Privacy Preserving Market Schemes for Mobile Sensing

Yuan Zhang, Yunlong Mao, He Zhang and Sheng Zhong

***Abstract*—To put mobile sensing into large-scale deployments, we have to take care of sensing participants' incentives and privacy first. In this paper, we study how to protect the sensing participants' privacy in the mobile sensing market where multiple sensing jobs reside in one consolidated place. Our problem is highly challenging due to the facts that incentives are introduced and we consider both the sensing job owner and the market administrator could invade the sensing participants' privacy. We propose two privacy-preserving market mechanisms that are able to protect the sensing participants' privacy to solve our problem. Experiments also demonstrate that our mechanisms have good efficiency.**

***Keywords*—mobile sensing; privacy; linkage attack**

I. Introduction

As one successful application of the crowdsoucing idea, the mobile sensing has been a topic of increasing interest. In a general mobile sensing scenario, a sensing job owner outsources its job to a crowd of mobile device users (e.g. smartphone users), and these users complete the sensing task using the sensors equipped on their devices and send back their sensing data to the job owner. Due to the pervasiveness of smartphone users, the mobile sensing has a giant pool of potential participants beyond comparison. Furthermore, nowaday smartphones often have a rich set of sensors such as the microphone, camera, accelerometer, and GPS equipped which make their users excellent sensing job executors. Due to these two major advantages, in recent years, a variety of mobile sensing applications have been developed and utilized in many areas, such as health care [1], [2], intelligent transportation [3], [4], environmental monitoring [5], [6] and etc.

Since most existing mobile sensing applications (and its other crowdsourcing application siblings) do not have a common supporting infrastructure and rely on a small number of volunteers to contribute data, it is difficult to put them into large-scale deployments. To change this situation, people start to build incentive-driven crowdsourcing markets or platforms (e.g. the Amazon Mechanical Turk [7], the Clickworker [8], the Helpdesk [9] and the Domywork [10]) that allow multiple jobs to be crowdsourced with payments in one consolidated place. For example, the Amazon Mechanical Turk [7] provides application programming interfaces (APIs) for those enterprises or individuals who need to outsource their jobs to find proper on-demand workers around the world, collect the results of their work, and pay for their labor.

Despite facilitating interactions between job owners and workers, sensing participants' privacy is a concern especially in *incentive-driven mobile sensing markets* where mobile sensing applications reside and incentives are introduced to attract workers' participation. Consider a research organization that uses the market to collect data from HIV patients' daily physical status as an example. First, the data needs to be processed with sufficient caution since it directly reveals a patient's health status. In addition, the identity of sensing participants who accept the job also needs to be protected. Otherwise, knowing that a person participates in this job directly reveals he or she has HIV. One may suggest to remove the sensing participant's identity information and thus completely anonymize his or her data. Even if we are able to carry out this "solution" and terminate all privacy threats, no sensing participants would be able to get any payment since no one knows who submitted the anonymized data.

In this paper, we aim to protect the sensing participants' privacy in incentive-driven mobile sensing markets. In fact, our problem is complicated since 1) we need to protect the privacy (which means to suppress identity information) in an incentive-driven system (where in general the identity information is required) and 2) the market system considered by us involves three kinds of parties (the job owners, the sensing participants and the market administrator) and both the job owners and the market administrator could be malicious adversaries. Most previous works on privacy protections in mobile sensing does not consider the incentive, thus cannot solve our problem. The recent works [11], [12] by Li and Cao firstly study the privacy protection problem in mobile sensing applications with incentives introduced. However, their works focus on a single mobile sensing application which involves two kinds of parties only the job owner and the workers. Their solutions do not apply in mobile sensing markets.

To solve our problem, we propose two privacy-preserving market schemes. The first scheme $PPMS_{dec}$ protects the sensing participants' privacy in sensing markets that allow arbitrary payments. We construct $PPMS_{dec}$ by adapting the Divisible E-cash scheme that was recently developed in the cryptography area. The second mechanism $PPMS_{pbs}$ is specially designed for sensing markets in which all jobs' payments per sensing participant are the same. We build $PPMS_{pbs}$ by constructing light-weight "digital coin" using the partial blind signature.

Our main contributions include:

- To the best of our knowledge, we are the first to study the privacy protection problem in incentive-driven mobile

Y. Zhang, Y. Mao, H. Zhang and S. Zhong are with the Computer Science and Technology Department, Nanjing University, Nanjing, 210023, China, and with the State Key Laboratory for Novel Software Technology at Nanjing University, Nanjing , 210023, China.
E-mails: zhangyuan05@gmail.com, njucsmyl@163.com, mattzhang9@gmail.com and sheng.zhong@gmail.com.

This work is supported by NSFC-61321491, NSFC-61425024, NSFC-61300235 and NSFC-61402223.

sensing markets. We identity the possible privacy attacks in the market systems and provide efficient solutions.

- We propose $PPMS_{dec}$ a privacy-preserving market mechanism that can protect the sensing participants' privacy in sensing markets that allow arbitrary payments. $PPMS_{dec}$ protects the *data-linkage privacy*, the *job linkage privacy* and the *transaction linkage privacy* of the sensing participants against both the job owner and the market administrator. (Please refer to Section III-B for detailed explanations of these privacy notations) As byproducts, $PPMS_{dec}$ also protects the identity of the job owner.
- We propose $PPMS_{pbs}$, a light-weighted privacy-preserving market mechanism that protects the sensing participants' privacy in sensing market where all jobs' payment are unitary. $PPMS_{pbs}$ protects the *data-linkage privacy* and the *job linkage privacy* of the sensing participants against both the job owner and the market administrator. In addition, it protects the *transaction-linkage privacy* against the job owner, while revealing sensing participants' transaction information to the market administrator (the bank). We note that removing the transaction privacy against the bank is actually required in many practical systems to thwart money laundering.
- We perform experiments to evaluate the efficiency of our mechanisms. Experiment results show both mechanisms have very good efficiency.

The rest of paper is organized as follows. Section II presents the related works and Section III introduces our system model, trust assumptions, privacy models, and the cryptographic primitives that we use. Sections IV and V presents our two mechanisms. Section VI evaluates the efficiency of our mechanisms. Section VII concludes this paper.

II. Related Works

Many schemes have been proposed for incentives or user privacy in mobile sensing, but only few of them have addressed these two problems simultaneously. Privacy problem has been focused earlier, many scheme [13]–[15] have tried to solve this problem from different aspects. For example, the scheme in [13] uses group signature to encrypt data report, and then aggregate them after verifying the correctness, so that sensing data can be collected from anonymous users. In [14] authors use a probabilistic algorithm, generating dummy messages to avoid monitoring and mixing real messages within dummy messages. And they choose exponential distribution to control dummy message generation. The case of reputation system of mobile sensing has been studied in [15], authors propose a scheme, combining blind signature with periodic generated pseudonyms. These schemes have their own advantage, but the incentive is not in their consideration. On the other hand, incentive has been studied in many recent work [16]–[18]. In mobile social networks, an incentive scheme based on heterogeneous belief values has been studied in [16] for a real-time sensing scenario. In [17], authors design incentive mechanisms for users in participatory sensing, which is based on a reverse auction. Authors then propose a mechanism to optimally solve this problem. And in [18], three online incentive mechanisms based on online reverse auction are proposed. All of them take users' nature, that of opportunistically occurring in an area of interest, into consideration. But, all of these incentive schemes do not achieve the goal of privacy preservation. The privacy-aware incentive scheme is a burning desire.

Recently, this kind of privacy preserving incentive scheme has drawn many researchers' attention. In [19], authors propose a privacy protection system in participatory sensing with incentives. However, their security system using symmetric key algorithm and SHA-1 is weak, and their system can not prevent dishonest users to abuse the system to earn credits. In [11], authors propose two incentive mobile sensing schemes, both of them are privacy preserving one with a trusted third party, while the other one is not. Their threat model only focuses on a curious service provider and one malicious user, while other roles in the market are not included.

Meanwhile, some researchers also give surveys [20], [21] on this topic exploring how to address all aspects of a privacy preserving incentive system, which is a multifaceted problem. Many factors are included in their work both user privacy threat and incentive threat are considered. As a result, they posit open issues to be addressed. Considerate as they are, however, the privacy issue of service provider(or job provider) is still out of plan.

III. Preliminaries

A. System Overview

In a typical mobile sensing market, there are three different types of parties: the *job owners* (JO), the *sensing participants* (SP) and the *market administrator* (MA). Each JO has a sensing job which requires mobile users to perform sensing and collect data for it. The SPs are a group of mobile phone users who are interested in trading their work for JO's payments. Both JOs and SPs have at least one kind of network channel (e.g. WIFI, WLAN, 3G/4G and etc.) that connects them to the MA. Through MA, a JO and an SP communicate with each other. For the ease of presentation, we refer to a JO or a SP as a *market resident* when we do not want to differentiate them in the rest of this paper.

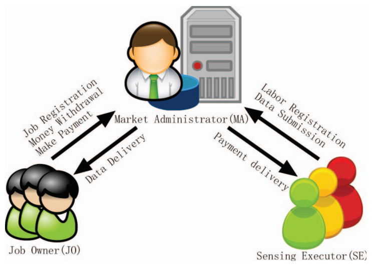

Fig. 1: System Model

The virtual currency/money is introduced to provide incentives in the market. The virtual currency is issued by a virtual bank that is also maintained by MA. To encourage SPs' participation, the JO pays them with virtual currency. The currency can be used to buy sensing services from other SPs, or converted to real-world rewards or even money. To earn or spend credits, a market resident has to open an *account* in the virtual bank. As with most real-world banks, here we consider the virtual bank also requires the resident to submit authentic identity information to open an account. In addition, we assume each resident is required to have at most one account only. Clearly, a resident's account ID (denoted by AID) is equivalent to the resident's real identity and needs to be protected in general.

Below we list the basic interactions that happen between the market resident and the MA in the market:

- *Job Registration*: The JO submits a job profile to the MA, and then the MA publishes the profile in the market. In general, the job profile consists of the job description, the amount of payment for each SP, and the job owner's identity information. MA publishes the profile using a bulletin board that can be accessed by all market residents.
- *Labor Registration:* The SP who decides to participate submits its identity information to the MA, and then the MA sends the information to the JO.
- *Data Submission/Delivery:* The SP who completes the sensing job submits its data as well as its identity information to the MA. Later, when the SP agrees, the MA delivers the data to the JO.
- *Payment Submission/Delivery:* The JO submits its payment and the payee's identity information to the MA, and later the MA delivers this payment to the corresponding SP.
- *Money Withdrawal/Deposit:* The JO withdraws a certain amount of virtual money from its own account in the bank. After receiving their payments, SPs deposit the money into their own accounts.

We note that for privacy considerations, a resident could provide a fake identity or just a random pseudonym as his or her identity information in the interactions above. Nevertheless, we point out that there are a few interactions that do require authentic identity information e.g. the money withdrawal and the money deposit.

Fig. 1 gives an overview about the system model of an incentive-driven mobile sensing market.

Finally, a *market mechanism* consists of a series of instructions or rules that specifies how the above interactions should be done.

B. Trust model and privacy model

1) Trust model: We consider a similar trust model as the one in [11]. In particular, we assume each market resident (JO or SP) and the MA have a pair of public key and private key that can be used to authenticate each other. The key pair can be generated by an offline certificate authority for example. By this assumption, we focus our attentions on the inside attacks. The attack could come from the market users or some external parties that have a market user compromised. In addition, we also assume that the communications between each JO/SE and the MA are anonymized on the networking level using IP/MAC recycling and/or Mix Networks.

2) Privacy model: In this paper, we aim to protect the privacy of every SP. In particular, we mainly consider three kinds of privacy issues.

- First, if any adversary can identify the data contributed by an SP, i.e. find the linkage between a piece of data and an SP's authentic identity, this is clearly unacceptable. We refer to this kind of privacy as the *data-linkage privacy*.
- Second, if any party can tell whether or not an SP participates in a job, i.e. find the linkage between a job and the authentic identity of the SP who participates in it, this could raise privacy problems as we have shown in the HIV research organization's example in the introduction section. We refer to this kind of privacy as the *job-linkage privacy*.
- Third, if any party can tell if a JO and a SP have ever made a money transaction, this may cause privacy problems in some cases. For example, if the adversary happens to know the only job that this JO outsources, the adversary then knows the SP's who have made a money transaction with the JO that participated in that job. We refer to this kind of privacy as the *transaction-linkage privacy*. We note that in some practical systems, the transaction-linkage privacy against the bank is deliberately removed to thwart money laundering.

C. Cryptographic Primitives

1) Divisible E-cash scheme: In our mechanism, we use a binary tree based Divisible E-cash scheme (DEC) [22]–[25] for coins in currency. This e-cash scheme has two distinct features, denomination of coins is the exponent of 2, and coins can be divided except the unitary ones.

Each coin can be described in the form of a binary tree. The coin with value of $w = 2^L$ is equivalent to a binary tree of $L{+}1$ levels. Every node of the tree is assigned to a denomination. The root node is with value w, and each of its child nodes is half the value of parent node, i.e.$w/2$. If a coin is unitary, then it is the only node in the entire tree.

In this paper, we denote the level of binary tree L. A node, denoted $N_{i,j}$ of a tree is in the i-th level of the tree, and $j \in \{0, 1\}$ indicates this node is the left child or the right child. Especially, the root node with value 2^{L-1} is N_0.

Some special cyclic groups which the DEC scheme is based on need to be generated in a $Setup(DEC)$ process. The first is $\mathcal{G}$ of order $o_{\mathcal{G}}$, and another cyclic group $\mathcal{G}_1 = \langle g_1 \rangle$, which is a subgroup of $\mathbb{Z}^*_{o_{\mathcal{G}}}$. And $\mathcal{G}_i = \langle g_i \rangle$, where $2 \le i \le L+1$, is a subgroup of $\mathbb{Z}^*_{o_{i+1}}$, where o_{i+1} is the order of $\mathcal{G}_{i+1}$. In order to guarantee subgroup's requirement, we choose $o_{i+1} = 2o_i + 1$. Besides, the MA needs to choose some generators randomly from those groups and whose discrete logarithms to their bases are unknown, respectively. When a user requests a coin with denomination 2^L, he will receive a coin $E(2^L)$ with bank's signature.

2) Blind Signature: Each coin withdrawn from bank should be signed, generally this signature should be unique. If a JO withdraws a coin with an unique signature from MA's bank, then deliver this coin to MA again to publish a job, MA can easily locate this JO's identity by linking original withdrawal account. So, we use blind signature scheme [26], [27] to obstruct MA's sight.

Just like an envelop, the blind signature can blind signer's sight from the content, but can still obtain a signature on it. In this paper, we use the signature scheme proposed in [27], denoted $CLSign$. With constructed public key pk_{CLS} and private key sk_{CLS}, the input of $CLSign$ is generally message m and a random r, and the output is a signature $\sigma = CLSign(m, r)$, which contains original message. Anyone can check whether this signature is valid by $Verify(\sigma, pk_{CLS})$.

3) Zero Knowledge Proof: To guarantee the validity of cash, JO needs prove its computation and signature to MA, preventing exploration at the same time. In this case, zero knowledge proof is the best choice.

IV. A PRIVACY-PRESERVING MECHANISM FOR MARKETS THAT ALLOW ARBITRARY PAYMENTS

In this section, we design a privacy-preserving mechanism that can protect the SP's data-linkage privacy/job-linkage privacy/transaction-linkage privacy, and the JO's job-linkage privacy/transaction-linkage privacy. To achieve our goal, we make use of the Divisible E-cash (DEC) primitive that is recently proposed in cryptography area. Although DEC provides perfect anonymity for the cash spender and the cash receiver, we cannot directly apply it in our problem. The main reason is that DEC presumes the bank cannot see the transaction between cash spender and the cash receiver. However in our problem, the bank is controlled by the MA who is directly involved in JO and SPs' transactions, DEC immediately fails. To deal with this issue, we propose cash breaking algorithms and adapt the Divisible E-cash scheme to make it fit in our problem. For the ease of presentation, we call our privacy-preserving mechanism $PPMS_{dec}$. Below, we explain our mechanism step by step.

A. Mechanism Design of $PPMS_{dec}$

1) Setup: The MA runs $Setup(DEC)$ to generate the cyclic groups that are required by the DEC. It also generates a random pair of Camenisch-Lysyanskaya signature's public key and private key: $(clpk_{MA}, clsk_{MA})$, and publish its public key as well as the public parameters of the DEC to all market residents.

Every JO generates a random pair of Camenisch-Lysyanskaya signature's public key and private key: $(clpk_{JO}, clsk_{JO})$, and binds $clpk_{JO}$ to its account by sending it to MA.

2) Job Registration: To register a job in the market, the JO first generates a random RSA key pair (rpk_{jo}, rsk_{jo}), and sends to the MA a job profile that specifies the job description jd, the amount of money it offers for each SP w, and rpk_{jo} as its identity information:

$$JO \rightarrow MA : jd, w, rpk_{jo} \tag{1}$$

After receiving the job profile, MA publishes it in the bulletin board BB so that all market residents could see the job:

$$MA \rightarrow BB : jd, w, rpk_{jo} \tag{2}$$

3) Money Withdrawal: JO runs the anonymous $DEC\_WITHDRAW$ protocol using its Camenisch-Lysyanskaya signature public key $clpk_{JO}$ with the MA, and gets a divisible e-cash with a denomination of 2^L:

$$MA \rightarrow JO : E(2^L) \tag{3}$$

4) Cash Break: JO breaks the denomination w into unitary denominations of 1s to thwart MA's denomination attack (Please read Sec IV-B for detailed explanations of this attack). In addition, to prevent an adversary from knowing the total value of all unitary "coins" by checking the length of all coins, JO generates $2^L - w$ fake coins with the same size of these coins. $E(0)$ denotes the fake e-cash. JO generates $E(0)$ by generating a random number whose bit-length equals the bit-length of $E(1)$. When an SP receives all 2^L coins, it can immediately identify all fake ones since they cannot pass the verification.

$$JO : E(2^L) \rightarrow \underbrace{E(1), \ldots, E(1)}_{w}, \underbrace{E(0), \ldots, E(0)}_{2^L - w}. \tag{4}$$

When JO wants to pay an SP, it sends all 2^L coins, including the fake ones and the good ones.

5) Labor Registration: To sign for a job, an SP randomly generates an RSA key pair (rpk_{sp}, rsk_{sp}) and sends the public key rpk_{sp} as its identity information to the MA.

$$SP \rightarrow MA : rpk_{sp} \tag{5}$$

After receiving the public key, MA forwards it to the JO.

$$MA \rightarrow JO : rpk_{sp} \tag{6}$$

6) Payment Submission: To pay an SP with money w, the JO first generates an RSA signature with its own private key rsk_{jo} on the SP's public key received from the MA in the labor registration phase:

$$sig = RSA\_SIG_{rsk_{jo}}(rpk_{sp}). \tag{7}$$

Then JO generates a "designated receiver of secret e-cash" that contains the e-coins generated in the cash break phase and its signature on the receiver's public key, and then sends it as well as the SP's public key to MA:

$$\begin{aligned} JO \rightarrow MA : RSA\_ENC_{rpk_{sp}}\big(&\underbrace{E(1), \ldots, E(1)}_{w}, \\ &\underbrace{E(0), \ldots, E(0)}_{2^L - w}, sig\big), rpk_{sp} \end{aligned} \tag{8}$$

7) Payment Delivery: When MA receives the JO's encrypted payment to the SP with public key rpk_{sp}, MA verifies whether the SP has submitted the data.

If yes, MA forwards the encrypted payment to the SP:

$$MA \to SP : RSA\_ENC_{rpk_{sp}}\big(\underbrace{E(1),\dots,E(1)}_{w}, \underbrace{E(0),\dots,E(0)}_{2^L-w}, sig\big). \quad (9)$$

Otherwise, MA waits for the SP to submit data and then deliver the encrypted payment as (9).

8) Money Deposit: After SP receives the encrypted message from MA, it first decrypts the message and gets the e-cash and JO's signature

$$\big(\underbrace{E(1),\dots,E(1)}_{w}, \underbrace{E(0),\dots,E(0)}_{2^L-w}, sig\big). \quad (10)$$

SP verifies the validity of the sig using the JO's public key. In addition, SP check whether there are w valid e-coin by verifying all e-coins. If both results are positive, SP confirms the payment to MA and asks MA to send its data to JO.

Next, SP waits for a random period of time and then starts to deposit all w e-coins one by one to MA. In addition, SP waits for a random period of time between two consecutive deposits of e-coin.

$$SP \to MA : AID_{sp}, \underbrace{E(1),\dots,E(1)}_{w} \quad (11)$$

When MA receives the deposit from SP, MA verifies the e-cash's validity. If the e-cash is valid, MA adds the money to SP's account.

9) Summary of $PPMS_{dec}$: To be complete, in Algorithm 1 we summarize our mechanism $PPMS_{dec}$.

Algorithm 1 $PPMS_{dec}$
```
1:  JO → MA:  $jd, w, rpk_{jo}$
2:  MA → JO:  $E(2^L)$
3:  JO:  $E(2^L) \to E(w_1), E(w_2), \dots, E(w_{L+2})$
4:  SP → MA:  $rpk_{sp}$
5:  MA → JO:  $rpk_{sp}$
6:  JO → MA:  $RSA\_ENC_{rpk_{sp}}(E(w_1), \dots, E(w_{L+2}), sig)$
7:  SP → MA:  data report
8:  MA → JO:  data report
9:  MA → SP:  $RSA\_ENC_{rpk_{sp}}(E(w_1), \dots, E(w_{L+2}), sig)$
10: SP → MA:  $AID_{sp}, E(w_{\rho(i)})$ for $i \in [1, L+2]$
```

B. Privacy Analysis

1) SP's privacy: First, we consider the transaction-linkage privacy of SP. Since we have shown it is impossible to link the JO to a transaction, it directly implies that no adversary can link a SP to the JO, otherwise it directly means the adversary knows the JO has made a transaction (with the SP). Thus $PPMS_{dec}$ protects SP's transaction privacy.

Next, consider the job-linkage privacy of SP. In Algorithm 1, the SP has used two piece of identity information. The first one is a random RSA public key rpk_{sp} that is used in the labor registration phase. SP's sensing data and the job is directly linked to rpk_{sp}. The second one is SP's account ID (AID_{SP}), which is used in the money deposit phase. It is easy to see rpk_{sp} and AID_{SP} are totally independent. No adversary can directly link AID_{SP} to rpk_{sp}. However, there is one piece of information that could provide some inference on the linkage of AID_{SP} and rpk_{sp} which is the amount of a job's payment. For example, if MA knows the job that the SP participates in, with rpk_{sp}, this makes a payment of 8 to each SP. Later, when a JO deposits its payment (assume JO does not break payment) to the bank which also equals 8, the MA may know it is likely the payment that comes from the job it previously saw. This could deteriorate SP's job-linkage privacy. In this paper, we refer to this kind of inference attack as "denomination attack". To thwart this attack, $PPMS_{dec}$ uses cash break technique to break the payment. After breaking JO's payment into k small payments, the payment received by the SP could come from a job whose payment has at most $\sum_{i=1}^{k}\binom{k}{i}$ different values. This could make the denomination attack less effective, or more prone to fail at most times. In our $PPMS_{dec}$, it is easy to see the possible sum of previous deposits could equal to every elements in $[1, w]$. As the number of jobs that the SP participates in become greater, the possible sum of previous deposits could cover all element in $[1, 2^L]$ which makes the denomination attack completely fail.

Finally, consider the data-linkage privacy of an SP. Since we have shown $PPMS_{dec}$ protects the SP's job-linkage privacy quite well, it directly follows $PPMS_{dec}$ also protects the SP's data-linkage quite well. Otherwise, the adversary could link the data in a job to the identity of an SP. This directly allows the adversary to link the job to the identity of an SP.

2) JO's privacy: As byproducts, $PPMS_{dec}$ also protects the identity privacy of the JO.

First, we consider the job-linkage privacy of JO. In Algorithm 1, the JO has used two pieces of identity information. The first one is its Camenisch-Lysyanskaya signature's public key $clpk_{JO}$ that is used for generating the e-cash in the money withdrawal phase. Since the DEC scheme guarantees the withdrawal process is anonymous, no adversary can see $clpk_{JO}$ in this phase. Also, the DEC scheme guarantees that as long as JO does not double-spend its e-cash, $clpk_{JO}$ cannot be derived from the e-cash. Thus $clpk_{JO}$ is safe in $PPMS_{dec}$. The other one is a random RSA public key rpk_{jo} that is used in the job registration phase. rpk_{jo} reveals no information about the authentic identity of JO. It is easy to see that no adversary can link the JO's real identity to the job it registers for. In other words, it is impossible to know if JO has ever made a transaction (as long as JO does not double spend its cash).

As for the transaction-linkage privacy, due to the perfect anonymity of DEC, no adversary can link the e-cash to its original owner. Thus $PPMS_{dec}$ protects both JO's job-linkage privacy and its transaction-linkage privacy.

C. Improving the efficiency of cash break

In $PPMS_{dec}$, we choose to break payment w into w unitary coins of value 1. Although this breaking scheme has a good performance in defending from the denomination attack, it makes the JO have to send 2^L e-cash and the SP have to receive and verify 2^L e-cash. When L is big, the efficiency of $PPMS_{dec}$ could become low. Thus, we provide two different cash break algorithms which have better efficiency. We designed our algorithms based on the following findings. If we break $2^L - 1$ into $2^L - 1$ unitary coins, the sum of coins' denomination could cover all elements in $[1, 2^L - 1]$. If we break $2^L - 1$ into L coins with denomination $1, 2, 4, \dots, 2^{L-1}$, the sum of coins also covers all elements in $[1, 2^L - 1]$. However, the latter scheme only needs to break it into L small coins, which yields much better efficiency compared with breaking it into $2^L - 1$ unitary coins especially when L is large.

We name the two algorithms as the Privacy-aware Cash Break algorithm (PCBA) and the Enhanced Privacy-aware Cash Break algorithm (EPCBA). PCBA breaks w directly following its binary representation, while EPCBA aims to break w into smaller denominations which include more elements in $\{1, 2, 4, \dots, 2^{L-1}\}$. In particular, JO runs the Algorithm 2 to determine the breaking plan and generate the $L + 2$ e-cash in small denominations $w_1, w_2, \dots, w_{L+2}$.

Algorithm 2 Privacy-aware Cash Break Algorithm

```
Require: $w$: the amount of credits to the SP $(1 \leq c \leq 2^L)$;
Ensure: $w_1, w_2, \dots, w_{L+1}$: $L + 1$ denominations whose sum equal $w$;
1: $a \leftarrow \Sigma_{i=1}^{L+1} B(w)[i]$;
```

Denote $B(\cdot)$ as the function that takes an integer in $[0, 2^L]$ as the input, outputs the integer's $(L+1)$-bit binary representation. In addition, denote $B(\cdot)[i]$ $(i \in [1, L+1])$ as the i-th least-significant bit of $B(\cdot)$.

Algorithm 3 Enhanced Privacy-aware Cash Break Algorithm

```
Require: $w$: the amount of credits to the SP $(1 \leq c \leq 2^L)$;
Ensure: $w_1, w_2, \dots, w_{L+2}$: $L + 2$ denominations whose sum equal $w$;
1: $a \leftarrow \Sigma_{i=1}^{L+1} B(w)[i]$;
2: $a' \leftarrow \Sigma_{i=1}^{L+1} B(w-1)[i]$;
3: if $a \leq a'$ then
4:     $w_i \leftarrow 2^{i-1} B(w-1)[i]$ for $i \in [1, L+1]$;
5:     $w_{L+2} \leftarrow 1$;
6: else
7:     $w_i \leftarrow 2^{i-1} B(w)[i]$ for $i \in [1, L+1]$;
8:     $w_{L+2} \leftarrow 0$;
9: end if
```

V. A light-weight privacy-preserving mechanism for markets of unitary payments

In this section, we consider a special type of sensing market in which all jobs' payments are unitary. We show how to design a light-weight privacy prserving mechanism for this kind of market without using e-cash primitives.

In particular, we allow the bank (MA) to know which two residents have made transactions with each other. However, we stress that our mechanism still protect the SP's data-linkage privacy and job-linkage privacy against both the JO and the MA, as well as the transaction privacy against the JO.

A. Mechanism design of $PPMS_{pbs}$

1) Setup: In $PPMS_{pbs}$, we use JO's signature as the digital coin. Each JO first generates a private RSA key pair $rpk_{JO} rsk_{JO})$, and sends the public key rpk_{JO} to MA to bind it to its own account. Similarly, each SP generates a private RSA key pair (rpk_{SP}, rsk_{SP}), and sends the public key rpk_{SP} to MA to bind it to its own account.

2) Job registration: To register a job in the market, the JO first generates a random RSA key pair (rpk_{jo}, rsk_{jo}), and sends to the MA a job profile that specifies the job description jd, and rpk_{jo} as its identity information:

$$JO \rightarrow MA : jd, rpk_{jo} \tag{12}$$

After receiving the job profile, MA publishes it in the bulletin board BB so that all market residents could see them:

$$MA \rightarrow BB : rpk_{jo} \tag{13}$$

3) Labor registration: To sign for a job, an SP randomly generates an RSA key pair (rpk_{sp}, rsk_{sp}) as well as a random serial number s, uses the JO' public key rpk_{jo} to encrypt rpk_{sp} and s, and sends the ciphertext to the MA.

$$SP \rightarrow MA : RSA\_ENC_{rpk_{jo}}(rpk_{sp}, s) \tag{14}$$

After receiving the ciphertext, MA forwards it to the JO.

$$MA \rightarrow JO : c = RSA\_ENC_{rpk_{jo}}(rpk_{sp}, s) \tag{15}$$

JO decrypts the c to get the SP's public key rpk_{sp} and serial number s. It generates an RSA signature on rpk_{sp} and s using rsk_{jo}, encrypts its public key rpk_{JO} and the signature using rpk_{sp}, and then sends the ciphertext to MA.

$$JO : (rpk_{sp}, s) = RSA\_DEC_{rsk_{jo}}(c) \tag{16}$$

$$JO : sig = RSA\_SIG_{rsk_{jo}}(rpk_{sp}, s) \tag{17}$$

$$JO \rightarrow MA : RSA\_ENC_{rpk_{sp}}(rpk_{JO}, sig), rpk_{sp} \tag{18}$$

After receiving the ciphertext, MA sends it to the SP with public key rpk_{sp}.

$$MA \rightarrow SP : c = RSA\_ENC_{rpk_{sp}}(rpk_{JO}, sig) \tag{19}$$

After receiving the ciphertext from MA, the SP decrypts the public key and verifies the signature using JO's publickey rpk_{jo}:

$$SP : (rpk_{JO}, sig) = RSA\_DEC_{rsk_{sp}}(c) \tag{20}$$

$$SPRSA\_SIGVERI_{rpk_{jo}}(sig) \tag{21}$$

If the verification passes, the SP proceeds in remaining steps; Otherwise, the SP aborts the entire process.

4) Payment submission: Knowing the JO's public key psk_{JO}, the SP helps the JO to generate a partial blind signature on its publickey rpk_{SP} and a pre-agreed serial number s.

$$SP \rightarrow JO : pbs = RSA\_PBSign_{psk_{JO}}(rpk_{SP}, s) \quad (22)$$

JO sends the partial blind signature pbs as well as the corresponding SP's public key rpk_{sp} to the MA.

5) Payment Delivery: When MA receives the data report from the SP, the MA sends it the corresponding partial blind signature:

$$MA \rightarrow SP : pbs \quad (23)$$

6) Money Deposit: After SP receives the partial blind signature pbs from MA, it first recovers the signature from it.

$$SP : sig = RSA\_PBSREC(pbs) \quad (24)$$

The SP verifies this signature using the JO's public key rpk_{JO}:

$$SP : RSA\_SIGVERI(sig, rpk_{JO}, rpk_{SP}, s) \quad (25)$$

If it passes, the SP lets the MA send the data report to the JO. After a random period of time, SP deposits the signature to the MA by sending the signature sig, its real public key rpk_{SP}, the JO's real public key rpk_{JO}, and the serial number s.

$$SP \rightarrow MA : sig, rpk_{SP}, rpk_{JO} \quad (26)$$

MA verifies the signature's correctness and the freshness of the serial number. If both two verifications pass, MA adds/deducts corresponding money to the corresponding accounts.

7) Summary of $PPMS_{pbs}$: Briefly, our second mechanism can be summarized as Alg.4.

Algorithm 4 Algorithm Summary of $PPMS_{pbs}$

```
1: MA → JO: $jid$
2: SP → JO: $pk_{pse}$
3: JO → SP: $RSA\_ENC_{pk_{pse}}(pk_{JO})$
4: JO: $PBSign(pk_{SP}, jid)$
5: JO → MA: $PBSign(pk_{SP}, jid)$
6: SP → MA: data report
7: MA → SP: $PBSign(pk_{SP}, jid)$
8: SP: $PBSign\_Verify(PBSign(pk_{SP}, jid))$
9: MA → JO: data report
10: SP → MA: $PBSign(pk_{SP}, jid), pk_{SP}, pk_{JO}$
```

B. Security analysis

First, we consider SP's transaction-linkage privacy against the JO. SP uses its public key rpk_{SP} only when it helps the JO to generate the partial blind signature in the payment submission step. Due to the blindness of the partial blind signature, the JO knows nothing about the SP's private public key, thus does not know whom it makes the transaction with. SP's transaction-linkage privacy against the JO is protected.

Next, we consider the SP's job-linkage privacy against the JO and the MA. Since SP's transaction-linkage privacy is protected, the JO does not know whom it makes the transaction with, thus does not know who have participated in its job. In addition, although the MA knows which JO and which SP have made transactions, it does not know which job this transaction corresponds to due to the JO not publishing the job using its real identity information, and all jobs' payments per sensing participant are the same. Thus, SP's job-linkage privacy against the JO and the MA is protected.

Finally, since SP's job-linkage privacy against the JO and the MA is protected, it directly follows its data-linkage privacy against the JO, and the MA is protected also.

VI. Performance Evaluations

As part of our work, we bring our scheme into reality. Considering this scheme may be applied in mobile devices, and to be more practical, we want it independent on particular operating systems. Putting all these together, we implement this scheme in Java rather than C++ and Crypto++, which is popular in cryptographic applications. As a matter of concern, we measure these two type implementations' efficiency. Results and the experiment's detail will be introduced in the last subsection.

A. Cunningham chain in Setup

In $PPMS_{dec}$, a series of computations need to be finished in the setup stage. The most expensive computation is finding cyclic groups satisfying subgroup constrain, where $o_{i+1} = 2o_i + 1$. In other words, it's difficult to find a prime number chain forming $o_{i+1} = 2o_i + 1$, which is more generally known as Cunningham chain of the first kind [28] [29]. The computational complexity of this chain is very high, not until March 2014, have second kind Cunningham chain with length 19 been found by Raanan Chermoni and Jaroslaw Wroblewski [30]. As for the first kind, length 17, having 2759832934171386593519 as its smallest beginning number, is the limitation so far [30]. Even a chain with length 7 has a 7-digits' smallest beginning number. In a word, it's unreasonable to compute this chain in setup stage for each time. So, we separate $PPMS_{dec}$'s setup stage from online executing. We will discuss both the setup time and executing time without setup stage later in this section.

B. Bilinear map in CLSign scheme

The precondition of Camenisch-Lysyanskaya signature scheme is a non-degenerate efficiently computable bilinear map e [27]. But to find a proper map is not easy. Fortunately, Ben Lynn and Dan Boneh give a solution to this question [31] [32]. And a Java paring-based cryptography library(JPBL) has been provided [33], which implements Lynn's method and provides bilinear maps' generators. However, it's also acceptable if anyone wants to map the multiplicative group into an additive group, in this case, a bilinear map is very easy to find, and the correctness of signature will still hold.

C. *Different types of proof of knowledge*

There are different types of zero-knowledge proof used in our mechanism: proving the knowledge of a discrete logarithm of a group element to a base [34], of a representation of an element to a series of bases [35], of a double discrete logarithm [36], of at least one out of the discrete logarithm of elements to the base [37] [38]. And for some situations, we need to combine two or more of them to achieve one new type of proof. We use Fiat-Shamir heuristic [39] to implement those proofs so that they can be finished in one round of interaction.

D. *Efficiency*

In order to prove that our mechanisms are really practical, we measure our implementations' efficiency. Several experiments are performed, so that we can have an intuition expression through comparing the results. In case that some unsteadiness of the running environment may exist, we perform every experiment for 100 times, and record the average time as performing time. And a matter to note is that JVM will optimize Java code and need initial time, therefore, the first experiment performed always takes a little more time to finish.

As mentioned above, our first mechanism $PPMS_{dec}$'s setup stage can be finished in advance. However, the setup executing time still need be shown. In this case, our attention is setup stage, so Ni is irrelevant and we fix $Ni = 0$. The time record of each level is shown in Fig.2. Obviously setup executing time is especially high when the level reaches 7, the reason is obvious too, for computing the prime chain.

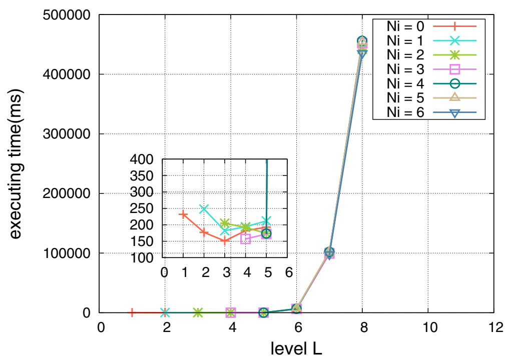

Fig. 2: Setup executing time of each level.

With the setup stage finished, we can use groups and parameters generated to process mechanism's main steps. In a real application, we can not predict which node may be used, so we assume that every node may be used in some deal, and measure every possible node in the divisible tree of every specific level. The result is shown in Fig.3 including all main steps after setup stage. If we fix the level (e.g. $L = 12$), and observe each Ni's executing time, the increasing phenomenon will be more obvious. But, we can tell from the result that the growth rate is acceptable. Even with $Ni = 10$, the executing time is still within 30ms.

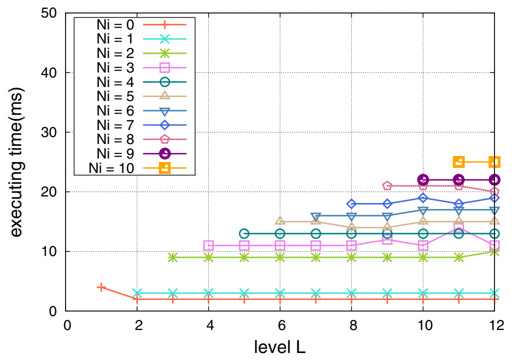

Fig. 3: Executing time of each possible node level.

One of the most important operation in $PPMS_{dec}$ is cash breaking. As mentioned before, we want to break a coin into several coins with cost as small as possible. To verify cash breaking is reasonable, the last experiment of $PPMS_{dec}$ is the measurement of braking executing time. We fix level $L = 12$, and use generated parameters and groups to calculate every child nodes and their path values to root. The results shown in Fig.4 proves our thought. With a fixed level, the deeper a child node is in the tree, the higher the cost.

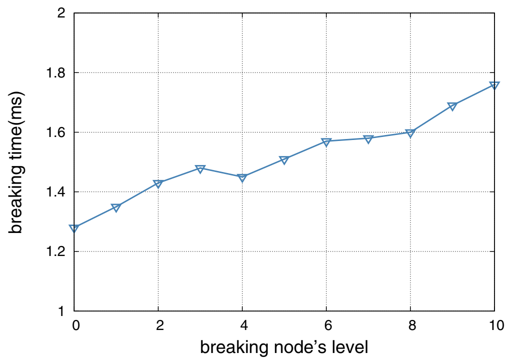

Fig. 4: Executing time of each breaking node.

As for the second mechanism $PPMS_{pbs}$ we introduced, the main component is partial blind signature. We use an RSA-based blind signature method [40]. Every round of deal involves one unitary coin. Our $PPMS_{pbs}$ makes a tradeoff between privacy and efficiency. In order to verify that the second mechanism is more efficient than the first one, we measured the average of multiple rounds of executing time of the two mechanisms, both including a setup stage. As shown in Fig.5, the result follows our expectation. With one single round costing less time, $PPMS_{pbs}$ has a much lower growth rate than $PPMS_{dec}$.

We have analyzed the core basic operation of our mechanisms. We focus on these core basic operations, hash function, encryption, decryption, and zero-knowledge proof. For the sake

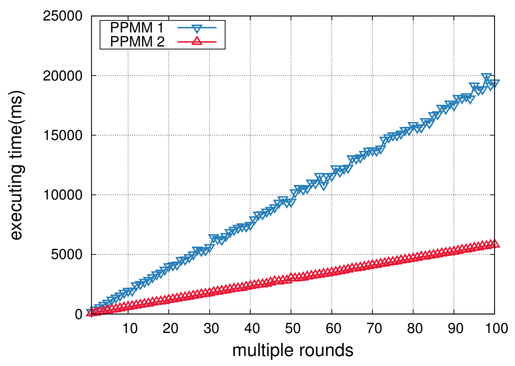

Fig. 5: Executing time comparing of multiple rounds.

TABLE I: core operation complexity comparing

mechanism	JO	SE	MA
$PPMS_{dec}$	(8+i)ZKP+4Enc+1Dec+1H	4Dec	1Enc
$PPMS_{pbs}$	2Enc+1H	2Dec+3H	1Dec+2H

TABLE II: communication traffic comparing

Scheme	JO (bytes)		SP (bytes)		Total (kb)
	Input	Output	Input	Output	
first	664	4864	3840	2176	11.27
second	256	784	768	384	2.14

of simpleness, we consider signature as encryption and verifying signature as decryption, which is the opposite to common sense, but this can make it easier to show the results in Table.I. We list four operations, ZKP for zero-knowledge proof, and Enc for encryption and signature, Dec for decryption and verifying, H for hash function. This table may not be accurate enough, because we ignore some operations such as exponent and random big prime generating, but this can reflect main computation complexity of our mechanisms.

Besides computation, we also compare the communication traffic involved in our mechanisms. We observe and count the main traffic flow through input and output. And the scenario set is the same as executing time comparing situation, where the level and node index of $PPMS_{dec}$ are minimum. The result can be found in Table.II. Because of the use of zero-knowledge proof, $PPMS_{dec}$ has more traffic than $PPMS_{pbs}$. But this is also acceptable, nowadays communication links generally have high transmission speed.

VII. Conclusion

It is of great interest and value to consider both privacy issues and incentive issues for mobile sensing markets. In this paper, we design two mechanisms that protect sensing participants' privacy in incentive-driven sensing markets. Our first mechanism $PPMS_{dec}$, is able to protect both the job owner and the sensing participants' identity privacy in markets that allow arbitrary amount of payments. Our second mechanism $PPMS_{pbs}$ is specifically designed to protect the sensing participants' identity privacy in the markets of unitary payments. Experiments demonstrate that both two mechanisms have good efficiency.

References

[1] M. Rabbi, S. Ali, T. Choudhury, and E. Berke, "Passive and in-situ assessment of mental and physical well-being using mobile sensors," in *Ubicomp*, J. A. Landay, Y. Shi, D. J. Patterson, Y. Rogers, and X. Xie, Eds. ACM, 2011, pp. 385–394.

[2] J. Dai, X. Bai, Z. Yang, Z. Shen, and D. Xuan, "Mobile phone-based pervasive fall detection," *Personal and Ubiquitous Computing*, vol. 14, no. 7, pp. 633–643, 2010.

[3] A. Thiagarajan, J. Biagioni, T. Gerlich, and J. Eriksson, "Cooperative transit tracking using smart-phones," in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '10. New York, NY, USA: ACM, 2010, pp. 85–98.

[4] K. Ali, D. Al-Yaseen, A. Ejaz, T. Javed, and H. Hassanein, "Crowdits: Crowdsourcing in intelligent transportation systems," in *Wireless Communications and Networking Conference (WCNC), 2012 IEEE*, April 2012, pp. 3307–3311.

[5] R. K. Rana, C. T. Chou, S. S. Kanhere, N. Bulusu, and W. Hu, "Ear-phone: An end-to-end participatory urban noise mapping system," in *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, ser. IPSN '10. New York, NY, USA: ACM, 2010, pp. 105–116.

[6] M. Mun, S. Reddy, K. Shilton, N. Yau, J. Burke, D. Estrin, M. H. Hansen, E. Howard, R. West, and P. Boda, "Peir, the personal environmental impact report, as a platform for participatory sensing systems research," in *MobiSys*, K. Zielinski, A. Wolisz, J. Flinn, and A. LaMarca, Eds. ACM, 2009, pp. 55–68.

[7] "Amazon mechanical turk." [Online]. Available: https://www.mturk.com/mturk/welcome

[8] "Clickworker." [Online]. Available: http://www.clickworker.com

[9] "Workforce support: Helpdesk." [Online]. Available: support.crowdsource.com

[10] "Domywork." [Online]. Available: http://www.domywork.net

[11] Q. Li and G. Cao, "Providing privacy-aware incentives for mobile sensing," in *Pervasive Computing and Communications (PerCom), 2013 IEEE International Conference on*. IEEE, 2013, pp. 76–84.

[12] Q. Li, G. Cao, and T. F. L. Porta, "Efficient and privacy-aware data aggregation in mobile sensing." 2014, pp. 115–129.

[13] C. Cornelius, A. Kapadia, D. Kotz, D. Peebles, M. Shin, and N. Triandopoulos, "Anonysense: Privacy-aware people-centric sensing," in *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '08. New York, NY, USA: ACM, 2008, pp. 211–224.

[14] M. Shao, Y. Yang, S. Zhu, and G. Cao, "Towards statistically strong source anonymity for sensor networks," in *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, April 2008, pp. –.

[15] D. Christin, C. Rosskopf, M. Hollick, L. Martucci, and S. Kanhere, "Incognisense: An anonymity-preserving reputation framework for participatory sensing applications," in *Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on*, March 2012, pp. 135–143.

[16] J. Sun, "An incentive scheme based on heterogeneous belief values for crowd sensing in mobile social networks," in *Global Communications Conference (GLOBECOM), 2013 IEEE*, Dec 2013, pp. 1717–1722.

[17] I. Koutsopoulos, "Optimal incentive-driven design of participatory sensing systems," in *INFOCOM, 2013 Proceedings IEEE*, April 2013, pp. 1402–1410.

[18] X. Zhang, Z. Yang, Z. Zhou, H. Cai, L. Chen, and X. Li, "Free market of crowdsourcing: Incentive mechanism design for mobile sensing," *Parallel and Distributed Systems, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2014.

[19] J. Zhang, J. Ma, W. Wang, and Y. Liu, "A novel privacy protection scheme for participatory sensing with incentives," in *Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on*, vol. 03, Oct 2012, pp. 1017–1021.

[20] D. Christin, C. Buchner, and N. Leibecke, "What's the value of your privacy? exploring factors that influence privacy-sensitive contributions to participatory sensing applications," in *Local Computer Networks Workshops (LCN Workshops), 2013 IEEE 38th Conference on*, Oct 2013, pp. 918–923.

[21] T. Giannetsos, S. Gisdakis, and P. Papadimitratos, "Trustworthy people-centric sensing: Privacy, security and user incentives road-map," in *Ad Hoc Networking Workshop (MED-HOC-NET), 2014 13th Annual Mediterranean*, June 2014, pp. 39–46.

[22] T. Okamoto, "An efficient divisible electronic cash scheme," in *Advances in Cryptology CRYPT0 95*, ser. Lecture Notes in Computer Science, D. Coppersmith, Ed. Springer Berlin Heidelberg, 1995, vol. 963, pp. 438–451.

[23] A. Chan, Y. Frankel, and Y. Tsiounis, "Easy come easy go divisible cash," in *Advances in Cryptology EUROCRYPT'98*, ser. Lecture Notes in Computer Science, K. Nyberg, Ed. Springer Berlin Heidelberg, 1998, vol. 1403, pp. 561–575.

[24] T. Nakanishi and Y. Sugiyama, "Unlinkable divisible electronic cash," in *Information Security*, ser. Lecture Notes in Computer Science, G. Goos, J. Hartmanis, J. van Leeuwen, J. Pieprzyk, J. Seberry, and E. Okamoto, Eds. Springer Berlin Heidelberg, 2000, vol. 1975, pp. 121–134.

[25] S. Canard and A. Gouget, "Divisible e-cash systems can be truly anonymous," in *Advances in Cryptology - EUROCRYPT 2007*, ser. Lecture Notes in Computer Science, M. Naor, Ed. Springer Berlin Heidelberg, 2007, vol. 4515, pp. 482–497.

[26] D. Chaum, "Blind signatures for untraceable payments," in *CRYPTO*, D. Chaum, R. L. Rivest, and A. T. Sherman, Eds. Plenum Press, New York, 1982, pp. 199–203.

[27] J. Camenisch and A. Lysyanskaya, "Signature schemes and anonymous credentials from bilinear maps," in *Advances in Cryptology CRYPTO 2004*, ser. Lecture Notes in Computer Science, M. Franklin, Ed. Springer Berlin Heidelberg, 2004, vol. 3152, pp. 56–72.

[28] A. Cunningham, "On hyper-even numbers and on fermat's numbers," *Proceedings of the London Mathematical Society*, vol. s2-5, no. 1, pp. 237–274, 1907.

[29] G. Lh, "Long chains of nearly doubled primes," *Mathematics of Computation*, vol. 53, no. 188, pp. pp. 751–759, 1989.

[30] "Cunningham chain records." [Online]. Available: http://primerecords.dk/Cunningham_Chain_records.htm

[31] B. Lynn, "On the implementation of pairing-based cryptosystems," Ph.D. dissertation, Stanford University, 2007.

[32] D. Boneh, I. Mironov, and V. Shoup, "A secure signature scheme from bilinear maps," in *Topics in Cryptology CT-RSA 2003*, ser. Lecture Notes in Computer Science, M. Joye, Ed. Springer Berlin Heidelberg, 2003, vol. 2612, pp. 98–110.

[33] A. De Caro and V. Iovino, "jpbc: Java pairing based cryptography," in *Proceedings of the 16th IEEE Symposium on Computers and Communications, ISCC 2011*. Kerkyra, Corfu, Greece, June 28 - July 1: IEEE, 2011, pp. 850–855.

[34] M. Girault, G. Poupard, and J. Stern, "On the fly authentication and signature schemes based on groups of unknown order," *Journal of Cryptology*, vol. 19, no. 4, pp. 463–487, 2006.

[35] J. Camenisch and M. Michels, "Proving in zero-knowledge that a number is the product of two safe primes," in *Advances in Cryptology EUROCRYPT 99*, ser. Lecture Notes in Computer Science, J. Stern, Ed. Springer Berlin Heidelberg, 1999, vol. 1592, pp. 107–122.

[36] M. Stadler, "Publicly verifiable secret sharing," in *Advances in Cryptology EUROCRYPT 96*, ser. Lecture Notes in Computer Science, U. Maurer, Ed. Springer Berlin Heidelberg, 1996, vol. 1070, pp. 190–199.

[37] R. Cramer, I. Damgrd, and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," in *Advances in Cryptology CRYPTO 94*, ser. Lecture Notes in Computer Science, Y. Desmedt, Ed. Springer Berlin Heidelberg, 1994, vol. 839, pp. 174–187.

[38] A. De Santis, G. Di Crescenzo, G. Persiano, and M. Yung, "On monotone formula closure of szk," in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, ser. SFCS '94. Washington, DC, USA: IEEE Computer Society, 1994, pp. 454–465.

[39] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Advances in Cryptology CRYPTO 86*, ser. Lecture Notes in Computer Science, A. Odlyzko, Ed. Springer Berlin Heidelberg, 1987, vol. 263, pp. 186–194.

[40] H.-Y. Chien, J.-K. Jan, and Y.-M. Tseng, "Rsa-based partially blind signature with low computation," in *Parallel and Distributed Systems, 2001. ICPADS 2001. Proceedings. Eighth International Conference on*. IEEE, 2001, pp. 385–389.