Anonymous Online Payment Systems

Sebastian Straub

2014-12-16

Hauptseminar Technischer Datenschutz WS14/15

***Abstract*—Contemporary online payment systems are fast and convenient, but not everyone is willing to accept the growing number of privacy interferences by popular providers like PayPal or Amazon. In this paper we will evaluate cryptocurrencies as an alternative to conventional payment services in terms of user privacy. We will review the privacy issues of the well-established cryptocurrency Bitcoin and give an overview of concepts and proposals that aim to tackle these issues.**

I. Introduction

Today there is a variety of digital payment systems in existence which have overtaken cash in terms of revenue years ago, for good reason: They are instantly available, offer suitable protection from theft and many of them operate worldwide.

Yet still, current digital payment systems are not superior to cash in every aspect: With cash, you can pay anonymously, without anyone tracking your purchase and without concluding any contracts beyond the purchase. There is already a number of privacy issues with the widely-used payment cards like credit or debit cards, where your bank can track your transactions, retailers can identify you by the credentials stored on the card and you have to agree to a variety of obscure contracts whenever you make a payment. It gets even worse with contemporary online payment systems, where not only the provider tracks and evaluates all your purchases but the terms and conditions often even allow the resale of certain personal information to third parties.

This outgrowth of excessive customer surveillance created a demand for payment systems that transfer the functionality of cash on the Internet. A system that works independently of institutions that control the data of their customers.

One of the more successful attempts to realize this goal are cryptocurrencies, a class of decentralized payment systems that rely on free software and cryptography to prove the intentions of it's participants. The first publicly traded cryptocurrency was Bitcoin, which launched in 2009 and is by far the most valuable cryptocurrency today with a market capitalization of more than $4 billion at the time of writing.

While today's cryptocurrencies are not controlled by central authorities and accounts are not associated with the names of their owners, most of them offer even less privacy than traditional payment systems. The predominant reason for the privacy problem is the public transaction record which is a requirement for the functioning of all contemporary cryptocurrencies, but there are also some other issues which will be discussed in this paper.

First, we will give a short technical review of Bitcoin as an example for the core functionality of cryptocurrencies. Based on this introduction we will then analyze the privacy issues of Bitcoin in particular which are also shared by most other cryptocurrencies today. Afterwards, we will give an overview of concepts that aim to increase privacy for users of cryptocurrencies before we draw a conclusion of the current state and the possible future of privacy in cryptocurrencies.

II. Bitcoin - a Technical Introduction

To get an understanding of the privacy implications in cryptocurrencies today as well as the proposals to solve some of them, a basic model of the functionality of cryptocurrencies is required. In this chapter, we will create such a model that should be detailed enough for our needs while hiding as much complexity as possible.

The model will be based on the popular cryptocurrency Bitcoin, which was publicized in 2008 under the pseudonym Satoshi Nakamoto [Nak08]. Though our model will specifically reference implementation details of Bitcoin, they can be generalized to match properties of the majority of cryptocurrencies in existence today.

We will explain, how user-generated addresses can be used to receive payments, how transactions are performed and what the purpose of mining is.

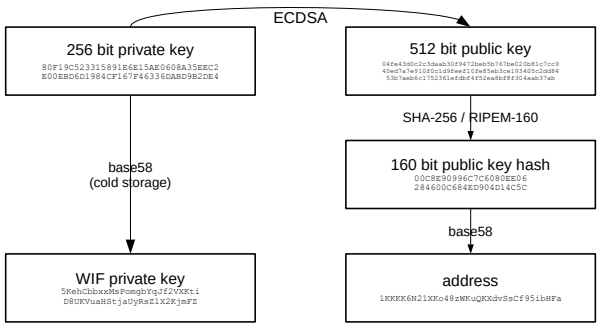

Figure 2. Bitcoin Addresses

A. Addresses

In order to receive payments, a user needs some kind of unique identifier that she may pass on to the sender, like a bank account number. Furthermore, the recipient needs to have the ability to prove that she and only she

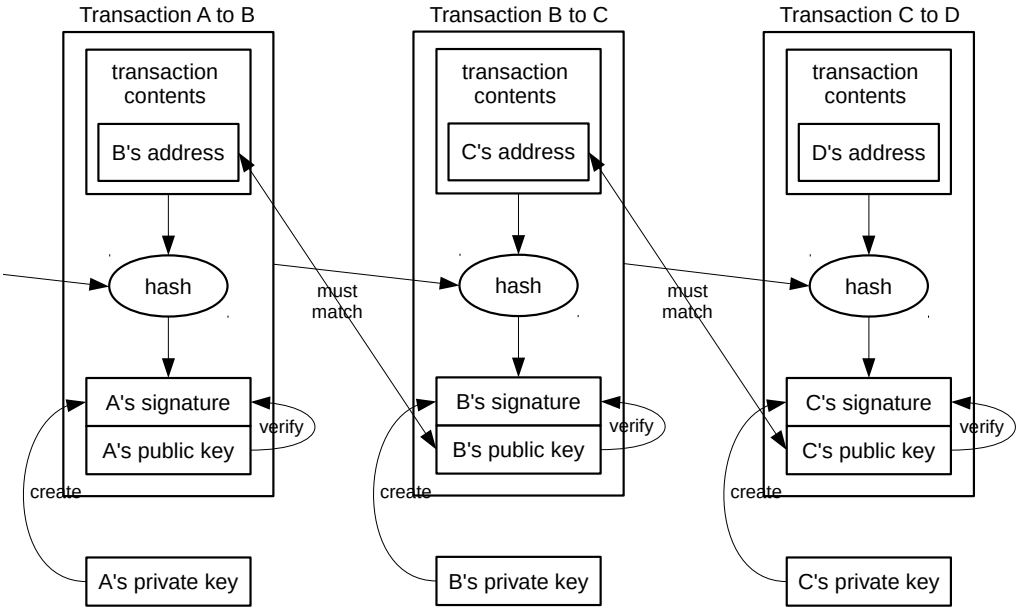

Figure 1. Bitcoin Transactions

is the rightful owner of that address. A design goal of cryptocurrencies is to work without a central entity like a bank that might assign account numbers and verify the authenticity of it's customers; therefore, public-key cryptography is used.

With a self-generated public and private key-pair, three problems are solved:

- the public key can be used as an account number that can be passed on to third parties
- the private key can be used to authenticate a user as the owner of an account (or rather, the corresponding public key)
- users can generate their own account numbers and authentication codes

In Bitcoin, in order to create a new address, a user first generates a 256 bit private key. By applying the elliptic curve digital signature algorithm (ECDSA), a corresponding 512 bit public key is derived.

This public key could already be used to receive payments, but in Bitcoin, an address consists of a hash value of the public key in a user-friendly base58-encoding. There are various reasons for this hashing, the predominant one being to create a shorter address which can be saved on paper if required. But this approach comes at a cost: The Bitcoin address cannot be used to verify the integrity of a signature. To check if someone is the owner of an address, the actual public key is still required. We will see in the next chapter, how this issue is circumvented.

B. Transactions

When we transfer physical cash from one wallet to another, individual coins and bills have to be moved, thereby giving the owner of a wallet access to the cash and permission to spend it.

In Bitcoin, the smallest unit of the currency is a *satoshi*, which corresponds to 10^{-8} BTC. But rather than transferring satoshis and other coins denoting multiples of a satoshi between addresses, Bitcoin relies on the concept of unspent transaction outputs (UTXO).

The output or destination of a transaction is called unspent when it has never been used as input or source for another transaction. Every UTXO points to a Bitcoin address and it is the privilege of the private key owner to that address to use this output in a new transaction. An UTXO that points to an address is what gives that address value and client software usually displays the sum of all UTXO as the *amount of coins in a wallet*, which is nothing more than a convenient abstraction.

Every transaction is part of it's own continuous chain that can be traced back to it's origin when the coins it contains have been mined (see next section). This chain is stored in a public record which makes it possible for anyone to verify the integrity of the chain and the validity of an attempt to spend an UTXO.

We will now have a look at how transactions are created and verified by the P2P network.

1) The most basic transaction: Let's assume Alice has created a transaction with one output that points to Bob's address, and Bob now wants to spend this UTXO and forward all it's value to Carol.

Figure 1 illustrates the core elements of such a transaction. Let's assume for a moment, that Alice's transaction is valid.

Bob can now create a new transaction that contains Alice's transaction as input and Carol's address as the only output. Bob calculates a hash value from the parameters of

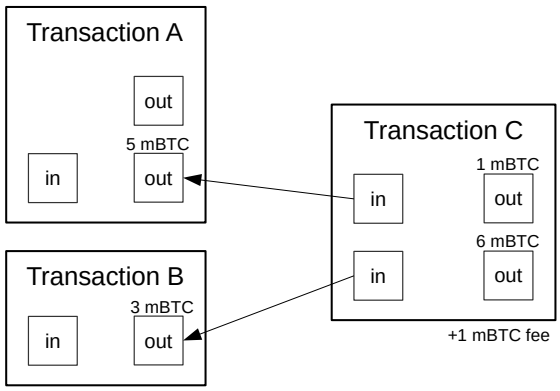

Figure 3. Transaction inputs and outputs

his own transaction which he just defined and the entire previous transaction from Alice that was used as input. Finally, Bob signs this hash value using his private key and appends this signed hash as well as his public key to the end of the transaction. Then he announces his transaction to the P2P network.

Now, any participant of the network can verify

- that Bob's address can be derived from the public key he provided (it must match the address Alice has defined as output in her transaction)
- that Bob is the owner of the address by validating his signature using the provided public key
- that the transaction was not tempered with by recalculating the hash value

Of course, before Bob attempts to create a new transaction, he would have used this technique to verify the transaction from Alice first, just as Carol can now check if Bob's transaction was valid.

2) Multiple in- and outputs: As we have seen, UTXO can only be spent as a whole, but we still want the possibility to send arbitrary amounts of coins. To achieve this, transactions with multiple in- and outputs are used.

Let's assume we want to send some coins, but no UTXO we control contains the exact amount of coins we want to send (which is very likely to happen). If however we have a UTXO with *more* coins than we want to spend, we can create a transaction that has two outputs: One with the exact amount we want to send to the destination address, and one with the change we want to keep and which we will send back to our own address.

If we don't have an UTXO with enough balance, but a number of smaller UTXO that together exceed the amount we want to send, we can combine them in a transaction that uses multiple inputs. Every transaction gives us the right to spend as many outputs as we want as long as the sum of outputs is less than or equal to the sum of inputs. Of course, this means that we can also apply the previous change-address pattern to multi-input transactions. If the sum of outputs is less than the sum of inputs, the leftover value is spent as transaction fee (see Mining).

Just as a side note: Current Bitcoin clients do not send the change back to the source address, but instead to a new address which is also controlled by the sender. In fact, clients try not to reuse addresses at all to increase privacy. All addresses are generated on-the-fly and are derived from the single master keypair that was originally created by the user.

3) Transaction scripting: All Bitcoin transactions are written in a scripting language that is simply called *script* [Wika]. It is a non-Turing-complete language which was specifically designed to allow for complex contracts between multiple parties while still allowing efficient verification through participants of the P2P network. Therefore, the instruction set was intentionally reduced so any script is guaranteed to terminate in finite time (less than a few milliseconds, in practice).

Transaction scripts allow to set up interesting contracts, like transactions that will only be executed if the majority of parties signs them, or a deposit system where the sender can be sure that his deposit is returned after a certain amount of time. In chapter IV we will see how transaction scripts can be used to increase the privacy of participants.

C. Mining

Though Bitcoin is a decentralized currency, there is still a need for consensus upon a transaction history that is mandatory for every participant. If this was not the case, there would be constant dispute on the network about who would own which transaction outputs, making the use of the cryptocurrency non-binding and insecure.

Another problem we haven't considered yet is money creation. How can we achieve a sufficient money supply based on a fair and preferably rewarding principle?

Mining [KDF13] is a process that solves both of these problems in an elegant manner. Bitcoin miners are ordinary participants of the P2P network that collect all transactions they receive form the network. The transactions are verified upon retrieval and if they turn out to be valid, they are included in a *block*. A block is a data structure that combines a set of valid transactions with some additional metadata. The idea is that miners create blocks in regular intervals, and each block has to contain the hash value of the previous block, therefore forming the transaction history, an unbroken chain of blocks: the *blockchain*.

What makes mining hard and competitive is not that forging a valid block would be very difficult. Whether a block is accepted by the network depends on the solution of a computationally very intensive crypto puzzle, which limits the amount of blocks that are found (or *mined*) to only a few blocks per hour. Bitcoin miners are working concurrently on new blocks, each on her own version, based on the transactions received. In the end, only those transactions that are included in a successfully mined block will become part of the blockchain and therefore *confirmed*. It is of course in the best interest of the miners to include as many transactions as possible in a block to

receive the transaction fees that were discussed earlier (see II-B2).

A block is considered to be *successfully* mined, if it fulfills two requirements

1) the content is valid (only valid transactions were included and the metadata is correct)
2) the SHA-256 hash of the block is smaller than a certain integer, the *target*

The first condition is easily fulfilled (at least in terms of processing time), but the second one is hard: The only way to compute a different hash code for the same block is to vary the content slightly without invalidating it. There is a special field in the metadata, the *nonce*, which only exists for that purpose. Still, there is no way to know which input will generate a hash code that meets the requirements, so the only known way to solve the problem is brute force. By changing the nonce, adding new transactions and recalculating the hash with every change, at some point a hash code will be found that is smaller than the target.

The task of calculating billions of hashes serves no end in itself (in fact, it is a huge waste of energy and processing power), but it ensures that

- block-generation is expensive and therefore unprofitable for attackers
- no central instance can dictate the contents of each block if enough miners compete (the probability of success only depends on the own processing power compared to that of the other miners)

To create an additional incentive, a reward is paid to the successful miner of a block. The reward was not part of the previous money supply, so this is a process of money creation. Every transaction that is part of the blockchain can be traced back to such a reward for a successfully mined block.

III. Review of Privacy in Bitcoin

After we have built our basic model of a cryptocurrency, it is not immediately obvious what serious privacy implications should come from it: Sure, the full transaction record is publicly available, but no Bitcoin address is directly linked to any personal information, addresses are not reused and there is no central entity which routinely tracks the activities all users.

This argumentation probably lead to the widespread claim that Bitcoin was *the* truly anonymous currency. The fact is that Bitcoin currently offers less privacy, at least towards the general public, than most online payment systems, for a number of reasons: The IP-Addresses of all participants are visible in the P2P network, Bitcoin addresses can be linked to user identities whenever a payment is made and all transactions are irrevocably part of the public blockchain which anyone is free to evaluate.

In this chapter, the mentioned privacy implications will be elaborated in more detail.

A. IP-Address Tracking

The heart of a decentralized cryptocurrency is an active P2P network. The network must be publicly accessible and each participant has to know the IP-addresses of a certain amount of peers to be able to effectively relay messages. But this does also mean that an observer can get a good overview of the network with a relatively small number of nodes.

If an observer can record the traffic of the majority of nodes in the network, she will be able to determine at which node a transaction has been published for the first time and therefore have linked an IP-Address to a Bitcoin-Address.

Of course, the IP-Address is not of much use, if it comes from the exit node of a VPN or an anonymity network. But an attacker can temporarily block users of anonymity networks like TOR [BKP14], thereby forcing the participants to either stop using Bitcoin until the ban is lifted or switching to a different IP address. This is accomplished by abusing the DoS protection schemes that are part of the Bitcoin protocol. If a client receives a malformed message, it will not be relayed and the sender of the suspicious message gets a penalty score. If the penalty exceeds a certain amount, the client will stop accepting messages from this peer for 24 hours.

It is possible to send a malformed message that is considered by standard clients to be so “wrong” that it will cause an immediate ban. An example of this would be a block with an empty transaction list, which is a message with a length of just 81 bytes. An attacker can connect to a TOR exit node and send one of these zero-tolerance messages to the most active nodes in the Bitcoin network, which would make it very difficult for another user of this exit node to connect to the Bitcoin network. If the attacker repeats this procedure for all TOR exit nodes, all TOR users will be unable to connect to the Bitcoin network for the next 24 hours.

Connecting to each TOR exit node and sending a few kilobytes of data is not a trivial task, as there are currently more than 1000 public exit nodes available, but as the number of connections from each exit node and the amount of data to send is relatively low, this attack is feasible. [BKP14]

B. Thin Clients

So far we have assumed that every Bitcoin-user is an active participant of the P2P network, using a *full-chain* client. A full-chain client holds a local copy of the entire blockchain and is therefore able to verify the integrity of each incoming transaction. Unfortunately, the firewall policy in many networks (including most home networks) prevents clients from accepting incoming connections. This means that the client will still be able to transmit her own transactions to a few participants which use a static IP-address, but it will not be possible to hide one’s own transactions in the constant flow of other messages on the network.

Many users avoid full-chain clients for various reasons: The local copy of the blockchain can grow to several gigabytes, the P2P network causes a lot of traffic and synchronization can take several hours if the blockchain was not updated in a while. *Header-only* clients [Wikb] are a popular alternative to circumvent these issues. This type of client does only store the header information of each block, stripped of all transaction contents, thereby reducing the blockchain size to only a few megabytes. But working without a copy of the blockchain does comes with significant drawbacks:

- no incoming transaction can be verified; the client can only check whether the transaction ID has been included in a block
- as the integrity of incoming transaction cannot be verified, header-only clients cannot participate in the P2P network
- it is trivial for an attacker to link IP-Addresses to transactions if a client does not relay network messages

Another popular client option for casual users are the *server-trusting* clients or *online wallets* [Wikb], as they are more commonly called. An online wallet is a web service where the user entrusts the server operator to handle all transactions for them. This can actually increase privacy and security for a user, if the service provider is trustworthy, as the operator usually runs a full-chain client and the transactions of multiple users are handled from one IP address, making it hard for an attacker to associate private IP addresses to Bitcoin transactions.

But having a single service provider also introduces a single point of failure which turns them into an interesting target for attackers. Using an online wallet also means storing (encrypted) private keys on a web server and giving the server access to the private key from time to time to sign transactions. All in all, online wallets may be convenient, but they are not designed in a way cryptocurrencies were meant to be used.

C. Transaction partners

There are various transactions one can make without revealing one's identity: making a purchase at a retail store or in a restaurant for example. There is an increasing number of stores that accept Bitcoins or other cryptocurrencies. But often one cannot avoid to reveal personal information, e.g. in online shopping or when a contract is closed, and therefore it cannot be avoided that certain public keys or addresses can be linked to one's identity.

This can be problematic, as we will see in the next chapter (Blockchain analysis), because other activities of a user can be derived from a single known address. This means that a single leaked address-identity pair can turn the entire payment history of a user into public knowledge.

Currency exchanges are another instance which most users of a cryptocurrency today can hardly avoid, but which are in most jurisdictions required to verify the identity of it's customers. There are some interesting alternatives like ATMs which can change cash into Bitcoins and vice versa, but there's only a few of them in the world and the transaction limits are very low.

D. Blockchain analysis

The blockchain is the public record of all transactions that have ever been included in a successfully mined block and a valuable source of information for anyone who wants to analyze the flow of coins in the network. Spagnuolo et al. have implemented a system called Bitiodine [SMZ14] which parses the entire blockchain and runs various search and clustering algorithms to make sense of the transaction history by identifying clusters of transactions that belong to a single user.

This is not a trivial task, as a user can generate a new address for each transaction and the most popular Bitcoin clients generate new addresses on the fly, whenever they are needed. Bitiodine applies a number of heuristics to determine which addresses belong to the same user. Two examples:

Multi-input transaction grouping: This heuristic exploits the fact that transactions with multiple inputs are usually created to combine several smaller transaction outputs in order to create a sufficient balance for a larger output. So for this kind of transaction, one can assume with a certain probability that all input transactions are controlled by the same user.

Shadow address guessing: This heuristic exploits the handling of change, which occurs in almost every transaction, as it is highly unlikely that a user has a transaction output available that matches the exact amount she wants to spend. For a transaction with two outputs, one can assume that one of them is the destination address and one is the sender's change address.

But which one is the change address? If the transaction has multiple inputs, the change address is probably the one with the lower value, or else there would not have been the need to combine multiple inputs in the first place. If one of the addresses is already part of the blockchain and the the other is not, then the so far unknown address is the change address (as it was probably just generated by the sender). And due to an interesting bug in the official Bitcoin client, which is responsible for the vast majority of transactions in the first years of Bitcoin, we know that the change address of almost all transactions before February 2013 is in the first output.

For transactions with just one input or more than two outputs or more sophisticated scripts these heuristics will fail, but they work very well with the majority of transactions that can be found on Bitcoin's blockchain today.

IV. Increasing Privacy

In chapter III we have seen that Bitcoin is not the anonymous currency which it has been praised for in the media. On the other hand Bitcoin is not broken by

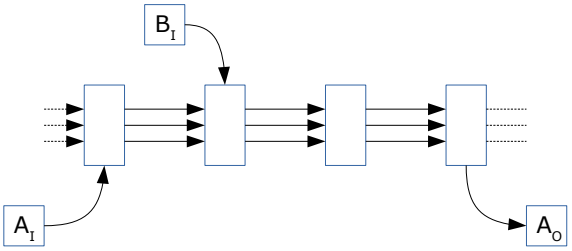

Figure 4. Mixing service

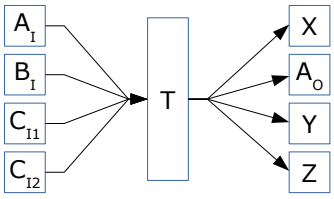

Figure 5. CoinJoin

design - in fact, using Bitcoin as it was intended by the developers provides far better privacy for all users than what is common practice today. This includes

- using a full-chain client
- active participation in the P2P network
- no reuse of addresses

As an additional measure, one can further increase privacy by using anonymity networks.

But there are a few reasons for concern that cannot be eliminated: The public transaction record is an integral part of any cryptocurrency, therefore the blockchain will remain a target for investigators. And it cannot be avoided that single entities will be able to link an address (or public key) to the identity of an individual.

Therefore, most attempts to increase privacy on a technical level target the blockchain. The goal is to hide the activities of users on the blockchain, which would prevent outsiders from creating user profiles and therefore would not expose a user's activities if a single address can be associated with a name.

A. Mixing Schemes

The problem of blockchain analysis was recognized very early by the Bitcoin community and there has been a variety of attempts to hide the origin of transactions by using the tools that the Bitcoin protocols provides. Mixing schemes rely on the characteristic of transactions that multiple inputs and outputs are not directly linked. The idea is to combine the inputs of several users in a single transaction and then to distribute the outputs in a way that it is hard to draw a line between a single input and output. Applying this process for several rounds generates transaction outputs with a highly diversified origin that cannot be linked to other addresses of a user.

1) Escrow: The so called *mixing services* are based on an escrow agreement between the service provider and her customers.

The operator of the service has a sufficiently large amount of transaction output under her control, which she uses to generate a constant flow of transactions that randomly select transactions as input and randomly generate new outputs.

If a customer wants to use the service, she creates a request and provides a new destination address that is controlled by the customer. The operator then assigns one of her own addresses to this customer and demands a payment to this address. The customer transacts the desired amount to the provided address and the operator includes this output into her mix of transactions. After a certain number of iterations, one of the outputs from the transactions will point to the destination address that was provided by the customer.

Möser, Böhme and Breuker [MBB13] investigated the effectiveness of such services and came to the conclusion, that some of the reviewed service providers were able to "successfully anonymize [their] test transactions".

A major disadvantage of escrow mixing services is that customers lose control of their transactions and therefore have to trust the provider to return their input at some point. Due to various reports of fraud and the predominant use for money laundering purposes, such mixing services have gained a bad reputation in the Bitcoin community.

2) CoinJoin: To mitigate the trust issues with escrow mixing services, CoinJoin was announced in 2013 on Bitcointalk [gma]. It is based on the same principle of combining transactions of multiple users, but does not rely on a service provider to do the mixing for the participants. Instead, transaction scripts are used that allow multiple participants to agree on a transaction where each participant can define her own inputs and outputs. This way, each participant never loses control of her coins and can still hide the own transaction among other in- and outputs.

Transaction scripts were already mentioned in section II-B3 - they allow multiple parties to set up contracts that work within the Bitcoin protocol. This is possible because each transaction input has to be signed separately. In a basic transaction, all inputs belong to the same user and are therefore all signed using the same private key, but in CoinJoin-transactions, the inputs of multiple users have to be combined and therefore the transaction only becomes valid after all inputs have been signed by the respective owners.

In practice, the participants of a transaction have to meet somehow and agree on a chairman that collects all desired transaction inputs and outputs of the participants. The chairman then creates a new transaction that contains all these inputs and outputs, signs his own transaction inputs and sends the unfinished transaction to the other participant. The other participants can now check if their desired inputs and outputs are part of the transaction and will only sign their own inputs if the contents match their expectations. After all inputs have been signed, the

transaction becomes valid and is ready to be included in the next block. If one or more participants are not satisfied with the transaction, they can deny their signature and the transaction will not become valid, meaning that all participants have to start over.

Because a single participant has the ability to cancel an entire transaction for all participants at basically no cost, this scheme is quite prone to trolling and DoS-attacks. That is why online services like SharedCoin [Shaa] have been created to take the role of the chairman and automate the CoinJoin process. SharedCoin is a free software [Shab] online wallet service where users trust the operator to include their intended transaction into a CoinJoin transaction. This means that participants do not have the ability to refuse their signature, as the SharedCoin service automates the process of combining and signing transactions. But the advantage of a convenient service is again overshadowed by an involved third party that all participants have to trust with their private keys.

Furthermore, recent research has shown that a single CoinJoin iteration is insufficient in providing good privacy to it's participants. [Atl14]

B. DarkCoin

In 2014 the new cryptocurrency *DarkCoin* [DH14] has been launched. It is based on Bitcoin code with a few topical enhancements, but the unique feature is the fully integrated CoinJoin mixing scheme called *DarkSend*, as part of the DarkCoin protocol. The idea is that every participant of DarkCoin's P2P network can constantly participate in CoinJoin mixing while the client is running, without the need to trust a third party. This way, certain transaction outputs can be "anonymized" before they are needed and can later be used even with less secure thin clients without revealing other user activities.

DarkSend relies on a number of *master nodes* that are regular participants of the P2P network but share a few additional responsibilities with the miners. They act as facilitators for the participants of each CoinJoin transaction and are awarded 20% of the output of each successfully mined block for the service they provide. However, each master node is required to make a large reserve payment so the network is not overrun by malicious master nodes and there are penalties for master nodes that provide harmful services.

If a DarkCoin user decides to use DarkSend, the client software would randomly select a master node and declare the desired transaction inputs and outputs. The master node waits for other requests and creates a new transaction for all participants to sign. The client checks if the transaction is correct and signs or refuses the signature. This process requires no user interaction and the private key always remains on the client side, the only trust involved is towards the free software client [Dar].

A malicious fork of the DarkCoin client could still be used to participate in DarkSend mixing and block the system by refusing any signature. Therefore a penalty system was introduced: Each participant has to provide a deposit for a DarkSend transaction or the master nodes will refuse the request. If a client leaves early or a signature is refused for no reason, the deposit get drawn in. To protect participants from malicious master nodes, the deposit can only be collected by the DarkCoin miners, and furthermore, the last two DarkCoin miners have to agree that a participant unjustifiably refused the signature. This should make DoS attacks very expensive and create sufficient protection from abuse through miners, as long as the majority of miners intends to defend the currency. Miners are usually organized in mining pools to stabilize their income and the idea is that miners would avoid pools that cash in deposits without justification, in order to keep the currency alive and therefore their own source of income.

C. HD Wallets

A different approach to impede blockchain analysis is the use of *hierarchical deterministic wallets* or *HD wallets* [Wui12]. As we have seen in chapter III-D, transactions with just one input and at most two outputs are hard to analyze. Therefore it would increase privacy if we had a convenient way to exchange sufficient amounts of addresses so that a combination of transaction inputs is not required any more and the use of change addresses is minimized.

The goal of HD wallets is to "set a standard for deterministic wallets that can be interchanged between different clients" [Wui12]. In Bitcoin, a user has one master key and one subkey for each Bitcoin address under her control. The subkeys are randomly generated and cannot be restored if the wallet is lost, even if a backup of the master key is available. HD wallets solve this problem by providing extended keypairs that can be used to deterministically derive subkeys, which themselves can be used to recursively derive further subkeys. Therefore, a backup of an extended private key is sufficient to derive a tree of all subkey pairs.

An interesting property of the specification is that there are not only extended private keys but also extended public keys which can be used to derive only it's public subkeys without access to any private key. So when a transaction partner is handed an extended public key, she can derive as many subkeys and therefore Bitcoin addresses as necessary to make a transaction without combining any transaction inputs.

Unfortunately, this also implies that by leaking the topmost extended public key, all transactions that belong to one of it's subkeys can be associated with the owner of that key. Therefore it should be ensured that only extended public keys at the lowest level of the hierarchy are used for transactions and never reused. In any way, a convenient client implementation is required that manages extended keypairs and handles transactions that use extended public keys, as well as user awareness for the privacy concerns related to public key leakage.

While HD wallets provide better privacy than regular Bitcoin transactions, the predominant design goal is

improved key management. Using just HD wallets will probably not provide sufficient privacy for sensitive users.

D. Zerocash

Zerocash [BSCG+14] is the specification of a currency that offers truly anonymous transactions based on zero-knowledge proofs. Originally specified as an extension to Bitcoin, the originators are now planning to release it as a separate cryptocurrency and are therefore implementing their own client based on the Bitcoin codebase [Zer].

Zerocash will combine two separate currencies in a single system:

- *Basecoin*: A base currency that will share most properties with contemporary cryptocurrencies.
- *Zerocoin*: An anonymous currency that will allow for private transactions between participants. Basecoins can be converted 1:1 in zerocoins and vice-versa.

Basecoins are converted to zerocoins using a so called *mint transaction*. The transaction consists of a cryptographic commitment to a new transaction output in zerocoin that resembles the value of the transaction output in basecoin. As with the blockchain, these commitments are stored in a public ledger and replicated on each node of the network so anyone can verify that the commitment is valid.

To make actually private transactions within Zerocoin, *pour transactions* are used. Such a transaction proves, in zero knowledge, that the user owns all input coins, that each of the inputs already occurred as mint or pour transaction and that the total value of the input coins equals the value of the output coins. Only the serial numbers of consumed zerocoin transaction outputs are stored in a public ledger, the addresses involved or the value of the transaction remains hidden. Anyone can verify that the zero-knowledge proof contained in a pour transaction is valid without access to sensitive information (private keys, zerocoin addresses, etc.).

V. Conclusion

In our short review of privacy issues in contemporary cryptocurrencies we have seen that there is a number of serious challenges that have not yet been resolved. Tracking of IP-addresses is an issue if participants are not using a full-chain client and connecting through anonymity networks with incoming connections allowed. We have seen that it cannot be avoided in day to day business that transaction partners will be able to link an identity to a single Bitcoin address and we have seen that it is possible through blockchain analysis to find clusters of addresses that likely belong to the same user. This means that a leak of a single address-identity-pair has the potential to reveal all past and future activities of a participant.

But there is also a slew of concepts with a few actual implementations to tackle these issues. We have discussed different types of mixing schemes which try to eliminate the correlation between two addresses that belong to the same individual. The conventional mixing service providers can be very effective in hiding correlations between transaction outputs, but they cannot eliminate the trust issues against the providers themselves, as they have the ability to keep transaction logs, and of course the customer has no guarantee that any funds will ever be returned.

CoinJoin turns out to be an interesting alternative, as it solely relies on transaction scripts to create contracts between a number of participants to create transactions where it is hard to associate inputs with outputs. Unfortunately, CoinJoin is prone to DoS attacks and requires too much user interaction and offers too little privacy to be an effective privacy enhancement. DarkCoin is a new cryptocurrency which raises CoinJoin to it's full potential: A tight integration of a constant CoinJoin mixing scheme into the DarkCoin protocol makes it possible to anonymize coins ahead of time. Therefore the user has always transaction outputs at hand which cannot be traced back to past activities. This does even allow the use of less secure thin clients, e.g. on mobile devices, as a leak of private information can only compromise a single address, not the entire transaction history.

We have seen that HD wallets can impede blockchain analysis by discouraging transactions with multiple inputs while providing a convenient way to split multi-input transactions into a number of single-input transactions. Yet the main purpose of the system is improved key management and it probably cannot prevent exposure through more sophisticated blockchain analysis techniques. Finally, we had a look a Zerocoin, a proposed cryptocurrency which could provide truly anonymous transactions, based on zero-knowledge proofs.

While the current state of cryptocurrencies in terms of privacy is moderate at the best, there is a number of alternative approaches, partly with working implementations, that offer a fair amount of privacy for it's users. Especially DarkCoin, which has only recently ascended in the top ten of cryptocurrencies by market capitalization, is an interesting new player in the field. And the authors of ZeroCash, which could be a candidate for the title "first *actually* anonymous cryptocurrency", have already announced that they are working on an implementation of their concept.

The originator(s) of Bitcoin decided to sacrifice some privacy to successfully exclude any central authority from the system. What we see now, is a community working on a process to reclaim that privacy without sacrificing the newly gained independence. Cryptocurrencies have the potential to become a worthy successor for cash in the digital age.

References

[Atl14] Kristov Atlas. Weak privacy guarantees for sharedcoin mixing service. 2014. http://www.coinjoinsudoku.com/advisory/ accessed at 2014-12-11.

[BKP14] Alex Biryukov, Dmitry Khovratovich, and Ivan Pustogarov. Deanonymisation of clients in bitcoin p2p network. *arXiv preprint arXiv:1405.7418*, 2014.

[BSCG+14] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *Security and Privacy (SP), 2014 IEEE Symposium on. IEEE*, 2014.

[Dar] Darkcoin at github. https://github.com/darkcoin/darkcoin accessed at 2014-12-13.

[DH14] Evan Duffield and Kyle Hagen. Darkcoin: Peer-to-peer currency with anonymous blockchain transactions and an improved proof-of-work system. 2014.

[gma] gmaxwell. Coinjoin: Bitcoin privacy for the real world. https://bitcointalk.org/index.php?topic=279249.0 accessed at 2014-12-10.

[KDF13] Joshua Kroll, Ian Davey, and Edward Felten. The economics of bitcoin mining, or bitcoin in the presence of adversaries. In *Proceedings of WEIS*, 2013.

[MBB13] Malte Moser, Rainer Bohme, and Dominic Breuker. An inquiry into money laundering tools in the bitcoin ecosystem. In *eCrime Researchers Summit (eCRS), 2013*, pages 1–14. IEEE, 2013.

[Nak08] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[OKH13] Micha Ober, Stefan Katzenbeisser, and Kay Hamacher. Structure and anonymity of the bitcoin transaction graph. *Future internet*, 5(2):237–250, 2013.

[Shaa] Sharedcoin. https://sharedcoin.com accessed at 2014-12-11.

[Shab] Sharedcoin at github. https://github.com/blockchain/Sharedcoin accessed at 2014-12-11.

[SMZ14] Michele Spagnuolo, Federico Maggi, and Stefano Zanero. Bitiodine: Extracting intelligence from the bitcoin network. In *Financial Cryptography and Data Security*, Lecture Notes in Computer Science, pages 457–468. 2014.

[Wika] Bitcoin Wiki. Script. https://en.bitcoin.it/wiki/Script accessed at 2014-12-07.

[Wikb] Bitcoin Wiki. Thin client security. https://en.bitcoin.it/wiki/Thin_Client_Security accessed at 2014-12-09.

[Wui12] Pieter Wuille. Hierarchical deterministic wallets, 2012. https://github.com/bitcoin/bips/blob/master/bip-0032.mediawiki accessed at 2014-12-13.

[Zer] Zerocash q&a. http://zerocash-project.org/q_and_a accessed at 2014-12-14.

List of Figures