Sybil-Resistant Mixing for Bitcoin

George Bissias A. Pinar Ozisik Brian N. Levine Marc Liberatore
School of Computer Science, University of Massachusetts Amherst
{gbiss, pinar, brian, liberato}@cs.umass.edu

ABSTRACT

A fundamental limitation of Bitcoin and its variants is that the movement of coin between addresses can be observed by examining the public block chain. This record enables adversaries to link addresses to individuals, and to identify multiple addresses as belonging to a single participant. Users can try to hide this information by mixing, where a participant exchanges the funds in an address coin-for-coin with another participant and address. In this paper, we describe the weaknesses of extant mixing protocols, and analyze their vulnerability to Sybil-based denial-of-service and inference attacks. As a solution, we propose Xim*, a two-party mixing protocol that is compatible with Bitcoin and related virtual currencies. It is the first decentralized protocol to simultaneously address Sybil attackers, denial-of-service attacks, and timing-based inference attacks. Xim is a multi-round protocol with tunably high success rates. It includes a decentralized system for anonymously finding mix partners based on ads placed in the block chain. No outside party can confirm or find evidence of participants that pair up. We show that Xim's design increases attacker costs linearly with the total number of participants, and that its probabilistic approach to mixing mitigates Sybil-based denial-of-service attack effects. We evaluate protocol delays based on our measurements of the Bitcoin network.*

Category and Subject Descriptors
C.2.0 [**Computer-Communication Networks**]: General-*Security and protection*; K.4.1 [**Computers and Society**]: Public Policy Issues-*Privacy*

General Terms Security, Measurement

1. INTRODUCTION

A growing number of people have turned to decentralized *virtual currencies* (VCs) such as Bitcoin [20] or Litecoin (`http://litecoin.org/`) for convenience, speculation, or as a potential source of financial privacy and security. For example, people generally lack control over the efforts of credit agencies and marketing firms to mine information from financial transactions. And merchants that accept credit and debit cards have a history of security failures (e.g., [13,22]).

The benefits of virtual currencies are many: low transaction fees, transactions over the Internet, and potentially, convenience and privacy. Unfortunately, VCs are not a complete solution to the problem of economic privacy. Imagine purchasing coffee from a café and revealing the balance of your bank account at the register. This scenario is the status quo when using Bitcoin and related VCs because they use a public transaction log (called a *block chain*) to prevent counterfeiting. As a result, user wallets (called *addresses*) are pseudonymous rather than anonymous: a user's actions cannot be linked to an individual, but multiple actions of the same user can be linked together. Although it is broadly recognized [5,17,23,23] that Bitcoin and similar VCs are not private as-is, there are to date no robust decentralized services for providing privacy that break the links among users, addresses, and transactions.

Currently, there are three mechanisms for *mixing*, which is the process of transferring funds between two address without recording their relationship to the public block chain.

First, there exist several *centralized* mixing services. Examples include SharedCoin [4] and DarkWallet [2]. Such centralized mixing services are analogous to providing Web browsing anonymity with a single anonymizing proxy, rather than using a more robust solution, such as Tor [11]. The central mixing agent must be completely trusted as it knows which users exchange funds with others.

Second, new VCs have been designed with explicit support for mixing, wherein users create new addresses and move coins among them to explicitly *unlink* the addresses and past transactions. For example, Miers et al.'s Zerocoin protocol [18] allows anonymous transactions based on zero-knowledge proofs, and Ladd [15] has proposed a scheme based on blind signatures. These schemes are incompatible with existing VCs, of which there are many and for which the most popular comprise an enormous amount of capital. These improvements as separate VCs are useful, but we assert a privacy solution for existing VCs is of critical importance.

Third, peers can rely on one another for mixing without a centralized or third party. For example, Barber et al. [7], CoinShuffle [24], CoinJoin [1], and others have proposed methods for two or more parties to directly mix their coin. These approaches obviate centralized trust and are compatible with existing Bitcoin-like currencies. Unfortunately, they are also limited in fundamental ways: none provide a complete, decentralized protocol that pairs specific partners while avoiding

WPES'14, November 3, 2014, Scottsdale, Arizona, USA.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3148-7/14/11 ...$15.00.
http://dx.doi.org/10.1145/2665943.2665955.

Sybils [12]; all are cost-free for participants, and therefore are trivially subject to denial-of-service attacks; none analyze performance over multiple rounds of mixing; and some leave information on the block chain about which peers paired to exchange coin, a significant vulnerability.

Contributions. We propose **Xim**: the first complete solution for two-party mixing that is compatible with Bitcoin and related VCs. It is the first decentralized protocol to *simultaneously* address Sybil attackers, denial-of-service attacks, and timing-based inference attacks. Xim is a multi-round protocol with significant privacy gains over single instances of CoinJoin or fair exchange protocols [6,7]; the gain is analogous to the use of multi-hop Tor circuits as an improvement over a single-hop communications mix or anonymous proxy. Our contributions are as follows.

- Xim includes a decentralized system for anonymously finding mix partners based on ads placed in the block chain, each incurring a participation fee. No outside party can confirm or find evidence of peers that pair up.
- We quantify the success rate of Xim's tunable, multi-round mechanism, which serves to thwart Sybil-based linking attacks. For example, when a Sybil attacker controls $x = \frac{1}{3}$ of all mixing peers, our protocol can be set to mix with a 99% success rate with low costs, and with a delay that does not increase with the amount of coin mixed.
- We show that because of Xim's participation fees, launching inference or DoS attacks based on Sybil identities are costly. For a given success rate, a Sybil attacker's costs *grow linearly with the number of mix participants*, while honest participants' costs remain small, fixed, and *constant.*
- Xim's probabilistic approach mitigates the effects of Sybil-based denial-of-service attacks. An attacker that is willing to pay to control a fraction x of the mix participants increases costs for victims by only a factor of $1 + x$. For example, if an attacker controls $x = 10\%$ of all peers, honest participants costs increase to 110%.
- Our protocol thwarts timing attacks by coordinating the actions of participants. Participants agree upon the number of rounds they seek to mix and when to start mixing.
- We empirically analyze the delay performance of our protocol if used on Bitcoin. We measured the commit delays of transactions on Bitcoin for 21 days. Our measurements show that Bitcoin transaction delays fit well to a heavy-tailed, log-normal distribution, which has implications beyond the operation of Xim.

We begin by analyzing previous work on mixing. We show that previous work is subject to timing attacks and trivial DoS and low- or no-cost Sybil attacks.

2. PROBLEM DEFINITION AND RELATED WORK

Alice is known to control a virtual currency address A. She would like to move the coin from A to a new address A', such that no one knows that Alice controls A'. In other words, she wants addresses A and A' to be *unlinkable*, despite the public visibility of transactions on the block chain.

2.1 Decentralized Mixing

Alice can achieve this unlinkable transfer of funds trivially with a trusted partner Bob. *Mixing* schemes allow Alice to trade coin-for-coin with Bob, using his addresses B and B'. Alice agrees to transfer coin from address A into address B', and Bob transfers the same amount of coin from his address B into A'. Now each pair (A, A') and (B, B') is unlinkable by a third party because there is no record of the partnership between Alice and Bob's address on the public block chain.

This approach has a problem: Alice may not trust that Bob will reciprocate her transfer. A solution to this problem is use of a trusted third party, in this case, a centralized mix to *(i)* pair participants and *(ii)* oversee and enforce the exchange and unlinking process.

The use of a trusted third party has well-known problems of its own: Not only must Alice and Bob trust that a centralized mix, Trent, will not steal their coins, but they must also trust that he will not help others — or be attacked by others — that want to link Alice and Bob's old and new addresses. Even if Trent has integrity, it's possible that others might launch a denial-of-service attack on Trent. A better solution is to remove Trent while still allowing the mutually untrusting Alice and Bob to find one another and to fairly and unlinkably exchange coin.

Thus, *our goal is to develop a decentralized protocol that allows Alice and Bob to discover, partner, and unlinkably but fairly exchange coin* with one another. Our solution, Xim, is a protocol that functions correctly in the presence of malicious mixing partners, forces Sybil attackers [12] to pay costs at least linear to the number of participants in the mix and is more robust against denial-of-service attacks than previous approaches. Xim also attempts to hide the IP address of the parties involved in mixing [14]. Our goal is not to defeat all possible inference attacks — attacks leveraging outside information to link many different addresses to the same owner — but we do show how Xim addresses both intersection and timing attacks.

2.2 Assumptions and Attacker Model

Our protocol is designed to work immediately with Bitcoin, but should work with any VC that provides a similar set of features. In particular, we assume a VC with a public block chain and addresses controlled by public-private key pairs. We assume all participants, including attackers, can create new, empty addresses at will, without cost, and without being linked to the old addresses. That is, participants and addresses are not one-to-one. We assume that Bitcoin's limited scripting language is used to create transactions.

We consider two types of attackers with differing objectives. Each operates within the confines of the VC and mixing protocols. We assume that the cryptographic mechanisms that support the currency are not a source of vulnerability. The first type seeks to link two addresses controlled by a single user either through information remaining on the block chain, or by participating as a peer in the distributed mix protocol. The second type is disruptive. He seeks to disrupt the mixing service or increase the cost of mixing for honest participants, even though the he won't profit from the delay or increase costs; he may even spend money on these attacks. This type of attacker is rational if there is an additional external or intrinsic value to the attacker from disabling the mix or from causing the other party to lose coins.

2.3 Related Work

There exist various protocols for mixing Bitcoin and related virtual currencies. We distinguish each protocol or service according to two primary mechanisms: how they find peers to partner with and how they fairly exchange funds.

- **Partner Selection:** Peers exchanging funds must find others and be partnered for later mixing. Some protocols use a centralized third party to select mix partners. Others use a public bulletin board and self organize. For Xim, we propose a Sybil-resistant peer-based protocol.
- **Fair exchange:** Once partnered, peers must fairly exchange funds. Some protocols use a trusted third party, and other protocols make use of CoinJoin [1]. For Xim, we use Barber et al. [7]; the secure multiparty computations proposed by Andrychowicz, et al. [6] would also serve.

First, we summarize the operation of CoinJoin and Barber et al. Each is restricted to fair exchange of coin, and neither proposes a partnering protocol. To our knowledge, there are no detailed proposals for partner selection to review here: all use a trusted third party, except CoinShuffle, which we describe below. Second, we enumerate complete protocols that include both partner selection and fair exchange. In Section 3, we identity vulnerabilities in these past works.

2.3.1 *Decentralized Fair Exchange*

Because of the decentralized architecture of Bitcoin and related VCs, it's appealing to provide a mixing service without involving a third party, and thereby avoid many vulnerabilities. The miners collectively are a third party, though the Bitcoin protocol curtails their ability to misbehave, as invalid blocks will be rejected by other miners. As third parties, the miners enable a fair exchange between two or more mixers; without a third party, fair exchange is impossible [21]. (A protocol is *fair* if no honest participant loses anything valuable if the other party does not behave according to the protocol [21].)

CoinJoin. A popular exchange technique known as CoinJoin [1] was described informally in a Bitcoin-related forum and has subsequently been deployed by commercial mixing services [2,4]. The protocol begins by identifying n participants, each with an existing input address that holds at least δ Bitcoin and a newly created output address. A single *swapping* transaction is published, which moves δ Bitcoin from each input address to one of the output addresses. Because multiple input addresses map to multiple output addresses, this transaction ensures that the n output addresses are indistinct by inspection of the block chain. Therefore, it's impossible to ascertain who controls the δ Bitcoin in any of the output addresses, and so they are unlinked from the input addresses. There are multiple weaknesses in this approach, as we detail in Section 3. CoinJoin itself is not a complete mixing protocol because it neither describes how participants should be selected, nor how the swapping transaction is actually formed. Several services build upon CoinJoin and share its vulnerabilities (e.g., DarkCoin).

Barber's Fair Exchange. Barber et al. [7] propose a protocol that both unlinks exactly two addresses, and cryptographically enforces a fair exchange between them. Although the protocol requires four transactions total (versus CoinJoin's single transaction), and does have limitations as we show in Section 3, Barber et al. provides a significantly stronger basis for Xim. For the remainder of the paper, we will refer to this protocol as **FairExchange**.

2.3.2 *Complete Pairing and Mixing Protocols*

Centralized Pairing or Mixing. We classify any protocol as centralized if it uses a trusted third party for partner discovery or centralized fair exchange. Whether pairing, mixing, or both, the central agent is a convenience, but it must be completely trusted as it knows which users exchange funds with others. As we discuss in Section 3, such centralized mixing services are analogous to providing Web browsing anonymity with a single anonymizing proxy rather than Tor.

MixCoin [10] allows centralized mixes to issue warranties, which users can redeem to damage the mix's reputation if it fails in its promises. The mix collects all-or-nothing fees according to a rate parameter ρ. Until recently Blockchain.info offered a centralized mixing service called Send Shared [3]. It has been replaced with SharedCoin [4], which uses a centralized server for both participant selection as well as CoinJoin transaction creation and publication. DarkWallet [2] operates similarly and imposes no fees on mixing participants (other than normal transaction fees).

Decentralized Pairing and Mixing. Only two protocols provide fully distributed pairing and mixing: our protocol Xim, CoinShuffle [24]. (ZeroCoin [18] and ZeroCash [8] are both decentralized; but neither is backwards compatible with Bitcoin or its related currencies.)

CoinShuffle is a variation on CoinJoin where $n \geq 2$ participants meet through a public bulletin board and agree to mix. Ownership of output addresses is disguised from participants by means of a cryptographically secure shuffling protocol in the spirit of communication mix networks. The swapping transaction is fashioned so that it must be signed by all n participants in order to be considered valid.

3. ANALYSIS OF EXISTING MIXES

In this section, our goal is to identify the vulnerabilities and limitations of existing Bitcoin-compatible mixing protocols so as to justify our design of Xim, which is detailed in Section 4. A summary of our results appears in Figure 1.

3.1 Centralized Mixes

Mixes can be centralized by a third party, but using such an architecture imposes costs and risks. Third parties may charge large fees for the services they provide, and there is no guarantee that they won't conspire with one of the parties or steal from both [17,19]. Bonneau et al.'s [10] mix reputation system reacts to that possibility but is centralized nonetheless. If a centralized mix logs information about partnering, it could later be used, by the mix or another party, to reveal the information the mixing hid. If that revelation is undisclosed, the reputation system is ineffectual. As we show in Section 4, Xim does not suffer these vulnerabilities because its parter selection and exchange algorithms are both decentralized.

3.2 Vulnerabilities in Decentralized Pairing or Mixing

DoS Vulnerability. Protocols such as DarkWallet, SharedCoin, and CoinShuffle do not exact fees from their participants to join the mix pool, so they are all susceptible to a simple denial-of-service (DoS) attack. Suppose that an attacker would like to disrupt one of these mixes. He creates many different identities, each associated with a different IP address and Bitcoin address. The attacker inserts each of these identities into the pool of mix participants at no cost. Any time he is selected as a mix participant he refuses to sign the swapping transaction. If the attacker can create enough identities, it's possible for him to participate in the majority of transactions, which would significantly affect mix

Protocol	Pairing Protocol	Exchange Protocol	Evidence of Pairing?	Attacker's per round Participation Costs	Attacker's per round DoS Costs
MixCoin [10]	Centralized with warranties	Centralized	On block chain	mixing fee rate ρ	N/A
SharedCoin [4]	Centralized	n-way CoinJoin	On block chain	tx fees per Sybil	None
DarkWallet [2]	Centralized	n-way CoinJoin	On block chain	tx fees per Sybil	None
CoinShuffle [24]	Centralized	n-way CoinJoin	On block chain	tx fees per Sybil	None
Xim	Anonymous Decentralized Pairing	Barber et al. [7]	None; timing attack required	τ coin + tx fees per Sybil	τ coin + tx fee per Sybil

Figure 1: Among Bitcoin-compatible mixing protocols, only Xim simultaneously has all desired properties: distributed pairing, fair exchange, no globally visible evidence, and per-Sybil and per-DoS fees.

performance. *The attacker loses no coin in any of the aborted swapping transactions.*

In fact, precisely because CoinJoin and its derivatives (including CoinShuffle) perform the entire mix operation in a single transaction, it's not possible to force participants to pay a fee up front. Any fee would need to be collected earlier using a separate protocol, one that is unclear how to design, and has, to our knowledge, not been proposed. Similarly, Barber et al.'s fair exchange protocol does not include any fees for participating. Abandoning the exchange, resulting in denial-of-service for the victim, does not cost the attacker at all. As we show in Sections 4 and 7, Xim does not suffer these vulnerabilities because it charges a participation fee, and aborting in the middle of a round of mixing (to cause a DoS) incurs a cost.

Intersection Attack. A practical consideration for any decentralized CoinJoin implementation is that the number of participants n in any single swapping transaction must remain relatively low. One reason is the quadratically increasing communication overhead, and another is increased susceptibility to the DoS attack outlined above. Additionally, there is a maximum transaction size.

On the other hand, when n is small, susceptibility to an intersection attack increases. Essentially, the attack consists of watching for repeated participation by a set of wallets that belong to a single participant or involved in a unique behavior, as discussed in Section 5.4. In CoinJoin (and CoinShuffle), the partnering peers are written to the public block chain, which makes the attack available *even to attackers that have not participated in mixing*; in analogy to anonymous communication systems, all attackers are global passive adversaries in these protocols. Xim does not allow for global passive adversaries because when using Barber et al., no evidence is written to the block chain; attackers must participate (and thus pay) to perform the intersection attack. Also, as argued in Section 5.4, the larger pool sizes used by Xim further lower the probability of attack success relative to CoinJoin implementations.

Sybil Attacks. One way to defeat any mixing protocol is to comprise a large fraction of the available participants. As the size of such a Sybil population increases, the chances increase that the parties selected to mix include the attacker. In a protocol that mixes pairwise, a Sybil attacker will know the destination of the funds in any address with which it is paired. DarkWallet, SharedCoin, and CoinShuffle are subject to a no-cost Sybil attack because none charge a fee to participants, and because addresses (i.e., Sybil identities) are free to create in Bitcoin and related VCs.

Participants in these protocols do pay standard transaction fees. These costs are practically negligible; but even if considered substantive, the cost to the attacker is only the sum of the instances where they are successful, rather than being proportional to the number of identities they created. As we detail in the next section, Xim does not suffer these vulnerabilities: Sybil attackers wishing to maintain a fixed success rate must pay for each participating address, and so their costs grow linearly with the total number of mix participants, while honest participants' costs are small, fixed, and constant with the number of participants.

4. XIM: ANONYMOUS PARTNERING AND MULTI-ROUND MIXING

In this section, we detail Xim, a decentralized protocol for anonymously finding partners and mixing with them. Its advantages are *(i)* it mixes *without leaving evidence on the block chain* of the exchanged control of funds; *(ii)* it works *without trusting a third party* to select or learn the pairing of peers; *(iii)* it *thwarts Sybil attacks* because the costs for attackers increases linearly with the total number of participants, but remaining constant for honest participants; *(iv)* it thwarts denial of service attacks because they incur costs for the attacker; and *(v) dishonest participants can never profit*, and honest participants never risk more than a small amount, much less than the amount they are mixing.

4.1 Overview

Xim is described at a high level as follows. Alice wishes to mix coin stored in address A. To begin, Alice commits a transaction that advertises her willingness to mix with a partner, tipping $\tau/2$ coin from A to the miners; she uses Tor to hide her IP address. She is contacted by several willing partners (also using Tor), and she chooses one, who then tips τ coin to the miners; no other peers can recognize their partnership. Finally, Alice confirms the partnership by releasing another $\tau/2$ coin to the miners.

Once Alice has a partner (call him Bob with address B), they swap funds: δ coin from A are transferred to an address B' controlled by Bob; δ coin from B are transferred to A', controlled by Alice. The transfer of funds is performed in a single logical step using Protocol **FairExchange** (See Section 2.3). Once this round of mixing is complete, Alice begins anew, advertising and finding a new partner. We expect that Alice has a total of $m\delta$ coin that require mixing. Alice will thus repeat the protocol m times, but this can be performed in parallel if desired.

No evidence on the block chain. Completing one round for mixing δ coin is sufficient for defeating attackers that aren't participating in mixing because, unlike many other protocols, a Xim pairing leaves no evidence on the block chain. Transactions are structured to include no indication of who is partnered for mixing.

Transaction conventions and notation. If Alice authorizes the transfer of δ coin from address A to address X, we denote the transaction as $\mathbf{T}\{A \overset{\delta}{\rightarrow} X\}$. Note that the *generated* transaction is not actually valid until Alice *publishes* it and it is *committed* by miners to the block chain.

Communication. Our protocols make use of a TEXT field that is included in the transaction's validating signature. In Bitcoin, the comment field is not included in data that gets signed. However, the scripting language that controls the validity of transactions can include the extra values we require with ease. We may encrypt or sign a plaintext message msg stored in the TEXT field. We denote an encrypted message as $\mathbf{enc}_{A.pk}(msg)$, where $A.pk$ is a public key controlled by Alice, who also holds a corresponding secret key $A.sk$. Similarly, we denote a signed message as $\mathbf{sig}_{A.sk}(msg)$. If desired, a probabilistic encryption scheme can be used.

Out-of-block-chain communication between mixers is facilitated through any publishing service that meets Xim's criteria for anonymous but public messaging. For example, a lightweight service could run by each participant as a Tor hidden service, and each round moved to a new .onion address. This service would accept Xim protocol messages from anyone, and would allow anyone to read messages posted by Alice. In our protocol, we denote the *location* where Alice can receive and post messages as α_A. Similarly, the location corresponding to Bob is α_B.

4.2 One Round of Mixing

Xim operates in two phases: discovery of a mix partner, and then the fair exchange with that partner. Participants Alice and Bob mix each unit of δ coins over $n \geq 1$ rounds. During any given round, δ coins are transferred from Alice's origin address A to her destination address A' as follows, and likewise for her partner in the current round, Bob.

PROTOCOL 2: Xim(δ)

```
1: Alice:       $\alpha_B \leftarrow$ Discover()
2: Bob:         $\alpha_A \leftarrow$ Discover()
3: Alice, Bob:  FairExchange($A, A', B, B', \delta$)
```

Protocol **Discover** is responsible for pairing Alice (listening for messages at location α_A) with a randomly chosen participant, Bob, who is listening at location α_B. By design, it is difficult for an adversary to partner herself with Alice in every round.

We introduce additional details in Section 7 that allow Xim to avoid timing attacks within a round, and allow it to operate over multiple rounds. To summarize, we require participants use an arbitrary *pool* $\mathcal{P}$; all participants in a given pool synchronize the start and end of a mix round using information visible in the block chain.

4.2.1 Peer Participation and Discovery

Peer discovery is a fundamental part of a distributed mix protocol and has its own set of challenges. It's important to ensure that a participant can easily find the full set of other participants without risk of being deceived into selecting from a pool of unscrupulous ones. Second, a single attacker must do approximately the same work as an honest participant for each Sybil-based identity she creates. We leverage the global consistency of the block chain itself to achieve these ends. Protocol **Discover** details the solution. Alice randomly alternates between acting as an advertiser and a peer that responds to advertisements.

The advertising protocol is designed so that Advertiser ($\mathcal{A}$) and Respondent ($\mathcal{R}$) will each spend τ on an advertisement at its successful conclusion; we discuss how to set τ in Section 5.2. These recurring costs are a stronger deterrent against Sybil attacks than a flat, one-time fee [16]. Further, because they are charged for joining the pool of participants rather than for actual mixing, they ensure DoS attacks incur a non-zero cost. Although the ads are public, no one can prove that the two parties are linked. However, if $\mathcal{R}$ aborts before paying τ, $\mathcal{A}$ can reuse her advertisement without losing her investment. Conversely, if $\mathcal{A}$ aborts after $\mathcal{R}$ pays τ, then $\mathcal{R}$ can prove he committed to working with $\mathcal{A}$ who ultimately aborted. With his proof made public, others will ignore $\mathcal{A}$'s ad and she will have to advertise again at a loss of $\tau/2$ coin. Thus, $\mathcal{R}$ cannot cause $\mathcal{A}$ to lose any coin. And while $\mathcal{A}$ can cause some fraction of respondents to lose τ coin, it will come at a cost to him of $\tau/2$ per victim; such attacks incur a small overhead overall, as we show in Section 5.3.

- In Steps 1–5 of **Protocol 2 Discover**, a participant uniformly at random selects the role of advertiser or respondent; the alternation is random to thwart inference attacks.
- In Step 6, as advertiser $\mathcal{A}$, a participant spends $\tau/2$ on an ad, and lists a location where others can leave messages for her. Messages left for her will be encrypted with the public key of the address she advertised with. The τ coin go to the miners; since no party can pre-select the winning miner, it's clear there can be no collusion. A nonce, N_a, uniquely identifies the ad, while parameter $\mathcal{P}$ states the desired mix pool.
- In Step 7, some number of peers will each send her an encrypted message at the given location. The message contains the ad that they are responding to N_a, their own nonce N_r, and a location α_R at which they receive messages to set up the fair exchange. The three values are chosen at random for each instance of an ad or response.
- In Step 8, the advertiser selects, at random, one of the respondents $\mathcal{R}$, and she commits to his response by sending to location α_A a signed message including N_a and the hash of N_r. Other respondents will then seek other advertisers. Once $\mathcal{R}$ sees this commitment, he is encouraged to place his own response ad on the block chain in Step 9. His ad costs $\tau + f$, and the included message, encrypted with $\mathcal{A}$'s public key, guarantees to her that the costs are in response to her ad only. (The extra f is to balance costs.)
- If both parties are honest, then in Step 14, $\mathcal{A}$ will publish a response ad on the block chain costing $\tau/2$ coin that broadcasts she is pairing her ad with a partner obfuscated as $h(N_r)$. Since the partner's hashed nonce is included, $\mathcal{A}$'s response ad can satisfy exactly one respondent.
- The remaining steps serve for failure recovery. If $\mathcal{R}$'s ad (Step 9) does not appear in the block chain by a deadline, then $\mathcal{A}$ can unpair and re-use her ad at no cost (Step 11). The deadline can be a small number of blocks beyond the current block on the public chain (in Section 6 we quantify block chain delays). On the other hand, if it's $\mathcal{A}$ that does

PROTOCOL 1: Discover

```
Role Selection
 1: if $rand(0,1) > 0.5$ then
 2:     Assume role of Advertiser $\mathcal{A}$ with address $A$ and location $\alpha_A$
 3: else
 4:     Assume role of Respondent $\mathcal{R}$ with address $R$ and location $\alpha_R$
 5: end if
 6: Advertiser:  PUBLISHES $\mathbf{T}\{A \xrightarrow{0} A, \text{tip} = \tau/2,\ \text{TEXT} : (\text{loc} = \alpha_a, \text{nonce} = N_a, \text{pool} = \mathcal{P})\}$
 7: Respondent: Randomly selects advertiser; STORES "$\mathbf{enc}_{A.pk}(N_a, N_r, \alpha_R)$" to location $\alpha_A$
 8: Advertiser:  Selects respondent, STORES "$\mathbf{sig}_{A.sk}(N_a$ paired to $h(N_r))$" to location $\alpha_A$
 9: Respondent:  PUBLISHES $\mathbf{T}\{R \xrightarrow{0} R, \text{tip} = \tau + f,\ \text{TEXT} : (\text{id} = \mathbf{enc}_{A.pk}(N_a, N_r))\}$
        Failure Recovery
        10: if Respondent's transaction is not committed to block chain by time $t_1$ then
        11:     Advertiser:  STORES "$\mathbf{sig}_{A.sk}(N_a$ unpaired from $h(N_r))$" to location $\alpha_A$
        12:     goto line 7 (wait for new respondents)
        13: end if
14: Advertiser:  PUBLISHES $\mathbf{T}\{A \xrightarrow{0} A, \text{tip} = \tau/2,\ \text{TEXT: } (\text{lock} = h(N_r), N_a)\}$
        Failure Recovery
        15: if Advertiser's transaction is not committed to block chain by time $t_2$ then
        16:     Respondent:  STORES "$N_a$ aborted $h(N_r)$;  proof: $N_r$" to locations $\alpha_A$ and $\alpha_R$
        17:     goto line 7 (contact a new advertiser)
        18: end if
19: return  Address of the opposite party, $\alpha_A$ or $\alpha_R$
```

not publish a response by a deadline, then $\mathcal{R}$ can prove $\mathcal{A}$ was dishonest by storing the following values to α_A and α_R for all to see: the id of $\mathcal{A}$'s ad N_a; the pairing message $h(N_r)$; and his knowledge of the true id N_r. Any third party can encrypt N_a and N_r, and match them to $\mathcal{R}$'s ad for verification. Thus, $\mathcal{R}$ can only make this claim if and only if he was $\mathcal{A}$'s respondent in Step 7 and he actually paid for $\mathcal{A}$'s ad.

It's important that peers take on both roles of advertising and responding between rounds. If Alice acts only as an advertiser, then an attacker could always request Alice's participation and make it very likely that she and Alice are partnered on every mixing round. If all peers act only as a respondent, then the protocol is not sustainable.

4.3 Mixing Fees

There are several fees associated with the mixing process. If Alice wishes to mix δ coin, then her origin address should start with a total of $\delta + n\tau + 5nf$ coin. The first $n\tau$ coin are required to pay advertisement fee τ for each of the n rounds. Every transaction also requires the payment of mining fee f so that it is committed to the block chain in timely fashion. Participants each publish either four or five transactions during the course of a round depending on whether they act as a respondent or advertiser respectively. Thus, $5nf$ coin must be reserved for mining fees (with the respondent adding an extra f to his tip). Following this scheme, at the end of n rounds, Alice will have an unlinked address holding exactly δ coin. (For clarity, we ignore transaction fees in the **FairExchange** Protocol.)

Transaction fees (f) are small and set by the network, whereas the tips (τ) is in our control and ideally quite small. It's ideal to keep δ relatively large so that a participant with a large amount of coin to mix is not overburdened by the advertising fee. On the other hand, δ is the smallest mix unit, so it needs to be small enough to attract a maximum number of participants. Satisfying the needs of the majority is crucial to creating a thriving mix.

Different mix pools could serve different communities of interest, but the values for τ, δ, and to a very large degree n, must be consistent within each community. We discuss these parameters further in the next sections.

4.4 Splitting Large Sums of Bitcoin

One round of mixing provides unlinkability for only δ coin, which is insufficient if the mixing partner is an attacker. We refer to each collection of δ coin as a *mix unit*; Alice may need to mix her coin at address A across m *mix units.* Each unit can be processed in parallel. To unlink all of her funds, Alice's goal is to find at least one honest partner out of the n participants (one each round) for each mix unit. We analyze the security of this procedure against attackers with varying resources in Section 5.1; in short, a mixer's probability of success grows exponentially with n, but also decreases exponentially with m.

5. ATTACKS ON XIM

The key questions that we address in this section are: What are the costs and effectiveness of attacks that gather information about participants? How does the advertising fee increase the costs to attackers? How does mixing decrease their ability to infer linkability?

5.1 Attacker's Cost of Sybil-based Linking

Xim is susceptible to variations of the Sybil attack. It is possible for one entity to present as multiple identities, which can lead a victim to believe they are mixing with an inflated number of entities. The following theorem relates the resources an attacker is willing to apply to this attack to the probability of his success.

THEOREM 1: It will cost an attacker a minimum of $(\frac{\tau}{2})Y(1-(1-p)^{1/m})^{2/n}$ coin to partner with Alice for every

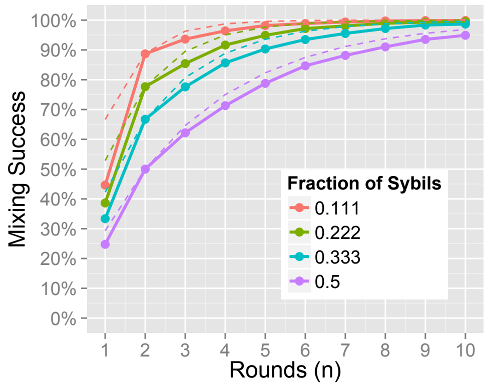

Figure 2: The probability of successful mixing one address over n rounds, given a fraction of Sybil attackers in the mix pool. The dashed lines are based on Eq. 1, while the solid lines from a simulation.

round of an n-round mix with probability of success p, given that participation costs τ per round, there are Y participants, and Alice mixes m mix units that are each worth δ coin per round.

The proof appears in a technical report [9].

As a concrete example, suppose that there are $Y = 1000$ participants and that τ is the equivalent of \$1. If Alice mixes $m = 10$ units over $n = 10$ rounds each, it will cost an attacker about \$341 to succeed in tracking at least one address with 80% probability.

The theorem demonstrates the robustness of our fee-based design: the costs to the attacker are $O(Y\tau)$, linear with the fee and the number of participants mixing. Without a per-round participation fee, the costs for attackers would not scale with the number of participants [16].

5.2 Multi-Round Mixing to Thwart Linking

Alice mixes $m\delta$ coin, separated into m mix units, each mixed over n rounds. Below, we derive the fraction of the m units she can expect to successfully mix per round in the presence of an attacker that controls a fraction of all advertisements in each round. Accordingly, she can select n to avoid such an attacker.

For each unit of δ, Alice must make sure that she has successfully avoided an attacker at least once. Alice alternates between contacting a mixer participant and being contacted. Therefore, in the best case, she has control over only half of the rounds ($\frac{n}{2}$), where she can freely choose her mix partner. It's possible that the rest of the rounds can be controlled solely by the attacker.

Suppose that the attacker represents a fraction x of the participants. Alice successfully mixes a unit if the attacker does not observe at least one of the $\frac{n}{2}$ rounds controlled by Alice per given δ. The probability that a given unit is unobserved in some round is one minus the probability that it is observed in every round, which is

$$1 - x^{n/2} \tag{1}$$

A plot of Eq. 1 appears in Fig. 2 as dashed lines for four Sybil populations: $\frac{1}{9}$, $\frac{2}{9}$, $\frac{3}{9}$, and $\frac{1}{2}$. The solid lines represent averages of a simulation of 1,000 trials of each point. (Confidence

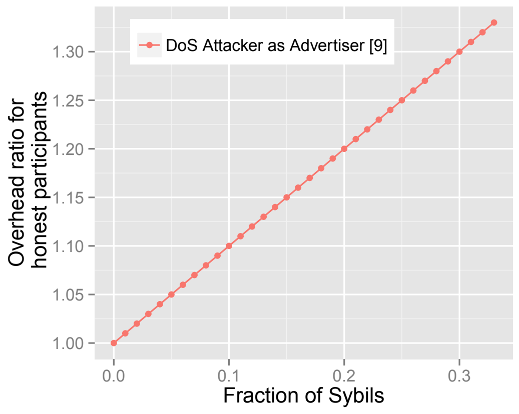

Figure 3: The overhead ratio for an honest participant from a DoS attack from a Sybil that comprises a fraction x of all participants is $1 + x$ (derivation in [9]). Overhead ratio is defined as the fees paid by an honest participant when there are Sybils over the fees paid when there are no Sybils.

intervals are not shown and negligible.) In the simulation, peers randomly select the role of advertiser or responder in each round. This strategy avoids an inference attack, but performs slightly worse than the simplified assumptions of the equation. Even with half the mix represented by a Sybil, mixing success reaches 95% after 10 rounds.

5.3 Denial of Service Attacks

Xim survives denial of service attacks well. An attacker with control of many identities can increase the fee overhead for honest participants. But, the amount of money spent by the attacker increases with the number of Sybil identities he controls, and success in pairing with honest participants is not guaranteed. The ratio of overhead participants under DoS attack must pay compared to when they are not attacked is $1 + x$ (see [9] for the derivation), which is plotted in Fig. 3. For example, when an attacker controls $x = 1/3$ of the participant pool, honest participants costs are multiplied by 4/3. In a technical report [9] we also show why the best strategy for the DoS attacker is to act as an advertiser.

5.4 Intersection Attacks

Xim is susceptible to an intersection attack, but has a small advantage over related work. Recall that we are concerned with the case where Alice exhibits unique spending behavior of her mixed coins. Imagine that an attacker is able to determine the entity controlling every input address, which could be accomplished for example by using the heuristics developed by Meiklejohn et al. [17]. He defines the *entity* E^A associated with Alice by the set of addresses she controls $E^A \equiv \{A_1, A_2, \ldots\}$, where A_i is part of the ith mix. The mixes may be distributed over a period of days (or even years). He does not know the associated output addresses $A'_1, A'_2, \ldots$ *a priori*, but can observe where these addresses are eventually spent. To launch the attack, he forms a sequence of entity sets $\mathcal{E} = \{E_1, E_2, \ldots\}$ and output address sets $\mathcal{O} = \{O_1, O_2 \ldots\}$, where E_i and O_i are the sets associated with mix i. Suppose that by analyzing the block chain, the attacker identifies a unique destination address

D that belongs to every set in the subsequence $\mathcal{O}' \in \mathcal{O}$. From the corresponding subsequence of entities $\mathcal{E}'$ he notes that $E^A \in \cap_{E \in \mathcal{E}'} E$. The attacker can now surmise that the probability that Alice is the entity making purchases from D is $1 \,/\, |\cap_{E \in \mathcal{E}'} E|$.

There is no simple defense for this attack, but note that its efficacy depends crucially on two factors. First, the entity churn rate between mixes. Churn is defined as the number of new entities joining each mix plus the number of old entities leaving. If churn is zero, then the intersection set $\cap_{E \in \mathcal{E}'} E$ will always be the same size no matter how long the attacker waits. Unfortunately, churn is entirely dependent on participant behavior, and thus it is difficult to predict what it will look like once Xim is deployed. The second factor is the number of entities participating in any given mix. Even when churn is low, the overall probability of attack success depends on the number of mixing entities. While existing mix protocols such as CoinShuffle and DarkWallet are limited to a small number of mix participants, Xim enables participants to refrain from mixing until the mix pool reaches a desired size (see Section 7). This greatly decreases our protocol's vulnerability to an inference attack relative to existing solutions.

5.5 Other Attacks

Xim is vulnerable to timing attacks. We argue in Section 7 that these attacks can be mitigated by loosely synchronizing mix participants, with the potential side benefit of mitigating other inference attacks by increasing the size of the mix pool.

Protocol **FairExchange** is susceptible to bribes because Alice and Bob exchange secrets in order to carry out an exchange. Either of them can choose to sell their secrets to a third party once they have mixed. Currently, there is no market for acquiring these secrets in the Bitcoin economy. But even if such a market emerged, the existence of a cryptographic receipt does not necessarily make bribes more commonplace. These receipts provide security to mix participants when interacting with an untrusted second party and we expect the benefits to outweigh the risks. Even without cryptographic proof, a third party still has the ability to obtain information about mix participants simply by asking them for information in return for money; although the evidence may not be cryptographic, it is still informative.

6. DELAY ANALYSIS

In this section, we analyze the delay overhead incurred by Xim if used on the Bitcoin network. We measured commit delays on Bitcoin for 21 days and characterize the delay distribution both for commits and for blocks to be settled up to 5 places back into the block chain. In sum, we observed 890,467 unique transactions posted to 2,517 unique blocks[1]. In addition to using the data to determine the delay overhead of Xim, we are also able to determine the correct setting for timelocks in the refund transactions of protocol FairExchange; we find there is a race condition if set incorrectly.

6.1 Xim Delay Overhead

Using five listeners on the Bitcoin broadcast network, we collected transaction times between April 9–29, 2014 (data available from `traces.cs.umass.edu`). We recorded each transaction's earliest broadcast time and the time when it appeared on the block chain. In the case of block chain forks, we recorded the time of the block after the fork was resolved.

[1]An average of 353 transactions per block; See `http://blockchain.info/charts/n-transactions-per-block`.

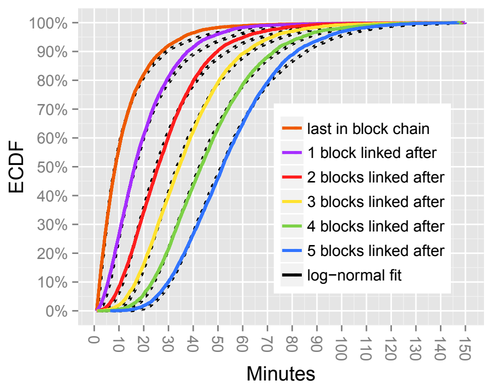

Figure 4: ECDF of the delay until transactions appear in the Bitcoin block chain. The black dotted lines fit to a log-normal distribution. This skewed distribution contrasts with the managed 10-minute average delay between block chain additions. (April 4–29, 2014.)

Fig. 4 shows the distribution of the delay between when a transaction is first broadcast and when it appears as the last (i.e., newest) block in the block chain. We also calculated the delay until the transaction appears 1–5 blocks behind the newest block. As Fig. 4 shows, these delays are highly skewed. The heavily skewed log-normal resulted in the best fit among many distributions. Bitcoin's miners are calibrated to post a new block to the block chain once every 10 minutes; but they are not calibrated to ensure that transactions are posted according to any policy. Miners do not post the largest number of transactions possible in each block for fear of losing the race to post a winning result (which requires all bytes posted and not just the resulting hash value). Accordingly, while 50% of transactions are committed to the block chain in at most 8 minutes, 25% take at least 15 minutes, and 5% take at least 35 minutes. Delays for waiting until a block is further into the chain are also log-normally distributed, seeing heavy-tail delays as well.

Let t^{90} and t^{99} be the number of minutes required to ensure that 90% and 99%, respectively, of mix transactions for a given stage have completed. For a transaction to be completed, it must be confirmed a minimum number of times (i.e., a minimum number of blocks must appear after it on the block chain). The number of required confirmations depends on the participants' risk tolerance: Fewer confirmations require less time but are vulnerable to double spending attacks.

We can use expected confirmation times to estimate the delay of Xim rounds. To keep mix participants synchronized, Section 7 specifies that each round should be delayed so that it takes $D_r(t^{90}, t^{99})$ minutes total. In the case that peers require just a single confirmation for transactions, we have $t^{90} = 27$ and $t^{99} = 64$ so that the inter-round delay is $D_r(27, 64) = 263$ minutes. When requiring two confirmations, the delay increases to $D_r(50, 88) = 426$. Most conservatively, a requirement that each transaction is confirmed six times incurs an inter-round delay of $D_r(84, 122) = 664$ minutes.

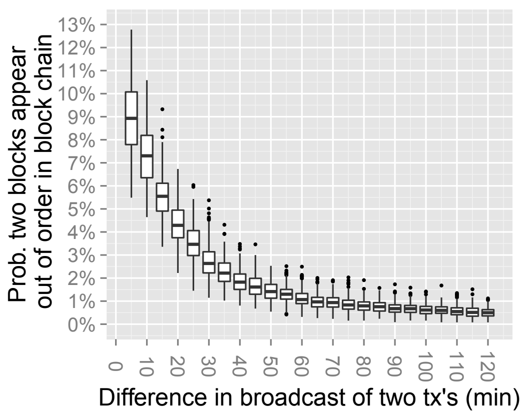

Figure 5: The probability that two transactions appear out of order on the Bitcoin block chain given the difference in times they are broadcast (the center of each box plot is the median). (April 4–29, 2014.)

While these delays of between 4.4–11.1 hours per round are certainly lengthy, note that there is no bound on δ, and that a participant can mix many addresses in parallel. And, Xim's success rate is very high after just 4 rounds: at 90% for a Sybil population of 1/3, and 99% for a population of 1/9.

6.2 Bitcoin Transaction Reordering

Our data also shows that, when configured incorrectly, Barber et al. (i.e., FairExchange) is subject to a race condition that can make it unfair, allowing one participant to both retain their coin and gain control of the other participant's coin. This race condition exists because the protocol assumes that transactions are processed in the order they are broadcast, including time locks. If the time lock on TxRefundA is not long enough, it's possible that Alice's refund appears on the block chain before Bob can claim his money. Therefore, if TxRefundA is committed on the block chain before TxClaimB, and TxClaimA is also committed, Alice would receive both her refund and Bob's money.

Fig. 5 shows the likelihood of a pair of transactions that were broadcast in a certain order to appear in reverse order on the block chain. On average there is a 7.5% likelihood of two transactions appearing out of order on the block chain if they are less than 20 minutes apart. Fortunately, the timelock value (l in [7]) can be set to start very distantly in the future without affecting the performance of honest peers. Our data indicates that the value should be at least 120 minutes to ensure a 99% likelihood of a correct ordering. The tradeoff is that aborting dishonest peers will cause honest peers to skip the current mix round and wait to recover their funds.

7. MIX COORDINATION

Xim is potentially vulnerable to timing attacks, as fair exchange transactions are distinct from the majority of typical Bitcoin transactions. In this section, we describe the timing attack, and show how loose mix coordination mitigates it.

We call two participants that pair for the purpose of mixing *mix partners*, and we say they form a *mix pair.* If Alice and Bob form a mix pair, then they pass through seven sequential *stages* each corresponding to the publication of one or more Bitcoin transactions (see Table 1). If they exchange coin at a time different from all other mixers, then they will leave a distinct signature on the block chain. This is because a majority of Bitcoin transactions look alike — the use of contracts is not prevalent — so the mix transaction will be obvious by examination of the block chain. For this reason, participants can loosely coordinate their mix times in large groups so that many participants will tend to be in the same stage simultaneously. It is this loose coordination that mitigates the timing attack.

	Part.	Action	Balance
1	Alice	Ad Solicitation	$\delta + 4f + \frac{\tau}{2}$
2	Bob	Ad Response	$\delta + 3f$
3	Alice	Ad Confirmation	$\delta + 3f$
4	Bob	TxCommB, TxRefundB	$\delta + f$
5	Alice	TxCommA, TxRefundA	$\delta + f$
6	Bob	TxClaimB	δ
7	Alice	TxClaimA	δ

Table 1: The seven sequential stages of a mix round.

Announcements. Any mix participant can post a *Mix Pool Commencement Announcement* to a public bulletin board. The announcement includes four pieces of information: the minimum number of participant pairs p, number of rounds n', a unique pool number $\mathcal{P}$, and a commencement time T. During Protocol **Discover**, any advertiser who wishes to participate in the mix will add $\mathcal{P}$ as his pool parameter. When time T has been reached, the mix proceeds for pool $\mathcal{P}$ provided that at least p advertisements list it as their desired pool. Otherwise, the mix for that pool is aborted and all associated advertisements (and responses) are nullified. Once a mix begins, participants commit to continuing for all n' rounds.

Synchronization. Throughout the mix, participants are careful to remain synchronized with the rest of the pool. Specifically, they all wish to pass each stage of each round at approximately the same time. But this is difficult to accomplish because completing a stage requires that all transactions from that stage are confirmed on the block chain. Section 6 demonstrates that two transactions published at the same time may each see confirmations at very different times. For example, 30% of the participants will see those confirmations in fewer than 40 minutes while another 30% will see them after more than 60 minutes. For this reason we introduce delays between both stages and rounds.

Each mix round unfolds over seven stages. Recall from Section 6 that t^{90} and t^{99} are the number of minutes required to ensure that 90% and 99%, respectively, of mix pairs have completed a single stage. Regardless of when their transactions happen to be confirmed, each mix pair will wait a fixed number of minutes $D_s(t^{90})$ before proceeding to the next stage and $D_r(t^{90}, t^{99})$ total minutes before proceeding to the next round. We show below that the following delays are appropriate for ensuring that half of all mix pairs remain synchronized.

$$D_s(t^{90}) = t^{90} \tag{2}$$

and

$$D_r(t^{90}, t^{99}) = 5D_s(t^{90}) + 2t^{99} \tag{3}$$

Intra-round delay. If all pairs wait a minimum of $D_s(t^{90})$ minutes between stages, then even if mix pairs never catch up once they've fallen behind, approximately half (0.9^7) of all participant pairs are expected to remain in the same stage throughout all seven stages of the round. Therefore, participants can be confident that by inspection of the block chain, their mix pair is indistinguishable from $0.5p$ other pairs. Any pairs that failed to complete a stage in time $D_s(t^{90})$ should continue to mix by proceeding immediately to the next stage. They should be aware, however, that an attacker will have an easier time linking them with their partner for that round.

Inter-round delay. At the end of a round participants again wait until a total of $D_r(t^{90}, t^{99})$ minutes have passed since the beginning of the round before proceeding to the next. This will ensure that all participants have completed seven stages with high likelihood. We choose this time for the following reason. The probability that a given mix pair completes at least 5 of 7 stages within t^{90} minutes each is $1 - \binom{6}{3}(1-0.9)^3 = 0.98$. For the stages that the mix pair does not complete in t^{90} minutes, the probability that they *do* complete within t^{99} minutes is at least $0.99^2 > 0.98$. Therefore, the probability that any given mix pair completes the seven stages in less than $5t^{90}+2t^{99} = D_r(t^{90}, t^{99})$ minutes is at least $0.98^2 > 0.96$.

Jilted mix participants. Even though mix pairs will likely finish in time to begin the next round, there is always a possibility that they will fail to complete all seven stages in time. It's also possible that one partner will go offline during the round. In such cases, jilted mix participants must withdraw from the current mix pool because the protocol requires uniform Bitcoin balances for all participants. There are, however, several opportunities for recourse. First, the participant may decide to mix no further. Second, he may choose to combine the funds in the jilted address with another address and enter a later mix. Third, he may enter a later mix that requires fewer rounds for completion and avoid recombining funds with a different address.

8. CONCLUSION

We analyzed current mixing schemes used in VCs, and shown that they are susceptible to Sybil, denial-of-service, and timing attacks. As a solution, we proposed and analyzed Xim, the first complete solution for two-party mixing that is compatible with Bitcoin and related VCs. Our protocol includes a decentralized solution for anonymously finding partners and is designed to be a multi-round system with arbitrarily increasing security. It is the first protocol to simultaneously address Sybil attackers, denial-of-service attacks, and timing-based inference attacks. Xim uses fee-based advertisements to pair partners for mixing, and provides evidence of the agreement that can be leveraged if a party aborts.

We demonstrated why the specific properties of our protocol increase economic privacy. Specifically, we showed that fees and multiple rounds make the protocol more robust against Sybil and cost-free denial-of-service attacks. Loosely coordinating the actions of participants thwarts timing attacks and linkability of addresses. Finally, using data we collected from Bitcoin, we showed that the heavy-tail delay distribution of transaction commitments impacts the efficiency of our protocol. Fortunately, because Xim can be operated in parallel and there is no restriction on the amount of coin that can be mixed per address, these delays do not increase with the amount of coin mixed.

Acknowledgements. We thank Yue Wang for collecting the Bitcoin transaction data.

9. REFERENCES

[1] Coinjoin. `http://bitcointalk.org/index.php?topic=279249.0`, May 2014.

[2] Darkwallet. `https://wiki.unsystem.net/index.php/DarkWallet/Alpha#Mixing`, May 2014.

[3] Send shared. `https://blockchain.info/wallet/send-shared`, May 2014.

[4] Sharedcoin. `https://sharedcoin.com/`, May 2014.

[5] E. Androulaki, G. Karame, M. Roeschlin, T. Scherer, and S. Capkun. Evaluating User Privacy in Bitcoin. In *Proc. Financial Crypto. & Data Security*, Apr 2013.

[6] M. Andrychowicz, S. Dziembowski, D. Malinowski, and Ł. Mazurek. Secure multiparty computations on bitcoin. In *Proc. IEEE Symp. Security & Privacy*, May 2014.

[7] S. Barber, X. Boyen, E. Shi, and E. Uzun. Bitter to Better—How to Make Bitcoin a Better Currency. In *Proc. Financial Crypto. & Data Security*, pages 399–414, Feb 2012.

[8] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza. Zerocash: Decentralized Anonymous Payments from Bitcoin. In *Proc. IEEE Symp. Security & Privacy*, pages 459–474, May 2014.

[9] G. Bissias, A. P. Ozisik, B. N. Levine, and M. Liberatore. Sybil-Resistant Mixing for Bitcoin. Technical Report UM-CS-2014-016, UMass Amherst, Sept 2014.

[10] J. Bonneau, A. Narayanan, A. Miller, J. Clark, J. Kroll, and E. Felten. Mixcoin: Anonymity for bitcoin with accountable mixes. In *Proc. Financial Crypto. & Data Security*, Mar 2014.

[11] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In *Proc. USENIX Security Symposium*, pages 303–320, Aug 2004.

[12] J. Douceur. The Sybil Attack. In *Proc. Intl Wkshp on Peer-to-Peer Systems (IPTPS)*, pages 251–260, Mar. 2002.

[13] E. Harris and N. Perlroth. For Target, the Breach Numbers Grow. *New York Times*, page B1, Jan 11, 2014.

[14] P. Koshy, D. Koshy, and P. McDaniel. An Analysis of Anonymity in Bitcoin Using P2P Network Traffic. In *Proc. Financial Crypto. & Data Security*, Mar 2014.

[15] W. Ladd. Blind signatures for bitcoin transaction anonymity. `http://wbl.github.io/bitcoinanon.pdf`, 2012.

[16] N. B. Margolin and B. Levine. Quantifying Resistance to the Sybil Attack. In *Proc. Financial Crypto. & Data Security*, pages 1–15, Jan 2008.

[17] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. Voelker, and S. Savage. A Fistful of Bitcoins: Characterizing Payments Among Men with No Names. In *Proc. ACM IMC*, pages 127–140, 2013.

[18] I. Miers, C. Garman, M. Green, and A. Rubin. Zerocoin: Anonymous distributed e-cash from bitcoin. In *Proc. IEEE Symp. Security & Privacy*, pages 397–411, May 2013.

[19] M. Möser. Anonymity of bitcoin transactions: An analysis of mixing services. *Münster Bitcoin Conference*, 2013.

[20] S. Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. `http://www.bitcoin.org/bitcoin.pdf`, 2009.

[21] H. Pagnia, H. Vogt, and F. Gaertner. Fair Exchange. *The Computer Journal*, 46(1):55–78, 2003.

[22] N. Perlroth. Target struck in the cat-and-mouse game of credit theft. *New York Times*, page B1, Dec 20, 2013.

[23] D. Ron and A. Shamir. Quantitative analysis of the full bitcoin transaction graph. In *Proc. Financial Crypto. & Data Security*, pages 6–24, Apr 2013.

[24] T. Ruffing, P. Moreno-Sanchez, and A. Kate. CoinShuffle: Practical Decentralized Coin Mixing for Bitcoin. In *Proc. ESORICS (to appear)*, Sept 2014.