Building Reliable Mix Networks with Fair Exchange

Michael K. Reiter[1], XiaoFeng Wang[2], and Matthew Wright[3]

[1] Carnegie Mellon University, reiter@cmu.edu
[2] Indiana University at Bloomington, xw7@indiana.edu
[3] University of Texas at Arlington, mwright@cse.uta.edu

Abstract. In this paper we present techniques by which each mix in a mix network can be paid for its services by message senders, in a way that ensures fairness and without sacrificing anonymity. We describe a payment mechanism for use in mix networks, and use this payment scheme in fair exchange mechanisms for both connection-based and message-based mix networks. In connection-based mix networks, our protocols achieve fairness in a weak sense: no player can benefit from stopping the exchange prematurely. In message-based mix networks, by taking advantage of each mix's next-hop neighbor as a rational third party, our exchange protocol guarantees strict fairness between initiators and mixes: either both parties successfully exchange payment and service or neither gains anything.

1 Introduction

Mixes are one of the most well-known and commonly-used privacy-preserving systems since their introduction by Chaum [9] over two decades ago. A *mix* is a server that accepts input messages, changes their form and outputs them to their destinations in a permuted order. This prevents an observer from linking inputs to the corresponding outputs. One can distribute trust in a mix system by transferring messages through a sequence of mixes (a *path*), each operated by a different entity. Whenever at least one mix in the path keeps the input/output relation secret, the privacy of the communication will be preserved. We consider both *message-based* mixes, in which each message is sent independently, and *connection-based* mixes, wherein each user maintains a path for a period of time and sends a stream of messages along that path.

The benefits of mixes depend on users trusting individual mixes. Any misbehavior, such as dropping messages or not correctly permuting messages, could jeopardize the credibility of the network of mixes, the *mix-net*, and thus drive away users. Previous research on reliability of mix networks has focused on how to *enforce* the proper operation of each mix. This has generally led to heavy-weight solutions [22, 1, 20], or solutions without a rigorous foundation that cannot provide even weak guarantees of mix server behavior [15, 14].

Most existing work assumes that individual mixes are either honest or malicious. The objective, then, is to identify these bad ones and separate them from the mix-net. However, recent research on the economics of anonymity systems [2] questions this assumption: instead of a black-and-white model, a mix server is better described as a *rational and self-interested* entity which strives to maximize its own profits. There are substantial costs involved in operating a server, keeping it running reliably, and providing bandwidth. A server operator, facing these costs and overheads, may drop connections to save bandwidth costs or perhaps simply neglect server performance as a low priority. Given sufficient economic incentives, the mixes can be *encouraged* to operate honestly. For example, a user can pay to use a mix-net, and tie the payments to good mix server performance. Such a scheme could motivate the mixes to properly relay messages, and thus increase the reliability of the mix-net.

Some incentive mechanisms [16, 18] have been proposed for anonymity protocols using digital cash. Unfortunately, they all miss a key issue: fairness in exchange of payment and service. Without proper handling of this problem, a self-interested mix may grab the payment without relaying users' messages, or a self-interested user may refuse to pay after having her messages transmitted. The rationale behind the incentive mechanism will collapse in these conditions.

Though fair exchange is a well-studied problem, a simplistic application of existing exchange mechanisms to anonymity systems might be problematic. The central concern here is how to preserve anonymity during the exchange process. In this paper, we address this problem and present the first design of a reliable mix network built upon fair exchange techniques. Specifically, we adopt the idea of *coin-ripping* for fair exchange to devise a protocol that encourages individual mixes to perform proper operations (forwarding messages, permuting the order of messages) under different scenarios, including connection-based mix-nets and message-based mix-nets.

The rest of the paper is organized as follows: Section 2 reviews related work. In Section 3, we model the fair exchange problem in mix-nets, explicate the assumptions we make in our research and describe our design of a coin-ripping protocol. Section 4 presents a fair exchange protocol for connection-based mix-nets. Section 5 describes a fair exchange protocol for message-based mix-nets. Section 6 analyzes privacy and fairness achieved by our protocols. Section 7 concludes the paper and proposes future research.

2 Related Work

This section reviews previous work related to this research. We first discuss the related concepts of reliability and *robustness* in mix networks and then discuss some of the relevant fair exchange mechanisms.

2.1 Reliability in Mix Networks

Significant attention in anonymity research has been focused on *robustness* in mix-nets. Robustness primarily refers to systems in which each mix is asked to provide a proof or strong evidence for its honest behavior [21]. Some robust mix-nets are also capable of successfully delivering messages even when $k < \ell$ out of the ℓ mixes on a user's path do not follow the protocol. Most of the proposed approaches have been built upon zero-knowledge proofs and secret sharing in re-encryption mix-nets. For example, Ogata, et al. present a robust mix-net based on cut-and-choose methods [22]. Both Abe [1] and Jakobsson and Juels [20], propose more efficient zero-knowledge proofs which achieve *universal verifiability* [1]. This property allows a third-party to verify the proof of correct behavior.

In this paper, we focus on *reliability*, a concept closely related to robustness. Reliability is the property that individual mixes behave honestly and provide service according to the mix protocol. Instead of enforcing this by requiring a proof of service or another strong form of evidence, as with the approaches for robustness, *we intend to motivate each mix to voluntarily perform correct operations*. That is, our approach does not provide robustness guarantees against malicious service providers who seek to undermine the system. We assume that the service providers who run mixes are self-interested, rather than potentially malicious, and we provide a carefully designed mechanism that pays service providers for correct performance. This ensures reliability as long as service providers prefer to be paid rather than disrupt service.

Dingledine et al. [14] propose a different (though related) concept of reliability. They implement a reputation system to record the mixes' performance. This helps users to improve their long-term odds of choosing a reliable path and avoid failing nodes [14]. This works well when mixes are run by volunteers who receive no compensation for their service, but not for self-interested service providers. For example, in a mix cascade network, their protocol has participating mixes report the failures of message transmissions, which will lead to a decrease in the reporter's reputation. If mix server operators are paid based on reputation or usage statistics, as in our model, a self-interested mix will never take such an action. This kind of reasoning is justified by research on the economics of mix-nets [2], which argues that incentives play a central role in the practical deployment of anonymity systems.

Incentive mechanisms for mix-nets have been investigated by Franz, et al. [18] and Figueiredo, et al. [16]. Both of them propose to use electronic payment to encourage mixes to behave honestly. The approach of Franz, et al., divides electronic payment and messages into small chunks and allows mixes and users to do the exchange step-by-step. This approach is very inefficient. Moreover, it also gives the party that is second to act a small, undue advantage during the exchange. Such an advantage may exceed the cost of the exchange at some moment, causing the party to stop the exchange prematurely. For example, if the mix server goes second, it may receive the last payment and then fail to deliver the last part of the message. If this is the case, the sender would not even provide the last payment. Recursively, we can infer that the transaction completely falls apart—no rational sender will provide any payment and no rational mix will send any part of the message.

Figueiredo et al., propose to wrap an electronic coin in every layer of the onion—i.e., the message constructed by the user through layered encryption, in which each mix reveals the next hop on the path by decrypting the outermost layer—thus encouraging mixes to do decryption. However, the mixes have no incentive to forward the onion after obtaining the payment, and self-interested mixes can save bandwidth by simply dropping the packet. An improvement of the scheme encrypts the payment in each layer with a secret kept in the next layer of the onion. Unfortunately, this gives users a chance to replace payments with junk data and thus avoid payment. In fact, the very anonymity that the system provides means that users can cheat without their identity being exposed. On the surface, our scheme is similar to their approach, but we provide solutions to these problems.

2.2 Fair Exchange

The essential problem in the above mentioned incentive mechanisms is the fairness in exchange of service for payment. Fairness, in the strictest sense, means that either both parties get the items they want in exchange or none of them get any thing useful. A weaker sense of fairness is that no party will benefit from cheating.

The most straightforward way of doing fair exchange is through a *trusted third party* (TTP) [13, 30]. However, this does not work in mix-nets: not only does such an online TTP constitute a trust bottleneck, but it becomes a traffic bottleneck as well. A better alternative is the use of an *offline TTP* that gets involved only when something wrong happened during the exchange process [6, 3]. The problem is that the TTP might still have access to sender/receiver relationships once it participates, which is highly undesirable in an anonymity system. Previous research shows that without a TTP, fairness only can be achieved in a weak sense [6, 8].

Jakobsson proposes *coin ripping* as an offline exchange protocol [19]. The idea is to split a digital coin, called a *rip coin*, into two halves. A customer may *rip spend* the coin by giving the vendor half of it before service is given. One she receives the service, she gives the vendor the other half. Either half by itself carries no value – in this way, it is similar to using a bill of currency (e.g., a $100 bill) that's been ripped in half. Once the vendor gets the second half of the bill, she can tape the two halves together to form the original bill. However, unlike the bill, if one half has been spent, the other half may not be used as another "first-half" payment.

This idea works well for fair exchange in mix-nets, as well will demonstrate. In mix-nets one knows the mixes she is going to use up-front and is likely to stick to the same set of mixes for a period of time [28]. In the next section, we present a novel and efficient rip coin protocol by taking advantage of this property.

2.3 Cooperation

A related concept to fair exchange is cooperation. One of our approaches requires additional cooperation between mixes.

Evolutionary game theory [5] extensively studies the mechanisms to encourage self-interested players to cooperate. In particular, much of the work in this field uses a concept called *evolutionary stable strategies* (ESS). An ESS is a convention that, upon being established in a population, becomes an individual player's best response to others, even in the presence of single deviation. A famous ESS is the tit-for-tat (TFT) strategy, which offers incentives to rational players to behave cooperatively in the *repeated prisoner's dilemma* (RPD) game.

Simply put, the prisoner's dilemma problem can be modeled as a game played between two players, in which in any particular game, an individual will prefer to defect if she knows the other player is going to cooperate. However, if the game is played repeatedly between the two players (hence *repeated* prisoner's dilemma), each will be better off if they cooperate.

An extended version of TFT is used effectively in the BitTorrent peer-to-peer protocol to encourage file uploading [12]. BitTorrent has become the leading P2P file sharing system, making up 35% of all traffic on the Internet [23], so it appears that TFT works even in dynamic environments with little trust between players.

Srinivasan, *et al.*, describe how to use generous TFT (GTFT) in ad-hoc networks to achieve the optimal throughput for each node. In GTFT, nodes sometimes cooperate with apparently uncooperative peers in order to keep temporary aberrations from halting cooperation in the long run.

In Section 5, we show how to model a key part of our protocol for message-based mix-nets as an RPD and how to use GTFT to ensure cooperation between mixes.

3 Fair Exchange in Mix Networks

This section explicates the assumptions made in this research and the fair exchange problem in mix-nets, before presenting an efficient rip coin scheme for mix-nets.

3.1 Descriptions of Assumptions and Research Problem

In this paper, we propose to use electronic payment to encourage individual mixes to operate properly. The rationale of our approach is built upon following assumptions:

1. *All players (including users and mixes) are rational and self-interested.* A rational player seeks to maximize her own profits. Here we do not consider irrational players who just want to hurt others' interests regardless of their own interests. A mix could turn into an irrational player once being tampered with by an adversary. However, since this will also compromise the profits

of the mix itself, we simply assume that the owner of the mix will take proper measures to prevent this from happening. In addition, we assume that every player in the mix-net behaves honestly when she is neutral between cooperation and defection.
2. *User payments exceed the cost of proper operation.* This assumption ensures that once a payment has been tied to the service, a rational mix's optimal strategy is to perform honestly.
3. *Adversaries have global observation.* We assume that the adversary can observe all input/output messages of every mix. Therefore, the exchange process is monitored by the adversary. We also allow adversaries to submit their own messages to the mix-net.

Under the above assumptions, the reliability problem is converted into a fair exchange problem, i.e., how to achieve the fairness in exchange of service and electronic payment anonymously. Formally, given a user U communicating with a destination D via a path composed of mixes $N_1, \ldots, N_\ell$, the fair exchange problem here is described as:

1. If all parties are honest, then U receives *service* and $N_1, \ldots, N_\ell$ receive payments.
2. If U operates honestly, then each of $N_1, \ldots, N_\ell$ gains nothing unless it honestly completes the service for U.
3. If $N_1, \ldots, N_\ell$ operate honestly, then U gains nothing unless she spends the expected amount of payment for the communication.
4. Anonymity has been preserved in the course of the exchange: N_1 knows U but not D; N_i $(1 < i < \ell)$ knows neither U nor D; N_ℓ knows D but not U; an outside observer does not know the relationship between U and D.

In (1), the concept of *service* has different meanings in different settings: in connection-based mix-net, service refers to the connection between U and D; in message-based mix-net, service means that D receives U's message and that $N_1, \ldots, N_\ell$ perform desired permutations on input messages. In (3), we do not specify whether mixes can get payment. If U can prevent mixes from getting payment after providing service (though doing so does not make U better off), we say that the fairness is *weak*. Otherwise, we say the fairness is *strict*.

In the next section, we present a design of electronic payment which helps build fair exchange protocols in mix-net.

3.2 A Rip Coin for Mix Networks

For making electronic payments, we use a form of electronic cash (ecash). Ecash has the properties of anonymity and untraceability, which mesh well with mix-nets. In an off-line ecash system, the payment process does not need to go through the bank each time, and thereby is very efficient in communications. However, such an approach faces the threat of double-spending, that is, illegal use of the same coin multiple times. Existing solutions (e.g., [19]) to this problem typically construct a digital coin in such a way that double-spending will disclose the spender's identity. However, they tend to be more computationally intensive, involving multiple discrete exponentiations.

Double-spending can be more efficiently avoided with *vendor-specific coins*, which can be deposited by only a specific set of vendors; the vendor compares the current coin with the previous ones to ensure that it hasn't been used before [25]. However, as Rivest and Shamir point out [25], the coins have limitations due to their inflexible nature: customers may not be able to predict which vendors they are going to patronize. In a mix-net, however, this is not a problem. Users of a mix-net must know ahead of time the path of mixes before constructing their onions. In our construction, the bank will not know which vendor the coin is for, so nothing is exposed by requesting vendor-specific coins. More importantly, previous research suggests that the users should stick to the same mix path to transmit their follow-up messages for a period of time, in order to prevent *passive logging attacks*, which allow an attacker to identify an initiator by logging observations over time [24, 28, 29]. This means that users should not need to interact with bank very frequently.

Since users will know their vendors in advance and maintain their paths, we can design a very efficient vendor-specific rip coin for use in mix-nets. We describe the construction of this new coin, and the related protocols (withdrawing, spending, depositing) in the following paragraphs.

Rip coins: A rip coin for mix N is a pair $c_N = (c_{N,1}, c_{N,2})$, where $c_{N,1}$ is the first half and $c_{N,2}$ is the second half. $c_{N,1}$ consists of a string x_N and the Bank B's signature $\{x_N\}_{SK_B}$. The string $x_N = N \| v_N \| r_N$ is a concatenation of the mix's identity, a *verification string* v_N, and a random string r_N. $c_{N,2}$ is a random string. It is associated with $c_{N,1}$ through a public one-way function H, such that $v_N = H(c_{N,2})$.

Withdrawing coins: To withdraw a coin, user U has to obtain the Bank's signature on the string x_N, known only to U. The Bank B's signature must not be useful in producing a signature of another legal message. The signing technology implemented here is a *blind signature* [10], which makes sure that the Bank cannot know which message was signed. For clarity of presentation, we do not present the details of the signature scheme here. Instead, we simply state that U holds a *blinding function* $f(.)$ and can unblind the signed message with another function $f^{-1}(.)$. In addition, we assume that every coin has the same publicly-known value. The process of user U withdrawing ℓ coins from Bank B, at which U has a balance, proceeds as follows:

1. User U first generates a sequence of random strings $c_{1,2}, \ldots, c_{\ell,2}$ and then computes strings $v_1, \ldots, v_\ell$ such that $v_i = H(c_{i,2})$, $1 \leq i \leq \ell$. Then, U generates another sequence of random strings $r_1, \ldots, r_\ell$ to produce $x_1, \ldots, x_\ell$ where $x_i = N_i \| v_i \| r_i$.
2. $U \longrightarrow B$: $f(x_1), \ldots, f(x_\ell)$.
3. $B \longrightarrow U$: $\{f(x_1)\}_{SK_B}, \ldots, \{f(x_\ell\}_{SK_B}$. The Bank will modify the balance of U's account correspondingly.
4. The user U removes the blinding by performing $f^{-1}(\{f(x_i)\}_{SK_B})$, $1 \leq i \leq \ell$.

Spending a coin: User U can send the coin to N_i in whole or in parts. However, N_i obtains the coin only after receiving both halves. On receiving the first half $(N\|v\|r, \alpha)$, N_i accepts it only if $N = N_i$ holds, v has never appeared before and α is B's signature on $N\|v\|r$. On receiving the second half β, N_i accepts it only if $v = H(\beta)$.
Depositing a coin:

1. $N_i \longrightarrow B$: a pair $((N\|v\|r, \alpha), \beta)$.
2. The Bank B verifies that $N = N_i$, v never appeared before, α is a correct signature on $N\|v\|r$, and $v = H(\beta)$. If so, B transfers the payment to N_i's account.

Note that in any known protocol that uses e-cash payments for anonymous communications, the bank is in a position to perform a weak intersection attack. Coins spent and coins deposited will match services used over time. However, this attack can be mitigated by users withdrawing a number of coins in advance and by the digitial cash being used for other purposes and merchants besides anonymous communications servers. Further, other forms of the intersection attack are much stronger, notably under our assumption of a global observer.

In the following sections, we will show how to use this rip coin protocol to design a fair exchange mechanism in a mix-net.

4 A Fair Exchange Protocol for Connection-based Mix-nets

In a connection-based mix-net, user U can set up a connection with D by sending an onion though a sequence of mixes $N_1, \ldots, N_\ell$. Then, each mix $N_{i=1,\cdots,\ell}$ transmits data between U and D until finally U tears down the connection. In this section, we present a weak fair exchange protocol for connection-based mix-nets, assuming that user U neglects the communication costs of transmitting the second half coins.

Fair exchange in a connection-based mix-net can take advantage of the fact that the user knows whether the connection gets through. A simple approach proceeds as follows: user U pays each mix on its path the first half of a coin in the connection-setup phase; the mixes provide connection service to U; U issues the second halves when disconnecting. This protocol is incentive-compatible. U does have the incentive to pay the first halves, otherwise her connection won't get through. The mixes have an incentive to provide service honestly, otherwise they won't get paid. After paying the first halves, U has lost all these coins, and thereby would be willing to complete the payment if the costs of distributing second halves are negligible. This is especially true since U would likely need to make a special effort, e.g. modifying the client software or physically disconnecting the computer from the network, in order to disrupt payment. We call this protocol *direct anonymous exchange*. In the following paragraphs, we present the details of the protocol, especially how to achieve anonymity in the exchange process.

Connecting:

1. An user U who plans to connect to a responder D chooses a sequence of mixes $N_1, \ldots, N_\ell$ and withdraws from the Bank a sequence of rip coins $c_1, \ldots, c_\ell$. Then U constructs an onion M_1 to wrap its connection message M. Here, we formally describe an onion M_i, $1 \leq i \leq \ell - 1$ as:
$$\begin{aligned} M_i &= \{N_{i+1}, c_{i,1}, M_{i+1}\}_{PK_i} \\ M_\ell &= \{D, c_{\ell,1}, M\}_{PK_\ell} \end{aligned} \tag{1}$$
In other words, U inserts a half coin at every layer of the onion constructed.
2. $U \longrightarrow N_1$: M_1.
3. $N_i \longrightarrow N_{i+1}$: M_{i+1}, where $1 \leq i \leq \ell - 1$. In other words, each mix peels off a layer of the onion, takes off the half coin, records the connection state information (such as connection ID) and then forwards the rest of the onion to the next hop.
4. $N_\ell \longrightarrow D$: M.
5. D and U complete the rest of connection procedure.

Communication: After establishing a connection, U and D exchange messages along the connection through $N_1, \ldots, N_\ell$.

Disconnecting:

1. After completing communication, U sends a message to D to tear down the connection. This message also signals to $N_1, \ldots, N_\ell$ that the connection is over, and final payment stage begins.
2. The user U sends a sequence of messages through the connection path (still to D) in the order $\rho_\ell, \rho_{\ell-1}, \ldots, \rho_1$, where ρ_i is the message containing half coin $c_{i,2}$.
3. Every mix N_i checks every message transmitting through the disconnecting connection: if a message ρ arrives, which includes β such that $v_i = H(\beta)$, N_i clears the connection state information and stops forwarding any message (including ρ) for this connection.

In Step 2, U distributes the second halves of the coins to mixes. U should send them out in a reversed order (from N_ℓ to N_1). This is because a rational mix will stop doing anything for the connection after receiving its payment. The central rationale of this protocol is that U would tend to neglect the overhead of a few more messages for completing the payment after accomplishing large amount of communication via the connection.

In the case that users do care about the overhead for sending second half-coin, an alternative is for the Bank to periodically force users to submit, via a DC-net [11], the second half-coins. Here we briefly sketch the idea. Users form a neighbor-relationship graph in which one is represented as a vertex and shares a long secret key with each of her neighbors. The long key is divided into many slots, each the length of a second-half coin. To send her report, a user first XORs all her secret keys together to get a string (report), from which she randomly chooses several slots according to the number of coins being used to XOR the corresponding second halves onto. After collecting reports from all users, the Bank XORs these reports together to obtain second halves.[4] This approach preserves full anonymity during the payment process, and permits the Bank to detect if a user fails to submit second half-coins.

5 A Fair Exchange Protocol for Message-based Mix-net

In this section, we describe a fair exchange protocol for message-based mix-nets, which can achieve strict fairness in exchange.

5.1 The Protocol

Previous research shows that without a trusted third party, strict fairness cannot be achieved in exchange [26]. In a message-based mix-net, the service a mix offers is to honestly process and forward a user's onion to the next hop. Therefore, we can use the next hop as a third party in the exchange of services and payment between the user and the mix. To implement this idea, we need to tackle two central issues. First, we expect a strict fair exchange. That is, either both parties get what they want or neither benefits. Therefore, the user should not give out her payment without ensuring that the mix will forward her message, and the mix will not forward the message without being assured that its successor can complete the payment for her. Second, the next hop itself is self-interested. We need an incentive mechanism to encourage it to follow the protocol.

We tackle the first issue as follows. The user U wraps inside every layer of her onion the first half of a coin, the second-half coin of the payment for the preceding hop and a certificate that proves its second-half coin is inside the next layer of the onion. This assures individual mixes that they will get the payment after forwarding the onion to the next hop. At the same time, it also assures U that her message will be forwarded. We further discuss a game theoretic strategy, which provides a strict incentive to every player to help its preceding neighbor. For this construction, we assume that the destination D is a participant in the protocol. D is willing to follow the protocol because D is interested in the message contents. We will further discuss this assumption below.

The main protocol:

[4] Using reservation and trap technology [11], the Bank can further detect and capture those who jam the communication by filling report with random bits.

1. User U, who intends to send a message M to destination D, first chooses a sequence of mixes $N_1, \ldots, N_\ell$, prepares a sequence of coins $c_1, \ldots, c_\ell$ and then constructs an onion M_1. We describe the onion wrapped inside ith layer, $1 \leq i \leq \ell - 1$, as:

$$\begin{aligned} M_i &= \{N_{i+1}, c_{i,1}, E_i, M_{i+1}\}_{PK_i} \qquad (2) \\ M_\ell &= \{D, c_{\ell,1}, E_\ell, M_D\}_{PK_\ell} \\ M_D &= \{M\}_{PK_D} \end{aligned}$$

   where E_i is an "envelope" that, intuitively, includes the second-half coin $c_{i,2}$ encrypted under PK_{i+1} in a way that N_i can verify this. How E_i is constructed is described in Section 5.2.
2. $U \longrightarrow N_1$: M_1.
3. $N_i \longrightarrow N_{i+1}$: E_i,M_{i+1}, $1 \leq i < \ell$. Before making this move, N_i verifies the Bank's signature on $c_{i,1}$ and the validity of E_i (see Section 5.2). If correct, N_i is convinced that U has completed her part of the payment and that N_{i+1} will be able to help her to obtain the rest of the coin.
4. $N_{i+1} \longrightarrow N_i$: $c_{i,2}$, after confirming the validity of E_i and M_{i+1} in a manner described in Section 5.2.
5. $N_\ell \longrightarrow D$: E_ℓ,M_D.
6. $D \longrightarrow N_\ell$: $c_{\ell,2}$.

The above protocol guarantees that no mix profits without performing step 3. Therefore, the protocol is fair in a strict sense. The problem is how to ensure all players honestly perform step 4. Here we present a mechanism that achieves this goal in a self-enforcing way.

The cooperation problem in step 4 can be modeled as a RPD game, presuming that users create paths in such a way that for mixes N and N', N immediately follows N' in a path with the same probability with which it immediately precedes N' in a path. For example, suppose users randomly choose paths of length ℓ from a total of m mixes, without replacement. Then, the probability that mix N precedes N' is $\frac{\ell-1}{m(m-1)}$. This suggests that after receiving a message from mix N', N will be in the position of sending a message to N' within expected $\frac{m(m-1)}{\ell-1}$ transmissions. As such, two neighboring mixes repeatedly interact with each other, with the same probability of acting as the next-hop mix in Step 4 of the protocol. If both honestly execute Step 4, they are better off than if they both defect. Although one may enjoy "free riding" on the other's cooperation without reciprocation, according to our assumption 2 (see Section 3.1), the free-rider will lose more if the other does the same to it in the future.

"Tit-for-tat" (TFT) has been deemed as an effective means to encourage cooperation in a RPD game. We now describe how to implement it in mix-nets.

Incentive mechanism for Step 4:

1. N begins by cooperating, always performing Step 4.

2. If N' failed to send back a correct second-half coin in the last interaction (the message M was from N to N'), then N does not send to N' its coin in this interaction. Otherwise, N honestly follows the protocol.

Essentially, the above mechanism says that each mix always follows its neighbor's behavior in the last interaction, cooperation or defection. It retaliates against the mixes deviating from the protocol, thereby removing their economic incentives to defect in future moves. On the other hand, it also shows some forgiveness: after defectors return to cooperating, other mixes will cooperate with them.

Previous research shows that TFT satisfies evolutionary stability [5]: if all mixes play this strategy at the beginning, an individual mix's optimal strategy is to follow the strategy afterwards; even if a few mixes follow other strategies (e.g, they were temporally captured by adversaries), they will be attracted back to this strategy. An interesting property of this mechanism is that messages still get through the path even when some mixes are retaliating against each other.

A similar strategy also works for the responder, D. If mix N_ℓ has not received its coin from D in the last interaction, it will refuse to deliver the message in the current interaction but resume cooperation in the future. If D values reception of messages above processing overhead of Step 6, its optimal strategy is never defecting.

A default assumption for TFT strategy is that the communication channel is reliable: No message will be lost after transmission. On an unreliable channel, mix N may falsely retaliate against N' after the message in Step 4 is lost, which will further trigger N' to retaliate against N, and thus both parties will not cooperate afterwards. A simple solution is given by modifying GTFT: let individual mixes choose retaliation with a large probability. That is, instead of retaliating for every defection, a mix may completely forgive a defector with a small probability. This guarantees that once such a misunderstanding happens due to the lossy channel, the mixes will resume cooperation eventually.

5.2 Envelope Construction

Strict fairness requires that individual mixes know that the initiator has already *completed payment* before forwarding her messages. This is achieved in the protocol described in Section 5.1 through the construction of envelope E_i. A properly constructed envelope E_i consists of (i) a ciphertext encrypted under the public key PK_{i+1} of the next mix M_{i+1} and (ii) an accompanying noninteractive zero-knowledge proof Π that the corresponding plaintext is the second half-coin $c_{i,2}$ that matches the first half-coin $c_{i,1}$ (i.e., $H(c_{i,2}) = v$ where $c_{i,1} = ((N_i \| v \| r), \alpha)$). Here we sketch one construction for E_i; others are possible, e.g., drawing from techniques for verifiable encryption of signatures (e.g., [4]).

For the encryption algorithm with which $c_{i,2}$ is encrypted under PK_{i+1}, a CCA-secure encryption is advisable since N_{i+1} acts as a decryption oracle for PK_{i+1} in the protocol of Section 5.1. Such cryptosystems were proposed by Shoup and Gennaro [27], for example, which are secure in the random oracle model assuming either the computational or decisional Diffie-Hellman assumption in a cyclic group $\mathcal{G}$. Rather than detail these encryption schemes here, we simply note that components of a ciphertext of a plaintext m using public key PK include elements of the form $\alpha = g^r$ and $\beta = m(PK)^r$ for public $g, PK \in \mathcal{G}$ (and secret r generated during encryption).

The noninteractive zero-knowledge proof Π can thus be easily constructed for certain choices of the one-way function H. For example, suppose $\mathcal{G}$ is chosen such that computing square roots is intractible in $\mathcal{G}$, i.e., for any realistic adversary A, $\Pr[x \overset{R}{\leftarrow} \mathcal{G}; z \leftarrow A(x^2) : z^2 = y]$ is negligible.[5] For such a group, an appropriate choice for $H : \mathcal{G} \rightarrow \mathcal{G}$ is $H(x) = x^2$. In this case, Π can be constructed as non-interactive zero-knowledge proof of discrete logarithm equivalence,[6] specifically that $\log_g(\alpha) = \log_{(PK)^2}(\beta^2 v^{-1})$. There exist such proofs in the random oracle model that are computationally as expensive as a digital signature, even if the order of $\mathcal{G}$ is unknown (e.g., [17, 4]).

6 Security Analysis

Here, we analyze anonymity and fairness achieved by the protocols presented in previous sections.

The exchange protocol presented in Section 4 fully preserves anonymity in connection-based mix-nets. In the protocol, the first-half coins are paid through the connection onion and the second-half coins are paid through the connection itself. Therefore, no extra information has been leaked to either mixes or external observers. This protocol also achieves fairness in a weak sense, given that users neglect the overheads of completing payment.

The exchange protocol presented in Section 5 is embedded in the original mix-net protocol. Therefore, no extra information is leaked out in either the forwarding process or the retaliation mechanism. The proposed protocol employs the successor of individual mixes as a third party in exchange of payment and service. An initiator U gives the first-half coin to the mix N_i via onion M_i, and encrypts the second-half coin in an envelope only accessible to the next hop mix N_{i+1}. On one hand, to collect the second-half coin, N_i has to honestly forward M_{i+1} (not a junk bit string) to the next hop. Otherwise, it will not get the right half coin from N_{i+1} because it does not know $\log_g(\alpha)$ and thus is unable to produce the signature on (α, β, M_{i+1}). On the other hand, N_{i+1} also cannot steal N_i's coin because it does not have access to the first half-coin. The envelope in M_i also convinces N_i that U has completed the payment. Therefore,

[5] A suitable such group $\mathcal{G}$ is the subgroup of squares in $\mathbb{Z}_n^*$, where $n = pq$ and each of p, q, $\frac{p-1}{2}$ and $\frac{q-1}{2}$ are prime. The computational and decisional Diffie-Hellman problems are also believed to be hard in this group [7]. Though the Shoup-Gennaro cryptosystems [27] are specified for a prime-order group, they can be modified trivially to work over this group.

[6] The sender also needs to take M_{i+1} as part of the inputs to generate a Fiat-Shamir random challenge for the non-interactive zero-knowledge proof. This discourages N_i from not delivering M_{i+1}.

if N_{i+1} behaves rationally, strict fairness is achieved between the initiator and the mix. We take an evolutionary stable strategy called tit-for-tat to engineer third parties' incentive. Once established in the mix-net, this strategy becomes every player's optimal strategy towards the others. In other words, acting as an honest third party becomes every player's best choice. Implementation of this strategy also exhibits a very interesting property in mix cascade networks and mix networks: even in the presence of some defecting mixes, rational mixes still deliver initiators' onions as long as the responder behaved properly in previous interactions. This is because mixes retaliate against each other by not shipping the second-half coins, while the onions will still be forwarded.

7 Conclusions

Reliability is a real problem in today's mix networks [14], and it is not likely to go away on its own. When we approach this problem, if we model nodes as either well-behaved or malicious, we end up with draconian, expensive solutions or heuristics that don't have strong properties. We believe that it is more useful to consider mixes and users as rational players who will act according to the incentives they can expect to obtain from their actions. Although we are not the first to consider a payment system in this model, we are the first to handle the crucial aspects of fair exchange in the payment process. Without fair exchange, incentives to provide service or provide payment will fail; rational agents do not enter agreements without fairness (unless they are the ones who can gain the unfair advantage).

Our constructions are not significantly more costly than existing protocols that do not have reliability, and are far more efficient than protocols with strong reliability guarantees. Furthermore, just as a business expecting income might spend money for cash registers and accountants, we expect that mix operators will be willing to do more work in a system that compensates them for the costs.

References

1. M. Abe. Universally verifiable MIX with verification work independent of the number of MIX servers. In *Proc. EUROCRYPT 1998*, 1998.
2. A. Acquisti, R. Dingledine, and P. Syverson. On the economics of anonymity. In *Proc. Financial Cryptography (FC '03)*, 2003.
3. N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. *IEEE Journal on Selected Areas in Communications*, 18(4):593–610, April 2000.
4. A. Ateniese. Efficient Verifiable Encryption (and Fair Exchange) of Digital Signatures. In *Proc. ACM Conference on Computer and Communications Security (CCS 1999)*, Nov. 1999.
5. R. Axelrod. *The Evolution of Cooperation.* Basic Books, 1984.
6. F. Bao, R. Deng, and W. Mao. Efficient and practical fair exchange protocols with off-line TTP. In *Proc. IEEE Symposium on Security and Privacy*, pages 77–85, 1998.

7. D. Boneh. The decision Diffie-Hellman problem. In *Proc. Third Algorithmic Number Theory Symposium*, volume 1423, pages 48–63, 1998.
8. L. Buttyan and J.P. Hubaux. Toward a formal model of fair exchange – a game theoretic approach. In *Proc. International Workshop on ecommerce*, 2000.
9. D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, February 1981.
10. D. Chaum. Blind Signatures for Untraceable Payments. In *Proc. Crypto'82*, pages 199–203, 1982.
11. D. Chaum. The dining cryptographers problem: Unconditional sender and receipient untraceability. *Journal of Cryptography*, 1(1):65–75, 1988.
12. B. Cohen. Incentives build robustness in BitTorrent. In *Proc. Workshop on Economics of Peer-to-Peer Systems*, May 2003.
13. R. Deng, L. Gong, A. Lazar, and W. Wang. Practical protocols for certified electronic mail. *Journal of Network and Systems Management*, 4(3):279–297, 1996.
14. R. Dingledine, M. J. Freedman, D. Hopwood, and D. Molnar. A Reputation System to Increase MIX-net Reliability. In *Proc. Financial Cryptography (FC '02). SpringerVerlag, LNCS*, 2002.
15. R. Dingledine, N. Mathewson, and P. Syverson. Reliable MIX Cascade Networks through Reputation. In *Proc. Financial Cryptography (FC '03)*, 2003.
16. D. R. Figueiredo, J. K. Shapiro, and D. Towsley. *Using Payments to Promote Cooperation in Anonymity Protocols*, 2003. Manuscript.
17. E. Fijisaki and T. Okamoto. Statistical zero-knowledge protocols to prove modular polynomial relations. In *Advances in Cryptology – CRYPTO '97 (LNCS 1294)*, pages 16–30, 1997.
18. E. Franz, A. Jerichow, and G. Wicke. A payment scheme for mixes providing anonymity. In *Proc. Trends in Distributed Systems for Electronic Commerce*, volume 1402, 1998.
19. M. Jakobsson. Ripping coins for a fair exchange. In *Proceedings of Advances in Cryptology—EUROCRYPT'95*, volume 921, pages 220–230, 1995.
20. M. Jakobsson and A. Juels. An optimally robust hybrid mix network (extended abstract). In *Proc. Principles of Distributed Computing - PODC '01*, 2001.
21. M. Jakobsson, A. Juels, and R. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In *USENIX'02*, 2002.
22. W. Ogata, K. Kurosawa, K. Sako, and K. Takatani. Fault tolerant anonymous channel. In *Proc. Information and Communications Security — First International Conference*, volume 1334, pages 440–444, 1997.
23. A. Pasick. File-sharing network thrives beneath the radar. http://in.tech.yahoo.com/041103/137/2ho4i.html, November 2004.
24. M. K. Reiter and A. D. Rubin. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, November 1998.
25. R. L. Rivest and A. Shamir. Payword and micromint — two simple micropayment schemes. *CryptoBytes*, 2(1):7–11, Spring 1996.
26. T. Sandholm and X. F. Wang. (Im)possibility of safe exchange mechanism design. In *Proc. National Conference on Artificial Intelligence*, 2002.
27. V. Shoup and R. Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. *Journal of Cryptology*, 15:75–96, 2002.
28. M. Wright, M. Adler, B. Levine, and C. Shields. An analysis of the degradation of anonymous protocols. In *Proc. ISOC Symposium on Network and Distributed System Security*, February 2002.

29. M. Wright, M. Adler, B. Levine, and C. Shields. Defending anonymous communication against passive logging attacks. In *Proc. 2003 IEEE Symposium on Security and Privacy, May 2003*, May 2003.
30. J. Zhou and D. Gollmann. A fair nonrepudiation protocol. In *Proc. IEEE Symposium of Security and Privacy*, 1996.