Mixcoin

Anonymity for Bitcoin with accountable mixes

(Full version)

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 using the Bitcoin currency system. We build on the emergent phenomenon of currency mixes, adding an accountability mechanism to expose theft. Unlike other proposals to improve anonymity in Bitcoin, our scheme can be deployed immediately with no changes to Bitcoin itself. We demonstrate that incentives of mixes and clients can be aligned to ensure that rational mixes will not steal from clients. We contrast mixing for financial anonymity with better-studied communication mixes, demonstrating important and subtle new attacks.

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 [4]. 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 market capitalization of over US$10 billion in early 2014. Bitcoin can be thought of as a public, distributed ledger that logs all transactions in order to prevent double spending. Using a proof-of-work system, the integrity of the ledger is maintained as long as a super-majority of the computing power[4] is contributed by honest participants [20].

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 2.2). This has led to the rise of *mixing services* (or *tumblers*[5]) which promise to take

[4] The amount of computing power needed to subvert the mining process is an active area of research [12].

[5] Bitcoin tumblers are named after machines used to wash physical coins.

a user's coins and randomly exchange them for other users' coins to obfuscate their ownership, though these come with no protection from theft.

The Bitcoin community is well aware of this issue, leading to much interest in the provision of stronger anonymity. One proposal, Zerocoin [18], provides strong cryptographic anonymity but cannot be deployed without modification to miners' transaction validation process. CoinJoin [16] allows a group of Bitcoin users to randomly permute their coins without entrusting them to a third party, but it is complex for participants to protect their anonymity from each other and the protocol can be easily stalled by a malicious entity. We provide more detail of these solutions in Section 2.3, but note that Bitcoin still lacks a robust protocol with strong anonymity properties that can be deployed immediately, which is the problem we seek to address. Our strategy is to build on the existing phenomenon of mixes, but to add an independent cryptographic accountability layer. Our main contributions include the following:

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 networks for Bitcoin. Similar to experience from anonymous communication networks, we show how chaining multiple mixes together both improves anonymity and guards against an adversary in control of some of the mixes. There are important differences from communication mixes, however, introducing subtle new attacks (Section 7). Mixcoin offers strong anonymity against an adversary who, in addition to being able to analyze the Bitcoin block chain, controls some mixes and can engage in transactions with the honest mixes.

Our design philosophy is a very general core protocol, allowing clients and mixes to specify a variety of free parameters. Yet we expect that, because anonymity loves company [9], 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 Bitcoin users may eventually mix their funds as a matter of course, finally making an anonymous, decentralized digital currency a reality.

2 Background and related work

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.[6] 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[7] 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.

2.2 Deanonymization in Bitcoin

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 [17,22,24,2].

Four major techniques have been explored. First, in multi-input transactions, shared spending authority is evidence of joint control. The rationale is that different entities are unlikely to be co-spenders of the same transaction. Second is the use of taint analysis to track flows. The taint between addresses κ_{in} and κ_{out} is defined as the percentage of the balance of κ_{out} that came from κ_{in}. Taint analysis can overcome simple manual obfuscation techniques. More generally, one can analyze Bitcoin's overall *transaction graph*, with each address as a node and each transaction as a weighted, directed edge between nodes. Various heuristics can be used for clustering addresses and tracking flows, such as detecting the "change address" in a multi-output transaction (the rationale being that the change address is probably controlled by the same entity as the input address). Third, there are various side channel attacks such as timing,

[6] This is a simplification for conceptual clarity. On a technical level, monetary value is assigned to transactions and not to addresses.

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

precise values, and network-layer information. Finally, an attacker can use using auxiliary information (e.g., in forums) to connect pseudonyms to identities.

The first three techniques have proved quite powerful. The fourth technique (use of auxiliary information) is necessary to link a Bitcoin user to a real-world identity. However, it is more difficult for researchers to carry out using public information than for a well-funded adversary who may compel disclosures from various service providers, so the academic literature has generally stopped short of this step. Nonetheless, the evidence suggests that the anonymity properties of Bitcoin cannot be relied upon in the face of a committed adversary.

2.3 Anonymity technologies for Bitcoin

There have been two prominent attempts to provide stronger anonymity in Bitcoin. One is an academic proposal called Zerocoin [18] which uses cryptographic techniques (specifically, an accumulator with a zero-knowledge proof of inclusion) to break the link between individual Bitcoin transactions without adding trusted parties. Unfortunately, deployment of Zerocoin (or related proposals, such as Pinocchio Coin [7]) would require modifications to Bitcoin which appear unlikely due to the computational overhead. Additionally, the cryptographic constructions proposed require a trusted setup phase, where knowledge of the trapdoor created during setup would allow an attacker to forge coins.[8]

A second approach, arising organically from the Bitcoin community, is *mixing*, 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[9] 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 [19]. 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.[10] 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.

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

[8] The trusted setup may be obviated using techniques like RSA-UFOs [26].

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

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

sellers, while some "eWallet" services promise that when users withdraw funds they will receive random coins from the provider's reserves.

Finally, a proposal called CoinJoin [16] enables n users to atomically transfer funds from their n input addresses to their n output addresses in a single transaction, signed by each key corresponding to the inputs, which is possible without modification to Bitcoin. Since no entity will sign the transaction if its own output address is not specified correctly, there is no risk of theft. However arranging the output addresses in a way that prevents the participants from learning the correspondence to input addresses (necessary for anonymity) introduces complexity, with most designs relying on external cryptographic protocols [6] and/or a third-party facilitator. An attacker can easily block the transaction by participating initially to form the transaction, but failing to sign the finalized transaction. A simpler alternative adds a trusted "facilitator" to perform the mixing, though this re-introduces trust issues inherent in mixing services.

Despite these problems, enthusiasm and interest in mixing is high. The recent press on "dark wallets" for Bitcoin using Coinjoin is an example [14]. There has also been vigorous discussion in the Bitcoin community on ideas for making mixing secure and viable. However, to the best of our knowledge, the ideas we introduce have not been explored previously.

2.4 Mix networks for anonymous communication

Mix networks were introduced by Chaum in 1981 for anonymous communication [3]. Significant research has analyzed the relationship between design parameters, such as route selection and flushing policies, and the resulting anonymity (see [21] for a survey), much of which is broadly applicable to financial mixing.

Verifiable mixing, beginning with Sako and Killian [25], 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 [13], 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

While there are some differences in how current mixes operate, we describe the model on which Mixcoin's design rests. 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 so 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. Ideally, this channel will also be deniable for clients, so that the mix cannot prove that any client contacted it about mixing funds.

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[11] 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 paying randomixed 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 public policy for acceptable terms.[12] Alice similarly has no obligation to transfer funds 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.

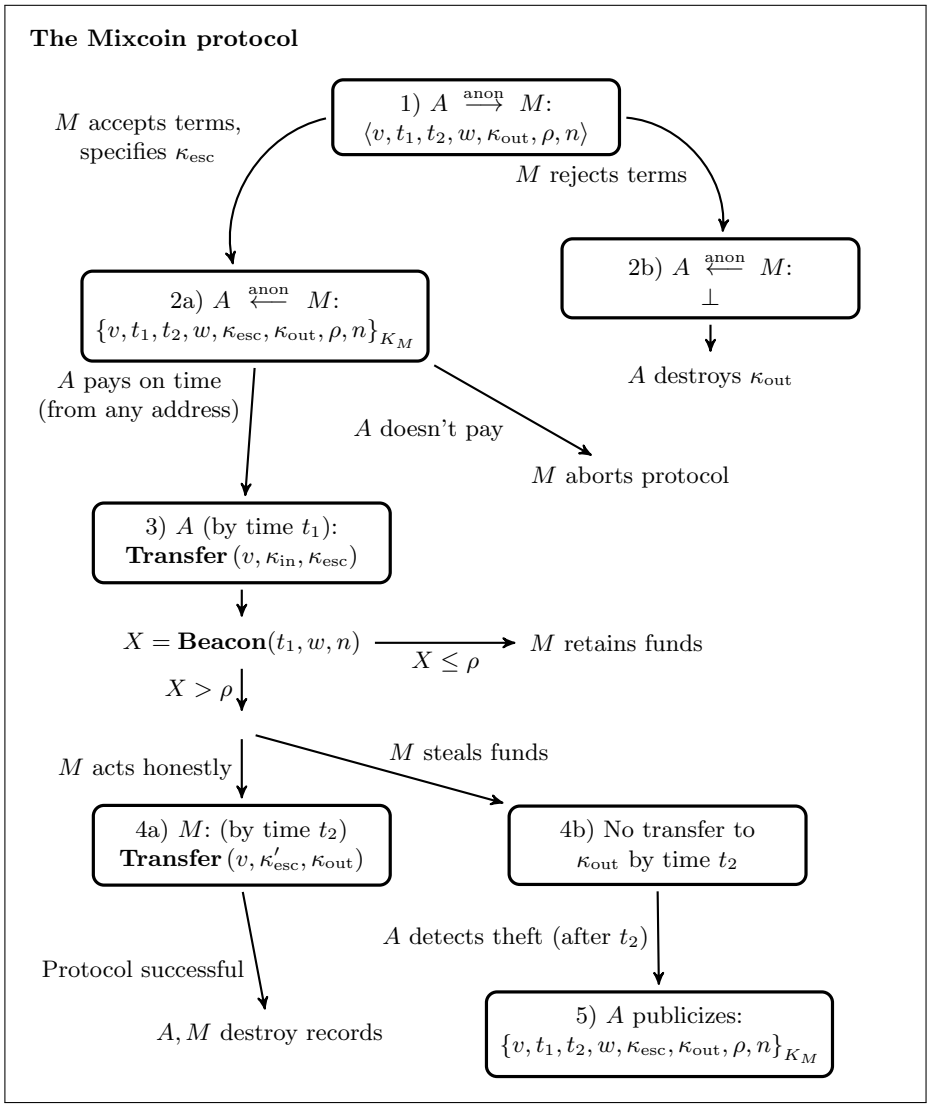
If Alice does transfer the agreed value v to κ_{esc} by the deadline t_1 (Step 3), then the mix is obligated to transfer an equal value to κ_{out} by time t_2 (unless the funds are retained as a mixing fee—see Section 4.4). If the mix does so faithfully (Step 4a), then both parties should destroy their 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),[13] 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 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

[12] As presented, this is an all-or-nothing negotiation with Alice proposing terms and the mix either accepting or rejecting. More complicated negotiations are possible but we expect this to be the simplest.

[13] 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.



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 ($\{\}$) indicate a digital signature.

parties should pick addresses with no other possible source of income so that the other party must themselves pay to fulfill the contract. This has the benefit of minimizing what Alice reveals if she publishes a warranty, as κ_{out} is useless if the mixing transaction it was created for ends up failing.

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 [23] used in some micropayment systems.[14]

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*. There are other options, such as a coin-tossing protocol between Alice and the mix, but these would introduce an extra round of communication.

If the beacon is external to Bitcoin (*e.g.,* NIST's beacon [1] or financial data [5]), warranties would need to be synchronized to real-world time to enable 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.[15] Because each block includes the value of the previous block, every transaction during a confirmation period of w blocks adds randomness.[16] 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

[14] 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.

[15] 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. We address the infeasiblity this attack in Appendix C.

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

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 that variance in the effective mixing fee rate increases for users mixing a small number of chunks. To address this, v should be kept as low as possible so that most users can mixing 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.[17] 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[18] for both 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 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. 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. It is also possible for mixes to collaborate and mix their reserves available for transaction fees using CoinJoin, which may be far more practical for mixes to organize without being subject to denial-of-service issues.

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

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

[18] In pipelined sequential mixing, which we will discuss in Section5, most mixes will need only pay one transaction fee.

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 $\kappa_{\mathrm{esc}_{i+1}}$ which she previously received from mix M_{i+1}. After obtaining N signed warranties,[19] 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 split 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.

5.1 Dynamic selection of mixing paths

Unlike traditional mix networks, an alternate approach is possible in which Alice doesn't negotiate an end-to-end path up front but receives her chunks back after each mix round at fresh intermediate addresses. This approach enables Alice to dynamically determine which mixes to use and how many, and would hide from each mix which order it is in the mixing sequence. The downside of this approach is added latency, as well as paying double the number of transaction fees.[20] Given these drawbacks, we anticipate that dynamic mixing is a less attractive option, though it could be used in hybrid form with Alice performing multiple sequential mixes to prevent negotiating a very long path up front.

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

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

[20] Transaction fees can still be paid by mixes to avoid the chunk size value declining, but this will require an increase in ρ which Alice will have to pay.

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 $\mathrm{E}[\mathsf{abscond}] = Q\bar{t}$.

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[21] at a rate r (per block), the net present value of indefinite honest behavior with a fee rate ρ is:

$$\begin{aligned}\mathrm{E}[\mathsf{continue}] &= \rho Q + (1-r)\mathrm{E}[\mathsf{continue}] \\ &= \rho Q + (1-r)\rho Q + (1-r)^2\rho Q + \dots \\ &= \frac{\rho}{r}Q\end{aligned}$$

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.[22] 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. Absconding might be slightly more favorable in practice due to several factors missing from a simple analysis, such as super-exponential time discounting by the mix, the risk that business may decline, or the fact that the mix can choose to abscond during a period of unusually high transaction flow.

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. Unfortunately Alice can't know what mixes' operating costs are, and therefore cannot verify that a rational mix will choose to stay in business at a given instant. Nevertheless, since absconding is a one-time action, she can gain a lot of comfort from the longevity of the mix.

---

[21] 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.

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

7 Anonymity properties

Given the large number of negotiable parameters in Mixcoin, our anonymity analysis is theoretical. The actual anonymity guarantees will depend on how the mixing ecosystem evolves. However, we can describe a threat model, quantify anonymity under a simplified model, discuss optimal strategies for mixes and clients, enumerate some attacks and how to avoid them. We draw many connections to the extensive literature on mix networks for communication, dating to the initial proposal of communication mixes in 1981 [3]. That said, financial mixing also differs from communication networks in important ways.

7.1 Threat model

Mixing literature typically analyzes the anonymity set of possible partners in a communications network. For unidirectional or message-based systems (as opposed to streams), the anonymity set may be considered separately for senders and receivers though the two often have symmetric properties. With Mixcoin the common case is a single linkable input address sending multiple chunks to distinct output addresses which never receive any other input. 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 model studied for communication mixes.[23] 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 [27]. Replay should be impossible in Mixcoin due to the double spending prevention in Bitcoin. Flooding is possible and we should assume attackers have this capability. Delaying Bitcoin transactions is possible, but not trivial. Any miner can refuse to include transactions in blocks they mine, but this is only as effective as the amount of mining power the attacker can control. An attacker might also try to flood miners with larger value transactions, but there is no way to do so selectively and this would thus amount to a denial of service attack on all of Bitcoin. Assuming an attacker doesn't control a substantial portion of the mining pool, blocking transactions indefinitely should be effectively impossible.

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

[23] Tor is notably not designed to withstand attack by a global passive adversary, as Tor relays provide no mixing of traffic [10].

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 indisinguishability may fail (which we will discuss in Section 7.3) but the anonymity offered is quite strong in this case.

Phantom mixing Mix indistinguishability brings about an interesting possibility, also with no precedent in communication mixes. Users may apparently gain some anonymity by "phantom mixing": sending their funds through escrow addresses they control themselves. To a passive observer this is indistinguishable from a user sending funds through a genuine third-party mix. Of course, if a significant proportion of users do this, these users will lose anonymity as attackers will be able to infer that the owner of a chunk's final output address is the same as the owner of the chunk's input address with elevated probability. Therefore we don't consider this to be a likely phenomenon.

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 to M. Observing the majority of links therefore appears to require an attacker generate a large portion of the mix's traffic. Even against an implausibly strong active attacker who can link every escrow address to its originating mix, the anonymity set still snowballs quickly with each mixing round as in traditional communication mixes. We demonstrate this via simulation in Appendix D.

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,[24] and will then uniformly at random choose a delay $\delta \in_R [w, \delta_{\max}]$ before forwarding Alice's chunk.[25] 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.[26] 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.

Whatever the mix's published policy is, Alice can easily verify that the random distribution of δ matches the mix's policy since we expect that over time she will send a large number of chunks to the mix. If the mix cheats and picks delays that don't match the distribution specified in its policy, it is not a violation of the warranty (as long as the mix adheres to its maximum delay). However, individual clients can detect this cheating and take their business elsewhere.

Latency/anonymity trade-off If we are willing to sacrifice mix indistinguishability, mixes can significantly decrease latency by skipping the normal confirmation delay of $w = 6$ blocks, batching all of their Q permuted chunks in a given block into one atomic transaction with Q inputs and Q outputs. In the event of a fork, if any input transaction to the mix is removed from the block chain the entire transaction will be invalid and can't be replayed, protecting the mix. Thus there is a trade-off between anonymity and latency, and much lower latency can be achieved at the cost of mix indistinguishability.

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

[24] Allowing different delays per client would open the possibility of free-riding and make anonymity analysis much more complex [11].

[25] 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.

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

those chunks sufficiently at the same time, then those chunks will have the same anonymity sets, and her payment is still unlinkable.

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 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. In Tor, there is a similar argument for why building circuits improves anonymity as opposed to routing each packet independently.

The selection of mixes for each chunk's path poses an interesting challenge. For anonymity, each chunk's path should be chosen independently at random from the same distribution. We envision *mix reputation lists* (MRLs) published by neutral evaluators which estimate legitimate mix transaction volume via the methods outlined in Appendix B. Such lists would include a suggested probability distribution over mixes based on reputation scores, limiting exposure to unreputable mixes without weighing too heavily towards a small set of the most reputable mixes which could collude to break anonymity. Users will want to share a random path selection algorithm to avoid leaking information, and therefore we expect a few dominant MRL services to arise.

It seems tempting to simply use the N most popular mixes permanently, since using more popular mixes improves the anonymity set. However, using popular mixes improves anonymity only against an very powerful active adversary, and popular mixes may be more attractive targets for the adversary to compromise leading to a *greater* probability of picking a completely compromised path. This question also impacts how many mixes we expect to see. If there is a strong preference for popular mixes, then the barrier to entry will be high and the first entrants will corner the market. If not, there will be a diversity of mixes. It is impossible to definitively answer this question, but our best guess is that there will be perhaps a dozen popular mixes followed by a long tail, and clients will use a combination of the two types in the paths they choose.

7.6 Convergence of free parameters

Our design intentionally left 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[27] to avoid splitting their anonymity sets based on parameter choices.

[27] 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.

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.[28] 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,[29] while decreasing v will require proportionately more runs of the protocol and more transactions in the block chain.[30]

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, as discussed in Section 7.5

Second and more importantly, all clients have an incentive to choose the most popular parameters in an application of the "anonymity loves company" principle [9]. 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, or possibly a small number of different "flavors" of Mixcoin, to maximize anonymity.

It is an open question how these parameters might change over time. For example, as Bitcoin experiences inflation or deflation the optimal chunk size will change. We expect that the ecosystem would react by introducing a new chunk size, with a transitional period where some mixes and clients use the old size and others use the new one, and the new chunk size gradually winning over.

7.7 Side channels

Financial mixing introduces several subtle side channels.[31] 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. 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

More subtle issues arise becaues 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 subet 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.[32] 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 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 sidechannels we have only briefly explored here.

Acknowledgements. 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 and Eran Tromer.

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

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. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM 24(2), 84–90 (1981)
4. Chaum, D.: Blind signatures for untraceable payments. In: CRYPTO (1982)
5. Clark, J., Hengartner, U.: On the Use of Financial Data as a Random Beacon. Usenix EVT/WOTE (2010)
6. Coutu, O.: Decentralized Mixers in Bitcoin. In: Bitcoin Conference (2013)
7. 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)
8. Diaz, C., Seys, S., Claessens, J., Preneel, B.: Towards measuring anonymity. In: PETS (2003)
9. Dingledine, R., Mathewson, N.: Anonymity Loves Company: Usability and the Network Effect. In: WEIS (2006)
10. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: USENIX Security (2004)
11. Dingledine, R., Serjantov, A., Syverson, P.: Blending different latency traffic with alpha-mixing. In: PET. pp. 245–257. Springer (2006)
12. Eyal, I., Sirer, E.G.: Majority is not Enough: Bitcoin Mining is Vulnerable. In: FC (March 2014)
13. Golle, P.: Reputable mix networks. In: PETS (2004)
14. Greenberg, A.: DarkWallet Aims To Be The Anarchist's Bitcoin App Of Choice. http://www.forbes.com/sites/andygreenberg/2013/10/31/darkwallet-aims-to-be-the-anarchists-bitcoin-app-of-choice/ (2013)
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. Maxwell, G.: CoinJoin: Bitcoin privacy for the real world. https://bitcointalk.org/index.php?topic=279249.0 (August 2013)
17. 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)
18. Miers, I., Garman, C., Green, M., Rubin, A.D.: Zerocoin: Anonymous Distributed E-Cash from Bitcoin. IEEE Symposium on Security and Privacy (2013)
19. Möser, M.: Anonymity of Bitcoin Transactions: An Analysis of Mixing Services. In: Proceedings of Münster Bitcoin Conference (2013)
20. Nakamoto, S.: Bitcoin: A peer-to-peer electionic cash system (2008)
21. Raymond, J.: Traffic analysis: protocols, attacks, design issues, and open problems. In: PETS (2001)
22. Reid, F., Harrigan, M.: An analysis of anonymity in the bitcoin system. In: Security and Privacy in Social Networks (2013)
23. Rivest, R.: Electronic Lottery Tickets as Micropayments. In: FC (1997)
24. Ron, D., Shamir, A.: Quantitative Analysis of the Full Bitcoin Transaction Graph. FC (2012)
25. Sako, K., Kilian, J.: Receipt-Free Mix-Type Voting Scheme - A Practical Solution to the Implementation of a Voting Booth. In: EUROCRYPT (1995)
26. Sander, T.: Efficient Accumulators Without Trapdoor. In: ICICS (1999)
27. 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 Aggregating theft reports

Essential to the success of Mixcoin is an external mechanism for client software to report theft by mixes. These reports will be aggregated and re-distributed to clients. Since it is impossible to falsely allege theft, this mechanism need not be trusted. Further, this aggregator could combine theft report data with estimates of mix volumes to publish mix reputation lists.

B.4 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.5 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.

A more rigorous version of this is for a reputable entity, such as Consumer Reports, to perform a survey of publicly known mixes by periodically carrying out small transactions and report the results. The success of this approach depends on the (network-level) anonymity of the surveyor's actions.[33]

B.6 Fidelity bonds

As we discussed in Section 6, once a mix steady state, it is rational to stay in business. Bootstrapping trust is a challenge for new, unknown mix service with many parallels in business and in life. In fact, the earliest modern banks faced the issue of getting clients to entrust their money. How could clients be sure the bank wouldn't disappear overnight? The banks solved this problem by signaling with huge, expensive buildings with gleaming facades that they were in it for the long haul.[34]

Effort and money expended by the operator of a new mix in advertising the service could serve as such a costly signal. But Bitcoin also has a mechanism for explicit signaling: *fidelity bonds.*Somewhat different from real-world fidelity bonds, Bitcoin fidelity bonds are a protocol for the owner of a Bitcoin address to sacrifice coins, *i.e.,* make them provably unspendable. We envision that mixes might optionally put up a fidelity bond to gain trust initially.

One possible game-theoretic explanation for why signaling by banks worked, and why fidelity bonds by mixes might work, involves *asymmetric information.* Let's say a business (a bank or mix) has done the research and concluded that the market is ripe for a new entrant. It therefore sets up shop with honest intentions. But consider clients who typically do not have access to this market research or other reasons for the business's belief in its long-term viability. To such a client, the business may very well have the intention of never reaching the steady state, but instead absconding as soon as a few tentative clients entrust them with money money. By undertaking a costly action, the business signals its belief that it can be viable. Since the sunk cost exceeds what the business might gain by absconding before reaching the steady state, a business that does not believe in its own viability will not undertake the costly action. Via a Bayesian argument, the client can infer that the business does not intend to abscond.

C Manipulating the beacon generated by the block chain

In Section 4.4, we assumed that values in the block chain are unpredictable for both Alice and the mix, and therefore can be safely used to determine which transactions the mix is able to retain as a fee. In reality, the mix might also

[33] As an aside, anonymity is an issue for survey organizations in physical-world transactions as well. Consumer reports had a policy of purchasing vehicles with cash to protect their identity, but had to abandon the practice when car dealers realized that they were the only entity to pay with cash.

[34] Other examples of costly actions undertaken for the purpose of signaling commitment are gang initiations and engagement rings.

control significant mining resources and therefore attempt to influence the value of blocks in the chain to collect a higher effective mixing fee rate than ρ.

This attack faces two major difficulties. First, it is very costly to influence blocks. Because the value of block hashes is the output of a proof-of-work scheme, the only effective means for influencing their value is to discard some valid blocks which M finds in hopes of finding an alternate block which increases the number of mixing fees M collects. Discarding otherwise valid blocks makes the attack very expensive, given the high value of mined blocks (currently around US$20,000) and expense of computing them.

Second, even given the ability to discard some blocks in the hope of finding a block which allows M to collect a higher mixing fee than usual, it is difficult to find a block which will significantly increase the number of mixing fees retained. Assuming the mix is holding Q transactions and is able to choose from a small number n of potential blocks, each potential block enables the mix to retain an independently random subset of the Q transactions with probability ρ each. The number of transactions retained is therefore a multinomial distribution with expected value ρQ and variance $Q(1-\rho)\rho$. Under reasonable assumptions for Q and ρ, for example $Q = 100$ and $\rho = 2\%$, even generating 1000 valid blocks (which would be absurdly expensive computationally) produces less than a 40% chance of finding a block which even doubles the mix's fee rate.

D Growth in anonymity against a strong attacker

As claimed in Section 7.3, even against an attacker able to identify which mixes correspond to escrow addresses in the block chain the probability distribution over the anonymity set of a chunk rapidly approaches the uniform distribution after multiple rounds of mixing. Assume that there are m mixes, and Q chunks are being mixed in N concurrent, synchronous rounds. In other words, in each round, each chunk is assigned to a given mix, and its anonymity set for that round is the set of all other chunks assigned to that mix in that round. We can quantify this by measuring the statistical distance (L_1 distance) from the uniform distribution. Figure 1 summarizes the results of simulation.

The first line shows $m = 100$ mixes, with each mix being equally popular (*i.e.,* in each round, each chunk is independently equally likely to be sent to each mix). We observe that the statistical distance of the PDF from uniform drops exponentially with the number of rounds. After 7 rounds it is under 0.1, and after 10 rounds it is about $2.4 \cdot 10^{-4}$.

The other lines show the situation when the mix traffic is skewed according to a power law distribution, *i.e.,* in each round, each chunk is independently sent to mix i with probability proportional to $\frac{1}{i}$. Here the distribution converges to uniform even faster. Decreasing the number of mixes has a similar effect. These are both illustrations of the fact that anonymity loves company. The *degree of anonymity* [8] converges to 1 similarly. With 100 mixes with power-law traffic, after 4 rounds it exceeds 0.99 and after 10 rounds it is about $1 - 2 \cdot 10^{-9}$.

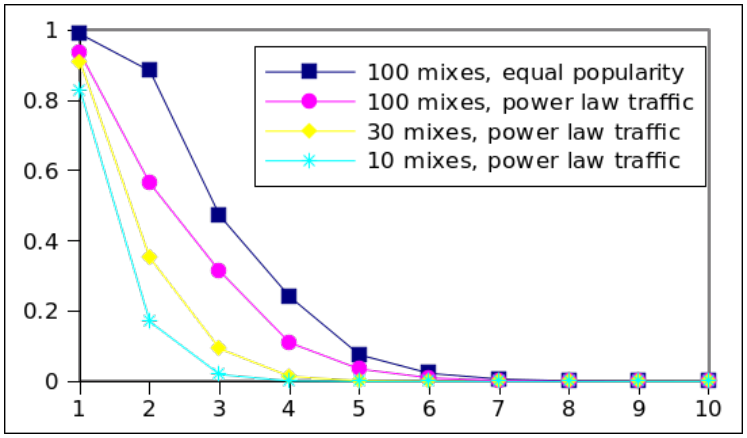


Fig. 1. Simulation. Statistical distance (L_1 distance) between PDF of a chunk's anonymity set and the uniform distribution, as a function of the number of rounds. $Q = 1000$ chunks in all experiments.

While this is a simplified model, we can infer a lot about the anonymity properties of Mixcoin. Qualitatively, after a few rounds Alice's chunk is uniformly mixed with all the other chunks that are being mixed contemporaneously. The chunk's anonymity set does not include chunks that were mixed at a different time, for example a month ago. If the attacker has compromised some of the mixes, as long as some of the rounds of mixing (say N') went through honest mixes, the anonymity properties are identical to using N' rounds of mixing with all honest mixes.