# Making Currency Inexpensive with iOwe

Dave Levin*, Aaron Schulman◇, Katrina LaCurts†, Neil Spring◇, Bobby Bhattacharjee◇
*HP Labs, ◇University of Maryland, College Park, †MIT CSAIL
*dml@hp.com, ◇{schulman,nspring,bobby}@cs.umd.edu, †katrina@csail.mit.edu

## Abstract

We introduce iOwe, a deferred compensation scheme that can be used in a broad range of decentralized systems. iOwe is reminiscent of a currency scheme backed by a precious commodity: network resources. iOwe does not require a central authority, proofs of work, continuous connectivity to currency issuers, or trusted storage. Instead, in iOwe, any principal may issue their own currency, in any amount, at any time. ”Currency” in iOwe, called iotas, is a promise of future work, bootstrapped by peers' trust of one another. We present several applications of iOwe, and an evaluation in the context of bootstrapping video streaming systems.

# 1 Introduction

Decentralized, peer-to-peer systems are built on the notion of peers providing resources to one another. When peers are selfish but mutually *interested* in what one another has to offer, the basic strategy is trivial: peers simply trade resources with one another [12, 27, 13]. However, peers in various distributed systems [38, 30] have varying demands across both time—Alice is interested in Bob today, Bob is interested in Alice tomorrow—and space—Alice is interested in something worth more than what Bob wants from Alice.

Digital currency[1] is a natural mechanism to equalize interest for the same reason physical currency has been useful for over 4000 years: it enables liquidity. Participants who do not simultaneously (or ever) have goods that they wish to exchange with one another can instead exchange currency. As long as a participant can then exchange this currency with someone who has a service of interest, a peer with currency is a peer of interest.

Unfortunately, digital currency has yet to experience widespread application in decentralized systems. We believe this is due to two fundamental properties of digital currency. First, digital currency is not backed by a good with intrinsic value. Historically, physical money was a precious commodity, such as a precious metal, whereas the legal tender of today is fiat money; it has value because a centralized authority deems it to be of value. Digital currency, like all fiat money, is subject to inflation. Furthermore, if the value of the digital currency is not directly linked to a good such as network or computational resources, it can be difficult to establish prices and to intuit about valuations of currency.

The second, and perhaps most important, property of digital currency schemes that make them difficult to apply in today's systems is that, much like physical currency, digital currency schemes are designed to make *owning* money the first-order privilege. So long as a participant can obtain a valid piece of currency, they can derive utility from it. A primary requirement of such approaches is therefore that it be non-trivial to obtain and particularly difficult to create money. Solutions to this problem rely on techniques that are typically not compatible with decentralized systems (§2).

We introduce iOwe, a new primitive to enable liquidity in decentralized systems. iOwe differs fundamentally from standard digital currency schemes in that the first-order privilege is not *owning* money, but rather the ability to *spend* money. We demonstrate that this difference makes iOwe applicable to a broader range of decentralized systems than standard schemes. Each iOwe user may mint an unbounded amount of their own money, called *iotas*, at any time. Iotas represent promises of future work—“This iota is good for one 500KB transfer”—and are thus backed by an intrinsic good: network or computational resources. In this sense, an iota is similar to an IOU. Also like an IOU, an iota is trivial to create, but has value only to those who are willing to accept it. iOwe relies on peers to evaluate whether or not they trust those who have owned a given iota in order to determine whether or not the iota's creator can be expected to perform the work promised in the iota.

The primary difference between iotas and IOUs is crucial: an iota does not specify to whom the promise is made, and is instead a promise to perform the work for whomever returns the iota first. An iOwe peer may therefore exchange iotas that it has received from other peers without requiring the iotas' creators in the transactions. This increases the liquidity that iOwe offers over standard, bilateral IOUs.

In the next section, we review related work in deferred compensation schemes and their application to decentralized systems. We introduce the iOwe design (§3), and demonstrate that, because it is purely decentralized, it can be applied to a wide range of systems (§4). We further demonstrate via simulation of multi-channel video streaming that iOwe quickly detects acts of misbehavior, and provides incentive to peers to maintain long-lived, weak identities (§5).

[1] By “digital currency,” we are referring to systems such as proofs of work [17] and tokens issued within a peer-to-peer system [42, 26], not, for instance, electronic transactions with a credit card.

## 2 Related Work

Digital currency schemes fall into one of two broad categories: those that seek to emulate physical currency by making it difficult to create money, and those that seek to emulate an IOU by restricting money exchange to some subset of the trust topology. In all cases, digital currency faces the challenge that it is easy to copy and, therefore, to double-spend. To be useful, currency must retain its value by limiting double-spending (or making it impossible to do so), and by ensuring that users cannot arbitrarily devalue currency by issuing too much.

The majority of prior work in providing currency-like liquidity has focused on making it difficult to create money. Indeed, work in this area is broad, with mechanisms spanning electronic cash [41, 2, 32, 18, 24], analogs to credit cards [29], privacy concerns [5, 23], trusted hardware [4, 8, 11, 14, 28], trusted intermediaries [39, 40, 21], resource-intensive proofs of work [17], and a wide array of other domain-specific problems [19, 23, 31, 10]. By making money difficult to manufacture, these systems make money easy to spend—peers are willing to accept any valid bill, regardless of who owns it—and they mitigate the attackers' ability to devalue currency by injecting a large amount of money. Unfortunately, the "difficult tasks" upon which these systems rely are typically not compatible with the goals and assumptions of decentralized systems. The above systems require either a central trusted "bank" to issue currency, trusted storage to maintain transaction histories, or proofs of work that are typically wasteful and weak against botnets. We seek a currency scheme that is itself decentralized and does not impose excessive resource costs.

There have been few proposals for currency schemes like iOwe, in which money is trivial to create but difficult to spend. Recent work by Dandekar et al. [15], who investigate a currency in which participants exchange bilateral IOUs with participants they trust, and demonstrate that this limited landscape of exchange has a relatively small impact on liquidity for several different trust topologies. In their system, peers form IOU chains, so that if peer $A$ owes $B$ who in turn owes $C$, then $C$ can redeem work from $A$ but only by going through $B$. Conversely, iOwe allows peers to exchange iotas without requiring prior owners to be online. We demonstrate in §3 the mechanisms and policies iOwe employs to achieve this property. Perhaps the most related work to iOwe is SHARP [20]. The mechanisms behind iOwe and SHARP are both based on signature chains and effectively creating an append-only log. The predominant difference between these two systems lies in the policy space. iOwe is designed to facilitate exchange between multiple resource-owners (e.g., BitTorrent peers), rather than to delegate resources among agents on the same resource (e.g., PlanetLab users on the same host). In this paper, we present a set of policies that inform a peer whether or not to accept an iota, in such a way that protects the user from acts of misbehavior. We suspect that these policies could be applied not only to iOwe and SHARP, but to other systems that require verifiable ownership chains, such as PeerReview [22].

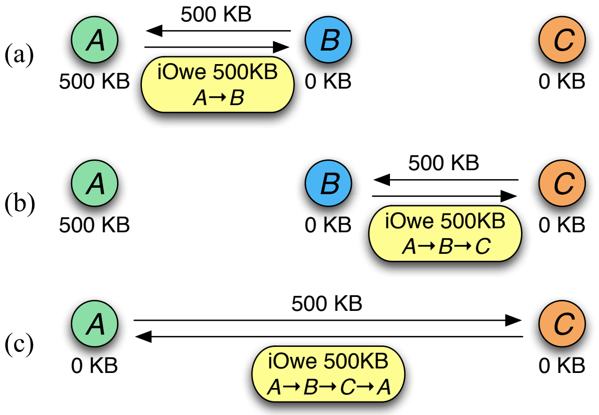

Figure 1: iOwe usage example. The value below each peer represents the amount of resources that peer has promised in iotas that have yet to be redeemed. (a) Peer $A$ issues an iota in exchange for a 500 KB transfer from $B$. (b) $B$ exchanges this iota for the same level of service from $C$. (c) Finally, $C$ redeems the iota at $A$.

## 3 iOwe Design

To be compatible with a wide range of decentralized systems, iOwe's mechanisms are designed to be lightweight in terms of computation and communication. As we will see, the mechanisms alone are enough to protect against acts of misbehavior. However, when combined with policies regarding what iotas to accept or reject, we demonstrate that iOwe is resilient to peers who double-spend, refuse to honor their promises, or fail to maintain long-lived identities. We present iOwe's mechanisms and policies in turn.

### 3.1 Mechanisms

iOwe exports three basic mechanisms: creating, spending, and redeeming iotas, as demonstrated in a high-level overview in Fig. 1. Each of these have negligible computation and communication costs, and they serve to facilitate the spending of iotas between principals by effectively implementing an append-only log.

#### 3.1.1 Creating iotas

Iotas are not proofs of completed work, but instead are *promises of future work*. Node $A$ issues an iota $\mathcal{I}$ by calling $\mathsf{issue}_A$(resource, expiry-time). $\mathsf{issue}_A$ returns an iota with the following form:

$$\mathcal{I} = \langle A, \text{resource}, \text{expiry-time}, \text{nonce } n\rangle$$

This iota states that any peer who redeems this iota at $A$, before the stated expiry time, will be provided the specified application-dependent resource. In a block-based trading system, for example, the resource could be a number of blocks that the issuer promises to provide. The nonce $n$ distinguishes iotas generated by $A$. Cryptographic operations at the time of issue are not required; nothing prevents $A$ from issuing an unlimited amount of currency. Note also that, unlike IOUs, iotas do not indicate ownership; rather, the iota represents a promise that the issuer will provide the stated work to whomever redeems the iota first.

### 3.1.2 Spending iotas

A peer *spends* an iota by signing the iota followed by the recipient's public key. $A$ (with private key $A_k$) spends iota $\mathcal{I}$ at $B$ (with public key $B_p$) by transferring $\mathcal{I}$ to $B$ (Fig. 1a). $A$ invokes $\mathsf{spend}_A(\mathcal{I}, B_p)$ to produce $\mathcal{I}_{A\to B}$ where:

$$\mathcal{I}_{A\to B} = [\mathcal{I}, B_p]_{A_k}$$

$B$ may further spend this iota at another peer $C$ without making promises of its own (Fig. 1b). $B$ invokes $\mathsf{spend}_B(\mathcal{I}_{A\to B}, C)$ to transfer the iota it received from $A$ to $C$.

These chains of signatures are crucial to iOwe's security and incentives properties, such as ensuring no peers double-spend iotas. The predominate cost of this approach is the lack of anonymity [9]; when a peer $B$ owns an iota, all peers who own that iota in the future will be able to verify that $B$ owned it, who gave it to $B$, and who $B$ gave it to.

The intended recipient may choose to reject an iota before it is transferred. For example, if $C$ had never interacted with $B$, then $A$ may not be able to accurately estimate whether the iota's issuer can be trusted to honor their promise, nor whether a prior owner of the iota had double-spent and already redeemed it, thereby eliminating any value of the iota. We discuss iota acceptance policies in §3.2.

### 3.1.3 Redeeming iotas

The final stage in an iota's lifetime is being *redeemed* at the issuer, in return for the service specified in the iota (Fig. 1c). Iota redemption is therefore simply a special case of spending an iota, with the property that the iota is returned to its issuer. As with spending, the recipient of the iota—in this case, the issuer—may refuse to accept the transfer. There are two reasons by which a peer is allowed to refuse to honor an iota he has created: if the iota has expired or if the issuer has already honored the iota's promise. To prove that he has already honored the iota, issuers store all redeemed iotas until their specified expiry time. Note that multiple attempts to redeem an iota can only occur if some peer who has owned the iota has *double-spent* it; we address this attack in §3.2.1. If an issuer refuses to perform work he has promised and is unable to provide such indemnifying proof, then that peer is said to perform a *step-omission* attack, which we address in §3.2.3.

## 3.2 Policies

iOwe's security and incentives properties derive predominately from user trust, and a set of policies defined over this trust. This is necessary—because iotas represent promises for future work, their valuation is intrinsically based on whether or not the peer issuing that promise can be trusted to honor it.

Trust establishment can, in general, be system-dependent, but takes the following basic form in iOwe. Two peers, $A$ and $B$, start without any trust at all. They then establish a history of successful *simultaneous transactions*, in which they exchange resources of mutual interest with one another, such as trading blocks in BitTorrent. This gives the peers the ability to measure one another's ability and willingness to perform work, but without giving either peer the advantage of getting significantly more resources before the other. Once $A$ and $B$ develop sufficient trust from the limited transactions, they can begin to exchange iotas for work. We call these *non-simultaneous transactions* because one peer's reward is delayed until he redeems the iota.

Although prior trust is a reasonable indication that peers will honor their iotas, it is not a guarantee. We identify three fundamental attacks that peers can launch in an attempt to gain more from the system: double-spending attacks, Sybil attacks, and step-omission attacks. In the remainder of this section, we present each of these attacks, and the policies that mitigate them, in turn.

### 3.2.1 Double-spending attacks

A peer may double-spend an iota by spending it at two (or more) peers. Because iotas, like all digital currencies, are trivial to copy, a double-spending attack is a simple attack that must be addressed in order to retain iotas' value.

The signature chains stored in iotas permit proof of double-spending. Peer $B$ double-spends the same iota $\mathcal{I}$ at $C$ and $D$ by issuing $\mathsf{spend}_B(\mathcal{I}_{A\to B}, C)$ and $\mathsf{spend}_B(\mathcal{I}_{A\to B}, D)$. This results in two signature chains that fork at $B$: $(A \to B \to C)$ and $(A \to B \to D)$. These two signature chains together constitute a proof of misbehavior (PoM) on behalf of $B$. Any peer which observes two instances of an iota with a fork in the signature chain can detect that a double-spending has occurred, and can identify the peer who committed the attack, namely, the final peer in the common suffix of the two signature chains. In the worst case, this will not occur until both iotas are redeemed at the issuer, $A$, but in our implementation, we also allow peers to compare recent iotas they have seen in order to detect double-spending more quickly. Thus, iOwe's mechanisms allow for easy detection of double-spending, but require an additional policy in order to mitigate the attack.

**Policy toward double-spenders:** When peer $A$ obtains a PoM implicating peer $B$'s double-spending, $A$ freezes $B$'s assets: $A$ never trusts $B$ again, and forwards this PoM to its trusted peers.

It is important to note that the above policy does not make double-spending *impossible*, rather it provides *strong disincentive* to do so. When peers apply the policy, the currency and reputation of a double-spender $B$ will be destroyed: peers with proof of $B$'s misbehavior will refuse to accept any iota that $B$ has created. Thus, although double-spending may provide a short-term benefit—extra currency until the double-spending is detected—the above policy outweighs this with a permanent punishment.

Each peer $A$ has incentive to inform other peers who trust $A$ of known double-spenders. Otherwise, double-spenders may flood the market with iotas, consuming peers' resources and making it difficult for an honest peer to spend iotas.

Interestingly, $A$ does not necessarily have incentive to inform peers who do *not* trust $A$. Because they would not accept an iota from $A$ anyway, $A$ gains no immediate benefit from informing them. Further, $A$ is essentially in competition with peers with whom $A$ cannot trade but with whom $A$'s trusted neighbors could. Keeping such peers uninformed may result in them being unable to take $A$'s trusted neighbors' iotas, increasing their availability for $A$.

### 3.2.2 Sybil attacks

iOwe is designed to be compatible with decentralized systems which do not provide strong identities. When applied to such systems, iOwe peers may attempt a Sybil attack [16]; that is, they may create cheap pseudonyms, or Sybils, in an attempt to gain more from the system than they would by maintaining a single identity. A pseudonym in iOwe is "cheap" if it performs no work for others, thereby earning no peer's trust.

iOwe's mechanisms limit the scope of potential attacks that Sybils permit. Because peers only accept iotas created by a trusted peer, Sybils cannot issue iotas. Also, since peers give iotas only in exchange for work, a peer gains no additional benefit from having its Sybils perform work to receive an iota rather than simply performing the work himself.

However, the mechanisms alone do not address attacks wherein a peer $p$ uses its Sybils as a relay of iotas that $p$ has either created or obtained. In particular, $p$ may spend iotas he owns at one of his Sybils, $s$, and then use $s$ to double-spend. Given the policy from §3.2.1, the double-spending attack will eventually be detected and $s$'s reputation will be compromised, at which point $p$ can simply replace that Sybil with another. As a result, $p$ will obtain the short-term benefit that double-spending attacks provide, while pinning all of the blame on a cheap pseudonym.

Such attacks are possible only if peers are willing to accept iotas held, but not created, by an untrusted peer. We address this with a policy that dictates that a peer $A$ will accept an iota only if $A$ trusts *all* peers who have ever owned the iota:

**Acceptance policy:** Peer $A$ accepts an iota $\mathcal{I}$ only if $A$ trusts the issuer and every prior owner of $\mathcal{I}$.

Combined with the policy toward double-spenders, any peer who double-spends will be unable to spend *any* iotas he owns, whether he created them or not.

### 3.2.3 Step-omission attacks

When a peer $B$ attempts to redeem an iota, its issuer $A$ may perform a *step-omission*, simply refusing to communicate with the peer. The initial, straight-forward reaction is to no longer value $A$'s currency:

**Policy toward known step-omitters:** When $B$ detects a step-omission by peer $A$, $B$ devalues $A$'s currency: $B$ no longer accepts iotas created by $A$.

Upon detecting a step-omission, $B$ has two options: attempt to spend the iota, or spread the word of $A$'s defection. An iota is not redeemed back to the issuer ($A$ is not given $\mathcal{I}_{A\rightarrow \ldots B \rightarrow A}$) until $A$ has provided the named service. Hence, if $A$ step-omits, $B$ will retain ownership of the iota, and may attempt to spend it at a peer who still trusts $A$.

Directly informing others of $A$'s silent defection is a more difficult matter. Contrary to double-spending, step-omission does not allow for nonrepudiable proofs of misbehavior. Instead, the peers who have experienced a step-omission from $A$ can choose to go public as a witness of this act. $B$ becomes a witness by "spending" the associated iota at a virtual $\mathsf{null}$ node: $\mathcal{X} = \mathsf{spend}_B(\mathcal{I}_{A\rightarrow \cdots \rightarrow B}, \mathsf{null})$. $B$ can then forward this "canceled" iota, $\mathcal{X}$, to the peers who trust $B$. In doing so, $B$ can no longer spend the iota: he has already informed its peers that it has transferred it to $\mathsf{null}$, and another transfer would be a double-spending.

Peers can interpret $B$'s canceled iota as $B$'s willingness to *pay* to lodge a complaint against $A$. After all, $B$ performed work to obtain the iota. Multiple complaints lodged against $A$ may indicate to $C$ that it, too, should devalue $A$'s currency. Each peer may have a varying parameter $\theta_C$ of how many complaints collectively indicate a step-omitter.

**Policy toward alleged step-omitters:** When $C$ obtains $\theta_C$ distinct iotas claiming defection from $A$, then $C$ devalues $A$'s currency: $C$ no longer accepts iotas created by $A$.

This policy is much less stringent than one toward the more egregious act of double-spending, because a step-omission may not necessarily be an act of misbehavior. For instance, the alleged step-omitting peer may have simply experienced a network fault. By waiting for additional evidence, $C$ can be more certain to devalue the currency only of those who consistently fail to uphold their promises.

## 3.3 Discussion

The policies discussed in this section are examples from a larger space. For example, the presented policy toward step-omitters is subject to an attack in which a rich participant can smear another by simply accruing (and subsequently burning) many of their iotas. An alternative policy would be to base decisions on the number of *peers* who file complaints, rather than the number of complaints; after all, one peer's extreme dislike of another may not be as strong a signal as a large contingent's. One could also envision using PoMs and step-omission allegations as input to a reputation system. To more quickly and reliably spread word of a step-omission failure, one could alternatively apply techniques from generalized accountability systems, such as Nysiad [25] and PeerReview [22]. Such systems attribute to each peer a set of *witnesses*; in the event of a step-omission failure, these witnesses attempt to contact their associated peer and, if they agree the peer is step-omitting, they inform others of this silent defection. Investigating other policies and their connections to existing systems is an area of future work.

# 4 Applications

iOwe's decentralized, lightweight design makes it suitable for a wide range of applications. We discuss three here, demonstrating that iOwe's necessary input—prior interaction among participants—naturally appears in many settings.

## 4.1 Bootstrapping

iOwe can assist peers in more quickly joining a system. Consider a peer-to-peer video streaming system [3, 6, 34, 33], in which peers exchange pieces of a live video stream. Efficient bootstrapping is critical in video streaming, as users wish to start watching the stream as soon as they tune in. However, bootstrapping times are notoriously poor in mesh-based systems, wherein a peer has no pieces of the stream to trade initially.

Users switching among multiple channels compounds bootstrapping, but this is the setting where iOwe thrives. As peers change channels, they build up history with others. This history serves as the necessary, a priori trust to iOwe. When user $A$ joins a channel, there may be another user $B$ in that channel with whom $A$ has interacted. $A$ may then issue (or exchange) iotas with $B$ in order to quickly fill her video buffer and start viewing the stream.

## 4.2 Seeder promotion in BitTorrent

Encouraging BitTorrent peers to stay in a swarm after completing a download is a challenging open problem. Proposed solutions exhibit parallels to that of traditional digital currency. For instance, BitTorrent communities [1] use a centralized coordinator to track peers' contributions as seeders, and BitStore [36] explicitly incorporates its own form of currency. One-hop reputations [35], on the other hand, limit IOU spreading to a single hop.

iOwe can be applied in this setting as follows. Peers who have completed downloading a file act as seeders, not by giving out pieces of the file for free, but by exchanging them for iotas. They can then exchange the iotas they received from seeding in the past for blocks from seeders in future swarms. This example demonstrates that iOwe can be applied to keep peers in a system even when they do not otherwise have a benefit to do so.

## 4.3 Payments among friends

Finally, we note that iOwe offers natural mechanisms that can be applied to a much wider set of applications than networked systems. iOwe can, for instance, facilitate transactions between friends by offering a simple and easy means of accounting for who owes what. Users may issue iotas to their friends when they borrow money or favors, and exchange these promises of repayment to others. iOwe thereby serves as a means of making concrete the social capital that exists in a nebulous form today. A natural substrate upon which to deploy iOwe in this fashion is online social networks, which opens the possibility for a rich set of policies, such as accepting an iota only if its prior owners are within no more than, say, two friend-of-friend hops.

| # owners | 1 | 2 | 3 | 4 | 5 | 6–9 |
|---|---|---|---|---|---|---|
| fraction | 0.315 | 0.160 | 0.290 | 0.190 | 0.041 | 0.002 |

Table 1: iOwe permits multi-hop exchanges.

# 5 Experimental Evaluation

We present a preliminary evaluation of iOwe in the context of bootstrapping a multi-channel video-streaming system, as discussed in §4.1. In our simulation, when a peer joins a channel, they are alloted up to 20 random *neighbors* also on the same channel. We model trust in a rough approximation of interaction as proof of work: once two peers have interacted for 60 seconds, they trust one another. Thousands of peers change among the 111 channels according to a distribution found in a recent study of TV viewer behavior [7], both in terms of which channels users switch to and how long they stay on a channel. In the simulation, we allow a single iota to be sufficient payment for bootstrapping, but find that the same results hold for varying bootstrapping costs. Our initial results show that iOwe not only allows peers to be bootstrapped quickly, but gives peers incentive to maintain weak identities and is robust against attacks such as double-spending.

**Iota lifetimes**

We begin our study by measuring how often iotas change hands. Table 1 shows the benefit iOwe adds from allowing peers to exchange others' promises; pairwise exchanges account for only 31% of all iOwe transactions. The particular values from Table 1 reflect the users' channel changing behavior from the underlying dataset. These are certain to vary from one system to another. The benefit of iOwe is its ability to adapt to various usage characteristics, as opposed to being limited to bilateral exchange or some fixed number of hops.

Prior work on a one-hop reputation system [35] focused on trading only between pairs of nodes, and seemed to demonstrate a contradictory result: that this limited scope of indirection offered sufficient liquidity. This discrepancy arises from the differences between the trust topologies in our simulation and in the one-hop reputations study. In our study, all peers change channels and trust their neighbors after 60 seconds of interaction, resulting in a mesh with high average degree, but with no set of nodes who have many more edges any other node. In the one-hop reputations study, however, there is a small core of BitTorrent users who have trust edges to virtually all other nodes. While it is possible that there are many different trust topologies that occur in deployed systems, we are skeptical of the existence of a small trusted group of BitTorrent users, as these are likely to be participating not to trade but to monitor [37].

**Weak identities are enough**

We next study whether peers have incentive to maintain their weak identities for long periods of time. Figure 2 shows the fraction of channel changes that can benefit from using iOwe (binned over 5 minute intervals). Each line represents 1000 viewers joining the system. Shortly after joining the system, more than half of a viewer's channel changes are to channels

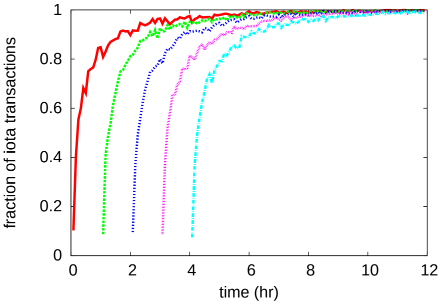

Figure 2: Viewers do not have incentive to give up their identity.

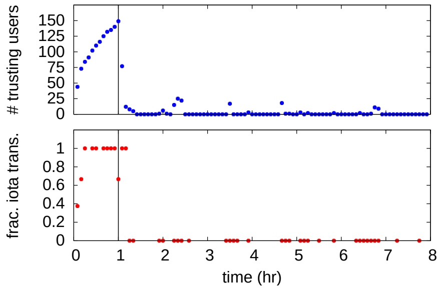

Figure 3: Viewers quickly freeze the assets of a double-spender.

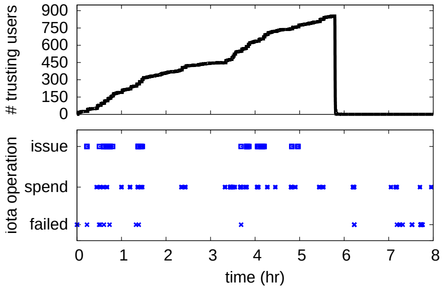

Figure 4: Viewers quickly devalue the currency of step-omitters.

at which more than 5 of its neighbors are willing to accept its iotas. At any point, were a user to shed her identity in favor of a new one, she would suffer from fewer channel-changes at which she could spend iotas. These results indicate that iOwe provides users incentive to maintain their identities.

### Culling double-spenders

Next we study how the system reacts to a *double-spending* attacker. Figure 3 shows our results run on a system consisting of 5,000 viewers. In this experiment, we simulate a peer double-spending after one hour. Although this peer double-spends only once, he quickly suffers the consequences of performing a provable act of misbehavior. Peers exchange the proof of misbehavior that the double-spending generated with the neighbors they trust. These neighbors forward the PoM to their neighbors and so on, and as peers change channels, the PoM propagates throughout the system quickly. Hours after this quick and large freezing of the double-spender's assets, the peer interacts with others who have yet to receive the PoM (predominantly because they had spent a long period of time on an unpopular channel), and thus begin to trust the peer. However, they too eventually obtain a copy of the PoM and the trust is lost. Thus, iOwe's policy toward double-spenders results in swift and long-lived disincentive to do so.

### Detecting step-omitters

Finally, we study the effect of a peer repeatedly refusing to honor the promises in the iotas he issued. Recall that, unlike double-spending, step-omission failures in iOwe do not result in a PoM, and instead peers infer misbehavior only after receiving some threshold number $\theta$ of "complaint" iotas, as described in §3.2.3. In this experiment, the threshold $\theta$ is set to 10, and the tenth step-omission occurs at around 3.7 hours. The upper plot in Fig. 4 shows the number of users that trust the attacker; in the lower plot, each point represents when the attacker changed channels, and whether he was able to issue a new iota, spend one he had received from someone else, or neither.

The first observation from Fig. 4 is that it takes longer to gather and disseminate sufficient evidence of step-omission than for double-spending. The drop-off in the number of peers begins as soon as one peer discovers that the attacker has made ten omissions, at which point information spreads quickly, as with double-spending. The second observation is that, once this attacker's misbehavior has been detected, the value of the attacker's currency is rendered worthless; the attacker is no longer able to issue his own iotas, and experiences a greater number of failed bootstrapping attempts after being caught. However, the fact that the peer did not honor his own promise of resources does not necessarily mean that the peer is a double-spender—he may, for instance, have simply experienced a spike in demand at around hour 3.7—and as such, peers continue to allow the peer to spend iotas that he did not create.

## 6 Conclusion

We have presented iOwe, a deferred compensation scheme for decentralized systems. The main insight behind iOwe is to make money *transfer* the privilege, not the money itself. This allows iOwe to avoid mechanisms such as centralized authorities, trusted hardware, and proofs of work, neither of which are compatible with decentralized systems. Further, because iOwe is based on promises for future work, the "currency" in iOwe is backed by an intrinsic good, which we expect to remove the problem of inflation.

We have demonstrated iOwe's utility via simulation in a multi-channel video streaming system. Our simulation shows that even though iOwe does not use strong identities, it provides sufficient incentive for its users to build and maintain trust with other peers. Additionally, peers that misbehave are punished quickly and thoroughly. We have also discussed other systems where iOwe is applicable, showing that it can be used as a mechanism for deferred compensation in many of today's decentralized systems.

## Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. This work was supported in part by NSF grants CNS-0643443, GRF-0645960, CNS-0917098, and IIS-0964541.

## References

[1] N. Andrade, M. Mowbray, A. Lima, G. Wagner, and M. Ripeanu. Influences on cooperation in BitTorrent communities. In *Proc. of P2PEcon*, 2005.

[2] H. V. Antwerpen. Electronic cash. Master's thesis, CWI, Neterlands, 1990.

[3] S. Banerjee, S. Lee, B. Bhattacharjee, and A. Srinivasan. Resilient Multicast Using Overlays. In *Proc. of ACM SIGMETRICS*, 2003.

[4] S. Brands. Untraceable off-line cash in wallet with observers. In *CRYPTO*, 1993.

[5] J. Camenisch, J.-M. Piveteau, and M. Stadler. An efficient electronic payment system protecting privacy. In *ESORICS*, 1994.

[6] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. SplitStream: High-Bandwidth Content Distribution in a Cooperative Environment. In *Proc. of ACM SOSP*, 2003.

[7] M. Cha, P. Rodriguez, J. Crowcroft, S. Moon, and X. Amatriain. Watching television over an ip network. In *IMC*, 2008.

[8] D. Chaum. Achieving electronic privacy. *Scientific American*, 267(2):96–101, 1992.

[9] D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *CRYPTO*, 1988.

[10] D. Chaum and T. Pedersen. Transferred cash grows in size. In *EUROCRYPT*, 1992.

[11] D. Chaum and T. Pedersen. Wallet databases with observers. In *CRYPTO*, 1992.

[12] B. Cohen. Incentives Build Robustness in BitTorrent. In *P2PEcon*, 2003.

[13] L. P. Cox and B. D. Noble. Samsara: Honor among thieves in peer-to-peer storage. In *SOSP*. ACM Press, 2003.

[14] R. Cramer and T. Pedersen. Improved privacy in wallets with observers. In *EUROCRYPT*, 1993.

[15] P. Dandekar, A. Goel, R. Govindan, and I. Post. Liquidity in credit networks: A little trust goes a long way. Tech. rep., arXiv:1007.0515, 2010.

[16] J. R. Douceur. The Sybil attack. In *IPTPS*, 2002.

[17] C. Dwork, M. Naor, and H. Wee. Pebbling and proofs of work. In *CRYPTO*, 2005.

[18] T. Eng and T. Okamoto. Single-term divisible electronic coins. In *EUROCRYPT*, 1994.

[19] N. Ferguson. Extensions of single-term coins. In *CRYPTO*, 1993.

[20] Y. Fu, J. Chase, B. Chun, S. Schwab, and A. Vahdat. Sharp: An architecture for secure resource peering. In *SOSP*, 2003.

[21] F. D. Garcia and J. henk Hoepman. Off-line karma: A decentralized currency for static peer-to-peer and grid networks. In *International Networking Conference (INC)*, 2005.

[22] A. Haeberlen, P. Kuznetsov, and P. Druschel. PeerReview: Practical accountability for distributed systems. In *SOSP*, 2007.

[23] B. Hayes. Anonymous one-time signatures and flexible untraceable electronic cash. In *AUSCRYPT*, 1990.

[24] R. Hirschfeld. Making electronic refunds safer. In *CRYPTO*, 1992.

[25] C. Ho, R. van Renesse, M. Bickford, and D. Dolev. Nysiad: Practical protocol transformation to tolerate Byzantine failures. In *NSDI*, 2008.

[26] K. Lai, L. Rasmusson, E. Adar, S. Sorkin, L. Zhang, and B. A. Huberman. Tycoon: an implemention of a distributed market-based resource allocation system. *Multiagent and Grid Systems*, 1(3):169–182, 2005.

[27] H. C. Li, A. Clement, E. L. Wong, J. Napper, I. Roy, L. Alvisi, and M. Dahlin. BAR gossip. In *OSDI*, 2006.

[28] C. Lim and P. Lee. A practical electronic cash system for smart cards. In *Korea-Japan Workshop on Information Security and Cryptography*, 1993.

[29] S. Low, N. Maxemchuk, and S. Paul. Anonymous credit cards. In *ACM Conference on Computer and Communications Security*, 1994.

[30] C. Lumezanu, D. Levin, and N. Spring. PeerWise discovery and negotiation of faster paths. In *HotNets*, 2007.

[31] G. Medvinsky and B. Neuman. Electronic currency for the internet. *Electronic Markets*, 3(9/10):23–24, 1993.

[32] T. Okamoto and K. Ohta. Universal electronic cash. In *CRYPTO*, 1991.

[33] V. Pai and A. E. Mohr. Improving robustness of peer-to-peer streaming with incentives. In *NetEcon*, 2006.

[34] V. Pai, K. Kumar, K. Tamilmani, V. Sambamurthy, and A. E. Mohr. Chainsaw: Eliminating trees from overlay multicast. In *IPTPS*, 2005.

[35] M. Piatek, T. Isdal, A. Krishnamurthy, and T. Anderson. One hop reputations for peer to peer file sharing workloads. In *NSDI*, 2008.

[36] A. Ramachandran, A. D. Sarma, and N. Feamster. BitStore: An incentive-compatible solution for blocked downloads in BitTorrent. In *NetEcon+IBC*, 2007.

[37] G. Siganos, J. M. Pujol, and P. Rodriguez. Monitoring the Bittorrent monitors: A bird's eye view. In *PAM*, 2009.

[38] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *ACM SIGCOMM*, 2001.

[39] N. Tran, J. Li, and L. Subramanian. Collusion-resilient credit-based reputations for peer-to-peer content distribution. In *NetEcon*, 2010.

[40] V. Vishnumurthy, S. Chandrakumar, and E. G. Sirer. KARMA a secure economic framework for p2p resource sharing. In *P2PEcon*, 2003.

[41] H. Youm, S. Lee, and M. Rhee. Practical protocols for electronic cash. In *Korea-Japan Workshop on Information Security and Cryptography*, 1993.

[42] S. Zhong, J. Chen, and Y. R. Yang. Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks. In *Infocom*, 2003.