Conditional E-Cash

Larry Shi[2] Bogdan Carbunar[2] Radu Sion[1]

[1] Network Security and Applied Cryptography Lab
Computer Science, Stony Brook University
(sion@cs.stonybrook.edu)
[2] Pervasive Platforms and Architectures
Motorola Labs
({larry.shi,carbunar}@motorola.com)

Abstract. We introduce a novel *conditional* e-cash protocol allowing future anonymous cashing of bank-issued e-money only upon the satisfaction of an agreed-upon public condition. Payers are able to remunerate payees for services that depend on future, yet to be determined outcomes of events. Once payment complete, any double-spending attempt by the payer will reveal its identity; no double-spending by the payee is possible. Payers can not be linked to payees or to ongoing or past transactions. The flow of cash within the system is thus both correct and anonymous. We discuss several applications of conditional e-cash including online trading of financial securities, prediction markets, and betting systems.

1 Introduction

Electronic cash (e-cash) instruments allow digital payment for goods and services. Desirable properties of such protocols include: the ability to effect anonymous payments, the detection and prevention of malicious behavior (e.g., double spending), as well as the transactional consistency of the participants' financial state. A multitude of e-cash protocols have been proposed in the recent past. The main desiderata in such efforts has often been achieving digitally, levels of similarity and ease of use comparable to physical cash.

There are scenarios however, where basic e-cash properties are not sufficient. Here we consider the case of payments conditional on unknown future outcomes. In such settings, payers require the ability to anonymously remunerate payees for items that depend on future, yet to be determined outcomes of events. Prominent examples include trading of financial market instruments such as futures and securities [7, 8, 23], and other online protocols involving deferred conditional payments such as betting.

Correctness assurances are essential. Payees need to be confident that payment *will* occur with certainty for favorable future event outcomes. Payers should be able to cash back un-cashed issued conditional payments for events with unfavorable outcomes. Overall monetary consistency needs to be preserved.

We note that trivial designs for such mechanisms can be envisioned, e.g., involving the e-cash issuing institution (i.e., bank) as a trusted arbitrator. Such assumptions, however, are rarely desirable. Requiring knowledge about the semantics of each and every considered future event at the bank is not scalable for even moderate transaction throughputs, considered events, and number of parties[3]. Moreover, an important concern in such scenarios is the privacy of participants. It is important to protect the privacy of interactions between payer and payee entities. Revealing identities should only be possible as a counter-incentive for faulty behavior (e.g., double spending) and specifically not during a correct run of the protocol.

Thus, one of the main challenges of a sound design is assuring participants' privacy while guaranteeing the conditional nature of payments. Payers and payees will naturally know each other, either by knowing each other's identity or at least by having access to a pre-authenticated channel through which to transfer public keys. No other party however should be able to associate them with each other and the conditional payments. While many existing e-cash protocols provide for participant anonymity, they cannot be directly deployed for payments of a conditional nature.

In this paper we introduce a new *conditional* e-cash protocol featuring the following properties. A payer can ask her bank to issue an anonymous payment token that can be cashed by any potential payee, *once* and if and only if a trusted publisher[4] will publish a specific secret (which only the publisher can do) in the future. In effect, payers are now able to remunerate payees (e.g., merchants) anonymously, for services that depend on future, yet to be determined outcomes of events. Once payment complete, any double-spending attempt by the payer will reveal its identity. Moreover, no double-spending by the payee is possible. Payers can not be linked to payees or to ongoing or past transactions. The flow of cash within the system is thus both correct and anonymous.

We explore a series of applications for conditional payments, including the online trading of securities, prediction markets, and online betting protocols.

The paper is organized as follows. We discuss the operational and adversarial models in Section 2. We introduce and analyze the payment protocol in Section 4 and explore several applications such as anonymous online betting in Section 5. We discuss related work in Section 3 and conclude in Section 6.

2 Model

A payer remunerates a payee by providing a payment token that can be activated and cashed at a specific bank, but only when a secret is published by a trusted publisher upon the completion of a certain agreed-upon event with a "favorable" outcome (e.g., stock price below given threshold, horse won race). Events with two possible outcomes will be considered ("favorable" – payment should be honored, and "unfavorable"). No other party but the publisher can generate the secret (under computational intractability assumptions). Without sacrificing generality, we will consider a single such event/secret combination,

[3] Additionally, arguably, very few banks would enter such an arbitration business.

[4] The publisher can be considered a "manager" of events – e.g., a stock market administrator, a race organizer.

but possibly many payees and payers exchanging conditional payments for one event. The protocol guarantees the following:

> **P1.** The bank is not able to associate previously issued conditional payments (to payers) with identities of principals (payers or payees) cashing them later.
> **P2.** Double spending by both the payers and the payees is prevented. Moreover, if a payer re-uses the payment token for a different payee, its identity is revealed to the bank.
> **P3.** The payer is able to cash back the payment token in the case of an unfavorable outcome.
> **P4.** Once the payee accepts the conditional payment from the payer, she will be able to cash it in with high probability in the case of a favorable outcome, when the publisher publishes the associated enabling secret. In this case, if the payer attempts to spend the payment token the payer's identity will be revealed to the bank (this is discussed in P2).
> **P5.** The publisher cannot infer any information about the existence of payer-payee-bank interactions solely through the protocol.
> **P6.** The bank cannot infer any event-specific details.
> **P7.** Neither the payer nor the payee should be able to prove to outside parties that they interacted in a conditional payment protocol (deniability).

2.1 Operational Model

Let A be the payer, C the payee, B the bank and T the trusted publisher. Factoring large composite numbers is hard. There exists a PKI infrastructure based on RSA. For any party X, we denote by $id(X)$ its identity, N_X its public RSA modulus, e_X its public key and d_X its private key. Network anonymizers [17] exist and can be deployed by both A and C to communicate to B. Let Mix be a notation for such an anonymizer. Whenever possible point-to-point communication will be encrypted semantically secure [5], including links passing through an anonymizer towards the bank. These will be encrypted with no forward security by using a session key generated by the anonymous party (e.g., C, when communicating with B). The meaning of all messages in the system is explicited as part of the message; we will not detail this in the protocols. The bank B manages client accounts and assists clients by generating or cashing traditional and conditional e-cash payments.

Let b denote the public "name" of the considered future event. Let t be the corresponding secret published by T in the case of a favorable (for payment) outcome. Without loss of generality we will consider b to be a large prime number, and $t = b^{-1} \ mod \ \phi(N_T)$, where $\phi()$ is Euler's totient (this is discussed further in Section 4.3). If the event's outcome is not favorable, T is trusted to immediately discard any information that could enable other parties to reconstruct

[5] With keys being generated using authenticated DH or equivalent.

t or portions thereof. We stress it is important for T to not collude with the payee to reveal the payer's identity by publishing t and allowing C to cause a payer double-spending condition. The publishing process of T could be as simple as maintaining an authenticated website. For scalability, outside of the publishing process, no interaction between T and other participants is required by the protocol.

2.2 Adversary

As discussed above we are concerned with a computationally bounded adversary. Because the message exchanges are encrypted, and the protocol only uses anonymizers when no authentication is required, we will consider here mainly the insider threat. Both the bank and the publisher should not be able to infer any additional information about ongoing or past conditional payment transactions. Specifically, without their direct cooperation, A and C should not be identifiable as conditional payment partners. Additionally, no subset of participants should be able to collude and violate any of the properties above.

2.3 Crypto Tools

For completeness, we will briefly discuss blind signature protocols.

Let a party A engage in a blind signature protocol with B (B is the signing party). At the end of a correct run of the protocol, A will be in the possession of a "well-formed" (e.g., "$10“) message signed by B, such that B does not know the message contents but is (sufficiently) confident of its "well-formed"-ness. It can be considered that B's signature semantics in fact speak only about the fact that the message is "well formed". Thus, the "blind" signature should not be interpreted to mean anything else. We now overview an instance, namely the **cut-and-choose** protocol [12–15].

Let $S_B(M)$ denote B's signature on message M. A generates n "well-formed" messages $\{M_1, \ldots, M_n\}$, such that any of them signed by B (i.e., any of $\{S_B(M_1), \ldots, S_B(M_n)\}$) would satisfy A as an end-result. A "blinds" all n messages with different blinding factors and sends them to B. A blinded message cannot be read unless the corresponding blinding factor is known. B requests $n-1$ randomly chosen blinding factors from A. It un-blinds the corresponding messages and verifies that they are "well formed". B is now convinced that with probability $1 - 1/n$, the remaining message is also well formed. By making n arbitrarily high, this confidence can also be made sufficiently high. B then signs the remaining blinded message M_j and sends it back to A, who simply un-blinds it. The blinding mechanism is designed such that a message first blinded by A, then signed by B, can be transformed into its simple signed (un-blinded) corresponding message $S_B(M_j)$ by A, knowing the blinding factor. We say that B blindly signed M_j for A.

For illustration purposes, we consider B's signature to be simple RSA exponentiation with private key d_B. The blinding mechanism of a message M can then be $M \times s^{e_B}$. The corresponding un-blinding process is simply division by the blinding factor s. We note that blind signature protocols can be run through anonymizers (with simple precautions).

3 Related Work

Prediction Markets. Prediction markets generate assets whose value is conditioned by specific events. Example markets include the Iowa Electronic Markets (IEM) [2], Intrade [1], and TradeSports [6]. IEM is an educational prediction market of University of Iowa, based on real money, where payoffs are based on real-world events such as political or economic outcomes. Intrade and TradeSports allow their members to speculate for real money on the outcome of a multitude of future events, ranging from politics to sports and pop culture.

Companies such as Hewlett-Packard, Eli Lilly, Microsoft and Google use internal prediction markets, where employees trade futures contracts on sales and profits, success of products or supplier behavior [21, 24]. The Iowa Health Prediction Market [3] attempts to forecast the future activity of a wide variety of infectious diseases and related phenomena, by using the unique and fresh knowledge of health-care workers. University of Miami released a Hurricane Futures Market in an attempt to better understand the information that people rely on when forecasting hurricanes.

Conditional payments will enable novel applications for prediction makers and companies with an interest in future outcomes of events. Prediction makers can receive rewards for accurate predictions, while allowing companies to purchase safety for important decisions.

Time release encryption. Dodis and Yum [18] introduce a novel problem called the *time capsule signature*. It allows for the construction of a signature that becomes valid at a time in the future when a trusted third party publishes a trapdoor associated with the current time. The time capsule signature allows the recipient of the signature to immediately verify its validity. Moreover, the third party has no interaction with the generator or recipient of the signature. It may seem possible to use the time capsule signature to solve the conditional payment problem. The payer could ask the bank to generate a time capsule signature on a blinded e-cash such that the capsule can be removed only if a certain event occurs. Besides the technical difficulty of the payer un-blinding the time capsule, this solution would require the bank's knowledge of the event, its publishing procedure and ultimately the identity of the publishing institution. However, for privacy reasons, the conditional payment problem requires the decoupling of the publishing institution from all other participants. In particular, the bank's operation should be oblivious of the nature of the event determining the condition.

Blake and Chan [9] propose a protocol for transferring time-encrypted messages between users. A message becomes valid only after a trusted server publishes a signed piece of information on a specific time value. Their solution requires no interaction between the trusted server and the users and also preserves the user's privacy from the server. Cathalo et al. [11] propose a more efficient solution for this problem, that also improves the user's anonymity. However, none of these schemes allows the receiver of a timed release message to verify its validity before release time, making them unsuitable for conditional e-cash transfers.

E-cash. The use of blind signatures and of the cut-and-choose protocol for providing untraceable electronic cash payments was proposed in [12–15]. Franklin and Yung [20] proposed the use of a trusted entity (trustee) that collaborates with the bank at withdrawal and deposit to provide a computation efficient online cash system. Trustees (either on-line or off-line) were proposed to provide variable degrees of anonymity for e-cash [10, 16, 19, 22]. Stadler et al. [22] introduced the notion of coin tracing and introduced several tracing mechanisms, requiring the trustee to be on-line at withdrawal. Camenisch et al. [10], Frankel et al. [19] and Davida et al. [16] proposed payer and coin tracing mechanisms using off-line trustees. In our work however, the payer and payee anonymity is essential and requires the bank to be unable to link the payer and payee even when colluding with one of them.

Simon [26] proposes a simple e-cash protocol in a network where anonymous communication is possible. The payer generates the e-cash by having the bank sign $f(x)$ where x is the payer's secret and f is a one-way function. The e-cash can be transferred by revealing x to the payee. The payee can then either cash the money with the bank or further transfer it by providing the bank with x and asking it to sign $f(y)$ for which it knows y. If the communication between the payee and the bank is anonymous, the payee remains anonymous and can transfer the money further. The bank can link the start and end points of a transfer chain, however, for long chains this information may be meaningless. Moreover, the end point of a transfer chain may repeat this protocol with itself, to artificially increase the length of the chain. Even though we also require the use of anonymizers, the solution of [26] does not provide support for conditional transfers. Even if conditional transfers would be provided, the payer could easily spend the e-cash transferred to the payee before the condition is satisfied – as the e-cash does not encode any information about the payer for anonymity reasons.

4 Conditional Anonymous Payments

The solution is composed of a set of logical sub-components: the generation of conditional payments, the validated transfer of the payments from the payer to the payee, and their spending by the payee in the case of a successful event outcome, or the cashing of the un-spent payments by the payer otherwise. All the above will also be designed to prevent double spending by both the payer and the payee. In the following we detail each of these components.

4.1 Payment Generation (PG)

Let n_1 and n_2 be security parameters. To generate the conditional payment, the payer A will contact the bank B as follows (A holds an account with B).

A generates $2n_1$ random numbers $X_1, .., X_{n_1}$ and $R_1, .., R_{n_1}$. Using a standard secret splitting algorithm [25], A constructs n_2 shares for each of the values $X_i \oplus id(A)$, for $i = 1..n_1$. We denote the j-th share corresponding to $X_i \oplus id(A)$ by $share_{ij}$ for $j = 1..n_2$. For any $X_i \oplus id(A)$, all its n_2 shares are required for its correct reconstruction.

A then constructs n_1 blocks, each of $n_2 + 1$ fields. The i-th block consists of

$$m_{iL} = [id(A), X_i, R_i, v, "left"], \quad m_{ijR} = [share_{ij}, R_i, v, "right"],$$

where v represents the value and currency of the payment (i.e. \$1). $"left"$ and $"right"$ are text messages used to differentiate between the m_{iL} value and the m_{ijR} shares.

Next, A asks the bank to blindly sign one of the n_1 message pairs using the cut-and-chose protocol discussed in Section 2.3. In this specific case however, the bank signature consists of a signature on both m_{iL}, and m_{ijR} as well as on each and every $share_ij$ in m_{ijR}. The bank will do so after verifying "well formed"-ness of $n_1 - 1$ random pairs as well as their associated shares. Specifically, the bank will verify

- that each set of n_2 shares in the $n_1 - 1$ "right" messages m_{ijR} can be used to reconstruct the corresponding $X_i \oplus id(A)$ values.
- that XOR-ing these reconstructed values $X_i \oplus id(A)$ with the second fields of m_{iL} yield indeed $id(A)$.
- that the third field of m_{iL} is equal to the second field of m_{ijR}. This value, R_i associates the messages later on.
- the correctness of the enclosed currency value (v).

If any check fails, B aborts the protocol. Otherwise, A's account is debited in an amount of v and A is able to retrieve (after un-blinding) the following payment document (signed by the bank B):

$$M_L = m_{lL}^{d_B} \; mod \; N_B, \quad M_{jR} = m_{ljR}^{d_B} \; mod \; N_B,$$

where $j = 1..n_2$ and $l \in [1, n_1]$ was randomly chosen by B.

Intuitively, M_L can be later used by A to cash any un-spent payment in the case of an un-successful event outcome (see Section 4.4), while the n_2 bank signed e-cash shares, M_{jR}, can be used by A for payments to potential payees such as C (see Section 4.3).

4.2 Preventing Double Spending (PDS)

Before we proceed with describing the actual transfer of these shares to payees, we will first discuss a simple token attribution mechanism designed as one of the tools we will use to prevent the payer from double spending. Specifically, A will be prevented from transferring the payment to more than one payee. Moreover, at the completion of this step, at most two participants, one being A, will be able to cash the payment.

To achieve this, B will issue two unique "use tokens" for each signed payment (identified so anonymously by its unique R_l value). Each of these tokens will be issued on-demand, in an online interactive protocol, through an anonymizer. Specifically, before interacting with C but after retrieving the signed payment document $\{M_L, M_{jR}\}$ from B, A will use the anonymizer Mix to send B the currency amount v and the R_l value occurring both in M_L and M_{jR}, $j = 1..n_2$.

B will respond with a fresh random token $token_L$. B will also store an association between R_l and this token $R_l : \{token_L\}$ for future reference. We call the payment "activated" once this happens. If B has already seen R_l it ignores the message.

Before transferring the actual payment document, A sends R_l and v to C. C then forwards R_l anonymously to B who proceeds as follows:

- if B does not find any record of R_l it notifies C and then simply ignores the message as the payment has not been activated yet.
- otherwise, if R_l is associated with a *single* token $token_L$, B generates a new random token $token_R$, associates it also with R_l ($R_l : \{token_L, token_R\}$), and sends it back to C (through the Mix). It is important to note that only C and B know $token_R$. C will use $token_R$ later to cash the payment upon a successful event outcome, as will be discussed later.
- if B already stores *two* tokens associated with R_l, it notifies C, who in turn then aborts the protocol, knowing that A attempts to double spend.

4.3 Conditional Transfer (CT)

The PDS protocol above allows C to assert the fact that the payment that will follow from A has been activated and has not yet been spent. In this section we discuss achieving the "conditional" properties of the protocol. We introduce here a randomized probabilistic solution.

The main idea is for A to generate a quantity that can both (i) convince C to accept this payment because it is indeed valid cash-able money signed by B, (ii) allow its cashing only if t is published by T. A uses event b and T's modulus N_T (see Section 2) to blind each $M_{jR} = m_{ljR}^{d_B} \ mod\ N_B$, $j = 1..n_2$, separately, by computing

$$S_j = M_{jR}^b \ mod\ N_T.$$

A and C then engage in a cut-and-chose protocol (see Section 2.3) through which C becomes convinced that with $1 - 1/n_2$ probability, all of the S_j values are indeed well formed and signed by B, as follows.

A sends all such S_j values to C, along with the R_l value and currency amount v. C selects a random one of them (e.g., S_u) and asks A to prove that all the remaining ones are indeed valid M_{jR} messages. To do so, A sends C all $M_{jR} = m_{ljR}^{d_B} \ mod\ N_B$ values for all $j \in [1, n_2] \setminus \{u\}$ and C can verify that indeed $S_j = M_{jR}^b \ mod\ N_T$ for these values.

At this point, C will verify the "well-formed"-ness of all revealed M_{jR} values. After removing B's signature from M_{jR}, C verifies that the fourth field of m_{ljR} equals the constant string "right" and that the second and third fields equal the R_l and v values previously sent by A for the present transaction. This verification prevents A from re-using shares from different protocol instances. C also verifies that there are no duplicates among the first fields ($share_{lj}$) of the $n_2 - 1$ m_{ljR} values recovered. As a reminder, all n_2 shares are required for the reconstruction of the corresponding $X_i \oplus id(A)$ value later on. If any of these checks fails, C aborts the protocol.

Later, for a successful event outcome, T will publish

$$t = b^{-1} \ mod \ \phi(N_T)$$

Since b is prime (see Section 2), it has an inverse mod $\phi(N_T)$. Only T can compute this inverse, knowing the factorization of N_T. Using t, C can retrieve the missing M_{uR} value as

$$M_{uR} = S_u^t \ mod \ N_T$$

By removing B's signature from M_{uR}, C yield the last unknown share, $share_{lu}$, to construct the secret $X_l \oplus id(A)$. We next discuss the payment cashing procedure.

4.4 Spending The Money (SM)

In the case of a favorable event outcome, C should be able to interact with B and get her account credited appropriately. To achieve this, we propose a three-stage protocol. In the first stage C contacts B anonymously and provides proof of credit. In the second stage, C and B engage in a blind signature protocol (see Section 2.3) in which B blindly signs an un-traceable piece of currency of equivalent value to the credit that was proven in the first stage. In the final stage, the payee C directly contacts the bank B through an authenticated channel and exchanges this piece of currency for credit to her account. For an unfavorable event outcome, to cash an un-spent payment, A proceeds identically.

We note that, technically, the three-stage anonymous protocol is apparently superfluous here for purposes of providing anonymity, as this has already been ensured by previous anonymization and the lack of any information about A's identity in the proof of credit. Nevertheless, we chose to discuss it here for ease of presentation. Its purpose will become apparent later when we discuss specific applications of conditional payments such as online betting.

We now detail the above. C uses the anonymizer Mix to send to B the message

$$token_R, M_{jR}, j = 1..n_2,$$

containing the n_2 shares recovered from A and T. Similar to C (see Section 4.3), B immediately verifies the validity of each share M_{jR}. If at least one share does not verify, B aborts the protocol. Otherwise, it uses the shares to recover $X_l \oplus id(A)$. B then verifies that $token_R$ is the *second* token associated with the R_l value contained in all M_{jR} shares. If the check fails, B aborts the protocol.

Next, B investigates potential double spending. If the M_{jR} shares have been previously spent, it simply drops the message, knowing that C double spends. If the left part of the payment, M_L has been spent (by A), B can immediately recover A's identity by computing the XOR of the first field of the corresponding m_{lL}, X_l with $X_l \oplus id(A)$.

At this point, B has proof to believe that C is entitled to a credit equal to the v value stored in the third field of M_{jR}. Now C and B can anonymously engage in a blind signature protocol in which B blindly signs an un-traceable temporary piece of uniquely identifiable currency of equivalent value to this credit.

Finally, the payee C directly contacts the bank B through an authenticated channel and exchanges this piece of currency for credit to her account. B will first verify if this currency has been already spent, credit C's account, and store the unique identifier of the currency for future double spending detection.

4.5 Analysis

In this section we informally discuss the security properties of the above protocol.

Double spending (P2). The payer could try to double spend during the PDS step by registering with the bank the same e-cash under different R_l values and transferring each value to a different payee. This is prevented during the CT step, by having the payee verify that the R_l value encoded in the e-cash matches the R_l value received during the PDS step.

Alternately, during the SM step, the payer could try to spend her e-cash (using M_L) even in the case of a favorable outcome published by T. However, once the payee performs her SM step, the payer's identity will be immediately revealed. The payer could also try to spend the e-cash she sends to the payee, before the payee has a chance to do it. For this, the payer would have to obtain the $token_R$ value associated with the unique R_l of the e-cash, shared by the payee and the bank. If the payer retrieves $token_R$ from the bank before the payee, the payee will be unable to get it and will abort the protocol.

The payee cannot double spend, since both her shares (M_{jR}) and the unique identifier generated at the end of the SM step (see Section 4.4) are recorded by the bank. The payer and the payee could try to collaborate in order to double spend e-cash without having their identities revealed. This is prevented by the fact that the e-cash generated during the PG step ensures w.h.p. $(1 - 1/n_1)$ the fact that spending both M_L and the M_{jR} shares reveals the payer's identity. Moreover, both M_L and the M_{jR} shares can only be spent once.

Guaranteed Payment or Rollback (P3,P4). During the cut-and-choose sequence of the CT step, the payee receives $n_2 - 1$ shares of its choice of the payee's e-cash. If event b occurs and the corresponding t value is published by T, the payee can recover the missing share and spend the e-cash. If event b does not occur, the payer is certain that the payee is unable to recover the e-cash. The payer can then safely cash back its payment, without fear of double spending. At this point T is trusted to never reveal the factoring of the current N_T value. We stressed before the existence of a collusion vulnerability: T can collude with the payee to reveal the payer's identity by publishing t and allowing C to cause a payer double-spending condition.

Un-linkability and deniability (P1,P5,P7). The payer obtains the payment signed by the bank, containing a R_l value that is unknown to the bank. Moreover, the payee cannot prove payment origin to other parties as no non-repudiable identification tokens are revealed in any steps outside of double spending. This prevents the bank from colluding with payees to trace payments to their payer.

The payer could collaborate with the bank and attempt to reveal the identity of the payee. To achieve this, the payer could spend her e-cash (M_L) or the payee's e-cash (the M_{jR} shares) in order to signal the bank the moment when her e-cash will be spent by the payee. However, before spending the e-cash in person, the payee performs two additional stages, both through an anonymizer (see Section 4.4). The second additional stage generates the anonymous e-cash the payee will spend then in person.

Since the publisher does not directly interact with any participants, except possibly for publishing event outcomes, property P5 is trivially satisfied.

5 Applications

In this section we briefly overview just a few of the application scenarios requiring conditional e-cash payments: financial securities, prediction markets, and anonymous online betting.

5.1 Securities Trading

A particularly relevant application scenario for conditional payments can be found in trade systems involving (atomic) securities. Securities are financial instruments that deliver future value as a function of event outcomes. A simple illustrative instance is the following contract:

> "The Smart Financial Group will pay the bearer of this certificate \$50 at the end of the current financial year, if and only if the DOW Jones will increase by 5%."

Financial institutions can now sell such securities online with full privacy and assurances of payment for their clients.

5.2 Prediction Markets

Yet another application for conditional payments is in prediction markets [1, 2, 4–6]. For example, manufacturers may use futures markets to direct investments. Additionally, a sense of confidence can be gained if conditional monetary transactions are involved. A prediction maker can express its confidence in a prediction by associating a payment to the manufacturer that is to take place if the outcome of the prediction is unfavorable. In return, the manufacturer agrees to reward the prediction maker if the outcome of the prediction is favorable.

For instance, the Smart Motors Company (SM) may propose the following trade to any willing prediction maker:

> "If crude oil is traded at under \$60 a barrel until the end of 2007, the Smart Motors Company will pay \$6. If the price goes above \$60, SM will be paid \$10. No money changes hands now."

SM and a prediction maker may sign as many of such contracts as they desire.

Manufacturers and prediction makers signing such contracts online are now able to preserve their interactions private, even from the financial institution handling the money. This is important in cases where manufacturers want to hide certain decisions from the competition and where prediction makers may posses insider information.

5.3 Online Betting

Interestingly, the conditional payment mechanisms discussed here can be deployed in the design of anonymous online betting protocols. We briefly outline how.

Without loss of generality, we will consider A as being the betting party and C the "bookie" (the party taking bets). Then, a simple online betting protocol can be constructed as a symmetrical conditional payment scenario. For example, A will provide a conditional (on a certain race outcome) \$1 to C, while C will reciprocate with \$10 conditional on the negated outcome. The race organizer T will publish different t values t_{win} and t_{lose} for a win or a loss respectively.

Even though the payments sent are conditional, either C or A may choose not to reciprocate if the other party sends its payment first. One simple (yet more costly) solution to address this issue is to break each payment into multiple smaller payments. For instance, for a 2:1 bet for \$100, A may initiate a 10 step protocol, by sending C a \$10 conditional payment. A then waits to receive a \$20 conditional payment from C before sending the next payment. While imposing a larger communication overhead, this ensures that no participant may loose more than 1/10th of the expected value. We also designed a few lower-overhead solutions (of increased exposition complexity) we will not discuss here.

Full Anonymity. The above solution provides a simple betting protocol geared towards achieving anonymity of both C and A with respect to B or T. Often however, online betting protocols would benefit from one additional property, namely full anonymity:

> **P8.** The payer and payee should not be required to know each other's identities nor should they be able to infer these identities from the betting protocol.

This is particularly important in hostile environments with concerns of collusion (of either C or A) with outside parties with incentives to reveal participation in the protocol of either the better or the bookie.

To achieve full anonymity we will require the interaction between A and C to be performed either through a special anonymous IP rendez-vous point, similar to the ones in Tor [17] or through IRC channels as follows.

C anonymously advertises its public key as well as the service it provides. C also registers its public key along with several introduction points in a lookup service (built to be censorship resilient [27]).

A finds the advertisements and then uses the lookup service to retrieve the introduction points of the bookie. It then chooses an anonymous rendez-vous point as the place where the transaction is to take place and registers its coordinates (encrypted with the public key of the bookie) on one or several of the introduction points. If the bookie decides to accept the better, it retrieves the bet anonymously from the rendez-vous point while it reciprocates with its own conditional payment or engages in a more complex multi-step *simultaneous* payment protocol as above.

A simpler idea is to use IRC channels and messages steganographed into posted media files to also achieve plausible deniability of participation claims in the case of compromised rendez-vous points.

6 Conclusions

In this paper we introduced a novel conditional payment protocol that allows future anonymous cashing of bank-issued e-money only upon the satisfaction of an agreed-upon public condition. We discussed a set of application scenarios including online trading of financial securities, prediction markets, and betting systems.

In future work we believe it is important to allow payees to further transfer their payment tokens to third parties. This is of interest for example in financial securities/options trading where securities and options are subject to multiple sell-buy cycles before maturation. It would also be interesting to pursue events with more complex, non-binary outcomes, e.g., a boolean formula of multiple variables. Additionally, we are currently working on lower overhead methods to enable conditional transfers. We have designed a few solutions based on bilinear maps that seem particularly promising.

References

1. Intrade: A trade exchange network company. `http://www.intrade.com/`.
2. Iowa electronic markets. `http://www.biz.uiowa.edu/iem/`.
3. Iowa health prediction market. `http://fluweb.biz.uiowa.edu/fluhome/index.html`.
4. Newsfutures. `http://us.newsfutures.com/home/home.html`.
5. Strategy page. `http://www.strategypage.com/prediction\_market/default.asp`.
6. Tradesports. `http://www.tradesports.com/`.
7. K. J. Arrow and G. Debreu. The existence of an equilibrium for a competitive economy. *Econometrica*, 22, 1954.
8. Y. Balasko. *Foundations of the Theory of General Equilibrium*. 1986.
9. Ian F. Blake and Aldar C. F. Chan. Scalable, server-passive, user-anonymous timed release cryptography. In *ICDCS '05: Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05)*, pages 504–513, Washington, DC, USA, 2005. IEEE Computer Society.
10. Jan Camenisch, Ueli M. Maurer, and Markus Stadler. Digital payment systems with passive anonymity-revoking trustees. In *ESORICS '96: Proceedings of the 4th European Symposium on Research in Computer Security*, pages 33–43, London, UK, 1996. Springer-Verlag.
11. Julien Cathalo, Benoît Libert, and Jean-Jacques Quisquater. Efficient and non-interactive timed-release encryption. In Sihan Qing, Wenbo Mao, Javier Lopez, and Guilin Wang, editors, *ICICS*, volume 3783 of *Lecture Notes in Computer Science*, pages 291–303. Springer, 2005.
12. David Chaum. Blind signatures for untraceable payments. In *Advances in Cryptology— Proceedings of Crypto '82*, pages 199–203. Plenum Press, 1982.
13. David Chaum. Security without identification: transaction systems to make big brother obsolete. *Communications of the ACM*, 28(10):1030–1044, 1985.

14. David Chaum. Privacy protected payments: Unconditional payer and/or payee untraceability. In *Proceedings of SmartCard 2000*, 1988.
15. David Chaum, Amos Fiat, and Moni Naor. Untraceable electronic cash. In *CRYPTO '88: Proceedings of the 8th Annual International Cryptology Conference on Advances in Cryptology*, pages 319–327, London, UK, 1990. Springer-Verlag.
16. George I. Davida, Yair Frankel, Yiannis Tsiounis, and Moti Yung. Anonymity control in e-cash systems. In *FC '97: Proceedings of the First International Conference on Financial Cryptography*, pages 1–16, London, UK, 1997. Springer-Verlag.
17. Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium*, 2004.
18. Yevgeniy Dodis and Dae Hyun Yum. Time capsule signature. In Andrew S. Patrick and Moti Yung, editors, *Financial Cryptography*, volume 3570 of *Lecture Notes in Computer Science*, pages 57–71. Springer, 2005.
19. Yair Frankel, Yiannis Tsiounis, and Moti Yung. "indirect discourse proof": Achieving efficient fair off-line e-cash. In *ASIACRYPT '96: Proceedings of the International Conference on the Theory and Applications of Cryptology and Information Security*, pages 286–300, London, UK, 1996. Springer-Verlag.
20. Matthew K. Franklin and Moti Yung. Secure and efficient off-line digital money (extended abstract). In *ICALP '93: Proceedings of the 20th International Colloquium on Automata, Languages and Programming*, pages 265–276, London, UK, 1993. Springer-Verlag.
21. Barbara Kiviat. The end of management? Time - Inside Business: `http://www.time.com/time/insidebiz/printout/0,8816,1101040712-660965,00.html`.
22. Stadler M., Piveteau J.-M, and Camenisch J. Fair blind signatures. In *Proceedings of EUROCRYPT*, pages 209–219, 1995.
23. Paul A. Samuelson. *Foundations of Economic Analysis*. Harvard University Press, 1947.
24. Bill Saporito. Place your bets! Time: `http://www.time.com/time/magazine/article/0,9171,1118373,00.html`.
25. B. Schneier. *Applied Cryptography: Protocols, Algorithms and Source Code in C*. Wiley & Sons, 1996.
26. Daniel R. Simon. Anonymous communication and anonymous cash. In *CRYPTO '96: Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology*, pages 61–73, London, UK, 1996. Springer-Verlag.
27. Marc Waldman, Aviel D. Rubin, and Lorrie Faith Cranor. Publius: A robust, tamper-evident, censorship-resistant, web publishing system. In *Proc. 9th USENIX Security Symposium*, pages 59–72, August 2000.