Have a Snack, Pay with Bitcoins

Tobias Bamert*, Christian Decker*, Lennart Elsen*, Roger Wattenhofer†, Samuel Welten*

*ETH Zurich, Switzerland {tbamert,cdecker,lelsen,swelten}@tik.ee.ethz.ch

†Microsoft Research wattenhofer@microsoft.com

***Abstract*—Cashless payments are nowadays ubiquitous and decentralized digital currencies like Bitcoin are increasingly used as means of payment. However, due to the delay of the transaction confirmation in Bitcoin, it is not used for payments that rely on quick transaction confirmation. We present a concept that addresses this drawback of Bitcoin and allows it to be used for fast transactions. We evaluate the performance of the concept using double-spending attacks and show that, employing our concept, the success of such attacks diminishes to less than 0.09%. Moreover, we present a real world application: We modified a snack vending machine to accept Bitcoin payments and make use of fast transaction confirmations.**

I. Introduction

Today, we are witnessing that an increasing number of payments in our economy are executed digitally and cashlessly. Entire businesses are founded upon e-commerce and established companies are looking for new ways to expand their existing payment methods. In the last years, several new payment systems like Google Wallet or PayPal simplified fast and mobile money exchange. These approaches have in common that they rely on a central trusted authority to process payments. In contrast, the Bitcoin currency and payment system offers a completely decentralized payment infrastructure based on a peer-to-peer network. Even though there is no central trust authority, the Bitcoin network can provide reliable international money transfer.

However, due to the decentralized nature of Bitcoin, transactions can only be confirmed if the majority of participating nodes accepts them. This transaction confirmation process can take several minutes. Although often touted as the digital equivalent of cash it is not fit for interactions that require fast clearing of transactions. While this delay is not problematic for most online purchases, it prevents the use of Bitcoin in situations where a transaction confirmation is required in the order of seconds, such as paying in a supermarket or at a snack vending machine.

In this paper, we present a concept that improves the trade-off between transaction speed and confirmation reliability in the Bitcoin network. In addition to our double-spending experiments that quantify this trade-off, we implemented the fast transaction mechanism in a common snack vending machine that now accepts bitcoins as a payment and dispenses the product within seconds.

II. Bitcoin

Bitcoin was introduced as a peer-to-peer-based digital currency in 2008 by Satoshi Nakamoto [1]. Traditionally, cashless transactions between two entities require an issuing/clearing authority which will act as a trusted third party. In Bitcoin, the entire computer network fulfills the role of the trusted third party for transactions between accounts, where nodes in the network propagate and verify transactions. The nodes implement a replicated ledger that keeps track of the account balances, verifies transactions against its current state and updates account balances accordingly. In contrast to other cashless payment systems, Bitcoin transactions are irreversible once they have been accepted by the network. As a consequence, Bitcoin has comparatively low transaction fees and no charge-backs, the downside being that money lost through theft or fraud is non-refundable.

A transaction in the Bitcoin network describes the transfer of a specific amount of bitcoins from one account to another. Transactions are represented as signed data structures that are broadcast into the Bitcoin network and recorded by the nodes. It is comprised of references to one or more previous transactions that funded the spending party and an assignment of a specific amount of bitcoins to one or more addresses. A transaction is atomic in the sense that the claiming of bitcoins from the previous transactions is inseparably linked to the transfer of these bitcoins to the receiving account. The references to previous transactions are referred to as *inputs* to the transaction, whereas the account and the amount of bitcoins that it ought to receive are called *outputs*. After the transfer, the owner of the receiving account has new funds at her disposal to spend in future transactions. The balance of an account is the sum of the values of all unspent outputs owned by that account.

It is important to note that as soon as an output is claimed and used as an input to a transaction it may not be reused. This means that a single output may not be used as an input to multiple transactions. The violation of this principle is referred to as *double-spending*.

When a user wants to transfer bitcoins to another user, she creates a transaction specifying outputs as well as inputs and signs the transaction. The transaction is then broadcast through the network using a flood broadcast. Each node verifies that (a) the output value of the transaction does not exceed the input value, (b) outputs are spent only once and (c) that the

978-1-4799-0521-8/13/$31.00

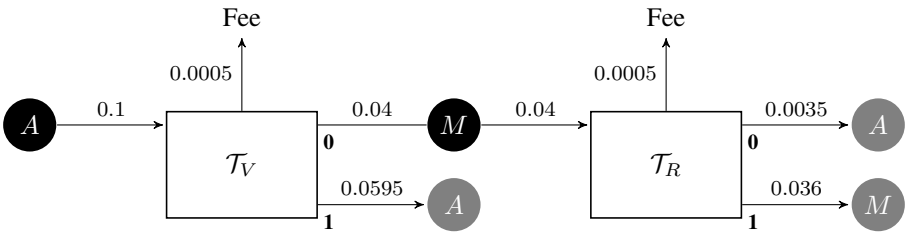

Fig. 1. Transaction chain for the return transaction $\mathcal{T}_R$ in the case of too many funds sent by the attacker. The attacker's wallet address is denoted by A, the current merchant's wallet address by M. The price of the good is 0.036 bitcoins and this amount would be effectively claimed by the merchant. The rest is returned to the attacker with fees deducted.

signatures match the sending account. If the transaction passes validation, the nodes will forward it to their neighbors.

Eventually, the network will confirm the transaction by including it into the public ledger. The transaction confirmation is a crucial step but also the most time-consuming. In expectation, it takes 10 minutes for the network to reach consensus about a set of transactions. A confirmation time in the order of minutes is undesirable for use-cases where purchased products are expected to be released to the customer immediately. However, merchants may choose not to wait for the confirmation of incoming transactions and release products as soon as they notice the transactions in the network. This is called a *fast payment*.

Fast payments build upon trusting a transaction to be eventually confirmed. It is a common suggestion by the Bitcoin community that the trade-off of not waiting for the definitive transaction confirmation should be accepted for low-priced goods where instant delivery is desired and possible loss of revenue marginal.

III. Related Work

Double-spends and fast payments were first analyzed by Karame et al. [2]. They found that double-spend attempts have a non-negligible probability of success. We expand on their result by considering a merchant that has a random sample of connected nodes in the network and does not accept incoming connections. Furthermore, we avoid isolating the merchant by not forwarding transactions destined for it and describe a method to securely return overpaid or incomplete payments.

The public ledger that tracks transactions in the Bitcoin network might be interesting for merchants as it would allow the creation of customer profiles. However, the initial publication by Nakamoto [3] claimed that transactions in the Bitcoin network are pseudonymous, which would make binding an account to its owner difficult. A first analysis of the anonymity was done by Reid and Harrigan [4], which used information from the publicly available ledger to connect multiple addresses to relinquish information about their owner. Ron and Shamir [5], among other things, attempted to infer patterns from individual transactions. ZeroCoin [6], a system that uses zero-knowledge proofs as a claiming condition of transactions, would allow truly anonymous transactions in which bitcoins can be acquired without a direct connection between sender and receiver.

With the recently published reports of the European Central Bank [7], the US department of treasury [8] and the FBI [9], Bitcoin received the needed legal status that allows its adoption at a large scale. Widespread adoption is a major requirement for merchants to begin using Bitcoin as a payment alternative. These reports were preceded by a first analysis of the legality of Bitcoin by Elias [10] in 2011.

IV. Securing Fast Payments

A merchant accepting fast payments incurs the risk of acting on a transaction that will not be confirmed by the network. This might result in a financial loss for the merchant.

An attacker attempting to defraud the merchant may try to double-spend the payment in order to receive the good or service from the merchant without paying for it. We assume that the attacker may connect to an arbitrary number of nodes in the network and broadcast any number of transactions claiming outputs in its possession. However, the attacker may not inhibit the communication between nodes and may not identify the neighbors of a node. To perform a double-spending attack, the attacker would create two transactions $\mathcal{T}_A$ and $\mathcal{T}_V$. Both transactions spend the same output and can therefore not both be valid. $\mathcal{T}_V$ denotes the transaction that transfers the required amount to the merchant, whereas $\mathcal{T}_A$ is a transaction that transfers the same amount back to the attacker. The attacker then attempts to convince the merchant about the validity of $\mathcal{T}_V$, while broadcasting $\mathcal{T}_A$ to the network at the same time. For the double-spending attempt to succeed two conditions have to be met: (a) the merchant should only see $\mathcal{T}_V$ until the good or service is released, and (b) $\mathcal{T}_A$ must be confirmed by the Bitcoin network, hence $\mathcal{T}_V$ would not be valid.

The risk for the merchant is further amplified by *information eclipsing* [11]. If the merchant forwards $\mathcal{T}_V$ to its neighboring nodes, they will verify and tentatively commit it to the local ledger. Should they later receive $\mathcal{T}_A$, it will not be considered valid as it conflicts with $\mathcal{T}_V$, and it will not be forwarded to the merchant. The merchant inadvertently shields itself against conflicting transactions like $\mathcal{T}_A$, and will be unaware of the double-spending attempt.

A. Countermeasures

To harden against the aforementioned attack, we propose several countermeasures. First and foremost, it has to be guar-

anteed that the attacker is not the only source of information. To this end, the merchant should connect to a sufficiently large random sample of nodes in the Bitcoin network. By doing so, the attacker cannot inject faulty transaction information to reach the merchant, because she does not know over which nodes the merchant communicates.

Secondly, the merchant should not accept incoming connections. Thus an attacker cannot directly send $\mathcal{T}_V$ to the merchant. Forcing the attacker to broadcast it over the network will ensure that $\mathcal{T}_V$ ends up in the local view of those nodes that forward it. Subsequent transactions using the same inputs, e.g. $\mathcal{T}_A$, would be duly ignored by those nodes, alleviating the risk that $\mathcal{T}_A$ ends up in the public ledger.

Furthermore, the merchant can effectively avoid isolation by not relaying transaction $\mathcal{T}_V$. This way, the attacker would have to shield all of the merchant's connected nodes from $\mathcal{T}_A$ in order to keep the merchant isolated. As soon as a single node is uninfluenced by the attacker, it will forward $\mathcal{T}_A$ to the merchant, thus informing the merchant of the attempted double-spend. In addition, by not relaying transactions, the merchant protects itself against timing attacks. An attacker could otherwise monitor and time relayed transactions in order to map out the merchant's connected nodes. The remainder of the network will ensure that the transaction is propagated to all nodes, hence not relaying the transaction has only a negligible effect on its propagation.

The merchant examines the *propagation depth* of $\mathcal{T}_V$, i.e., a listening period before the transaction is accepted by the merchant. Throughout this period, it would be imprudent of the attacker to attempt the double-spend, as any of the merchant's connected nodes could detect it. After the listening period, the legitimate transaction $\mathcal{T}_V$ will have propagated through a sufficiently large part of the network. If the attacker were to broadcast $\mathcal{T}_A$, only a minority of nodes in the network would forward it, thus drastically reducing the probability that $\mathcal{T}_A$ is ever included in the public ledger.

B. Return Chaining

The merchant needs a mechanism to return money to the customer. We can distinguish three possible transaction scenarios in which money is returned. Firstly, a transaction can be underspent and the insufficient funds are returned. Secondly, a transaction can be overspent, in which case the customer is entitled to change. Lastly, the customer could cancel the transaction, which would also make her entitled to the amount of money she already provided to the merchant.

In the presence of double-spending attacks, it is not only important how many bitcoins have to be returned, but also where they originated from. The merchant may simply create a transaction $\mathcal{T}_R$ from its account, that sends the amount to return to the customer's account. This transaction, however, would be valid independently from whether the customer's transaction $\mathcal{T}_V$ is valid or not. An attacker could attempt a double-spending attack and trigger a return payment. In case of an unsuccessful attack, the attacker would receive the return without losing anything. If the attack is successful, the attacker would receive the double-spent amount plus the return payment.

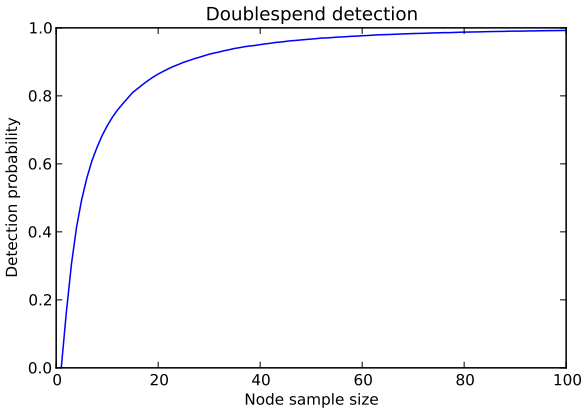

Fig. 2. Probability of the merchant detecting a double-spend attempt.

In order to avoid being vulnerable to this type of attack, the merchant has to ensure that if $\mathcal{T}_V$ is not valid, then also $\mathcal{T}_R$ is not valid as well. This automatic invalidation of the return transaction is possible by using a mechanism called *return chaining*. Return chaining means that the merchant claims the output created as a result of $\mathcal{T}_V$ and uses it as an input to $\mathcal{T}_R$ (cfr. Figure 1). This enables the merchant to immediately return change without the need of a confirmation. Should the attacker succeed in double-spending $\mathcal{T}_V$, the network would eventually reach consensus that $\mathcal{T}_V$ is not valid, the outputs would not be created and thus could not be claimed by any transaction. As $\mathcal{T}_R$ is a transaction claiming one of the outputs of $\mathcal{T}_V$ it would automatically be invalid.

V. Evaluation

To evaluate the accuracy of the proposed system to detect and reject double-spends we implemented the attacker and the merchant as fully automated systems. The double-spender generates two conflicting transactions by claiming a single output, hence double-spending it. It uses two network clients that are connected to a random set of nodes in the Bitcoin network to release the conflicting transactions. The nodes at which the transactions are released are chosen at random before each double-spending attempt in order to avoid favoring any particular configuration.

The merchant connects to a large random sample of nodes in the Bitcoin network and collects a network trace, i.e., it logs incoming transaction announcements from its peers, but does not relay any transaction in order to avoid isolating itself. The trace is used in the evaluation by selecting a random subset of the actually connected nodes, hence simulating multiple configurations and samples as they could have been observed in the real network. This allows a large number of evaluations while minimizing the number of double-spending attacks on the live network.

During the measurements 1922 double-spending attempts were initiated. The merchant was connected to 1024 nodes

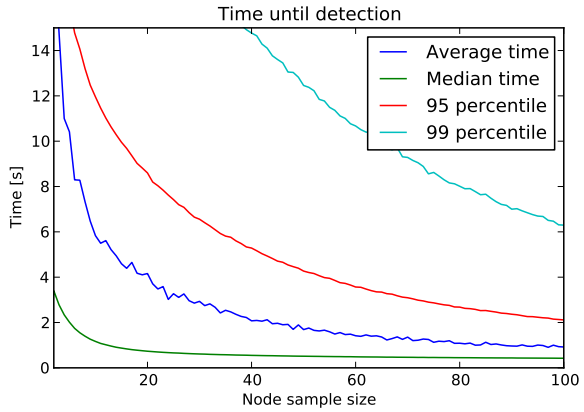

Fig. 3. Average time until the merchant detects the double-spending attempt. This does not include undetected attempts (see Figure 2)

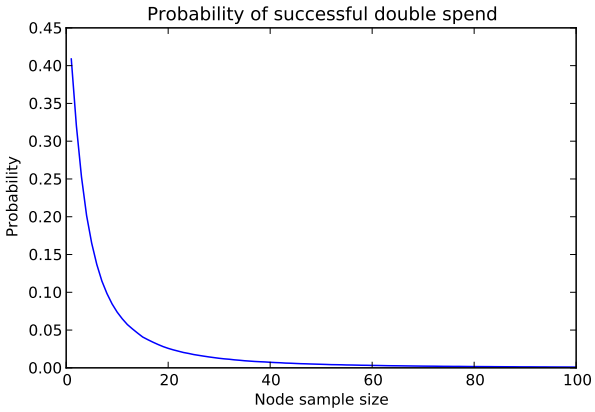

Fig. 4. The probability of a double-spending attempt being successful in relation to the node sample size of the merchant.

on average. Each additional connection causes an average of 104 bytes/second of additional bandwidth.

Although the transactions were released simultaneously and at the same number of points into the network, the ratio between the transactions observed by the merchants was not $1/2$ for either of the transactions. As soon as the balance between nodes seeing one transaction or the other is tipped to one transaction's favor, the other will be slowed down further so that the ratio the merchant observes deviates significantly from what was released into the network. The average ratio between the least distributed transaction to the most distributed transaction was 31.78%.

Furthermore, we observed that not all nodes announced transactions. Even though it is normal that nodes announce only one of the two conflicting transactions, there were nodes that did not announce either of them. On average 26.09% of peers did not announce either of the two transactions in a double-spending attempt. This lack of announcements is likely due to nodes not being up to date with the current ledger, and trying to synchronize the ledger state with the rest of the network. During the ledger synchronization the nodes do not relay incoming transactions as they might already have been confirmed.

To analyze the influence of the node sample size, we simulated the measurements by evaluating the double-spend detection repeatedly with varying subsets of the actually connected nodes. Each simulation was repeated 1000 times, resulting in 192,200,000 individual evaluations. Figure 2 plots the probability of the merchant eventually detecting a double-spend attempt, i.e., the probability of receiving an announcement for both conflicting transactions from its neighbors. At 100 nodes the merchant will not learn of a double-spending attempt in only 0.77% of all attempted double-spends.

As not all nodes announce transactions and we do not want to wait indefinitely until the transaction is confirmed by the network, a threshold has to be chosen after which to accept the payment. Introducing a threshold increases the probability of accepting double-spends as it reduces the window in which we attempt to detect the double-spend. Our measurements indicate, however, that the probability of not seeing both transactions decreases exponentially with the number of announcements received. The 99 percentile for the detection of a double-spend corresponds to 37 announcements. A combination of time-based and announcement-based thresholds should yield the best result.

As can be seen from Figure 3 the time until the merchant detects the double-spending attack quickly decreases for larger sample sizes. The 99 percentile is at 6.29 seconds for 100 peers. Hence, we set the threshold for the announcement count to 37 and the time to 6.29 seconds. The transaction is said to be accepted if both conditions are fulfilled.

So far, we have analyzed the probability that the merchant is not aware of a double-spending attempt. However, for the double-spending attempt to be successful two additional conditions have to be met. Apart from the merchant only seeing one transaction, that transaction also has to be $\mathcal{T}_V$, i.e., the one that transfers the bitcoins to the merchant. Finally, $\mathcal{T}_A$, which is the transaction that transfers the bitcoins to the attacker must be the one which is later confirmed by the rest of the network. With over 40 sample nodes, the probability that the first seen transaction will be confirmed is 73.64%. As we do not differentiate between $\mathcal{T}_V$ and $\mathcal{T}_A$ while sending, the probability of either being the first seen is $1/2$.

Hence, the probability of an attacker succeeding with a double-spending attempt is the product of the probabilities of the merchant first seeing $\mathcal{T}_V$, only seeing $\mathcal{T}_V$ and of $\mathcal{T}_V$ not being confirmed later. Figure 4 shows the probability that an attacker can successfully execute a double-spending attack against a merchant that listens to a random sample of nodes in the network. Taking into consideration the 99 percentile we used above to determine the thresholds, the attacker is left with a 0.088% chance of performing a successful double-spending attack. This means that the merchant has to hedge against a loss of one purchase in one thousand.

VI. Practical Example

To test the concept of secure fast payments in a real world scenario, we realized a working system, where people can actually pay in a few seconds using bitcoins. A fitting example for a venue where fast transactions are essential is a snack vending machine where purchases are largely spontaneous impulse purchases. This impulse is significantly suppressed should the drink or candy not be dispensed immediately. The classical Bitcoin transaction confirmation scenario is thus out of place.

Our solution is divided into a verification server that connects to the Bitcoin network and manages the company wallet, and a vending machine interface that connects to the server and handles the user interaction. It is noteworthy that a single server may verify transactions for any number of clients, keeping the impact on the Bitcoin network low. The vending machine interface is implemented as an Android smartphone application as this offers a display to interact with the user, enables communication with the verification server over mobile internet and, using the IOIO prototyping platform, allows to communicate with the vending machine.

When the customer selects a product in the vending machine, the vending machine interface displays a QR-code with the payment information, such as the price and the merchant's account. The customer scans the code and issues the payment. A progress bar informs the customer about the verification progress and, upon completion, the product is dispensed.

The server provides the central interface between the Android client and the Bitcoin network. For each vending session the server submits a payment address to the client that is displayed as a QR code. Once the customer transfers the necessary funds to this address, the server starts verifying the transaction by monitoring the Bitcoin network. As soon as the fast verification is complete, the result is sent to the client (i.e. smartphone in the vending machine).

The server's wallet holds all the addresses that are generated through payment requests over time. Moreover, it has a balance of bitcoins associated to it based on the public ledger. Newly generated addresses are intended to be one-time use, which means that the wallet will register transactions to a different address each time a payment is made by a customer. One obvious reason is that it enables to track orders much easier. Another reason is aiming at anonymity, in the sense that it is not possible to easily identify the server as a major payment recipient.

For the practical implementation of the fast payment verification, the server uses a threshold T to decide when the desired propagation depth has been reached. It will listen to incoming transactions to its wallet. As soon as such a transaction is first announced in the network, the server will begin to continuously monitor the transaction with respect to how many of the connected nodes in the random node sample have seen it. If the number of nodes that announced the transaction reaches T, the server will consider the transaction to be valid. In the meantime, should the transaction be included in the public ledger, it would immediately be considered valid by the server. In order not to keep customers waiting for too long, we considered verification times of ten seconds or less to be preferable. Based on our measurements in the Bitcoin network we chose $T = 40$ which is usually achieved in less than 10 seconds.

Fig. 5. The snack vending machine that accepts bitcoins. Note the display on the right that can show the according QR code or transaction information.

The vending machine is working correctly and our experiments show that not a single double-spending attack was successful against the prototype. A planned large scale test, together with the manufacturer, in Switzerland will show whether such a deployment could be successful and gauge interest in the wider public.

VII. Conclusion

We have shown that Bitcoin can be used as a reliable alternative for fast cashless payments. The low transaction fees of the network (compared to traditional centralized cashless payment processors) and the instant availability of the money to the merchant might render bitcoins interesting for vending machine operators.

References

[1] Satoshi Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash system," tech. rep., 2008.
[2] G. Karame, E. Androulaki, and S. Capkun, "Two bitcoins at the price of one? double-spending attacks on fast payments in bitcoin," in *Proc. of Computer and Communication Security*, 2012.
[3] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system,"
[4] F. Reid and M. Harrigan, "An analysis of anonymity in the bitcoin system," in *SocialCom*, 2011.
[5] D. Ron and A. Shamir, "Quantitative analysis of the full bitcoin transaction graph,"
[6] I. Miers, C. Garman, M. Green, and A. D. Rubin, "Zerocoin: Anonymous distributed e-cash from bitcoin," 2013.
[7] "Virtual currency schemes," tech. rep., European Central Bank, 2012.
[8] "Application of fincen's regulations to persons administering, exchanging, or using virtual currencies," tech. rep., Financial Crimes Enforcement Network, US Department of the Treasury, 2013.
[9] "Bitcoin virtual currency: Unique features present distinct challenges for deterring illicit activity," tech. rep., Federal Bureau of Investigation, 2012.
[10] M. Elias, "Bitcoin: Tempering the digital ring of gyges or implausible pecuniary privacy," *Available at SSRN 1937769*, 2011.
[11] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in *IEEE Internation Conference on Peer-to-Peer Computing (P2P), Trento, Italy*, September 2013.