Mixcoin
Anonymity for Bitcoin with accountable mixes

Joseph Bonneau[1], Arvind Narayanan[1], Andrew Miller[2], Jeremy Clark[3], and Joshua A. Kroll[1] and Edward W. Felten[1]

[1] Princeton University
[2] University of Maryland
[3] Concordia University

Abstract. We propose Mixcoin, a protocol to facilitate anonymous payments in Bitcoin and similar cryptocurrencies. We build on the emergent phenomenon of currency mixes, adding an accountability mechanism to expose theft. We demonstrate that incentives of mixes and clients can be aligned to ensure that rational mixes will not steal. Our scheme is efficient and fully compatible with Bitcoin. Against a passive attacker, our scheme provides an anonymity set of *all* other users mixing coins contemporaneously. This is an interesting new property with no clear analog in better-studied communication mixes. Against active attackers our scheme offers similar anonymity to traditional communication mixes.

1 Introduction

Protecting the privacy of financial transactions has long been a goal of the cryptography community, dating at least to Chaum's work on anonymous digital cash using blind signatures [6]. Despite initial excitement, anonymous digital payments have not seen mass adoption. One reason is that traditional electronic cash requires a central, trusted entity, typically called a bank.

By contrast, Bitcoin is a relatively young decentralized currency that has rocketed to popularity with a monetary base worth over US$6 billion in early 2014. Bitcoin can be thought of as a public, distributed ledger that logs all transactions in order to prevent double spending [23]. Using a proof-of-work system, the integrity of the ledger is maintained as long as a majority of the computing power is contributed by honest participants [17].

Bitcoin does not provide true anonymity: transactions involve pseudonymous addresses, meaning a user's transactions can often be easily linked together. Further, if any one of those transactions is linked to the user's identity, all of her transactions may be exposed. A small but growing body of academic literature has found that Bitcoin offers only weak anonymity in practice (see Section 2.1). This has led to the rise of *mixing services* (or *tumblers*) which promise to take a user's coins and randomly exchange them for other users' coins to obfuscate their ownership, though these come with no protection from theft by the service.

The Bitcoin community is well aware of this issue, leading to much interest in the provision of stronger anonymity. We provide more detail in Section 8, but

existing proposals can be thought of in two main groups. First are proposals which provide strong anonymity but require advanced cryptography and substantial modifications to Bitcoin, like Zerocoin [21], or even a completely new currency as with Zerocash [4]. Second, there are proposals such as CoinJoin [18] or CoinSwap [19] which are backwards-compatible with Bitcoin but have practical complications and may provide smaller anonymity sets. Our goal is to enable strong anonymity in a simple scheme that can be deployed immediately. Our strategy is to build on the existing phenomenon of mixes, but to add an independent cryptographic accountability layer. Our main contributions include:

Accountability. Mixcoin mixes issue signed warranties (Section 4) to users which roughly state: "if Alice sends me v coins by time t_1, I will send v coins back to her by time t_2." A user can then confidently send funds to the mix, knowing that if the mix misbehaves she can publish this warranty, damaging the mix's reputation and (presumably) its business model.

Randomized mixing fees. We show how paying mixes for their services incentivizes honest behavior (Section 6), yet fixed fees undermine anonymity when coins are mixed multiple times. Instead we apply randomized, all-or-nothing fees in which mixes retain the entire value from a small percentage of transactions. We show how to generate the requisite randomness in a fair and accountable manner using the unpredictability of the Bitcoin block chain itself.

Mix indistinguishability. Although users interact with specific mixes, single-use mix addresses enable a surprising property that passive adversaries can't determine which mix a user is interacting with. The anonymity set in this case is then the set of all users interacting with *any* mix at the same time.

Mix networks for Bitcoin. Against an active attacker who can break mix indistinguishability, we draw on the experience from anonymous communication networks to demonstrate how chaining multiple mixes together can still provide strong anonymity. There are important differences from communication mixes, however, which we discuss in Section 7).

Our core protocol is a very general design, allowing clients and mixes to specify a variety of free parameters. We expect that, because anonymity loves company [11], these parameters will converge to global values (Section 7.6). In particular, we expect mixing to complete in a few hours with mixing fees of less than 1% (Section 6). Given this modest overhead and the fact that Mixcoin can be deployed immediately with no changes to Bitcoin itself, it is our hope that all Bitcoin users will have the opportunity to mix their coins, making strong financial privacy practical in a decentralized digital currency.

2 Background

In this section we provide a basic model of Bitcoin. We focus on the properties required for Mixcoin, which could be implemented on top of any distributed

currency system similar to Bitcoin in these basic respects. We then model today's nascent Bitcoin mixes and the attacks they are vulnerable to.

2.1 Bitcoin

Bitcoin can be thought of as a decentralized system which tracks a mapping between *addresses* and monetary value denominated in coins. An address, which we denote κ, is simply a public key. Addresses are pseudonymous: anybody can create an arbitrary number of addresses for free with no verification. Control of an address's private key provides "ownership" of all coins mapped to that address. The simplest[4] Bitcoin *transaction* is essentially a statement that an address κ_{in} would like to transfer some value v to an address κ_{out}, signed by κ_{in}.

A distributed consensus protocol maintains a global history of all transactions to prevent double spending. Transactions are grouped into *blocks* for efficiency, which are chained in a linear structure called the *block chain*. The chain represents (probabilistic) consensus; at present most Bitcoin users will consider a transaction confirmed if it appears in a block with at least $w = 6$ blocks following it. New blocks are generated roughly once every ten minutes.

Creating new addresses is trivial, but this does not make Bitcoin anonymous as all transfers are globally (and permanently) visible in the block chain. Several recent papers have studied ways to link a user's addresses to each other and to an external identity [20,25,27,2].

2.2 Current Bitcoin mixes

To preserve their privacy, some Bitcoin users exchange their coins using *mixes*, directly analogous to the concept in communication networks. In the common implementation a mixing address receives coins from multiple clients and forwards them randomly to a fresh address for each client. Several such services have arisen, typically charging commissions in the 1–3% range and requiring manual interaction through a website[5] to arrange transactions. A small-scale study of three mixing services found that in one case, taint analysis was immediately sufficient to link the input and output [22]. In the other two cases, taint analysis did not succeed but the transaction graph showed rich structure, leaving open the question of more sophisticated linking attacks. Anecdotal evidence from user forums include complaints slow mixing times of up to 48 hours and low transaction volumes leading to users frequently receiving their own coins in return.[6] Reports of theft by mixes are also a significant concern, with the popular Bitcoin Wiki warning: *...if the mixing output fails to be delivered or access to funds is denied there is no recourse. Use at your own discretion.*

[4] Bitcoin transactions may feature multiple inputs and outputs. Bitcoin also features a limited scripting language allowing more complicated transactions.

[5] Some mixing services are only accessible as Tor hidden services.

[6] Receiving one's own coins back from a mix is not necessarily a vulnerability. This will happen with probability $\frac{1}{N}$ in a random permutation of N participant's coins.

In contrast to dedicated mixing services, some services with a high preexisting trust requirement have deployed implicit mixing successfully. For example, the Silk Road marketplace mediated and mixed all transactions between buyers and sellers [7], while some "eWallet" services promise that when users withdraw funds they will receive random coins from the provider's reserves.

2.3 Mix networks for anonymous communication

Mix networks were introduced by Chaum in 1981 for anonymous communication [5]. Significant research has analyzed the relationship between design parameters, such as route selection and flushing policies, and the resulting anonymity (see [24] for a survey), much of which is broadly applicable to financial mixing.

Verifiable mixing, beginning with Sako and Killian [28], aims to provide accountability by mixes issuing a proof that their output is a permutation of their input, particularly important when users cannot trace their own input through the mix. In reputable mixing, beginning with [14], each mix provides proof that each output corresponds to some input, as opposed to the mix itself originating the message. Unfortunately these lines of research are largely orthogonal to the risk of theft in a financial mix. In communication mixes, messages can be resent, which is not possible in Bitcoin as transactions are irreversible.

3 A simple model of mixing

We start with a client Alice (A) who owns some number of Bitcoins at an address κ_{in} which we assume is linkable to her real world identity. Alice wishes to transfer some of her funds to a fresh address κ_{out} in such a way that it is difficult to link κ_{out} to κ_{in} (and hence Alice herself), in exchange for a mixing fee.

Alice will send some of her coins to a mix M, a for-profit entity which will hold Alice's funds in escrow for an agreed time period before sending an equal value to κ_{out}. We don't require M to have any real-world reputation or assets, only to maintain the same digital identity long enough to build a virtual reputation. Alice is exposed to two major threats:

Theft Because mixes routinely send funds to fresh addresses with no transaction history, it is possible for a malicious mix to send Alice's funds to its own secret address κ_M instead of κ_{out} as requested. Though Alice can publicly complain about the theft and attempt to undermine M's reputation, there is no way for observers to determine which of A or M owns κ_M and therefore Alice's claim could be libelous. For-profit mixes may rationally attempt to undermine trust in their competitors through false accusations of theft. Because allegations of theft cannot be proven, it is difficult to determine which mixes are honest.

Deanonymization Because the mix learns that the same party owns both addresses $(\kappa_{\text{in}}, \kappa_{\text{out}})$, Alice's anonymity depends on the mix keeping this pairing secret forever. A mix which is malicious, compromised, or subpoenaed might share its records and undermine Alice's anonymity. Alternately, the mix could send coins in a non-random manner which reveals the connection to observers.

4 The Mixcoin protocol

Our goal with Mixcoin is to provide a protocol for mixing with *accountability*. Prior to mixing, the mix gives Alice a signed warranty which will enable her to unambiguously prove if the mix has misbehaved. Dishonest mixes will quickly have their reputation destroyed and lose business. Security against theft thus reduces to properly aligning economic incentives of mixes and clients.

However, there is no way to prove that a mix is not storing records sufficient to deanonymize its clients. Similarly to mix networks for communication, Alice can mitigate this risk by relaying coins through a series of mixes which must all collude in order to deanonymize her final output address.

4.1 Assumptions

We assume the availability of multiple mixes M_i, each represented by a warranty-signing key K_{M_i}. As for-profit enterprises, mixes are motivated to build and maintain a reputation in K_{M_i}, so it must be used consistently. Unlike mixes, Alice does not need to maintain any long-term public key nor any public reputation. Alice must be able to negotiate with the mix over an anonymous and confidential channel. In practice this will likely be realized by mixes running a dedicated Tor hidden service, but this is out of scope of the Mixcoin protocol itself.

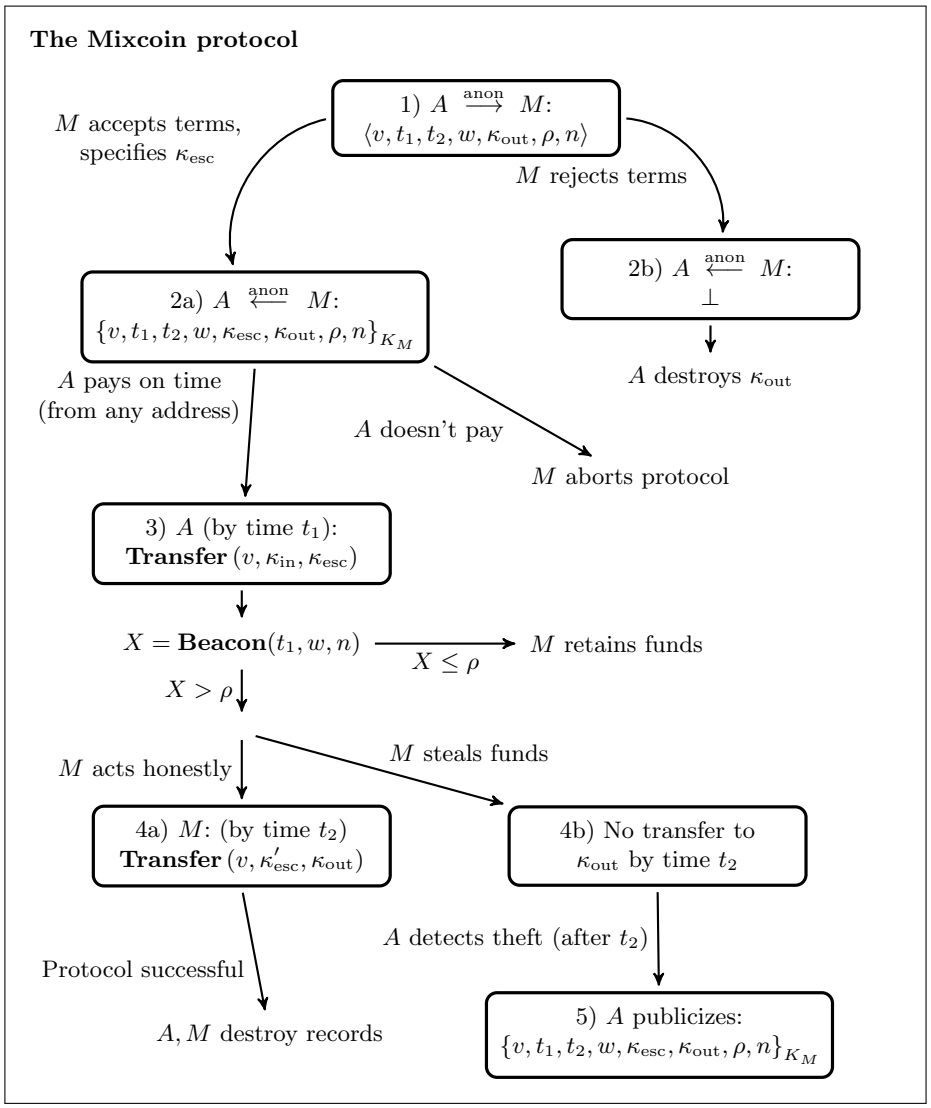
4.2 Core protocol

We outline the core Mixcoin protocol in Construction 1 which mixes a single "chunk" v of Alice's funds. For effective anonymity, chunk sizes should be standardized, as discussed in Section 7.6. While the core protocol can stand on its own, typically Alice will need to split her funds into multiple chunks and perform multiple sequential rounds of mixing for each.

The key accountability mechanism is Alice's receipt of a signed warranty prior to mixing. In Step 1 Alice contacts the mix over an anonymous channel and proposes a set of mixing parameters:

v	the value (chunk size) to be mixed
t_1	the deadline[7] by which Alice must send funds to the mix
t_2	the deadline by which the mix must return funds to Alice
κ_{out}	the address where Alice wishes to transfer her funds
ρ	the mixing fee rate Alice will pay
n	a nonce, used to determine payment of randomized mixing fees
w	the number of blocks the mix requires to confirm Alice's payment

If the mix accepts these terms (Step 2a) it generates a fresh escrow address κ_{esc} and sends back a warranty containing all of Alice's parameters plus κ_{esc}, signed using K_M. The mix may also reject Alice's request for any reason (Step 2b), though in practice we expect that a reputable mix will abide by a published policy for acceptable terms. Alice similarly has no obligation to transfer funds



Construction 1: A single mixing round between client A and mix M. A owns the addresses κ_{in} and κ_{out} and M owns κ_{esc} and κ'_{esc}. The random value $X \xleftarrow{\text{R}} (0, 1)$ is computed using **Beacon**, a pseudorandom function using the Bitcoin block $t_1 + w$ plus the nonce n, and compared to the fee rate ρ. Times t_1 and t_2 are blocks in the block chain. Curly brackets ($\{\}_K$) indicate a digital signature under a signing key K.

after receiving a warranty. If Alice declines (or forgets) to do so by the deadline t_1 the mix may delete its records and move on.

If Alice does transfer the agreed value v to κ_{esc} by the deadline t_1 (Step 3), then the mix is obligated to transfer an equal value to κ_{out} by time t_2 (unless the funds are retained as a mixing fee—see Section 4.4). If the mix does so faithfully (Step 4a), then both parties should destroy their records to ensure forward anonymity against future data breaches. If the mix fails to transfer the value v to κ_{out} by time t_2 (Step 4b),[8] then Alice publishes her warranty (Step 5). Because the warranty is signed by the mix's long-term key K_M and all Bitcoin transactions are publicly logged, anybody can verify that the mix cheated.

4.3 Freshness of addresses

Both the mix's escrow address κ_{esc} and Alice's output address κ_{out} should be fresh addresses created specifically for this mixing. This is required because warranties include neither κ_{in} nor κ'_{esc}, so they will appear to be satisfied as long as v is transferred on time to κ_{esc} and then κ_{out} from *any* address. Thus both parties should pick addresses with no other possible source of income so that the other party must themselves pay to fulfill the contract.

4.4 Mixing fees

A simple approach is to specify a fixed mixing fee rate ρ and have the mix return $(1-\rho)\cdot v$ to κ_{out} instead of the full v . However, this is problematic for sequential mixing, as the smaller output value $(1-\rho)\cdot v$ cannot be the input to a subsequent round of mixing with the same v. This could be addressed by using diminishing transaction sizes $v_i = (1-\rho)^i \cdot v$ for each round i, but this would undermine the goal (Section 7.6) of indistinguishable transfers and limit the anonymity set in each round to only other transactions at the same round of mixing.

Our solution is *randomized* mixing fees, whereby with probability ρ the mix retains the entire value v as a fee, and with probability $(1-\rho)$ takes no fee at all. This produces an expected mixing fee rate of ρ and leaves κ_{out} with either nothing or a full v which can be directly re-mixed. This solution is related to the idea of electronic lottery tickets [26] used in some micropayment systems.[9]

The mix must use a publicly verifiable mechanism to randomly choose which chunks to retain as mixing fees. Specifically, the mix must generate a $(\rho, 1-\rho)$-random bit which neither party can predict but can be audited afterwards for fairness. This can be done with a public source of randomness called a *beacon*.

If the beacon is external to Bitcoin (*e.g.,* NIST's beacon [1] or financial data [8]), warranties would need to be synchronized to real-world time to enable

[8] There is no way in Bitcoin to guarantee a transaction will be included in any specific block. Therefore in practice mixes will likely require a safety margin of several blocks to t_2 to ensure they can include the transaction before that time.

[9] Our motivation to use randomized fees is different from the case of micropayment systems, which do so to avoid transaction costs from many low-valued payments.

auditing. Alternatively, randomness can be extracted from future Bitcoin blocks, assuming the exact set of future transactions included in each block (as well as the random nonce used to solve the proof-of-work puzzle) is unknown.[10] Because each block includes the value of the previous block, every transaction during a confirmation period of w blocks adds randomness.[11] The warranty also includes a nonce n specified by Alice to ensure that the mix will compute an independent value for all transactions it is managing. Specifically, the mix computes $X = \mathsf{Beacon}(t_1, w, n) = \mathsf{PRNG}\,(n||B_{t_1+w})$, where B_i is the Merkle root of block i in the block chain and PRNG is a cryptographic pseudorandom number generator which outputs a value uniformly drawn from the range $(0, 1)$.

The mix retains Alice's funds only if $X \leq \rho$. Because this computation can be performed by anybody if Alice's warranty is published, cheating by the mix is detectable. Furthermore, in normal operation Alice's warranty (containing n) is kept secret so observers can't tell which transactions were retained by the mix.

A drawback of randomized fees is increased variance in the effective mixing fee rate for users mixing a small number of chunks. To address this, v should be kept as low as possible so that most users can mix at least $\frac{v}{\rho}$ coins.

4.5 Transaction fees

In addition to mixing fees, Alice may have to pay transaction fees to Bitcoin miners to ensure her transactions are included in the block chain.[12] Fixed transaction fees pose the same problem for anonymity that fixed mixing fees would, but paying miners randomly would require changes to Bitcoin itself.

Given a source of anonymous coins, Alice could address the problem of decreasing chunk sizes by "topping up" each chunk after it is mixed using her pool of anonynmized coins. However, it doesn't work for Alice to simply mix one chunk perfectly and then use it top up many other chunks, as this would publicly link each of those topped up chunks as belonging to the same party. Thus Alice would need a large number of mutually unlinkable addresses holding transaction-fee sized values useful for topping up. Acquiring these through mixing becomes a recursive problem though, as they themselves would require an even greater number of unlinkable addresses for their mixing!

Instead, mixes can effectively pay transaction fees both[13] for the transfer from κ_{in} to κ_{esc} and from κ'_{esc} to κ_{out}. Assuming miners require a minimum transaction fee τ (with $\tau \ll v$), Alice can transfer v from κ_{in} of which the mix will receive $v - \tau$ at address κ_{esc}. The mix can then form an output transaction

[10] A mix might also be a miner, in which case it may attempt to influence the block. However, such an attack is highly uneconomical given the high reward for mining a block compared to mixing fees.

[11] Though in practice $w = 6$ is a common standard, we include w as a negotiable parameter in the warranty to enable flexibility.

[12] Some transactions are accepted today without fees, though miners may change this at any time, which may occur as the minting rate decreases.

[13] In pipelined sequential mixing, which we will discuss in Section 5, most mixes will need only pay one transaction fee.

with $v - \tau$ from some κ'_{esc} and 2τ from a third address κ^*_{esc} which the mix previously retained as a mixing fee, ensuring that κ_{out} receives a full v while the miners still collect a fee of τ for each transaction. Of course, the mixing fee rate ρ must be increased to cover the mix's expected outlays on transaction fees.

This poses a problem for mix indistinguishability, which we'll discuss further in Section 7.2, as at the mix must use the same κ^*_{esc} to cover transaction fees for multiple chunks which will then all clearly come from the same mix.

5 Sequential mixing

Given the above Mixcoin protocol for interacting with a single mix, Alice will most likely want to send her funds through N independent mixes to protect her anonymity against the compromise of an individual mix. To do so, Alice can choose a sequence of N mixes $M_1, \ldots M_N$ and execute the Mixcoin protocol with each of them *in reverse order*, instructing each mix M_i to forward her funds to the escrow address $\kappa_{\text{esc}_{i+1}}$ which she previously received from mix M_{i+1}. After obtaining N signed warranties,[14] Alice then transfers her chunk to κ_{esc_1} and if any mix in the sequence fails to transfer it she can prove it with the appropriate warranty. One subtlety is that each mix can likely determine which number it is in the sequence based on timing information, as the later mixes will be contacted further in advance from when mixing will actually take place.

In practice, Alice most likely wants to transfer some value kv by splitting into k separate chunks. This means she will need to negotiate a total of kN warranties with mixes. An important consideration is that each chunk should travel through an independently-chosen random sequence of mixes. Otherwise, Alice's chunks would be exchanged for each other more frequently than would happen via chance, which would leak information to a potential attacker.

6 Mix incentives and mixing fees

Establishing the mixing fee rate ρ requires considering the dual roles of mixing fees. First, they can cover direct expenses for mixes such as Bitcoin transaction fees and electricity bills. Second and most importantly, they provide a mechanism for mixes to profit from honest behavior and disincentivize mixes from ceasing operations and absconding with users' funds. Because higher fees more strongly incentivize honesty, an interesting property arises that users should avoid mixes charging less than some *minimum* acceptable value of ρ.

In a steady-state model, the mix has two choices for any given block in time: continue to operate honestly until the next block, or abscond and retain all user funds it holds in escrow. The expected value of either choice scales linearly with Q, the average amount of money flowing into (and out of) the mix during any one

[14] Unlike in traditional communication networks, an onion routing approach doesn't seem possible due to the interactivity required in Mixcoin.

block. If $\bar{t}$ is the average time period (in blocks) that the mix holds funds during a mixing round, then the expected payoff of absconding is $\mathbb{E}[\mathsf{abscond}] = Q\bar{t}$.

The expected payoff from choosing to continue would properly be defined recursively, since the mix is able to play the same game again. However, under steady state conditions the optimal decision will be the same in every round, so if the mix initially chooses to continue it will do so indefinitely. Assuming the mix is exponentially discounting future earnings[15] at a rate r (per block), the net present value of indefinite honest behavior with a fee rate ρ is $\frac{\rho Q}{r}$.

Incentivizing honest behavior therefore requires that $\frac{\rho}{r} > \bar{t}$. With the interpretation that r for a rational mix is equivalent to the highest available risk-free rate of return available, this condition is simply that the expected value of fees collected by a mix during the time it holds funds is greater than the amount those funds would yield during the same time period if invested.[16] This can be explained by considering that we want an honest mix to continually decided to "invest" its potential earnings $Q\bar{t}$ from absconding into continuing to serve as a mix, earning a return of ρQ during every block.

We can estimate that relatively low mixing fees should suffice to incentivize honest behavior. Assuming a very attractive rate of return of $r \approx 20\%$ annually is available to the mix, a mix time of $\bar{t} \approx 1$ hour gives a lower bound of $\rho_{\min} \approx 2^{-15}$. Even considering a chunk taking a path through 10 consecutive mixes, this still leaves only an effective fee rate of $\approx 2^{-12}$ necessary to discourage absconding. This suggests that very low mixing fees may be sufficient to cover the risk of theft.[17] Still, actual mixing fees will be dominated by operating costs, suggesting that any mix which has been operating for a non-trivial period of time is turning a profit and is unlikely to abscond.

7 Anonymity properties

We can draw many connections to the extensive literature on mix networks for communication, dating to the initial proposal of communication mixes [5].

7.1 Threat model

We focus on an attacker who wants to gain as much information as possible about the *anonymity set* of possible pre-mixing input addresses which may have been the source of the funds held by a final output address κ_{out}.

Because the Bitcoin block chain is a permanent, public record of all transactions, every attacker is trivially a *global passive adversary*, a common attack

[15] The exchange rate of bitcoins may of course be drastically different in the future. We assume mixes have no private information about the future value of bitcoins and therefore use its current market price in calculating the net present value.

[16] This equivalence ignores the effects of compounding interest, though r and $\bar{t}$ are both low enough that $(1+r)^{\bar{t}} \sim 1 + r\bar{t}$.

[17] In practice, absconding may be slightly more appealing due to super-exponential time discounting by the mix or the risk that business may decline.

model studied for communication mixes.[18] Mixing literature also considers extended attacker capabilities, such as compromising mixes, delaying or blocking messages, replaying old messages, or flooding the network with dummy messages [29]. Replay should be impossible in Mixcoin due to the double spending prevention in Bitcoin, but flooding and delaying may be possible.

7.2 The passive adversary's view with mix indistinguishability

The best-case scenario for Mixcoin is a passive adversary. We assume this adversary can reliably determine with high probability which Bitcoin transactions are mix traffic, given their size v and their use of one-time escrow addresses. However, due to their one-time nature, this simple adversary may be unable to link escrow addresses to specific mixes, a novel property with no apparent precedent in communication mixes which we call *mix indistinguishability*.

If this is the case, the adversary is left to observe a sea of apparently identical escrow addresses and the system appears to function as one universal mix consisting of all participants using the chunk size v. There are several scenarios in which mix indistinguishability may fail (which we will discuss in Section 7.3) but the anonymity offered is quite strong in this case.

7.3 Active adversaries and distinguishable mixes

There are several ways that an active attacker might be able to distinguish which escrow addresses correspond to which mix and hence which mixes are involved in a chunk's mixing path. Observe that when Alice sends a chunk from κ_{in} to M via κ_{esc}, the client who ultimately receives this chunk will learn that κ_{in} interacted with M. Similarly, the client who sends the chunk to κ'_{esc} which is eventually sent to κ_{out} will also learn that Alice interacted with M. An active adversary can exploit this in a flooding attack, learning up to two other addresses interacting with the same mix for each chunk sent through that mix.

A second attack vector, if mixes are forced pay transaction fees, is that when a user's chunk is retained as a mixing fee by mix M it may might be used by M to pay transaction fees on many other transactions, all of which can then be linked to M. The effectiveness of this attack depends on the ratio of transaction fees per chunk τ to average mixing fees per chunk ρv. Mixes will have to spend a proportion $\frac{\tau}{\rho v}$ of their mixing fee revenue on transaction fees, so if mixes allocate a constant proportion of each retained chunk to transaction fees each retained chunk will pay fees on $\frac{1}{\rho} \cdot \frac{\tau}{\rho v}$ other transactions. Since each chunk is retained with probability ρ, the expected number of transactions identifiable by a given input transaction is just $\frac{\tau}{\rho v}$, which is maximized at 1 if mixing fees are only high enough to cover transaction fees. Thus for each mixing transaction an active attacker performs with M, she can link up to $(1-\rho) \cdot 2 + \frac{\tau}{\rho v}$ other transactions

[18] Tor is notably not designed to withstand attack by a global passive adversary, as Tor relays provide no mixing of traffic [12].

to M. Observing the majority of links therefore appears to require an attacker generate a large portion of the mix's traffic.

Finally, the attacker may be the mix themselves, or be able compromise the mix or subpoena its records, which would reveal all input/output pairs.

Against such a strong active attacker who can link every escrow address to its originating mix, the system appears similar to be a traditional communication mix network with mixes behaving as *stop-and-go* mixes [16] with limited pooling due to the block size. Stop-and-go mixes suffer from low guarantees of anonymity in periods of low traffic, but implementing other strategies such as threshold mixes appears very difficult to achieve with our warranty systems as significantly more information (including the entire set of transactions sent to a mix) would need to be available to enforce warranties.

7.4 Anonymity sets and mix delay

Regardless of mix distinguishability, there is a trade-off between mixing chunks with many mixes for a short escrow period each or few mixes with a longer escrow period. The escrow period is limited by t_2 and t_1 as specified in the warranty, with a maximum delay of $\delta_{\max} = t_2 - t_1$. Mixes will also require a minimum delay of $\delta_{\min} = w$ (typically 6 blocks) to protect against double spending. Picking the smallest possible $t_2 = t_1 + w$ allows Alice to afford more rounds of mixing in a given time period. But this also means that Alice's anonymity set for the round consists only of other chunks that were mixed at time exactly t_1.

We assume that individual mixes will only issue warranties with a specific $\delta_{\max}$ as a matter of policy,[19] and will then uniformly at random choose a delay $\delta \in_R [w, \delta_{\max}]$ before forwarding Alice's chunk.[20] Thus each mixing step adds $\lg(Q(\delta_{\max} - w + 1))$ bits of entropy to Alice's anonymity set, at a delay of $\delta_{\max}$ blocks.[21] In other words, the entropy of her anonymity set grows by $\frac{\lg(Q(\delta_{\max} - w + 1))}{\delta_{\max}}$ per block. It turns out that for $w = 6$ this expression is maximized for $\delta_{\max} = 6$ for $Q \geq 128$ (and $\delta_{\max} = 7$ for $13 \leq Q < 128$) so it appears minimal delays and longer mixing chains are preferable.

7.5 Mixing multiple chunks

So far we have considered each chunk individually. However, if Alice combines many mixed chunks to make a payment, her anonymity set will be reduced to the intersection of the anonymity sets of all chunks. As long as she mixed those chunks sufficiently at the same time, then those chunks will have the same anonymity sets, and her payment is still unlinkable.

[19] Allowing different delays per client would open the possibility of free-riding and make anonymity analysis much more complex [13].

[20] Non-uniform distributions such as an exponential distribution are possible, but they make it difficult to provide a firm bound on the delay as required by the warranty.

[21] Because Alice must have already negotiated her mixing warranty for the next round, each warranty must be delayed by the maximum $\delta_{\max}$ blocks.

However, if even one of the chunks travels through a path consisting entirely of compromised mixes, Alice's entire payment completely loses anonymity. If each chunk is routed independently, then with say 25% of mixes compromised, there is a 2^{-20} chance of routing a chunk through a chain of 10 compromised mixes, which may be acceptably low. However this probability increases rapidly if a greater fraction of mixes are compromised. One way to avoid this would be to randomly pick a set of mixes for each batch of funds to mix, and to use a random permutation of that set for each chunks in the batch.

7.6 Convergence of free parameters

Our design intentionally leaves many parameters free, such as the chunk size v, the time delay $t_2 - t_1$ and the number of rounds N. Our philosophy is to avoid embedding these into the protocol as the optimal choices may drift over time as the mixing ecosystem evolves and the underlying parameters of Bitcoin change. Yet it is critical for anonymity that a large number of users choose the same values[22] to avoid splitting their anonymity sets based on parameter choices.

As a case study, consider the effect of two different common values of v. Each will be clearly identifiable in the block chain and hence the anonymity set for each chunk is limited in the best case to those users who mixed a chunk of identical size in the same time period. We could attempt to ameliorate this slightly by hoping that all users mix chunks of both sizes regularly, but this is quite fragile.[23] The best-case scenario for anonymity is if all users choose the same chunk size. Yet there is an inherent trade-off: setting v too high will exclude users owning less than v coins,[24] while decreasing v will require proportionately more runs of the protocol and more transactions in the block chain.[25]

Still, we expect v and other parameters to converge in practice to a common value (or a small set) for two reasons. First, like with Bitcoin itself most clients will likely use one of a small number of software implementations which include reasonable parameters and a popular mix reputation list.

Second and more importantly, all clients have an incentive to choose the most popular parameters in an application of the "anonymity loves company" principle [11]. Unilateral variation in a user's transaction sizes, for example, could leak information which would help Eve deanonymize Alice's coins. Thus we expect Mixcoin users to relatively quickly converge on a global set of parameters.

---

[22] Note that the mixing fee rate ρ is unobservable and hence should have no impact on anonymity and can be chosen independently by different mixes.

[23] For example, if chunk sizes α and β are common, a user mixing $x = k_1\alpha + k_2\beta$ will have her anonymity set limited to other users mixing at least k_1 chunks of size α and at least k_2 of size β, instead of all users mixing at least x.

[24] Additionally, with randomized mixing fees (see Section 4.4) users owning only a small multiple of v may face unacceptably high variance in their fee rate.

[25] The Bitcoin community frowns on creating large numbers of low-value transactions (referred to as *dust*) because it places a higher verification burden on miners.

7.7 Side channels

Financial mixing introduces several subtle side channels.[26] The most obvious is payment sizes: If Alice receives a very specific amount of Bitcoins at her long-term address, is observed mixing them, and a day later an equal quantity of mixed chunks are combined to make a payment, the adversary might plausibly infer that Alice made the payment.[27] This can be addressed if Alice mixes her incoming funds as soon as she receives them and not immediately prior to making a payment. Of course, this requires Alice to always carry a balance of mixed funds and never pay them all out at once.

More subtle issues arise because mixed chunks carry an implicit timestamp of when they were last mixed. Suppose Alice immediately mixes three large, equal-sized quantities of income on three specific dates and then later combines a random subset of her mixed chunks to make a payment. Eve can trace the outgoing payment to Alice if it contains a mix of chunks from these times and Alice was the only person mixing at each of them.[28] The attack might work even if Alice wasn't the only person mixing: if Alice picks a random set of her mixed chunks, then the proportion of chunks from each time period in the outgoing payment will correspond to the amount Alice mixed in each time period.

Thus, even perfect mixing can leave Alice's transactions linkable without further obfuscation. One defense is for Alice to only make payments using chunks that were mixed contemporaneously. This works if payments are small enough. Second, Alice could re-mix all of her chunks every time she receives income. This destroys the timing information, but is expensive. Third, if Alice has advance notice before needing to make a payment, she can employ *input/output mixing*. Alice mixes her funds as soon as she receives income. When she needs to make a payment, she mixes a set of (already mixed) chunks totaling the amount she owes. It introduces a delay in payment equivalent to mixing time, which is why Alice must have advance notice. Finally, in Appendix A we introduce *continual mixing*, a more complex approach which can provide stronger guarantees of anonymity.

8 Related Bitcoin anonymity technologies

Several academic proposals have aimed to provide strong anonymity cryptographically. Most prominent is Zerocoin [21], which uses a cryptographic accumulator with zero-knowledge proofs of inclusion to implement a global currency pool from which users can deposit coins and withdraw random coins without any trusted parties. Unfortunately Zerocoin and related proposals [10,4] require modifications to Bitcoin which appear unlikely due to the computational overhead. Mixcoin, by contrast, can be deployed immediately.

[26] Network-level side channels are out of scope. As noted earlier, we assume that Mixcoin clients always communicate using a secure anonymity network such as Tor.

[27] This is analogous to a *packet counting attack* in communication mixes.

[28] This is analogous to an *intersection attack* in the mixing literature.

An alternate line of research, mostly arising from the Bitcoin developer community, is to remove the trust requirement from mixing using more complicated (but already supported) Bitcoin transaction scripts. For example, Barber et al.'s "fair exchange" protocol [3] or Maxwell's CoinSwap [19] allow two parties to anonymously swap coins with no risk of theft using a multi-step protocol and at least 4 transactions (compared to 2 in Mixcoin). Both of these protocols could be used as an alternative to Mixcoin to facilitate mixing with no risk of theft and mix indistinguishability against a passive attacker and our anonymity analysis would still apply, including the loss of mix indistinguishability against a flooding attack. Incorporating transaction fees is another open problem in these protocols and there doesn't appear to be a simple way to apply our randomized approach.

Finally, CoinJoin [18] enables k users to atomically transfer funds from their k input addresses to their k output addresses in a random permutation. Since the transaction is atomic and requires every participant to sign, there is no risk of theft. The transaction functions as an implicit mix between the participants. However arranging the output addresses randomly without users learning the correspondence for other users' coins introduces complexity. Overall we expect CoinJoin might be useful for small-scale mixing but the anonymity offered may be lower due to the lack of mix indistinguishability.

9 Conclusion

Despite significant interest in providing strong anonymity for Bitcoin, the design of a robust protocol with that can be deployed without modifications to Bitcoin has remained an open question. In this paper we proposed Mixcoin, which we believe meets these goals. Our key innovations are cryptographic accountability, randomized mixing fees, and an adaptation of mix networks to Bitcoin. We look forward to engaging with the academic community and the Bitcoin community to further refine the design and to progress toward implementation and deployment.

We also provide an initial treatment of mixing for financial privacy, a research area which we expect will be as deep and challenging as mixing for communication privacy. Many basic properties of communication mixes, such as the ability to pad or replay messages, don't exist in a financial setting. Yet interesting new properties, such as the possibility of indistinguishable mixes, arise. We expect that ensuring financial privacy, regardless of the underlying mixing protocol, will require careful consideration of some of the higher-level side channels we have only briefly explored here.

Acknowledgments. We thank our anonymous referees and all who read drafts and contributed valuable suggestions to this work, especially Aaron Johnson, Ian Miers, Roger Dingledine, George Danezis, Peter Eckersley, Peter Todd and Eran Tromer. Joshua Kroll was supported by the National Science Foundation Graduate Research Fellowship Program under grant number DGE-1148900.

References

1. NIST Randomness Beacon. http://www.nist.gov/itl/csd/ct/nist_beacon.cfm
2. Androulaki, E., Karame, G.O., Roeschlin, M., Scherer, T., Capkun, S.: Evaluating User Privacy in Bitcoin. In: FC (2013)
3. Barber, S., Boyen, X., Shi, E., Uzun, E.: Bitter to Better—How to Make Bitcoin a Better Currency. In: FC (2013)
4. Ben-Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., Virza, M.: Zerocash: Decentralized anonymous payments from Bitcoin. In: Security and Privacy (SP), 2014 IEEE Symposium on. IEEE (2014)
5. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM 24(2), 84–90 (1981)
6. Chaum, D.: Blind signatures for untraceable payments. In: CRYPTO (1982)
7. Christin, N.: Traveling the silk road: A measurement analysis of a large anonymous online marketplace. In: Proceedings of the 22nd International Conference on World Wide Web. pp. 213–224. International World Wide Web Conferences Steering Committee (2013)
8. Clark, J., Hengartner, U.: On the Use of Financial Data as a Random Beacon. Usenix EVT/WOTE (2010)
9. Coutu, O.: Decentralized Mixers in Bitcoin. In: Bitcoin Conference (2013)
10. Danezis, G., Fournet, C., Kohlweiss, M., Parno, B.: Pinocchio Coin: Building Zerocoin from a Succinct Pairing-Based Proof System. In: Language Support for Privacy-Enhancing Technologies (PETShop) (2013)
11. Dingledine, R., Mathewson, N.: Anonymity Loves Company: Usability and the Network Effect. In: WEIS (2006)
12. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: USENIX Security (2004)
13. Dingledine, R., Serjantov, A., Syverson, P.: Blending different latency traffic with alpha-mixing. In: PET. pp. 245–257. Springer (2006)
14. Golle, P.: Reputable mix networks. In: PETS (2004)
15. Jolly, G.M.: Explicit estimates from capture-recapture data with both death and immigration-stochastic model. Biometrika 52(1/2), 225–247 (1965)
16. Kesdogan, D., Egner, J., Büschkes, R.: Stop-and-go-mixes providing probabilistic anonymity in an open system. In: Information Hiding. pp. 83–98. Springer (1998)
17. Kroll, J.A., Davey, I.C., Felten, E.W.: The Economics of Bitcoin Mining, or Bitcoin in the Presence of Adversaries. In: WEIS (Jun 2013)
18. Maxwell, G.: CoinJoin: Bitcoin privacy for the real world. https://bitcointalk.org/index.php?topic=321228 (August 2013)
19. Maxwell, G.: CoinSwap: Transaction graph disjoint trustless trading. CoinSwap: Transactiongraphdisjointtrustlesstrading (October 2013)
20. Meiklejohn, S., Pomarole, M., Jordan, G., Levchenko, K., McCoy, D., Voelker, G.M., Savage, S.: A fistful of bitcoins: characterizing payments among men with no names. In: IMC (2013)
21. Miers, I., Garman, C., Green, M., Rubin, A.D.: Zerocoin: Anonymous Distributed E-Cash from Bitcoin. IEEE Symposium on Security and Privacy (2013)
22. Möser, M.: Anonymity of Bitcoin Transactions: An Analysis of Mixing Services. In: Proceedings of Münster Bitcoin Conference (2013)
23. Nakamoto, S.: Bitcoin: A peer-to-peer electionic cash system (2008)
24. Raymond, J.: Traffic analysis: protocols, attacks, design issues, and open problems. In: PETS (2001)

25. Reid, F., Harrigan, M.: An analysis of anonymity in the bitcoin system. In: Security and Privacy in Social Networks (2013)
26. Rivest, R.: Electronic Lottery Tickets as Micropayments. In: FC (1997)
27. Ron, D., Shamir, A.: Quantitative Analysis of the Full Bitcoin Transaction Graph. FC (2012)
28. Sako, K., Kilian, J.: Receipt-Free Mix-Type Voting Scheme - A Practical Solution to the Implementation of a Voting Booth. In: EUROCRYPT (1995)
29. Serjantov, A., Dingledine, R., Syverson, P.: From a trickle to a flood: Active attacks on several mix types. In: Information Hiding (2003)

A Continual mixing

A more in-depth defense against some of the side-channel attacks introduced in Section 7.7 is *continual mixing*, which does not require advance notice of payments. In addition to avoiding the timing side channel, it actually increases Alice's anonymity set. The core idea is that Alice continues mixing her coins until she is ready to spend them, but at a greatly reduced rate (*e.g.,* one round per month). Let Δ_A be a time period such that Alice is prepared to keep her coins for time Δ_A between receiving them and spending them. Then the continual mixing algorithm for a chunk c for which initial mixing completes at time t_0 is as follows:

- generate $\Delta_{A,c} = U[0, \Delta_A]$
- mix c at time $\Delta_{A,c}$ and thereafter at Δ_A intervals
- mark c as spendable after the first continual mix round

It is easy to verify that regardless of the timings of the payments received by Alice, the distribution last mixing times for each of her spendable chunks is always $U[0, \Delta_A]$. This nullifies the timing channel, except for the matter of picking Δ_A. If Alice makes a payment with a random subset of her spendable chunks, Eve can infer Δ_A with high accuracy.

Picking Δ involves a trade-off. From the point of view of a business, if Δ is too high, it adds latency to the operating cycle and decreases cash flow. If Δ is too low, it leads to a higher depreciation rate of long-term assets due to the mixing fees incurred by continual mixing. Further, clients must consider each others' choices in picking Δ, since anonymity loves company and highly unusual values of Δ will help Eve.

Given these constraints, we propose several globally fixed values of Δ: for instance, a day, a week, a month, and a quarter; each client is free to pick the value that best suits their operating patterns. Alice can now expect her anonymity set to be the set of all Mixcoin clients who have the same value of Δ.

Some inference attacks are hard to prevent with any mixing system. For example, if Alice owes Bob a highly unique amount of money, and neither Alice nor Bob transacts with any other users, this information is sufficient to link Alice's outflow with Bob's inflow. Unlikely as such situations are for most users in the real world, they pose a problem for analysis of anonymity of our system.

B Improving mix trustworthiness

If a mix cheats, the cheated client can ensure that the mix gets a poor reputation. But how can a mix build a reputation for trustworthiness? Even if there are no theft reports against it, it might simply be because the mix doesn't have much volume yet. Further, to the extent that more popular mixes may offer better anonymity (Section 7.3), clients would like to estimate mix transaction volumes.

In this section we discuss ways to better measure, as well as prove, mix trustworthiness, and even a mechanism for recourse against cheating mixes. These are all "out-of-band" and do not require modifications to the Mixcoin protocol.

B.1 External reputation

While some mix operators may choose to be anonymous, others may be comfortable revealing their real-world identity. A bank or trusted community member could leverage their external reputation to increase trust in their mix service.

B.2 Throttling

Throttling, or rate limiting by the client, lets Alice limit her exposure to a given mix at any given time. If Alice wants her maximum exposure to M to be E, she transacts with M at the average rate of $\frac{E}{\delta_{\max}}$ per block, where $\delta_{\max}$ is the maximum mix delay that she picks for M. If she stops transacting with M as soon as she detects misbehavior, then M can steal at most E of her coins.

B.3 User reports

To estimate volume, client users could publish through out-of-band channels, such as forums, logs containing aggregate statistics about their usage of various mixes (*e.g.,* "Alice mixed 10,000 chunks through mix M_1 in August"). If these are reputable members of the community (for example, with longstanding active accounts), observers can be reasonably confident that they are not sybils. Such reports provide lower bounds on mix volume.

B.4 Mark and recapture

The mark-and-recapture method for estimating wildlife populations (e.g., [15]) could be used to estimate a mix's escrow reserves and hence its volume. The method involves engaging the mix in n transactions over a short period, and observing what fraction of these get forwarded among the set of corresponding return transactions. If the transaction volume of the mix is Q, then at any time the escrow pool contains Q transactions, and the expected number of corresponding returns is approximately n/Q when n is much smaller than Q. The mix may attempt to inflate this measurement by simulating transactions of sybil clients and contributing its own funds to the escrow pool. To defeat sybil detection by transacting with other mixes would incur fees proportional to the inflated volume. Thus, to inflate the apparent volume to twice the actual amount, the mix would have to forego its entire profits.