: platforms, applications, and design patterns," in *Financial Cryptography and Data Security*, 2017, pp. 494–509. DOI: 10.1007/978-3-319-70278-0_31. arXiv: 1703.06322. [Online]. Available: http://arxiv.org/abs/1703.06322.

[10] F. Armknecht, G. O. Karame, A. Mandal, F. Youssef, and E. Zenner, "Ripple: Overview and outlook," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9229, 2015, pp. 163–180, ISBN: 9783319228457. DOI: 10.1007/978-3-319-22846-4_10. arXiv: arXiv:1506.07739v2.

[11] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, "Research Perspectives and Challenges for Bitcoin and Cryptocurrencies," *IEEE Symposium on Security and Privacy*, pp. 104–121, 2015, ISSN: 1081-6011. DOI: 10.1109/SP.2015.14.

[12] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," p. 9, 2008, ISSN: 09254560. DOI: 10.1007/s10838-008-9062-0. arXiv: 43543534534v343453. [Online]. Available: https://bitcoin.org/bitcoin.pdf.

[13] M. Rosenfeld, "Analysis of Hashrate-Based Double Spending," pp. 1–13, 2014. arXiv: 1402.2009.

[14] C. Pinzón and C. Rocha, "Double-spend Attack Models with Time Advantange for Bitcoin," *Electronic Notes in Theoretical Computer Science*, vol. 329, pp. 79–103, 2016, ISSN: 15710661. DOI: 10.1016/j.entcs.2016.12.006. [Online]. Available: http://dx.doi.org/10.1016/j.entcs.2016.12.006.

[15] G. O. Karame, M. Roeschlin, A. Gervais, S. Capkun, E. Androulaki, and S. Čapkun, "Misbehavior in Bitcoin: A Study of Double-Spending and Accountability," *ACM Transactions on Information and System Security (TISSEC)*, vol. 18, no. 1, p. 2, 2015, ISSN: 1094-9224. DOI: 10.1145/2732196.

[16] M. Carlsten, H. Kalodner, S. M. Weinberg, and A. Narayanan, "On the Instability of Bitcoin Without the Block Reward," *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security - CCS'16*, pp. 154–167, 2016, ISSN: 15437221. DOI: 10.1145/2976749.2978408.

[17] C. Lee, *Litecoin*, 2011. [Online]. Available: https://litecoin.org/.

[18] bitcoinwiki, *Block timestamp*, Jun. 1, 2016. [Online]. Available: https://en.bitcoin.it/wiki/Block_timestamp (visited on 06/01/2016).

[19] ——, *Protocol rules*, Aug. 25, 2017. [Online]. Available: https://en.bitcoin.it/wiki/Protocol_rules (visited on 08/25/2017).