Blindcoin

Blinded, Accountable Mixes for Bitcoin

Luke Valenta[1] and Brendan Rowan[2] *

[1] University of Pennsylvania lukev@seas.upenn.edu
[2] University of Maryland browan@cs.umd.edu

Abstract. Mixcoin is a Bitcoin mixing protocol proposed by Bonneau *et al.* which provides strong accountability guarantees [13]. However, in the Mixcoin protocol, the mapping from a user's input to output address is visible to the mixing server. We modify the Mixcoin protocol to provide guarantees that the input/output address mapping for any user is kept hidden from the mixing server. In order to achieve this, we make use of a blind signature scheme [14,23] as well as an append-only public log. The scheme is fully compatible with Bitcoin, forces mixes to be accountable, preserves user anonymity even against a malicious mix, is resilient to denial of service attacks, and easily scales to many users.

1 Introduction

Bitcoin is a decentralized electronic cash system that has become increasingly popular since its introduction in 2008 [27]. The base unit of currency, the *bitcoin* or *BTC*, is defined in terms of *transactions*, which are transfers of BTC from a sender address to a receiver address, where addresses are simply public keys. All transactions are stored in a public ledger, the *block chain*. Since all transactions are public, user anonymity in Bitcoin relies on pseudonyms. There has been much work on de-anonymizing users in the bitcoin block chain by linking together addresses [10,22,26,28,30,31]. If any one of the linked addresses can be mapped to the true identity of a user, then all past and future transactions involving the linked addresses can be associated with that user, compromising their anonymity. While a discussion of the costs and benefits of anonymity is beyond the scope of this work, there is much debate over the importance of this property [5,20,29].

1.1 Mixing Services

To alleviate this risk of deanonymization in Bitcoin, one can use a *mixing service* or *mix*. Mixes for anonymous communication were originally introduced by Chaum [15], but many recent services have adapted this idea to the financial setting. These services allow users to exchange their bitcoins for "clean" bitcoins that—within some anonymity set—cannot be linked to their current addresses

* This work originated as a project in a computer networks course at the University of Maryland

and identities by an adversary (See Sect. 4.1). To participate in a mixing operation, a user supplies some amount of bitcoins from an input address, and specifies an output address that they wish these bitcoins to end up in eventually. Although many such mixing services exist, current designs lack certain important features. We now state properties that we believe an ideal system should offer.

- *Accountability.* When a user sends funds to the mix, they should be confident that if a theft occurs, they can present a proof of the mix's misconduct.
- *Anonymity.* The user should be the only entity that knows the mapping from their input address to their output address.
- *Resilience to Denial of Service Attacks.* The system should not be vulnerable to attacks by a small number of malicious parties that could potentially DoS the exchange [24, 34].
- *Scalability.* The system should be efficient enough to be able to scale to a large number of users and large anonymity sets.
- *Incentive for Participation.* There should be a mechanism for the mix to collect mixing fees fairly that will incentivize it to provide the service. On the user end, the service should come at a reasonable cost and should be convenient, so that a large number of users participate.
- *Backwards-Compatibility.* The system should be compatible with the current Bitcoin system to allow for practical integration.

1.2 Current Bitcoin Mixing Services

Mixcoin. Mixcoin achieves all of the above properties, except for anonymity against a malicious mix [13]. In this protocol, a mix has access to the mapping from a user's input to output address and can store this information. At any point in the future, the mix could reveal this information and compromise the anonymity of its users.

CoinJoin. CoinJoin is a decentralized protocol in which all users must sign a joint transaction [24]. Users remain anonymous and their funds cannot be stolen, since a user will only provide its signature if they agree to the transaction. However, CoinJoin is vulnerable to a DoS by a single user if they refuse to provide a signature during the signing round of the protocol. Thus, Coinjoin is not resilient to misbehaving users.

CoinShuffle. The protocol CoinShuffle is built on CoinJoin [32]. CoinShuffle is decentralized and uses a multiparty sorting protocol [34] to achieve anonymous mixing. It ensures that a joint transaction will eventually go through by including a blaming process in which misbehaving users can be eliminated from future mixing attempts by the honest participants. There is no financial commitment required (i.e. mixing fees) for a user to participate in or DoS a round of the protocol, so a large-scale Sybil attack [19] would be a cheap way to delay a round of mixing. Overall, this seems to be a promising approach.

CoinSwap/Fair Exchange. These protocols allow users to exchange coins anonymously with no risk of theft [7, 11]. However, it is not clear how mixing fees could be collected in an anonymous manner to incentivize the mix to provide the service.

Altcoins. Another way to mix one's bitcoins is to exchange them for some alternate currency, or *altcoin*, and at some later time exchange the altcoins back for bitcoins. Some protocols designed for this purpose are Zerocoin and Zerocash [12, 25]. These techniques can achieve strong anonymity properties, but require additional infrastructure or modifications to the bitcoin protocol.

Deployed Services. Several mixing services that are used in practice have no provable guarantees of anonymity or theft protection [1–3, 8] For example, some Bitcoin Fog users claim that their coins were stolen by the service [4], and Möser *et al.* are able to link input/output transactions in the graph of BitLaundry [26].

1.3 Our Contribution

We modify the Mixcoin protocol to prevent the mix from learning the input/output address mappings of participating users. The Mixcoin authors, as well as [24], posit that such a scheme could be possible using blinded tokens as described in Chaum's original digital cash scheme [14]. We show that this is indeed possible and present a protocol that achieves this goal, and at the same time preserves the accountability property of Mixcoin and the mechanism for collecting fair, randomized mix fees. The main modifications that we make are the introduction of an append-only public log that is used to keep the mix accountable, and the utilization of a blind signature scheme to hide the mapping between a user's input and output addresses from the mix.

Our proposed system meets all of the above goals for a mixing service. For accountability, we use a warranty scheme that allows the user to provide evidence against the mix if it misbehaves, similar to [13]. Anonymity against the mixing service is provided by using a blind signature scheme to hide the input/output address mappings of participants. The system is resilient to DoS attacks by a single user refusing to sign a joint transaction, as certain schemes are susceptible to [24, 34]. This is possible since the mix deals with each user individually, and can exclude users from the exchange on a case-by-case basis. The system is scalable since it does not require any computationally complex cryptography as do some other systems [34]. The scheme employs fair, randomized mixing fees to incentivize mixes to provide the service to users at a reasonable cost and within a relatively short timespan. We expect the mixing of a single "chunk" of coins to take on the order of a few hours, and multiple chunks can be mixed simultaneously. Finally, the system is backwards compatible with Bitcoin, requiring no changes to the current protocol.

2 Background

2.1 Mixcoin Summary

We give a summary of the properties and contributions of Mixcoin.

Chunk Size. Each user sends a fixed quantity v of bitcoins to the mix. The intuition behind this is as follows: suppose some amount X BTC is tranferred to some output address. Then, it is easy to link this output address to the set of all input addresses that spent X BTC within that particular time interval. We want this set to be as large as possible, since "anonymity loves company" [17].

Accountability. Mixcoin forces mixing servers to be accountable for their actions. That is, when a user sends funds to a mix, they are provided with a warranty such that, in the event that the mix steals the user's coins, the user can present proof (verifiable by any party) of the mix's cheating. This warranty consists of a signed agreement from the mix that if the user pays funds to an escrow address specified by the mix by a certain time, then the mix will transfer an equal amount of funds to the output address specified by the user before an agreed-upon deadline.

Sequential Mixing. In the Mixcoin model, the mixing server learns the input/output address mapping, and must be trusted to keep this information private. In order to remain anonymous in the event that the mix is compromised, users are encouraged to mix coins through multiple rounds of independent mixes, which would be costly in terms of both time and mixing fees. Our proposed changes obviate the need for these extra rounds of mixing, since the mixing servers are blinded to the input/output address mappings.

Mixing Fees. Another important contribution that Mixcoin makes is the introduction of *randomized mixing fees*. The authors argue that the collection of mixing fees incentivizes mixes to act honestly. If the mixing fees are sufficiently high, then a rational mix looking to maximize profits will not risk cheating, lest they get caught. A warranty proving their dishonesty would damage their reputation and hurt their business model. Furthermore, the Mixcoin authors argue that the strongest anonymity properties are achieved when mix fees are *all-or-nothing*, instead of some fixed rate per chunk. In essence, rather than keeping ρ fraction of each chunk, the mix will keep an entire chunk with ρ probability. To determine which of the inputs to keep as a mixing fee, the mix calculates a **Beacon** function [16], which depends on a public source of randomness (e.g., future blocks of the Bitcoin block chain). The function maps a random nonce (unique to each of the inputs) to a value in the interval $[0, 1]$. If this output falls below ρ, the mixing fee rate, then the mix claims the corresponding coins as its mixing fee. According to the Mixcoin paper, a typical value for ρ might be less than 0.01, although this will depend on the market.

2.2 Blind Signatures

The idea of blind signatures originated with Chaum in 1983 [14]. In [23], Fuchsbauer gave the first efficient implementation of a blind signature scheme with round-optimal issuing [21], meaning that only one message from the user and one message from the signer is required for the user to obtain the signature. A blind signature scheme works as follows: a user wishes to obtain a signature

on some message without the signer knowing the message contents. To achieve this, the user computes some *commitment* function (which can be thought of as encryption) on the message and sends that along with a *randomization* of the message to the signer. The signer then makes a "pre-signature" and sends it back to the user. From this, the user can compute an actual signature on the message by adapting the randomness. The blind signature is then a proof of knowledge of a signature on the message. The scheme achieves the following requirements: *Blindness*: Given a set of blind signatures, the signer cannot relate the signatures to their issuings. *Unforgeability*: Given a set of blind signatures, an adversary cannot compute a valid signature on a new message.

3 Blindcoin Description

3.1 Model

In addition to the two entities of the Mixcoin protocol—the user and the mixing server—our protocol requires another entity, the public log. We summarize the entities present in the system below:

Public Mix. The public mix, M, is assumed to have a long standing public key M_{pub} and associated private key M_{priv}, used for signing. The model assumes that there are many such mixes that will compete. Mixes that have poor reputations will be less likely to be chosen by users, since users want to reduce their likelihood of being cheated. Thus, a mix is incentivized to participate in the protocol without allowing any proof of cheating to be made public.

The User. The user, A, is a party that has an amount of Bitcoins in an address k_{in} (possibly linked to their true identity) and wishes to transfer the coins to a different address k_{out}, such that it is hard for any adversary to link the addresses k_{in} and k_{out}. The user is also able to post anonymously to the public log with an alternate identity, A'. This could be achieved through Tor [18], for example. The user must be careful not to allow A and A' to be linked in any way.

Public Log. The public log is a public, append-only log used for third party verification purposes. If a party breaches protocol, the public log will contain enough information to incriminate the misbehaving party. Anyone can post to the log, and messages can never be erased once they are posted. Each message is also associated with a timestamp. One possible way to implement this would be within the Bitcoin block chain. To send a message, the sender could just transfer a small amount of BTC to one or more addresses corresponding to the message. Of course, if the user wishes to post to the log anonymously, the coins must be sent from an address that cannot be linked back to the user's identity. An example of a service that provided this capability is the Bitcoin Message Service [6].

3.2 Protocol

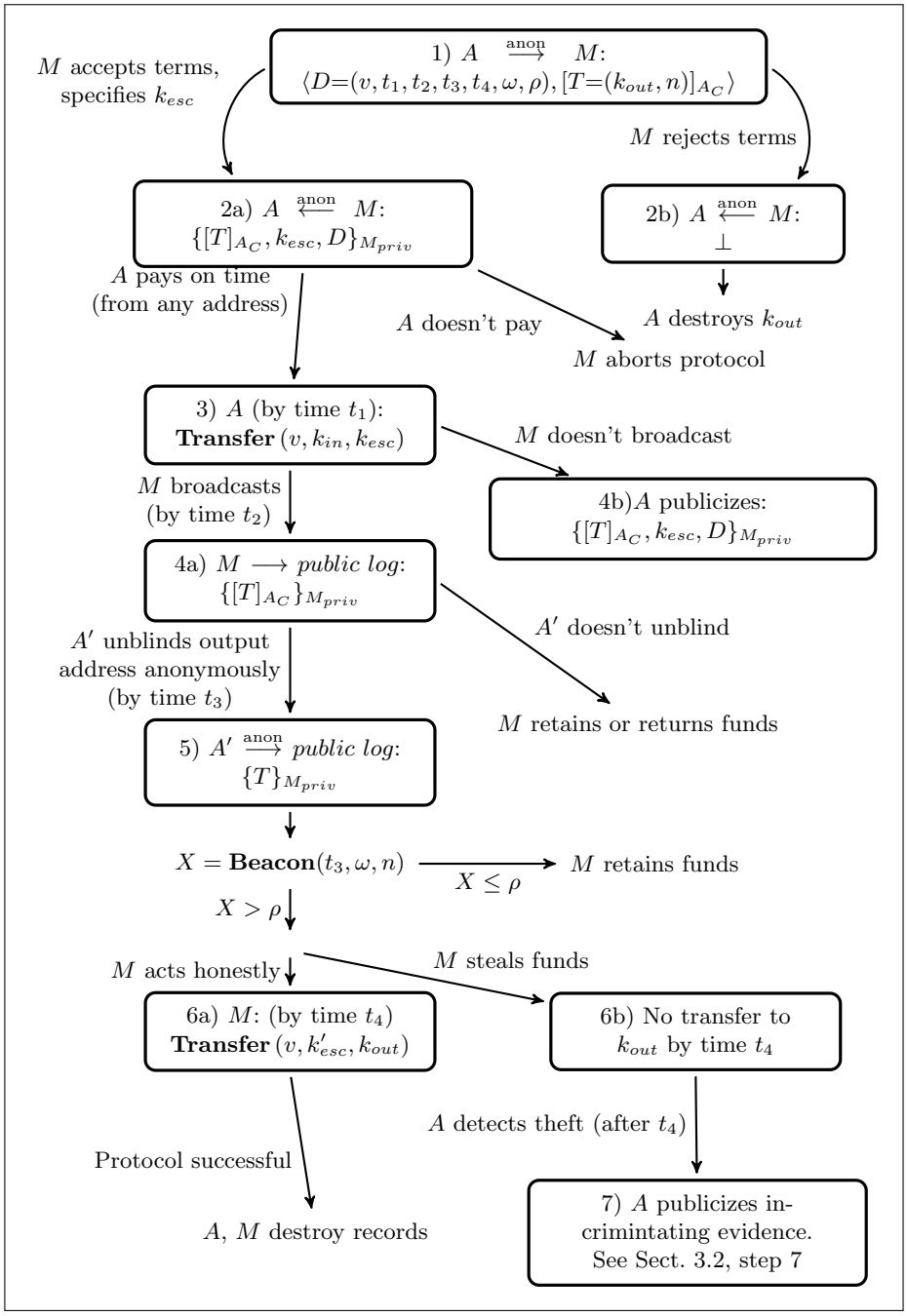
In this section, we describe the core protocol for the mixing of a single "chunk" of A's funds. At a high level, the protocol proceeds as follows: the mix announces mix parameters, the user opts in to the mix, the mix sends a partial warranty back to the user, the user transfers funds, the mix completes the warranty by posting to the public log, the user unblinds the output address, and finally the mix transfers funds to the output address. See Fig. 1 for an illustration of the protocol and to trace the protocol steps. We also indicate the differences in Blindcoin and Mixcoin at each step in the protocol. For ease of presentation, we use the notation $[x]_k$ for a message x committed with commitment function k, and the notation $\{x\}_k$ to represent a signature on message x with signing key k. We also assume that whenever a signature on a message is sent, the entire message x is sent as well in case the contents of the message need to be recovered.

k_{in}	the address from which the A pays, possibly linked to A's true identity
k_{out}	the address to which the user wishes funds transferred
k_{esc}	an escrow address, unique for each user, that M provides for A to pay
k'_{esc}	an escrow address that M uses to pay to k_{out}
A'	an anonymous identity that A can use to post to the public log
M_{pub}	the public key of M
M_{priv}	the private signing key of M
A_C	a secret commitment/encryption function of A
$A_{C'}$	the inverse of A_C
ω	the number of blocks M requires to confirm A's payment
n	a per-user nonce, used to determine payment of randomized mixing fees
v	the value (chunk size) to be mixed
ρ	the mixing fee rate A will pay
T	the token, which is the triple (k_{out}, n)
t_1	the time by which A must v BTC to k_{esc} in order to participate in the mix
t_2	the time by which M must post the token T to the public log
t_3	the time by which A' must unblind the output address via the public log
t_4	the time by which the mix must transfer v BTC to k_{out}
D	the mix parameters, a tuple $\{t_1, t_2, t_3, t_4, v, \omega, \rho\}$
Beacon	the beacon function, a publicly verifiable random function

Setup. The mix M publishes a set of *mix parameters*, D, to the public log. This data includes times (t_1, t_2, t_3, t_4) by which different steps of the protocol must be completed; the chunk size, v, which is the amount of BTC that each user inputs into the transaction; ρ, the fraction of user inputs that will be kept as a mixing fee; and ω, the number of blocks that the mix requires to verify the users payments. All of these parameters are part of the original Mixcoin protocol, except for two additional time deadlines.

User Sends Offer. See step (1). The user A opts in to the mix by sending its *offer* to M. The offer includes the mix parameters followed by a blinded *token* T. The token consists of the output address and a private, randomly selected

Fig. 1. The Blindcoin Protocol



nonce n which is used for fee collection. In order to keep these values hidden, the token is encrypted using a commitment function A_C known only to A (who also knows the inverse $A_{C'}$). The offer has the following form: $(D, [T]_{A_C})$, where $T = (k_{out}, n)$. In Mixcoin, the output address is not blinded, and the offer has the form: $(v, t_1, t_2, \omega, k_{out}, \rho, n)$.

Mix Sends Partial Warranty. See step (2a). If M accepts the offer, it sends a *partial warranty* back to the user. The partial warranty consists of the blinded token, an escrow address for the user to pay to, and the mix parameters. This is all signed with M_{priv}. The partial warranty has the form $\{[T]_{A_C}, k_{esc}, D\}_{M_{priv}}$. Note that given this partial warranty the user cannot recover the signed token directly, since other fields were included in the signed message. Also, the escrow addresses that the mix provides should be unique to each user, so that the mix can verify which users have paid. In Mixcoin, the mix simply sends back the warranty $(\{v, t_1, t_2, \omega, k_{out}, \rho, n\}_{M_{priv}})$

Mix Rejects Offer. See step (2b). If the mix rejects the offer, the user destroys the output address. This step is the same as in Mixcoin.

User Pays. See step (3a). A then transfers v coins from any input address k_{in} to k_{esc} by time t_1. This step is the same as in Mixcoin.

User Fails to Pay. See step (3b). If the user fails to tranfser the funds on time, then both parties abort the protocol. This step is the same as in Mixcoin.

Mix Completes Warranty. See step (4a). Once the user has transferred the funds, M must *complete the warranty* by signing the blinded token and publishing it to the public log by time t_2 (which should be long enough after t_1 to allow the transaction to be at least ω blocks deep into the chain). By publishing a user's token, the mix is publicly acknowledging that the user did indeed transfer their funds to the escrow address on time. The fact that this is public allows any third party verifier to check that the mix has completed the warranty by time t_2. The signed blinded token has the form $\{[T]_{A_C}\}_{M_{priv}}$. Mixcoin does not have an equivalent step, since the user already has the warranty.

Mix Fails to Complete Warranty. User Publishes Incriminating Evidence. See step (4b). If M fails to publish a user A's token by t_2, A can publish information to incriminate M. A can present the following evidence: the partial warranty $\{[T]_{A_C}, k_{esc}, D\}_{M_{priv}}$, the transaction $Transfer(v,\ k_{in},\ k_{esc})$ present in the block chain before time t_1, and the fact that the signed token was not published to the public log before t_2 (this may require an enumeration of all messages in the blockchain between times t_1 and t_2). Any third party verifier can see that M has signed the partial warranty, confirm that someone has transferred funds to k_{esc}, and verify that indeed no token of the correct form was published to the public log before t_2, proving that M deviated from the protocol. Mixcoin does not have an equivalent step.

User Anonymously Unblinds Output Address. See step (5). Once their signed blinded token of the form $\{[T]_{A_C}\}_{M_{priv}}$ has been published to the public log, A can apply $A_{C'}$ to recover the signed unblinded token $\{T\}_{M_{priv}} = \{k_{out}, n\}_{M_{priv}}$. The user connects anonymously as user A' and unblinds k_{out} by posting the signed token to the public log. The mix M can verify that the output address is valid, since the signed token is a proof-of-knowledge that the mix signed the corresponding commitment. If A' fails to publish the unblinded token to the public log by time t_3, M can choose to either refund the coins back to A or retain them. Since A breached the protocol, it cannot produce evidence to incriminate M, so M can do as it pleases with the funds. Mixcoin does not have an equivalent step.

Mix Computes Beacon Function. If the user unblinds their output address by time t_3, M computes a beacon function **Beacon**(t_3, ω, n) for each (k_{out}, n) pair to determine which output addresses to collect mixing fees from (by not sending any coins to them). The beacon function is a publicly verifiable function that uses entropy collected from the block chain to produce a number uniformly in the range $[0, 1]$ (see [13,16]). If the value for a particular input is less than or equal to the value ρ from the mix parameters, the chunk destined for that output address is kept by M as a mixing fee. Otherwise, the protocol proceeds to the next step. This step is equivalent to Mixcoin's **Beacon** step, except that in Blindcoin the mix does not know which input addresses correspond to which (k_{out}, n) pairs.

Mix Pays to Output Address. See step (6a). If M acts honestly, then before time t_4 it will transfer v BTC to all unblinded output addresses that have passed the **Beacon** function. The mix does not know which input and output addresses are from the same user, so the mapping from input to output addresses does not matter. Mixcoin has an equivalent step.

Mix Steals Coins. See step (6b). M steals the funds and fails to transfer a chunk to each of the output addresses by time t_4. Again, Mixcoin has an equivalent step.

User Detects Theft and Publishes Incriminating Evidence. See step (7). The user A detects the theft at time t_4 (since no funds were transferred to its output address), and can publish information to incriminate M. The user publishes the following: the commitment function A_C and its inverse $A_{C'}$, the partial warranty $\{[T]_{A_C}, k_{esc}, D\}_{M_{priv}}$ in the public log, the transaction $Transfer(v,\ k_{in},\ k_{esc})$ present in the block chain before time t_1, the signed token $\{k_{out}, n\}_{M_{priv}}$, and the fact that no such transaction $Transfer(v,\ k'_{esc},\ k_{out})$ is present in the block chain before time t_4. Any third party can verify that the token was signed with M_{priv} and recover the contents of the signed token with $A_{C'}$. Then, the verifier can check the public log and the block chain to see if both parties followed the protocol by transferring funds and posting to the public log by the correct deadlines. In Mixcoin, the incriminating evidence that the user publishes consists of the the warranty $(\{v, t_1, t_2, \omega, k_{out}, \rho, n\}_{M_{priv}})$, the transaction $Transfer(v,\ k_{in}, k_{esc})$ present in the blockchain before time t_1, and the fact

that no transaction $Transfer(v, k'_{esc}, k_{out})$ exists in the blockchain before time t_2.

4 Analysis

In this section, we discuss properties of Blindcoin, evaluate its efficiency and usability, and discuss side channel attacks.

4.1 Properties

The Blindcoin protocol inherits many properties from the Mixcoin system. Below, we outline the properties of both systems and discuss any changes that occur due to our modifications.

Accountability. This propery is inherited from Mixcoin. In our modifications, we ensure that this property is preserved.

We define *mix accountability* to be the property that if the mix deviates from the protocol, their cheating can be provably exposed. A mix can deviate from the protocol and steal the user's funds, but the user can then publish their mix-signed warranty along with other evidence to show that the mix has cheated. A mix can steal user funds at several points in the protocol, but at each point the user is "protected" by their warranty.

The warranty is slightly more complex in Blindcoin than in Mixcoin. When the mix agrees to a user's (blinded) offer, they issue a partial warranty back to the user, which is an agreement saying "if the user pays, the mix will publish the signed blinded token to the public log by the warranty deadline." The mix waits until the user has paid to publish the token to the public log so that it can make sure that only valid, paid-for tokens are in the log. The reason that the token must be published to the log is so that third party verifiers can check that the mix has acknowledged a user's payment. As described in Sect. 3.2, if the mix does not publish the signed blinded token to the public log before the agreed-upon deadline, the user can present a proof of misconduct, which would be damaging to the reputation of the mix.

Another point in time that the mix could potentially cheat is when determining which output addresses to send funds to. If users have correctly followed the protocol and published their signed, unblinded tokens to the public log, then any party can see that the mix has signed the token. Furthermore, after the mix computes **Beacon** to determine the random collection of fees, it is possible for any party to verify that it has correctly computed the function, since **Beacon** is public. If the mix fails to transfer funds to an output address that has passed **Beacon**, then the user can again present a proof of misconduct.

Anonymity. Mixcoin provides strong anonymity guarantees for its users except for the case in which the mix is the adversary. If a mix stores records, they could potentially release them to deanonymize users. In Blindcoin, we extend many of the guarantees provided by Mixcoin and show that anonymity is possible even

against a malicious mix. As in the Mixcoin protocol, we assume than an adversary wishes to link an output address k_{out} to the corresponding input address k_{in}.

The first adversarial model we consider is that of a *global passive adversary*. Since the Bitcoin block chain is public and anyone can access all Bitcoin transactions (and the public log in Blindcoin), this is the weakest adversarial model that we consider. Both Mixcoin and Blindcoin provide guarantees that no passive adversary can link input/output address pairs within a particular mix. Thus, Blindcoin achieves *k-anonymity* [33] within the set of all non-malicious users participating in the mix simultaneously. There are no constraints on the number of users that can participate in a mix simultaneously except for the mix server's resources, so the more participants there are, the larger the anonymity set.

A second adversarial model is one of an active attacker who is able to compromise some subset of the input/output address pairs for a particular mixing operation. For both Mixcoin and Blindcoin, the anonymity set for a user remains the number of non-compromised address pairs. An active adversary can also carry out a Sybil attack [19] to convince a user that their anonymity set is larger than it really is. Although there is no real solution to this type of attack, mixing fees make it expensive to carry out.

We also consider the model where the mix is the adversary. This could occur if a mix is compromised or coerced into revealing its records. In the case of Mixcoin, all input/output mappings are revealed, since the mix has access to this information. This problem can be alleviated if the user sends their coins through multiple independent mixes, but a strong adversary could potentially compromise all of them. This also causes a significant degradation in performance and increase in price since user must repeat the entire mixing procedure multiple times. However, a compromised mix does not weaken the anonymity guarantees for Blindcoin since mixes are blinded to the mapping from input to output addresses through the use of the blinded token scheme. This is the main contribution of Blindcoin.

One property that Mixcoin provides is *mix indistinguishability*. This property is unique to Mixcoin, and no other mixing services provide this same guarantee [13]. Against a global passive adversary, this property extends a user's anonymity set to *all* users participating in *different* mixes that use the same parameters for mixing. Unfortunately, this does not hold for Blindcoin, since in the process of unblinding their output addresses, users must publish their signed tokens to the public log. This allows an adversary to link the output addresses to the signing key of the mixing server, defeating mix indistinguishability. We note that mix indistinguishability breaks for Mixcoin when adversaries are active or the mix is malicious. From a user perspective, the loss of mix indistinguishability implies that their anonymity set is limited to only other users participating in the same mix simultaneously (the typical case for Bitcoin mixes), so users must ensure that this set is large enough for their anonymity purposes. With Blindcoin, a

user can easily check the public log *a posteriori* to determine the number of users that participated in their mix.

If enough care is not taken, side channels can leak user information that may lead to deanonymization and address linkage [10, 13, 30]. These side channels include timing, precise values, network layer information, and interactions with the public log.

- *Timing.* To avoid timing analysis attacks, users should not make their interactions with the mix predictable. For example, if a malicious mix posts a particular signed blinded token to the public log and then an output address is unblinded immediately after, the mix may be able to link the token to the output address.
- *Precise Values.* The size of a user's anonymity set is equal to the number of (non-compromised) users participating in a mixing exchange with the same mix parameters. If each user negotiated its own set of parameters with the mix, then it would be trivial for anyone to link a user's input and output addresses together. For example, if users had unique unblinding deadlines, then an observer could easily map an unblinded output address back to its associated input address.
- *Network-Layer Information.* For example, two identities connecting with the same IP address could be linked. We assume that users connect via a secure anonymity network such as Tor [18] and keep their connecting identities A and A' separate and unlinkable. In practice, keeping these connecting identies from being linked may be hard against a well-provisioned adversary, but our system does not have a solution to this problem and we consider it outside the scope of this paper.
- *Public Log.* If transactions fees are required to post messages to the public log, users must be careful not to pay the fees from linkable addresses.

Resilience to Denial of Service Attacks. This property is inherited from Mixcoin. In many distributed mixing protocols, a single user can DoS the entire exchange by participating in the protocol up to a certain point, and then refusing to transfer funds, causing the whole operation to fail [24, 32, 34]. However, in both Mixcoin and Blindcoin, each user interacts only with the mixing server, and refusing to comply with the protocol does not affect any other users or slow down the mixing process. One could also attempt to carry out a DoS attack to prevent transactions from entering the blockchain on time or to prevent messages from being posted to the public log. However, these attacks would be difficult to carry out in practice, since they would require the attacker to control a large portion of the Bitcoin block mining pool.

Scalability. This property is inherited from Mixcoin. It is efficient to add more users to a mixing operation since users interact only with the centralized mix and not each other. Further, if having a single server proves to be a bottleneck, it is possible to load balance the function of the mix onto several different servers, all operating with the same cryptographic keys.

Incentive for Participation. Both users and mixes are incentivized to participate in the system. For a small fee, users are able to safely mix their bitcoins. Although there is some delay associated with a mixing operation in Blindcoin, the fact that one does not need to trust any party clearly makes this a valuable tool. Mix servers are also incentivized. The fees collected by the mix can be set so that they are sufficient to cover operational costs, and make it a worthwhile to provide the service.

Backwards-Compatibility. Blindcoin does not require any changes to the existing Bitcoin protocol, so it can be easily deployed.

4.2 Overheads

A successful Blindcoin mixing operation requires two message directly between A and M, two messages (one from each) posted to the public log, and two Bitcoin transactions. The two messages to the public log are the extra overhead of our protocol over Mixcoin. Each of these new messages come with a deadline by which they must be posted. The time in between deadlines must allow the previous message to be at least ω blocks deep into the block chain to prevent double spending (a typical value might be $\omega = 6$). The expected time between blocks is 10 minutes, so on average it would take one hour for a message to be 6 blocks deep into the chain. However, the time deadlines should be set much further apart to allow for variations in the time it takes to mine each block. Since the protocol requires four deadlines, and the gap between deadlines is the dominating factor in how long the protocol takes to run, our estimate of the total time for a mixing operation is on the order of a few hours.

Posting messages to the public log (assuming it is implemented in the Bitcoin block chain) costs extra in transaction fees, in addition to the fees paid for the funds transfer. According to [9], the typical transaction fee rate is 0.0001 BTC per 1000 bytes. Although the exact message size depends on the implementation, we believe 5000 bytes is a reasonable estimate, for a total cost per message of 0.0005 BTC per message. Overall, the financial cost to a user is composed of the mixing fee and the transaction fees. For a chunk size of 0.1 BTC, a fee rate of 0.01, and a transaction fee of 0.0005 BTC per message, the total cost to the user is around 0.002 BTC or 2%, which we believe is a reasonable price to pay for the anonymity benefits.

To summarize, the main downsides of Blindcoin are the loss of mix indistinguishability, the necessity to maintain two unlinkable identities A and A', and the additional costs and delays associated with posting messages to the public log. The benefits are that the mixing server does not learn the input/output address matching, so multiple rounds of mixing are not required to achieve an adequate level of anonymity.

5 Conclusion

De-anonymization of the Bitcoin block chain is a problem that has attracted much attention. Many different designs of Bitcoin mixing services have been proposed, but we find that each of these systems lacks certain desirable features. In order to make steps towards an ideal system, we propose a list of properties that a mixing system should offer. We believe an ideal mix should be all of the following: accountable, anonymous, scalable, resilient, incentivized, and compatible with the original system.

We present a mixing system that meets the above requirements. The proposed system, Blindcoin, modifies the Mixcoin [13] mixing protocol by using blind signatures [14, 23] and a public append-only log. The log makes it possible for a third party to verify the validity of accusations when blind signatures are used. The system retains many of the benefits of the Mixcoin system, while also relaxing the constraint that the mix must be trusted to keep the input/output address mappings of users hidden.

Acknowledgements. We would like to thank the anonymous reviewers, as well as Nadia Heninger, Andrew Miller, Dave Levin, and Bobby Bhattacharjee for their helpful comments and input.

References

1. Bitcoin fog. http://www.bitcoinfog.com/.
2. Bitlaundry. http://app.bitlaundry.com/.
3. Bitmixer.io. https://bitmixer.io/.
4. The current state of coin-mixing services. http://www.thebitcoinreview.com/site.php?site_id=759.
5. Online anonymity is not only for trolls and political dissidents. https://www.eff.org/deeplinks/2013/10/online-anonymity-not-only-trolls-and-political-dissidents.
6. Bitcoin message service, October 2011. https://bitcointalk.org/index.php?topic=47283.0.
7. Coinswap, Oct 2013. https://bitcointalk.org/index.php?topic=321228.
8. blockchain.info, Oct 2014. https://blockchain.info/.
9. Transaction fees, March 2014. https://en.bitcoin.it/wiki/Transaction_fees.
10. Elli Androulaki, Ghassan O Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun. Evaluating user privacy in bitcoin. In *Financial Cryptography and Data Security*, pages 34–51. Springer, 2013.
11. Simon Barber, Xavier Boyen, Elaine Shi, and Ersin Uzun. Bitter to better–how to make bitcoin a better currency. In *Financial Cryptography and Data Security*, pages 399–414. Springer, 2012.
12. 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.
13. Joseph Bonneau, Arvind Narayanan, Andrew Miller, Jeremy Clark, Joshua A Kroll, and Edward W Felten. Mixcoin: Anonymity for bitcoin with accountable mixes. *IACR Cryptology ePrint Archive*, 2014:77, 2014.

14. David Chaum. Blind signatures for untraceable payments. In *Advances in cryptology*, pages 199–203. Springer, 1983.
15. David L Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–90, 1981.
16. Jeremy Clark and Urs Hengartner. On the use of financial data as a random beacon. *IACR Cryptology ePrint Archive*, 2010:361, 2010.
17. Roger Dingledine and Nick Mathewson. Anonymity loves company: Usability and the network effect. In *WEIS*, 2006.
18. Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. Technical report, DTIC Document, 2004.
19. John R Douceur. The sybil attack. In *Peer-to-peer Systems*, pages 251–260. Springer, 2002.
20. Tony Doyle and Judy Veranas. Public anonymity and the connected world. *Ethics and Information Technology*, pages 1–12, 2014.
21. Marc Fischlin. Round-optimal composable blind signatures in the common reference string model. In *Advances in Cryptology-CRYPTO 2006*, pages 60–77. Springer, 2006.
22. Michael Fleder, Michael S Kester, and Sudeep Pillai. Bitcoin transaction graph analysis. *HYPERLINK http://people. csail. mit. edu/spillai/data/papers/bitcoin-transaction-graph-analysis. pdf http://people. csail. mit. edu/spillai/data/papers/bitcoin-transaction-graph-analysis. pdf*, 2014.
23. Georg Fuchsbauer. Automorphic signatures in bilinear groups and an application to round-optimal blind signatures. *IACR Cryptology ePrint Archive*, 2009:320, 2009.
24. G Maxwell. Coinjoin: Bitcoin privacy for the real world, August 2013. https://bitcointalk.org/index.php?topic=279249.
25. Ian Miers, Christina Garman, Matthew Green, and Aviel D Rubin. Zerocoin: Anonymous distributed e-cash from bitcoin. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 397–411. IEEE, 2013.
26. Malte Möser, 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.
27. Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Consulted*, 1(2012):28, 2008.
28. Micha Ober, Stefan Katzenbeisser, and Kay Hamacher. Structure and anonymity of the bitcoin transaction graph. *Future internet*, 5(2):237–250, 2013.
29. Jacob Palme and Mikael Berglund. Anonymity on the internet. *Retrieved August*, 15:2009, 2002.
30. Fergal Reid and Martin Harrigan. *An analysis of anonymity in the bitcoin system*. Springer, 2013.
31. Dorit Ron and Adi Shamir. Quantitative analysis of the full bitcoin transaction graph. In *Financial Cryptography and Data Security*, pages 6–24. Springer, 2013.
32. Tim Ruffing, Pedro Moreno-Sanchez, and Aniket Kate. Coinshuffle: Practical decentralized coin mixing for bitcoin. *HotPETS.(July 2014)*.
33. Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.
34. E.Z Yang. Secure multiparty bitcoin anonymization, July 2013. http://blog.ezyang.com/2012/07/secure-multiparty-bitcoin-anonymization/.