POLA: A Privacy-Preserving Protocol for Location-Based Real-Time Advertising

Yiming Pang, Yichen Chen, Peiyuan Liu, Fudong Qiu, Fan Wu and Guihai Chen*
Department of Computer Science and Engineering, Shanghai Jiao Tong University, China

***Abstract*—The increasing popularity of smartphones, equipped with GPS, provides new opportunities for location-based service (LBS). Among all kinds of LBSs, targeted advertising based on users' locations takes great advantage of the rich location data to improve the accuracy of advertising and thus potentially increase the sellers' profits. However, location-based advertising (LBA) has raised significant privacy concerns, since the location information used in such kinds of services is private information, which the users may not be willing to expose. In this paper, we present POLA, which is a Privacy-preserving prOtocol for Location-based real-time Advertising. In this protocol, we not only preserve the privacy of the location data, we also take the values of advertisers into consideration which is also regarded as private information. We show the privacy-preserving properties of POLA in details. Furthermore, we have conducted simulations to evaluate the performance of POLA. Evaluation results show that POLA achieves privacy preserving LBA with relatively low overhead.**

I. Introduction

Location-based advertising (LBA) is a newly emerged way of selling and buying ads. Compared to the traditional ways of advertising, LBA is able to utilize the location data of the mobile users to display more relevant ads to the users, which significantly improves the profits the ads can bring. More relevant ads are expected to attract more attentions from the users, *e.g.*, Alice is walking down a street while browsing some web pages, then an advertisement of a nearby restaurant is more likely to attract her attention than that of a restaurant 5 blocks away. In other words, this kind of relevancy can result in high click through rates and high profits, and hence a number of advertisers flow into the location-based advertising platform.

To better match the advertisement and the user, ad exchange appears as an emerging way to sell and buy display ads on the Internet effectively. RightMedia, AdECN and DoubleClick Ad Exchange are such examples. Ad exchanges offer a place for publishers (the one who controls the web pages) and advertisers (the one who wants to display the ads) to negotiate and transact for ads. Once the publisher notifies the ad exchange of a free advertisement inventory, the ad exchange contacts interested advertisers. After acquiring their bids, the ad exchange chooses a winner, displays his advertisement on the slot and charges him if the advertisement is clicked.

Although LBA not only brings convenience to the mobile users, but also brings high profits to the advertiser, it induces great concerns, particularly among privacy advocacy groups. In order to use the LBA, a mobile user has to expose his/her current location to the advertiser, which is not expected by most users. Several protocols (*e.g.*, [1]–[7]) have been proposed to deal with the privacy concerns. Despite the advancing progress in the protection of privacy, two deficiencies are found in current models. One problem is that they all introduce a trusted third party to help them anonymize the users' identities. However, the existence of a trusted third party is questionable, since it is usually costly and once being attacked, the whole system will be crashed. The other problem is that the privacy of the advertisers is not protected in current models. In fact, the way advertisers price the advertisement is private information of advertisers, since the prices reflect their marketing strategy. Yet few works notice this problem.

In this paper, we introduce a system that can protect the location privacy of the users without the involvement of a trusted third party and allow the auction of selling advertisements to proceed rationally while preventing the ad exchanges from knowing the bids of the advertisers. To the best of our knowledge, we are the first to get rid of the two deficiencies simultaneously.

Our main contribution is that we design a system that:

- The advertisers can display relevant advertisement based on the location information of the mobile users without revealing the users' location privacies.
- The location information will not be exposed to anyone in the model including the advertisers and the ad exchange, assuming that the ad exchange and the advertisers do not collude with each other. Even the advertisers collude with a small number of advertisers, the probability that the location information is leaked is still very low.
- An advertiser's value of the specific location information, which is the advertiser's private information, will not be learned by anyone including the users and the ad exchange except the winner's charging price, assuming that the ad exchange and the user do not collude with each other.
- The users and the advertisers cannot gain utilities by colluding with each other to manipulate the auction.

The remainder of this paper is organized as follows. In Section II, we briefly review the related work. In Section III, we present some technical preliminaries. In Section IV, we present the system model and the detailed design of POLA. In Section V, we analyze the properties of our system and show how the system is robust against several possible attacks, followed by the performance evaluation in Section VI. Finally, we conclude our work in Section VII.

II. Related Work

Juels [1] is the first one who gives some examples of modern advertising models such as the keyword-based bidding. Juels's focus is on the private distribution of the advertisements and does not take other aspects such as the click report or the auctions into consideration. Juels proposed a full mix net between the client and broker, thus effectively solved the problem of collusion. However, the problem is that this privacy model is too strong. Therefore, it comes at the price of efficiency.

The privacy related issue in online advertising has received much attention in recent years [2]–[6]. Guha *et al.* proposed *Privad* [3] to protect privacy through unlinkability and used a dealer mechanism to ensure it. Toubiana *et al.* proposed *Adnostic* [2] as a browser extension that is able to generate the data for targeted advertising at the client side. *Adnostic* also provided a privacy-preserving accounting tool to ensure that the private information of the user would not be leaked to the outside world. Götz and Nath [5] designed a differentially-private distributed protocol that can protect the privacy of the users with the help of differential privacy and generalization. *Koi* [6] provided privacy-preserving LBS based on an exchange protocol among the user and 2 non-colluding servers. Similar to *Privad* [3], *Koi* provided privacy preservation based on unlinkability.

The works mentioned above all take users' privacy into consideration in some way, however, all of them require a trusted third party in their system, which is a very strong assumption. In [7], Lu *et al.* proposed *PLAM* which is a privacy-preserving framework for LBS. But it mainly concerns about location-based services generally instead of location-based advertising specifically, which means the work overlooks the ad auctions. In [8], Angel and Walfish discussed the privacy issues in ad auction. However, the work mainly focuses on the verifiability of the auction instead of privacy.

In our work, we jointly consider the privacy-preserving process without the involvement of a trusted third party and the privacy of the advertisers, which make the privacy model of the system much more stronger.

III. Preliminaries

In this section, we will briefly introduce the two cryptographic tools that we use in our protocol. One is the Kushilevitz-Ostrovsky protocol [9] for private information retrieval. The other is the Boneh-Goh-Nissim cryptosystem [10].

A. Private Information Retrieval

Private information retrieval is the problem that a user wants to retrieve the i^{th} item of the database with N entries, revealing no information about his choice to the database owner. It is proposed by Chor *et al.* in [11]. The protocol is mainly based on the quadratic residuosity assumption.

Quadratic residuosity assumption (QRA) is proposed in the pioneering work of Goldwasser and Micali [12].

Definition 1 (Quadratic Residue)**.** *An integer q is called a quadratic residue modulo n if there exists x such that:*

$$x^2 \equiv q(mod\ n) \tag{1}$$

We then have the following lemma [13]:

Lemma 1. *Given the factorization of a composite integer n, we can decide whether $q \in Z_n^*$ is a quadratic residue mod n within $O(n^3)$*

When the factorization of n is unknown, we can still obtain some information about whether a number is a quadratic residue from the Jacobi symbol. Assume $n = p_1p_2$ where p_1 and p_2 are two prime numbers. The Jacobi symbol $(\frac{a}{n})$ is defined as $(\frac{a}{n}) = (\frac{a}{p_1})(\frac{a}{p_2})$, where $(\frac{a}{p})$ is defined for all integers and odd primes p by

$$(\frac{a}{p}) = \begin{cases} 0 & \text{if } a \equiv 0(\text{mod } p) \\ +1 & \text{if } a \not\equiv 0 \wedge \exists x, a \equiv x^2(\text{mod } p) \\ -1 & \text{otherwise} \end{cases} \tag{2}$$

Despite the fact that the Jacobi symbol is defined through the factorization of n, $(\frac{a}{p})$ can be computed within polynomial time when the factorization is unknown. When $(\frac{a}{n}) = -1$, a should be a quadratic nonresidue mod n. Yet, if $(\frac{a}{n}) = 1$, then either a is a quadratic residue mod n or a is a quadratic nonresidue modulo for both the prime factors of n. Therefore, we can know whether some specific numbers are quadratic nonresidues from the Jacobi symbol.

When discussing the quadratic residuosity assumption, we are interested in those elements $a \in Z_n^*$ that $(\frac{a}{n}) = 1$. Therefore, we have the following set:

$$Z_n^1 = \{x | x \in Z_n^* \text{and } (\frac{x}{n}) = 1\} \tag{3}$$

To formally state the intractability assumption of the quadratic residuosity problem, we will first introduce the predicate Q_n and the "hard set" H_k. For all $x \in Z_n^1$, the predicate Q_n is defined as:

$$Q_n(x) = \begin{cases} 1 & \text{if } x \text{ is a quadratic residue mod } n \\ 0 & \text{if } x \text{ is a quadratic nonresidue mod } n \end{cases} \tag{4}$$

On the other hand, H_k is defined as

$$H_k = \{n | n = p_1p_2, \text{ where } p_1, p_2 \text{ are } k/2\text{-bit primes}\} \tag{5}$$

Now, we are ready to introduce the quadratic residuosity assumption. Informally, quadratic residuosity assumption states that there is no polynomial-size circuits that can compute the predicate $Q_n(y)$ better than guessing. Formally,

Definition 2 (Quadratic Residuosity Assumption (QRA))**.** *For every constant and every family of polynomial-size circuits $C_k(.,.)$, there is an integer such that for all $k > K$*

$$Prob_{n \in H_k, y \in Z_n^1}(C_k(n, y) = Q_n(y)) < \frac{1}{2} + \frac{1}{k^c} \tag{6}$$

Based on QRA, we can present the Kushilevitz-Ostrovsky protocol for private information retrieval. It is a single database scheme whose communication complexity is $O(n^{0.5}k)$ where k is the security parameter. We will view the database x as a

$s \times t$ matrix of bits, denoted by M. The user wants to privately retrieve the bit x_i of the database, which is the (a, b) entry of the matrix M. The basic scheme will be divided into several parts.

1) The user randomly selects two $k/2$-bit prime numbers and then multiplies them to get the parameter N. The user then sends N to the database but keeps its factorization secret.
2) The user chooses uniformly at random t numbers $y_1, ..., y_t \in Z_n^1$ such that y_b is a quadratic nonresidue and y_j for all the other j that does not equal b, is a quadratic residue. It then sends these t numbers to the database.
3) The database, when receiving those t numbers, will compute for every row r a number $z_r \in Z_n^*$ as follows. It will first compute

$$w_{r,j} = \begin{cases} y_j^2 & \text{if } M_{r,j} = 0 \\ y_j & \text{if } M_{r,j} = 1 \end{cases} \tag{7}$$

then the database will compute,

$$z_r = \prod_{j=1}^{t} w_{r,j} \tag{8}$$

4) The database will send $z_1, ..., z_s$ to the user.
5) The user will only consider the number z_a. This number will be a quadratic residue if and only if $M_{a,b} = 0$. Since the user knows the factorization of the number N, it can efficiently check whether z_a is a quadratic residue and therefore get the bit $M_{a,b}$.

This protocol will enable the user to privately retrieve the information from the database with a reasonable time complexity $O(n^{0.5}k)$.

B. Homomorphic Encryption

Homomorphic encryption is an encryption scheme that enables specific types of computations to be carried out on ciphertexts, and obtains a new ciphertext, which can be decrypted to match the result of computations applied directly on the original plaintexts. In this paper, we will use Paillier's cryptosystem [14], which belongs to partially homomorphic cryptosystems. It supports computation of unlimited number of additions. The work is based on the computational problem called the Composite Residuosity Class Problem [14].

First, we define $L(u)$ as

$$L(u) = \frac{u-1}{n} \tag{9}$$

Then we set $n = pq$ where p and q are prime numbers and randomly choose a base g which can be done quickly by checking whether

$$gcd(L(g^{\lambda} \bmod n), n) = 1 \tag{10}$$

We publish (n, g) as the public parameters and the factorization (p, q) as the private key. The encryption and decryption scheme is described below:

Encryption: The plaintext is m, which is a number smaller than n. We first select a random $r < n$. Then we compute the ciphertext as $c = g^m \cdot r^n \bmod n^2$.

Decryption: The ciphertext is $c < n^2$, which is a number smaller than n^2. Then we compute the plaintext as $m = \frac{L(c^{\lambda} \bmod n^2)}{L(g^{\lambda} \bmod n^2)} \bmod n$ based on λ Carmichael's function taken on n.

The cryptosystem has the following properties:

Theorem 1. *For any $m_1, m_2 \in Z_n$ and any $k \in N$, we have*

$$D(E(m_1)E(m_2) \ mod \ n^2) = m_1 + m_2 \ mod \ n$$
$$D(E(m)^k \ mod \ n^2) = km \ mod \ n$$
$$D(E(m_1)^{m_2} \ mod \ n^2) = m_1 m_2 \ mod \ n$$

These properties enable us to do operations on the ciphertext instead of the plaintext, enabling the ad exchange to compare the bids without knowing them.

IV. System Design

In this section, we present POLA, which is a privacy-preserving protocol for location-based real-time advertising. We formalize the system model, security model and illustrate how the protocol works.

A. System Model

In our model, we consider a typical advertising pattern. There are three roles in the model: mobile users, advertisers and ad exchange. Note that we omit the role of the publisher which is the owner or controller of the web page, the user can directly communicate with the ad exchange without the involvement of the publisher for privacy concerns. The only job the publisher can do in this protocol is to notify the ad exchange of the ad slot on the web page which can be neglected in this protocol. Therefore, the role of publisher is omitted.

Mobile users: Mobile users roam in a given area and visit some web pages. The users can obtain their current locations with the help of GPS. Once a user visits some web page, the publisher of the web page will notify the ad exchange. After that, the user and the ad exchange can communicate with each other directly.

Advertisers: Advertisers are those who want to display their advertisements on the advertising slots and are willing to pay for them.

Ad exchange: Ad exchange brings together advertisers and users. It is responsible for finding the suitable ads for the web page and initiates the auction to get a winner. When the advertisement displayed on the user's web page is clicked, the advertiser will pay the ad exchange and the ad exchange will pay the publisher.

B. Security Model

In our security model, we mainly consider the privacy requirements of each role in our system model.

Mobile users: The location information of the user will not be learned by any advertiser or the ad exchange assuming that

the advertisers do not collude with each other. Even if a small number of advertisers collude with each other, the probability of the location information being leaked is still very low.

Advertisers: The advertiser's values of the ad slots, which is the advertiser's private information, will not be learned by anyone including the users and the ad exchange except the winner's charging price assuming that the ad exchange and the user do not collude with each other.

We note that this protection is stronger than k-anonymity [15] protection model. In the k-anonymity model, the user's location will not be distinguished from $k-1$ locations. The attackers still have the ability to decide the location sets that the user's location will be [16]. In our model, the location will be fully protected. Furthermore, we do not require a trusted third party to help us remove the identity information of the users.

C. Privacy preserving comparison

In order to secretly compare bids without permitting the ad exchange to pry out useful information, we have designed a light-weighted privacy preserving bid comparison protocol, which is used in the system. We will first present our protocol.

We consider two l-bit number $a = (a_l, a_{l-1}, ..., a_1)$ and $b = (b_l, b_{l-1}, ...b_1)$, where a_1 and b_1 denote the least significant bits, and a_l and b_l denote the most significant bits. For each $k \in \{1, 2, ..., l\}$, we define

$$\theta_k = a_k + b_k - 2a_k b_k \tag{11}$$

$$\tau_k = r_k(a_k - b_k + 1 + \sum_{t=k+1}^{l} \theta_t) \tag{12}$$

where $r_k \in Z^+$ is a random positive number. Then, we have the following lemma.

Lemma 2. *For any $a, b \in Z_n$, we have $a < b$ if and only if there exists exactly one $k \in \{1, 2, ..., l\}$, where $\tau_k = 0$.*

Proof. We will first prove that if $a < b$ then there exists exactly one $k \in \{1, 2, ..., l\}$, where $\tau_k = 0$. Next we will prove that if $a \geq b$, there is no $k \in \{1, 2, ..., l\}$ satisfying $\tau_k = 0$.

Case 1: $a < b$. First, notice that θ_k is the XOR function of a_k and b_k. When $a_k = 0$ and $b_k = 0$ or $a_k = 1$ and $b_k = 1$, $\theta_k = 0$. Otherwise, $\theta_k = 1$. Without loss of generality, we assume that

$$\forall j \in \{l, l-1, ..., i+1\}, a_j = b_j \text{ and } a_i < b_i$$

for some i. Consider θ_k for $k \in \{l, l-1, ..., 1\}$.

- When $k > i$, from the definition of θ_k, we know that $\theta_k = 0$. Therefore, $\tau_k = r_k \neq 0$.
- When $k = i$, we have $a_k - b_k + 1 = 0$ and $\sum_{t=k+1}^{l} \theta_t = 0$. Therefore, $\theta_i = 0$.
- When $k < i$, because $\theta_i = a_i + b_i - 2a_i b_i = 1$, we have $\sum_{t=k+1}^{l} \theta_t > 0$. Therefore, $\tau_k \geq r_k(\sum_{t=k+1}^{l} \theta_t) > 0$

The proof of case 1 is over.

Case 2: $a \geq b$. When $a = b$, it is obvious that the claim stands. We assume that

$$\forall j \in \{l, l-1, ..., i+1\}, a_j = b_j \text{ and } a_i > b_i$$

Consider θ_k for $k \in \{l, l-1, ..., 1\}$.

- When $k > i$, from the definition of τ_k, we know that $\theta_k = 0$. Therefore, $\tau_k = r_k \neq 0$.
- When $k = i$, we have $a_k - b_k + 1 = 2$ and $\sum_{t=k+1}^{l} \theta_t = 0$. Therefore, $\theta_i \neq 0$.
- When $k < i$, because $\theta_i = a_i + b_i - 2a_i b_i = 1$, we have $\sum_{t=k+1}^{l} \theta_t > 0$. Therefore, $\tau_k \geq r_k(\sum_{t=k+1}^{l} \theta_t) > 0$

In this step, our proof is over. □

Note that when computing the encryption of τ_k, we do not need to know the value of a_k and b_k. By using homomorphic encryption, we just need the encryption of a_k, b_k and $a_k b_k$. When we decrypt the value of τ_k, we can know whether a or b is larger. Yet we have no idea what the a and b is. This motivates our design.

D. Design Details

The protocol is divided into four phases.

1) Initialization: Once the user visits the web page, the user and the ad exchange will set up the parameters. The user will first pick randomly a k-bit number $n_1 \in H_k$ where k is the security parameter and send it to the ad exchange, while keeping the factorization of n_1 secretly. After receiving the number n_1, the ad exchange will choose a k-bit number $n \in H_k$ and randomly select a base g. After that, the ad exchange will publish (n, g) as public parameters and keep the factorization of n secretly. Denote the Paillier encryption function to be $E(a)$ and the decryption function to be $D(a)$. Every time we use the encryption function, we will select a new random number r by default. As the value of this random number is not important, we will neglect it in our notation for convenience.

2) Bidding: When a mobile user visits the web page with an advertisement slot, the user can get the current location as (x, y) which can be regarded as the coordinate of the location. Then the publisher will launch a request of advertisement and send it to the ad exchange. After receiving the request from the publisher, the ad exchange contacts the interested advertisers. Every advertiser will generates a $s \times t \times l$ matrix M. The element $(i, j, k) \in M$ represents the k-th bit of the value profile that the advertiser has for the position (i, j). Note that $(i, j, 1)$ is the least significant bit for the profile and (i, j, l) is the most significant bit. Then, each advertiser will generate l random bijection mappings for $1 \leq k \leq l$

$$f_k : Z_2 \mapsto Z_2$$

Then the ad exchange will act as the intermediary between the user and the advertisers. The current location of the user is (x, y). The ad exchange will choose uniformly at random t numbers $y_1, ..., y_t \in Z_{n_1}^1$ such that y_b is a quadratic nonresidue

and y_j for all the other j that does not equal to b is a quadratic residue just as what we have discussed in Section III.

The advertiser, when receiving those t numbers, will compute for every row r and every bit k of the value profile a number $z_{r,k} \in Z_{n_1}^*$ as follows. It will first computes

$$w_{r,j,k} = \begin{cases} y_j^2 & \text{if } f_k(M_{r,j,k}) = 0 \\ y_j & \text{if } f_k(M_{r,j,k}) = 1 \end{cases} \tag{13}$$

and then the advertiser will compute,

$$z_{r,k} = \prod_{j=1}^{t} w_{r,j,k} \tag{14}$$

Next, the advertiser will in sequence send to the user via the ad exchange

$$\begin{aligned} &z_{1,1}, z_{2,1}, ..., z_{s,1}, E(f_1^{-1}(0)), E(f_1^{-1}(1)) \\ &z_{1,2}, z_{2,2}, ..., z_{s,2}, E(f_2^{-1}(0)), E(f_2^{-1}(1)) \\ &... \\ &z_{1,l}, z_{2,l}, ..., z_{s,l}, E(f_l^{-1}(0)), E(f_l^{-1}(1)) \end{aligned}$$

Note that the ad exchange has the ability to use the encryption function. When the ad exchange receives the ciphertext such as $E(f_1^{-1}(0))$, it will do the permutation

$$\begin{aligned} \text{Ciphertext} &= E(f_1^{-1}(0)) \times E(0) \\ &= E(f_1^{-1}(0) + 0) = E(f_1^{-1}(0)) \end{aligned}$$

After the permutation, the ad exchange will send the sequences listed above to the user.

The reason why the advertiser multiply the ciphertext sent by the advertiser by $E(0)$ is to prevent the user from recognizing the advertiser. If some advertiser colludes with the user, then the user might recognize the advertiser through the message the advertiser sent to the user. After the permutation, the user will not be able to recognize the advertiser. Meanwhile, the value of the ciphertext the user send will not change.

After receiving the sequences generated by the advertisers, the user considers only the number $z_{x,k}$ for $1 \leq k \leq l$ where x is the first coordinate of the user's current location. The numbers are quadratic residues mod n_1 if and only if $f_k(M_{x,y,k}) = 0$ where x, y are exactly the user's location. So if $z_{x,k}$ is a quadratic residue mod n_1, which can be easily verified by the user with the factorization of n, the user will know that the encryption of the k-th bit of the value profile $M_{x,y,k}$ will be

$$E(f_k^{-1}(0)) = E(f_k^{-1}(f_k(M_{x,y,k})) = E(M_{x,y,k}) \tag{15}$$

Similarly, the user will also know the encryption of the value as $E(f_k^{-1}(1))$ when $z_{x,k}$ is a quadratic nonresidue mod n_1.

Note that in this case, the user does not know the value profile of the advertiser because the user has no knowledge of f_k and the decryption function. The only thing he can confirm is the encryption of the ad value which is related to the user's location. On the other hand, the ad exchange cannot get knowledge of the value profile of the advertiser, since he cannot decide whether $z_{x,k}$ is a quadratic residue mod n_1 or not effectively without the factorization of n_1, though he knows f_k and the decryption function. Therefore, the privacy of the advertisers is protected. It is also obvious that the user's privacy is protected in this case, because neither the ad exchange nor the advertisers know what x and y is.

3) Comparing: So far, the user receives the encryption of all the value profiles of advertisers. Now he is ready to compare those profiles. The comparing process will proceed at most $2m-3$ rounds, where m is the number of advertisers. In the first round, the user will randomly select two advertisers a and b and notify his selection to the ad exchange. In the bidding phase, the ad exchange already knew the encryption of all the bits of the value profile. Denote the k-th bit to be a_k and b_k, when the ad exchange receives a and b from the user, it will send back to the user the following sequence which can be computed effectively by the properties of homomorphic encryption:

$$\begin{aligned} &E(-2f_{a,k}^{-1}(0) \times f_{b,k}^{-1}(0)), E(-2f_{a,k}^{-1}(0) \times f_{b,k}^{-1}(1)), \\ &E(-2f_{a,k}^{-1}(1) \times f_{b,k}^{-1}(0)), E(-2f_{a,k}^{-1}(1) \times f_{b,k}^{-1}(1)). \end{aligned}$$

for $1 \leq k \leq l$ where $f_{a,k}^{-1}$ is the inverse function of advertiser a's k-th bit random bijection.

After receiving the data, the user can compute the encryption of θ_k referred in section IV-C as

$$E(\theta_k) = E(a_k + b_k - 2a_k b_k) = E(a_k) \times E(b_k) \times E(-2a_k b_k) \tag{16}$$

Then, the user computes τ_k as

$$\begin{aligned} E(\tau_k) &= E(r_k(a_k - b_k + 1 + \sum_{t=k+1}^{l} \theta_t)) \\ &= E(a_k)^{r_k} \times E(b_k)^{-r_k} \times E(1)^{r_k} \times \prod_{t=k+1}^{l} E(\theta_t)^{r_k} \end{aligned} \tag{17}$$

Then the user will send $E(\tau_k)$ for $1 \leq k \leq l$ in permutation to the ad exchange. The ad exchange then decrypts the ciphertext. If it finds 0 in one of the plaintexts, the ad exchange knows that $a < b$. Otherwise, the ad exchange knows that $a \geq b$, which will be sent back to the user. In the next rounds, the user will choose the larger one got in the previous round and compare it with other value profiles. It is easy to see that within at most $2m - 3$ rounds, we can know the advertisers with the largest value and the second largest value.

4) Opening: In this phase, the user knows the advertiser with the largest value and the advertiser with the second largest value. She then will send the encryption of every bit of the second largest value to the ad exchange and display the advertisement of the highest bid. After receiving the value, the ad exchange can decrypt the value with the decryption function and charge the corresponding advertiser when the advertisement is clicked. Also, the ad exchange will notify the publisher of the winner and pay the publisher when the ad is clicked.

V. Security Analysis

In the previous section, we propose a privacy-preserving protocol for location-based real-time advertising. In this section, we will discuss the security of the whole system and prove that our system successfully achieves the four design goals we brought about in Section I. In the discussion, we can see why every component in the system is necessary in preventing the possible attacks.

A. Utilization of User's Location Information

For the first property of the system, we claimed that the advertisers can use the user's location when deciding the value of the slots on the user's web page without revealing the users' location privacies. From the design of the system, we can see that the user will retrieve the advertiser's value on the user's location based on the theory of private information retrieval. Afterwards, the user will start the protocol with the ad exchange to decide which bid is larger. Therefore, the bids the ad exchange uses to decide the winner are the values the advertiser has on the specific location. So the advertiser uses the location when deciding the value which satisfied the basic requirement of location-based advertising.

B. Privacy Analysis of the Users' Location Information

In the proposed POLA protocol, the location will not be learned by anyone including the advertisers and the ad exchange assuming that the ad exchange and the advertisers do not collude with each other. Actually, our system is robust against slight collusion unless the ad exchange colludes with most advertisers to fabricate the whole auction which will be verified later.

In the bidding phase, what we achieve is that the user can get the encryption of the advertisers' values on their current location without letting the advertisers know their current location. We can construct a transformation function to show that we can reduce the assumption that the advertiser can distinguish different queries to an algorithm that can effectively compute the quadratic residuosity predicate. Due to the limitation of the space, we omit the proof here.

Now we will consider the privacy of the users with the possible collusion between advertisers and the ad exchange.In the bidding phase, as we proved before, the location of the user will not be learned by anyone even if ad exchange colludes with all the advertisers. The only information the ad exchange can get from the user is the relationship of values from different advertisers and the second largest value, which is the payment of the winner. One way that the location information is leaked is through the relationship of values from different advertisers. As we said before, there are $s \times t$ different locations. And n colluded advertisers can identify $n!$ different locations by their value relationship. In our case, there are at least 10^6 different locations, which require at least 9 advertisers to collude with each other. Yet, those advertisers' relationship will not be easily known because the number to be compared is decided by the user instead of the ad exchange. Once a number is compared to be less than another, it will not be compared again. Therefore, the probability that the advertiser can accurately determine the relationship between all these 9 advertisers will be very small. Another attack for the ad exchange is to let the advertiser bid the highest in each location. However, this requires at least $s \times t$ advertisers, which is not realistic in our case.

C. Privacy Analysis of the Advertisers' Values

The advertisers' values, which are advertisers' private secrets, will not be learned by anyone including the users and the ad exchange except the winner's charging price assuming that the ad exchange and the user do not collude with each other. As we can see from the protocol, there are two levels of cryptography tools needed to know the value of the advertiser. The value is first encrypted by the Paillier's homomorphic encryption cryptosystem. Then the value is further encrypted using the quadratic residuosity assumption. Therefore, in order to know the value, the attacker should first have a way to decide whether the number the advertiser sends back to the user is a quadratic residue or not. Only after he successfully decides whether the ciphertext is a quadratic residue or not can he gain the right homomorphic encryption the value. He can obtain the homomorphic encryption of the value from $E(0)$ and $E(1)$ the ad exchange sends to the user. Note that the ad exchange cannot always send the homomorphic encryption of the values in order, i.e., ad exchange first sends $E(0)$ and then sends $E(1)$ as in this case, the user will know the value by the order of the ciphertext instead of breaking the ciphertexts. Therefore, we add a random bijection $Z_2 \mapsto Z_2$ to randomly send the order $E(0)$ and $E(1)$. Therefore, the user has to know the decryption function to decrypt the right homomorphic encryption. Note that this mapping is clear to the ad exchange because the ad exchange can decrypt the mapping and know the mapping. Then the user should have a way to decrypt the ciphertext got by the Paillier's cryptosystem to get the value. We know that the user starts the Kushilevitz-Ostrovsky protocol with the advertiser to privately get the encryption of the data. Therefore, only the user can break this level of encryption. And the user has no benefits to give his private key to others because this protocol also protects his location information from leaking out to other entities. What's more, the user has no idea of the private key of the homomorphic encryption system. Therefore, he cannot decrypt the ciphertext to know the exact value of the bid. We have shown before that the user cannot use other methods rather than decrypt the ciphertext because $E(0)$ and $E(1)$ is sent in random order to him. Therefore, as long as the user and the ad exchange don't collude with each other, the advertisers' values are safe.

D. No one can gain profits by manipulating the auction

The users and the advertisers cannot gain utilities by colluding with each other to manipulate the auction. In this protocol, we assume that the users represent the benefits of the publisher, which is a rational speculation. It is quite easy for the user to manipulate the auction as only the user, other than the advertiser, knows the encryption of the value and all know the encryption function. The user could easily fabricate the

value of some advertisers simply by replacing the ciphertext with the ciphertext the user generates by himself. However, we will show that the user has no way to associate the identity of the advertiser with its value profile, therefore providing no benefits to him if he manipulates the auction. One method to associate the identity with the value profile is through the order the ad exchange sends the ciphertexts to the user. However, the ad exchange will permute the order of different advertisers before sending them to the user. Therefore, there is no way that the advertisers' values can be associated with their identities. Another way is by using the content of the ciphertext. If the ad exchange does not change the ciphertext, then the user and the advertiser can communicate with each other to know the ciphertext the advertiser sends. Therefore, the user can identify the advertiser by the ciphertext. However, in our design, all the ciphertexts the advertisers send to the user will be modified by the ad exchange. The ad exchange will add the number by 0, which will not change the value but change the ciphertext. Therefore, the user cannot learn any more information from the ciphertext. Hence, as the user cannot know the identity of each profile, there is no benefit for him to manipulate the auction.

In conclusion, all the four goals are met in our design.

VI. Evaluation

We have implemented our design using the GMP library. In this section, we discuss how we implement our system. Furthermore, we show the computation time needed by the system in bidding phase and in comparing phase. Finally, we show the communication cost of our system.

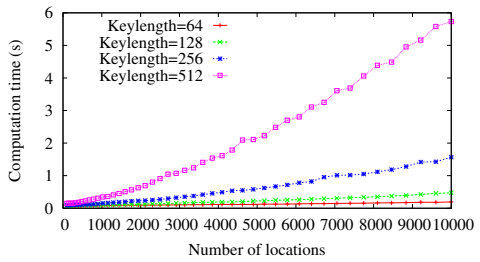
A. Methodology

We implemented our system as we described in our system design. The environment we use is Linux Ubuntu and we use a two-core 1.8GHz CPU. This type of CPU is rather slow, which is slower than common laptops by a factor of 4. If we compare the computation power with a server, the factor might increase to 10 or more. So our system should be better in reality than what the experiment suggests. We divide the locations into a $s \times s$ matrix. Each entry represents a specific location. The security parameter is represented by *KeyLength*. The value length is represented by l. The number of advertisers is represented by *adv*.We implemented our system using C language based on the usage of the GMP library.

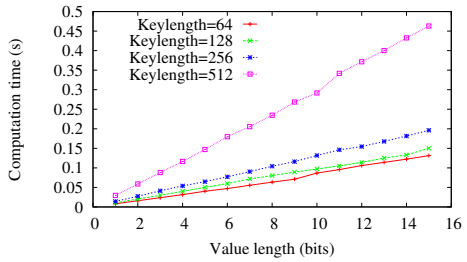
The security of the system is described before. Therefore, the focus of our simulation is on the efficiency. We will divide the analysis into three parts, namely, the computation time needed for the bidding phase, the computation time needed for the comparing phase and the communication cost of the system.

B. Simulation Results

Fig. 1 shows the computation time needed for the bidding phase when set to different location numbers and value lengths under different *KeyLength*. Fig. 1(a) describes the relationship between the computation time and the location number when l is set to 10. Obviously, the computation time is not very large



(a) Computation time with different number of locations



(b) Computation time with different lengthes of values

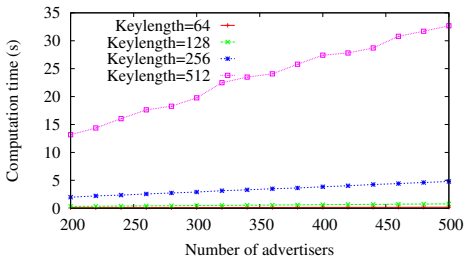
Fig. 1. Computation time in bidding phase.

which can be easily handled by most machines. Considering the most extreme situation where the security parameter is set to 512 and the location number is set to 10000, the computation time is still less than $6s$. Since the computation time is computed as one advertiser and computation for different advertisers can be pipelined, the whole system can be considered as efficient.

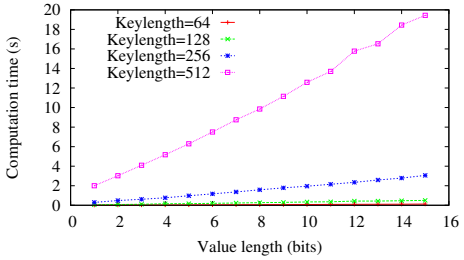
Fig. 1(b) shows the relationship between the computation time and the value length. It can be easily observed that the computation time is linear to the value length under each *KeyLength*. Usually, 10-bit value is really more than abundant for an advertiser. Therefore, we can decrease the length of the value by half, which means there are 32 values to choose from, to increase the efficiency.

Fig. 2 illustrates the computation time needed for the comparing phase. Fig. 2(a) describes the relationship between the computation time and the number of advertisers when l is set to 10. We can see that the computation is linearly associated with the number of advertisers and the computation time is also linear regarding the value length from Fig. 2. Therefore, cutting the value length by half can dramatically decrease the computation time. The number of advertisers is not something that should be decreased. However, it is showed that there are not much advertisers in reality, implying that 500 advertisers are suitable in most situations.

Fig. 3 is the communication cost regarding the number of locations and the number of advertisers measured in KB. The communication cost in the bidding phase will be $k \times l \times (s + 2) \times adv$, which in the comparing process will range from $(adv-1)\times k\times (l+4)$ to $(2adv-3)\times k\times (l+4)$ because the user chooses the value to be compared randomly. From the figure we can see that with the increment of the number of locations, the communication cost does not increase sharply. However,

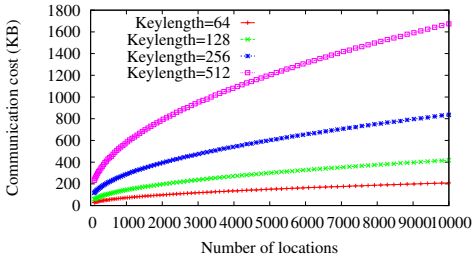


(a) Computation time with different number of advertisers

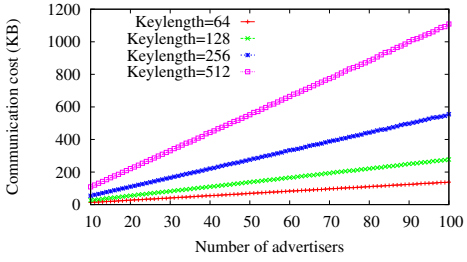


(b) Computation time with different lengthes of values

Fig. 2. Computation time in comparing phase.



(a) Communication cost with different number of locations



(b) Communication cost with different number of advertisers

Fig. 3. Communication cost

the threshold of the communication complexity is very large. In other words, the cost will be rather large even when there is a small number of locations. This is due to the inherent limitation of the private information retrieval scheme and the homomorphic encryption scheme. However, we believe that the complexity is enough for the current use.

In our evaluation, the max key length is set to 512. However, in today's frameworks, the key length usually have a length of 1024 even 2048. The observation we made here is that the key length is enough to block out the curious users. The location of a user is not a top security that will make a big significance to the society. There are millions of users and if we can break the ciphertexts using supercomputer within 2 hours (which means that it is unsecure), we still need much time to associate the location information with the user's identity. The cost of breaking is so much and the attackers can just fabricate an auction to pry out the user's location which is much more simpler. In a word, our system blocks out curious ad exchanges but not malicious ones.

VII. Conclusion

In this paper, we have proposed a privacy-preserving protocol for location-based real-time advertising, called POLA. The proposed POLA protocol utilizes private information retrieval and homomorphic encryption to protect the privacy of both users and advertisers without the involvement of a trusted-third party. Detailed security analysis shows that our protocol can protect the privacy under possible attacks and collusion which makes the protocol much stronger. Finally, the evaluation shows that the overhead in this protocol is acceptable which means that this protocol is feasible in real-time advertising.

References

[1] A. Juels, "Targeted advertising... and privacy too," in *Topics in Cryptology–CT–RSA 2001*, pp. 408–424, Springer, 2001.
[2] V. Toubiana, A. Narayanan, D. Boneh, H. Nissenbaum, and S. Barocas, "Adnostic: Privacy preserving targeted advertising," in *NDSS*, 2010.
[3] S. Guha, B. Cheng, and P. Francis, "Privad: practical privacy in online advertising," in *NSDI*, 2011.
[4] A. Reznichenko, S. Guha, and P. Francis, "Auctions in do-not-track compliant internet advertising," in *CCS*, 2011.
[5] M. Götz and S. Nath, "Privacy-aware personalization for mobile advertising," in *CCS*, 2012.
[6] S. Guha, M. Jain, and N. Padmanabhan, Venkata, "Koi: A location-privacy platform for smartphone apps," in *NSDI*, 2012.
[7] R. Lu, X. Lin, Z. Shi, and J. Shao, "Plam: A prvacy-preserving framework for local-area mobile social networks," in *INFOCOM*, 2014.
[8] S. Angel and M. Walfish, "Verifiable auctions for online ad exchanges," in *SIGCOMM*, 2013.
[9] E. Kushilevitz and R. Ostrovsky, "Replication is not needed: Single database, computationally-private information retrieval," in *FOCS*, 1997.
[10] D. Boneh, E.-J. Goh, and K. Nissim, "Evaluating 2-dnf formulas on ciphertexts," in *Theory of cryptography*, pp. 325–341, 2005.
[11] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *Journal of the ACM*, vol. 45, no. 6, pp. 965–981, 1998.
[12] S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of computer and system sciences*, vol. 28, no. 2, pp. 270–299, 1984.
[13] S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of Computer and System Sciences*, vol. 28, no. 2, pp. 270–299, 1984.
[14] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *CRYPTO–EUROCRYPT*, 1999.
[15] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002.
[16] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," in *ICDE*, 2006.