A Practical System for Privacy-Aware Targeted Mobile Advertising Services

Jinghua Jiang, Yifeng Zheng, Zhenkui Shi, Xingliang Yuan, XiaoLin Gui, and Cong Wang

Abstract—With the prosperity of mobile application markets, mobile advertising is becoming an increasingly important economic force. In order to maximize revenue, ads are recommended to be delivered to potentially interested users, which requires user targeting, i.e., analyzing users' profiles and exploring users' interests. However, collecting user personal information for targeted mobile advertising services raises critical privacy concerns. Although some solutions like anonymization and obfuscation have been proposed for privacy-aware targeted advertising, they undesirably face the issues of security, efficiency, and/or ad relevance. In this paper, we propose a practical system enabling secure and efficient targeted mobile advertising services. It allows the ad network to perform accurate user targeting, while ensuring strong privacy protection for mobile users. Specifically, we show how to properly leverage a cryptographic primitive called private stream searching to support secure, accurate, and practical targeted mobile ad delivery. Moreover, we propose secure billing schemes to enable the ad network to charge advertisers in a privacy-preserving manner. The security strength of our system is thoroughly analyzed. Through extensive experiments, we show that our system achieves practical efficiency on mobile devices.

Index Terms—Targeted mobile advertising services, private ad delivery, private ad billing, user privacy, mobile computing

1 Introduction

Mobile phones are ubiquitous today because of their greatly improved capabilities in computation, communication, and sensing [2], [3], etc. According to the report of ComScore [4], the number of global mobile users exceeded the number of global PC users in 2014. The booming smartphone market attracts more and more developers to create mobile apps [5]. For mobile apps, mobile advertising is the major revenue source, especially for those free ones. To maximize revenue, ads need to be delivered to potentially interested users. Targeted advertising has been deployed in commercial mobile advertising systems like AdMob [6], iAd [7], and Microsoft Mobile Advertising [8].

Targeted mobile advertising services rely on personal information of users to infer users' habits, activities, social relations, etc [9]. In current mobile advertising systems, device information and location are usually collected to ad networks for user targeting [10]. In order to fully exploit the potential of targeted mobile advertising, more user personal information are expected to be utilized by the ad network, such as keywords and demographics [10]. However, collecting such private user data for accurate targeting undesirably poses security and privacy risks: the collected data might allow the identification of the user [11]; the ad network may sell private user data to other parties [10] for profit; a curious insider working at the ad network could peek at the private user data [12]. Therefore, addressing the privacy issue is pivotal for targeted mobile advertising services.

In the literature, several approaches have been proposed for privacy-aware targeted advertising services. Roughly speaking, there are mainly three kinds of approaches. The first one is to let the ad network send a certain set of ads to the user device which then performs targeting locally (e.g., [13], [14]). The main limitation of such an approach is the high communication cost because the number of ads sent to the user for local targeting could be very likely large. The second one is to leverage intermediate party/trusted hardware to anonymize user data and operations (e.g., [15], [16]). However, anonymization based approaches may be prone to re-identification attacks [17], [18]. The third one is to adopt obfuscation techniques to protect user privacy (e.g., [11], [19], [20]). Nevertheless, this line of approaches suffers from low targeting accuracy and would force irrelevant ads to be delivered to users. To our best knowledge, privacy-preserving targeted advertising is still challenging, and to design a secure, practical, and targeted advertising system, especially for mobile devices with limited resources [21], [22], remains to be fully explored.

In this paper, we explore a new privacy-aware system architecture for targeted mobile advertising services, which aims for accurate and practical mobile user targeting with strong privacy protection. Our system aims to address the following challenges simultaneously: 1) security, i.e., how to

- *J. Jiang is with the School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China, and the Department of Computer Science, City University of Hong Kong, Hong Kong, China. E-mail: jjinghua2-c@my.cityu.edu.hk.*
- *Y. Zheng, Z. Shi, X. Yuan, and C. Wang are with the Department of Computer Science, City University of Hong Kong, Hong Kong, China, and the City University of Hong Kong Shenzhen Research Institute, Shenzhen 518057, China. E-mail: {yifeng.zheng, zhenkui.shi}@my.cityu.edu.hk, xyuancs@gmail.com, congwang@cityu.edu.hk.*
- *X. Gui is with the School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China. E-mail: xlgui@mail.xjtu.edu.cn.*

Manuscript received 14 June 2016; revised 13 Feb. 2017; accepted 10 Apr. 2017. Date of publication 24 Apr. 2017; date of current version 12 June 2020. (Corresponding authors: Xiaolin Gui and Cong Wang.)
Digital Object Identifier no. 10.1109/TSC.2017.2697385

well protect user private information, 2) ad relevance, i.e., how to make sure that the ads delivered to the user are relevant to her interests, and 3) efficiency, i.e., how to make the system efficient in terms of bandwidth, computation, and energy, especially for the resource-limited mobile devices. We first consider how to achieve secure and practical targeted ad delivery. In particular, we leverage the cryptographic technique called private stream searching (PSS) [23], [24] as our basis. With PSS, the untrusted ad network is allowed to process encrypted user profiles over the ads, and return encrypted matched ads without knowing the underlying contents. The ad network only learns which user it interacts with due to the absence of anonymous server, but it learns nothing about users' profiles and the matched ads.

However, as directly applying PSS does not enable a practical system design, we further present several optimization techniques to make our system suitable and practical for mobile devices. This includes representing the user preferences with hierarchical structure and leveraging ads auction to reduce bandwidth cost, utilizing prefetching and caching to minimize ad loading latency, and properly narrowing the search range of matched ads to save more computation for the ad network. The practical performance of private targeted mobile ad delivery in our system is demonstrated via comprehensive experiments. To our best knowledge, no prior works on private targeted ad delivery have tackled all the challenges simultaneously as we did.

In addition to ad delivery, billing is also an essential part of targeted mobile advertising service systems [25]. Therefore, we then consider how to support privacy-preserving billing in our system. Our practical system enables the ad network to record the number of views for each ad accurately, but know nothing about which ad was viewed by a particular user. Similar to prior arts [13], [15], our system design resorts to a third party, which is assumed not to collude with the ad network. Our system design requires the third party and the ad network to jointly calculate the number of views for each ad, while the ad network will not learn which ads a particular user viewed and the third party will not learn the number of views per ad. Our contributions can be summarized as follows:

- We propose a new privacy-preserving system architecture for secure and efficient targeted mobile advertising services, as well as deliver the implementation of a system prototype deployed on real mobile device to demonstrate the practical performance.
- We show how to leverage the PSS technique properly to deliver targeted ads to users while protecting their private interests. We also further present several practical optimization mechanisms such as categorized user interests, prefetching and caching, and ads auction, to make our system mobile-friendly.
- We propose secure and efficient billing schemes based on homomorphic cryptosystem, enabling the ad network to correctly charge the advertisers while concealing user's ad view history.

The rest of the paper is organized as follows. Section 2 presents our problem statement. Section 3 gives some preliminaries. Section 4 presents the design of secure and efficient targeted mobile ad delivery in our system. Section 5 gives our design of privacy-preserving billing. Section 6 presents the experiments. Section 7 describes the related work. Section 8 concludes the whole paper.

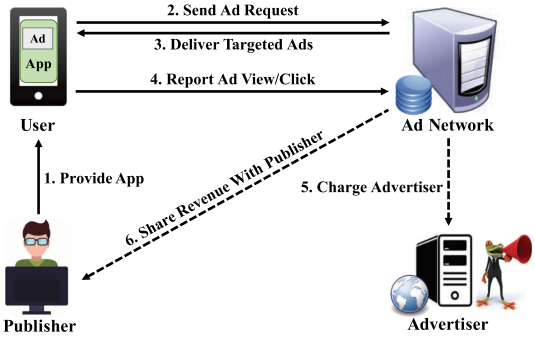


Fig. 1. The service flow of current targeted mobile advertising systems.

2 Problem Statement

2.1 Targeted Mobile Advertising

Targeted mobile advertising service systems such as AdMob [6] contain four major entities, namely, the user, the advertiser, the publisher, and the ad network, as illustrated in Fig. 1. We describe them in detail below.

- *User:* a party who downloads a mobile application and installs it on her mobile device. Afterwards, ads are served for the user and the user may click on her interested ones.
- *Advertiser:* a party who expects to deliver targeted ads to users across mobile applications, and is willing to pay for this service.
- *Publisher:* a party who releases mobile apps and is willing to place ads in some assigned screen "real estate", and expects to be paid for this service.
- *Ad network:* a party who connects the advertiser and the publisher. The ad network collects ads and their metadata from advertisers and delivers targeted ads to users via registered mobile apps of publishers.

To work with a particular ad network, the publishers register their apps at the ad network and embed an ad library within the apps. Hereafter, we refer to the ad library as the client. The client enables the ad network to collect the user's personal information, such as device properties, location, demographics, and interested keywords.

When an application is launched by the user, the client interacts with the ad network. Based on the collected user's behavioral data, the ad network selects relevant ads for that user. When the ads are received, the client selects one or several most relevant ads to display. When a user clicks on an ad, she will be redirected to the ad loading page served by the advertiser. Meanwhile, the ad network tracks the ad view/click conversation for future targeting and collects the view/click report for billing. At the end of a billing cycle, the ad network charges the advertiser and shares revenue with the publisher. Typically, the advertiser will be charged either for each ad view in the "charge-per-view" model or for each ad click in the "charge-per-click" model.

2.2 Adversary Model

Throughout the paper, we consider a semi-trusted ad network as the adversary in our system, which is consistent

with existing works on secure advertising [13], [15], [19], [26]. The ad network delivers targeted ads to users honestly, yet is curious in learning private information of the users, by exploiting the information/leakage it may obtain from the interactions with users. Specifically, we consider the following information that the user may wish to keep private: source data that are used for profiling, behavioral profile, and ad view history (i.e., a list of ads that have been displayed to the user). The ad network expects to collect the user's private information as much as possible either for performing targeting accurately, or for other purposes, such as selling the information to other parties. We do not consider the threats of micro-targeted ads [27] and malicious ads [28]. The works defending against these threats consider different adversary models and are complementary to ours.

2.3 Design Goals

Our ultimate goal is to design a practical system for privacy-aware targeted mobile advertising services. In particular, we have the following design goals:

- *Privacy*. The user private information should be protected against the ad network. In particular, our system aims to enable the ad network to provide relevant ads with no knowledge of the user's source data, behavioral profile, and ad view history. We do not hide the ad click history from the ad network for the following reason: all the ad clicks are visible to the advertisers, who are highly motivated to share such information with the ad network, so as to improve the quality of ads placement. Therefore, even if we hide the ad clicks from the ad network, it will probably derive such information anyhow from the advertisers. We note that similar treatment exists in [13].
- *Utility*. For the user, the delivered ads should be relevant to her interests. For the ad network, its goal is to display the ads which can maximize the expected revenue. The ad network delivers the ads according to the relevance and the bidding information of the ads [6]. In addition, the ad network should be able to charge the advertisers correctly according to the ad view reports.
- *Efficiency*. Targeted mobile advertising service system should be efficient in terms of bandwidth, computation, and energy. Due to the limited battery and expensive cellular network, mobile users expect that ads can be loaded with as less bandwidth and computation consumption as possible. Besides, the system should also minimize the latency of ad fetching which highly affects the user experience.

3 Preliminaries

Private Stream Searching. Private stream searching is first introduced by Ostrovsky et al. [23] and formally described by Bethencourt et al. [24], [29]. PSS allows a client to retrieve files matched with some search criteria from an untrusted server, while preventing the server from learning the search criteria and the results. The PSS system consists of four main algorithms: KeyGenerate, QueryConstruct, PrivateSearch, and FileReconstruct. The definitions of the algorithms are described below:

- $(K_{pub}, K_{priv}) \leftarrow \mathsf{KeyGenerate}(\lambda)$. This algorithm is run by the client to generate a public key and a private key. It takes as input a security parameter λ and outputs a key pair: a public key K_{pub} and a private key K_{priv}.
- $Q \leftarrow \mathsf{QueryConstruct}(K_{pub}, \epsilon, m, W, D)$. This algorithm is run by the client to prepare an encrypted query for search. It takes as input a public key K_{pub}, a correctness parameter ϵ, an upper bound on the number of matched files m, a set of plaintext keywords W, and a public dictionary of keywords D. Here, the correctness parameter ϵ is used to select various algorithm parameters and the upper bound m is used to initial the size of the buffer of matched ads, so as to ensure the matched files will be correctly retrieved with high probability. The set of plaintext keywords is the search criterion, and the public dictionary of keywords contains all possible searching keywords. This algorithm outputs an encrypted query Q.
- $R \leftarrow \mathsf{PrivateSearch}(K_{pub}, Q, FS)$. This algorithm is run by the server to perform search over a stream of files based on the encrypted query. It takes as input a public key K_{pub}, an encrypted query Q, and a stream of files along with the corresponding keyword sets, i.e., $FS = \{(f_1, W_1), (f_2, W_2), \ldots, (f_t, W_t)\}$, and outputs an encrypted buffer of matched files R.
- $F \leftarrow \mathsf{FileReconstruct}(K_{priv}, R)$. This algorithm is run by the client to reconstruct matched files from the encrypted buffer R. It takes as input an encrypted buffer R and a private key K_{priv}, and outputs a set of matched files $F = \{f_i \big| |W \cap W_i| > 0\}$.

There are two assumptions in the PSS scheme [24]. First, a public dictionary of potential keywords D is assumed to be available. Second, the user is assumed to be able to estimate the number m of files which match the query. We refer readers to [24] for more details about the algorithms.

4 Private Targeted Mobile Ad Delivery

In this section, we propose our design of private targeted mobile ad delivery. For ease of presentation, we focus on how to enable the ad network to deliver targeted ads to the users in a secure and efficient way, and defer the system design of private billing to the next section. We first present our basic design, analyze its security guarantees, and discuss the limitations. Then, we propose our main design, which takes into account both security and efficiency.

4.1 Notations

Before giving our basic system design of secure targeted mobile ad delivery, we first introduce some notations in a mobile advertising service system. In particular, we write D to denote a fixed public dictionary, which contains all attribute values and is assumed to be available for clients and the ad network. We denote by P the user profile which is represented as a set of attributes such as user preferences, demographics, and locations, i.e., $P = \{A_1, A_2, \ldots, A_\alpha\}$. And the ith attribute A_i has β keywords, i.e., $A_i = \{a_1, a_2, \ldots, a_\beta\}$. This reflects the status of the user under different attributes, e.g., user preferences, age, location. For example, the attribute "user preferences" may contain several keywords to

Ad Request Construction:

1. The client estimates the upper bound m on the number of matched ads.
2. The client constructs the encrypted ad request: $Q \leftarrow \mathsf{QueryConstruction}(pk_u, \epsilon, m, P, D)$.

Private Ad Searching:

1. The ad network initializes a buffer R with encrypted zeros under the public key pk_u.
2. The ad network performs private searching on all ads and produces an updated encrypted buffer: $R \leftarrow \mathsf{PrivateSearch}(pk_u, Q, AD)$.

Ads Reconstruction:

1. The client reconstructs matched ads from the encrypted buffer: $M \leftarrow \mathsf{FileReconstruction}(sk_u, R)$.
2. The client calculates the matching degree d between the user profile P and the profile of each reconstructed ad ad_i P_i: For each attribute $A_j \in P$ and $A'_j \in P_i$, the client first calculates the matching degree d_j=1 when $A_j \cap A'_j \neq \varnothing$, otherwise d_j=0; then the average matching degree d=$\mathsf{Average}(d_1, d_2, ..., d_\alpha)$.
3. The client ranks all the matched ads M according to the matching degrees, and displays the highest ranking one.

Fig. 2. The workflow of private targeted mobile ad delivery in our basic design.

represent user interests, such as computer, security, system; the attribute "location" may contain one keyword to represent user's current location, such as China, United States, and Australia. We denote by AD the set of all ads and corresponding targeting profiles, i.e., $\{(ad_1, P_1), (ad_2, P_2), \ldots, (ad_n, P_n)\}$. We use M to denote the set of matched ads which are reconstructed from the returned encrypted buffer R.

4.2 Basic Design

The basic design enables the ad network to obliviously deliver targeted ads to the users via processing their profiles in the encrypted domain. It contains the following phases:

Setup. The publishers (i.e., app developers) register their apps at the ad network. The advertisers register their ads at the ad network, along with the corresponding target profiles. Besides, a fixed public dictionary D is assumed to be available for clients and the ad network. It is used to construct ad request for users and enables the ad network to perform private search. Each client runs the algorithm KeyGenerate to generate a key pair (pk_u, sk_u).

User Profiling. In order to protect the user privacy, our system performs user profiling on the local device. User profiles can be inferred from multiple streams of private information, such as browsing history, email, and SMS [20]. This ensures that the private information used for profiling are kept confidential. In particular, a user profile P is represented as a set of attributes such as user preferences, demographics, and locations. Note that the attributes can also be refined by the user manually.

Ad Delivery. We now show how to leverage PSS in the targeted mobile advertising service system for private targeted ad delivery. The workflow of private targeted ad delivery is illustrated in Fig. 2. When a user launches an app, the client estimates the upper bound m of the number of matched ads, and constructs the encrypted ad request via $Q \leftarrow \mathsf{QueryConstruct}(pk_u, \epsilon, m, P, D)$. Then, the client sends the encrypted request Q to the ad network, along with m and its public key pk_u. Upon receiving the request, the ad network conducts search over all the ads. Specifically, the ad network initializes a buffer R with encrypted zeros under the public key pk_u, and performs private searching on all ads and produces an updated encrypted buffer: $R \leftarrow \mathsf{PrivateSearch}(pk_u, Q, AD)$, which is then returned to the client. Upon receiving the encrypted buffer R, the client reconstructs the matched ads using its secret key: $M \leftarrow \mathsf{FileReconstruct}(sk_u, R)$. Then, the client calculates the matching degree d between the user profile P and the profile P_i of each reconstructed ad ad_i as follows. First, for each attribute $A_j \in P$ and $A'_j \in P_i$, the client sets the matching degree $d_j = 1$ when $A_j \cap A'_j \neq \varnothing$, and otherwise $d_j = 0$. Then, the final matching degree is computed as $d = \mathsf{Average}(d_1, d_2, \ldots, d_\alpha)$. The client ranks all the matched ads according to the matching degrees, and displays the highest ranking one.

Security Guarantees. The security of the above basic design is guaranteed by that of the underlying cryptographic building block PSS. In the design, user profiling is performed locally, which ensures that all the user source data never leave the user's device. The ad network receives encrypted user profiles and produces an encrypted buffer of matched ads. Note that our basic design treats both the matched ads and non-matched ads equally at the ad network side. The ad network learns neither user profiles nor whether an ad matches the ad request or not. Consequently, our basic design is as secure as PSS.

Note that the security of our basic design requires that the ad network should return the same number of ads regardless of how many ads match the ad request. If this is not the case, the ad network could easily mount a dictionary attack using the algorithm PrivateSearch to determine the exact user profiles according to [24]. As a result, like all other proposals for private stream searching, our basic design for privacy-preserving ad delivery requires a priori upper bound m on the number of matched ads to fetch.

Remark. In the above basic design for private targeted ad delivery, we propose to exploit the PSS technique to enable the user to accurately retrieve ads that match her interested keywords while keeping the keywords confidential. The correctness of PSS ensures that if a user has a private keyword like Nike, then Nike will be involved in the encrypted delivered ads. So, this basic approach does not cause a revenue/ad relevance trade-off. However, as shown by the following analysis about the limitations of the basic design, it would incur an efficiency trade-off. The limitations of directly applying PSS in private targeted ad delivery are as follows.

- *Client-Side Cost.* The size of the encrypted ad request is $O(|D|)$, where $|D|$ is the size of the public dictionary D. This may incur cumbersome computation and bandwidth cost for the client due to enormous ad keywords in the public dictionary, say hundreds of millions [30].
- *Server-Side Cost.* When the number of ads at the ad network is large, it is impractical to search all the

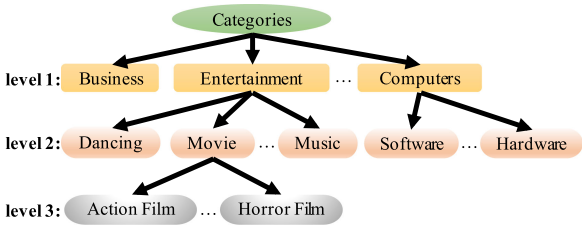


Fig. 3. The hierarchical structure of Google Ads Preferences categories.

ads, because searching each ad requires expensive cryptographic operations. Consequently, searching all the ads may incur heavy computation burden for the ad network, as well as make the client wait too long to fetch the matched ads. In particular, according to [24], at least one modular exponentiation (14.7 ms) is needed when the ad network conducts private search on each ad. And the majority of search time is due to the modular exponentiation [24]. As a result, searching all the ads is very time consuming, e.g., for the more than 1 million ads in the AdMob [6], it would take at least 4.1 h.

- *PSS Limitation.* Without knowing the information about ads, it is hard for the user to have an appropriate estimate of the upper bound m. If the upper bound is underestimated, the buffer will overflow and no information can be recovered from the returned encrypted buffer. If the upper bound is overestimated, it will lead to bandwidth wastage. Although the ad network can assist the client to estimate the upper bound via pre-interaction [24], this will increase the computation burden of both the client and the ad network.

4.3 Secure and Practical Targeted Mobile Ad Delivery

4.3.1 Design Rationale

Before giving our system design of secure and practical targeted mobile ad delivery, we first describe our design rationale based on our observations from the system implementation perspectives. First, we observe that users' interests can be represented by hierarchical preference categories. For example, users can use the Google Ad Setting tools to choose hierarchical categories, so as to control which ads the users expect to see on Google services and websites [31]. Compared with enormous (say hundreds of millions [30]) ad keywords, the number of categories is much smaller, say several hundreds [31], so the computation and bandwidth cost of the mobile client can be significantly reduced. Note that such efficiency improvement is achieved with a revenue/ad relevance trade-off, as the fine-grained ad keywords are replaced by relatively coarse-grained category information. Second, we observe that in order to maximize the revenue, the ad network usually conducts auction over the ads according to the bidding information, global information (e.g., Click Through Rate), etc [32]. Only those highest ranking targeted ads will be selected for delivery.

Based on the above important observations, we introduce several mechanisms to enable a practical service system built on our basic system design. First, in order to reduce the size of ad requests, we adopt hierarchical preference

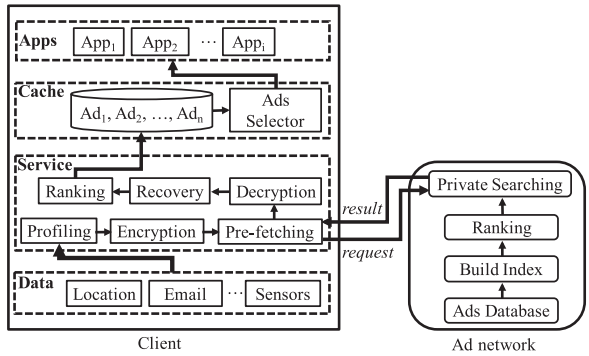


Fig. 4. Secure and efficient targeted mobile advertising architecture.

categories to represent user profiles, such as the public Google Ads Preferences [31] as shown in Fig. 3. In this way, the size of ad requests is proportional to the size of public preference categories instead of the huge public dictionary of potential keywords. Besides, it also reduces the overall computation and bandwidth cost.

Second, we build category indices and conduct auction over the ads of each category at the ad network. The ad network conducts private search over the highest ranking ads. Finally, several top-level categories can be sent to the ad network, which can help reduce the computation cost of the ad network by narrowing the search range. The top-level categories are considered to be less sensitive than the bottom-level categories. Note that this is a balance between privacy and efficiency. In particular, when user preferences are represented into level-i categories, and the level-j categories can be provided as plaintext, where $j < i$. For example, as shown in Fig. 3, the user preferences are Action Film and Horror Film (level-3), so the broad category Movie (level-2) or Entertainment (level-1) can be provided for the efficiency on mobile devices and the ad network.

4.3.2 Our Construction

Our practical system design of secure and practical targeted mobile ad delivery is illustrated in Fig. 4. It contains the following phases:

Setup. The ad network uses hierarchical preference categories to represent the ad profiles. The ads preference categories can be either provided by the advertisers, or be inferred from the content and keywords of the ads by the ad network. Besides, the ad network builds dictionaries and indices from the collected ads by calling BUILDINDEX which is illustrated in Fig. 5.

First, the ad network initializes the dictionaries and the posting lists. Then, the dictionary set $Dict$ and the index set I are built for the ads according to the hierarchical preference categories. Meanwhile, the ad network sorts the ads of each category according to the bidding information, global information, etc. As a result, the ads in each category are stored in order. In addition, each client runs the algorithm KeyGenerate to generate a key pair (pk_u, sk_u).

User Profiling. Different from our basic system design in which the user profile P is represented as a set of attributes, we now construct the user profile as hierarchical preference categories. First, the client stores the public hierarchical preference categories structure locally. Second, the preference categories of the user profile are inferred from the user'

Input:
- Structure of hierarchical preference categories.
- Ads and their preference categories: $\{(ad_r, P_r)\}_{r=1}^{n}$, where n is the number of ads.

Output:
- Dictionary set $Dict$ and index set I.

Procedure:
1. Initialization:
   1) For each level-i, initialize a dictionary $Dict_i$.
   2) For each category $C_{i,j}$ at level-i, initialize a dictionary $Dict_{i,j}$ and a posting list $I_{i,j}$.
2. Build dictionaries and posting lists:
   1) For each $C_{i,j}$ at level-i,
      - $Dict_i.add(C_{i,j})$.
      - For each subcategory $C_{i+1,k}$ of $C_{i,j}$, $Dict_{i,j}$.add($C_{i+1,k}$).
      - For each ad_r, $I_{i,j}.add(ad_r)$, where $C_{i,j} \in P_r$.
      - Sort($I_{i,j}$).
3. Output the index set I and dictionary set $Dict$.

Fig. 5. The details of the BUILDINDEX algorithm for our practical system design of private targeted mobile ad delivery.

interested keywords which are extracted from her online behavior information and smartphone sensor data. Finally, the hierarchical preference categories of the user can be built with reference to the locally stored hierarchical structure.

Note that a user may have multiple preference categories. In this case, inspired by [33], we can estimate the weight of each preference category and then select the most important preference categories for targeting. The weight of each inferred category is determined by the weights of the corresponding keywords. And the weight of each keyword is estimated by its source, recency, frequency and duration, etc. For example, the messages from friends are more important than that from strangers, the information generated recently is more important than past information, and the information accessed by the user frequently is more important than that accessed infrequently.

Ad Delivery. After the user preferences are inferred, the client constructs an encrypted ad request to fetch ads from the ad network. As the user preferences are represented as a hierarchical structure of categories, the client first determines which level of categories is non-sensitive, i.e., the categories that can be directly provided, and which level of categories should be encrypted. Then, the client encrypts those sensitive categories to generate the ad request.

The client sends the encrypted ad request to the ad network, along with the expected number of matched ads per preference category, the number of her preference categories, and the non-sensitive broad interest categories. Here, the expected number of matched ads per preference category and the number of preference categories help the ad network to determine the size of the encrypted buffer. The information of non-sensitive broad preference categories can reduce the size of ad request and narrow the search range. Note that the user can decide how to provide generalized non-sensitive category information with the help of existing tools like entropy-based mechanisms [34]. In addition, the client should tell the ad network which level of categories is used to construct the encrypted ad request, so as to specify the level of categories to be searched.

The ad network utilizes the non-sensitive broad preference categories to narrow the search range. In particular, suppose that the user's preference categories are at level-i, the ad network first locates all level-i categories in the sub-hierarchical structure rooted at each broad preference category. Then the ad network conducts private searching on the corresponding level of categories using the encrypted ad request, dictionaries of each level-(i-1) category in the sub-hierarchical structures, and indices of each located level-i category. As the ad network has built indices for the ads of each level-i category, we pack the highest ranking ads of each level-i category into a block. Consequently, the algorithm PrivateSearch is performed over the packed block rather than each ad. After processing each index built with the level-i category in the narrowed search range, the ad network returns an encrypted buffer to the client. Then the client reconstructs the matched ads as in our basic design.

4.4 Practical Considerations

Prefetching and Caching. According to [35], mobile applications usually refresh their ads every 12-120 seconds. Such frequent ad fetching will lead to serious energy consumption in both communication and computation for the mobile devices [35]. Therefore, we introduce a prefetching and caching mechanism in our system, which enables the ads to be fetched from local cache promptly rather than from the remote ad network. That is, before the user lunches the mobile app, ads are prefetched in bulk via our proposed approach of private ad delivery and be served locally. In practice, to ensure that the prefetched ads are related to the user's future preferences, we may leverage ad prefetching models [36] to first predict future user context by mining past context history, and then leverage this context prediction to retrieve a comprehensive corpus of relevant ads.

Previous studies showed that prefetching and caching can be effective in reducing the bandwidth cost because 0.58 percent of ads are fetched for more than 50 percent of the times [36]. Besides, prefetching can help save energy consumed by ad request construction and matched ads reconstruction in our practical design. Furthermore, prefetching also reduces the ad loading latency which highly affects user experience, because the ads are now fetched from the local cache promptly rather than from the remote ad network.

Note that ads prefetching from the ad network can be conducted periodically or when user preferences change. In addition, the ads can be prefetched when mobile devices are being charged and in WiFi networks. Therefore, it is very likely that the prefetching mechanism also saves the battery capacity (which is limited) and the expensive cellular bandwidth. We also note that such a prefetching mechanism may result in revenue loss, either because a prefetched ad is not shown within its deadline or is shown more times than required. However, we can minimize the revenue loss by building client models to predict how many ads that the user is able to consume in the future [35].

Ad Selection. In our system designs, we use a general ranking mechanism to determine which ads to be displayed. This might not always be suitable for the current mobile context. To cater for the current context, we can further take into account additional information such as app name/category and current session page information, so as to improve the relevance of displayed ads. This is also known as mobile contextual advertising [30], which can help the client to fetch more relevant ads according to the user's current interests and increase the probability of ad click.

4.5 Security Analysis

In our practical system design, all the sensitive user data are stored locally for inferring user preferences. Besides, the sensitive preference categories sent to the ad network and the returned matched ads are all encrypted by the user's public key. Therefore, the ad network learns nothing about user's private preferences and matched ads.

In our practical system design, the information of non-sensitive broad preference categories provided by the client can make the secure targeted mobile ad delivery more efficient, which will be evaluated in Section 6. We must note that such efficiency improvement is achieved with a privacy trade-off, as the generalized high-level category information might leak partial information about user's preferences. However, it is worth noting that in our practical system, the users can determine whether they provide the information of non-sensitive broad preference categories. To mitigate the impact of such privacy leakage, we can employ l-diversity [37] to protect the privacy of a user's sensitive low-level category information [11], so that the adversary cannot infer the user's interests accurately.

In addition, the client tells the ad network which level of categories the ad request adopts. The indication of the level of categories doesn't compromise user privacy, because the ad network cannot accurately infer users' preferences from the level of user preference categories. Moreover, the client provides the number of preference categories to the ad network. This may leak partial information, as the ad network may infer user preferences from the possible combination patterns of preference categories. A simple and effective solution is to let the client provide a fixed number a of preference categories to the ad network. When the actual number of user preference categories is larger than a, the client selects the a user preference categories with higher weight. Otherwise, the client fills the preferences with random categories.

5 Privacy-Preserving Billing

5.1 Design Intuition

As mentioned in Section 2.1, the advertiser is billed either for each ad view in the "charge-per-view" model or for each ad click in the "charge-per-click" model. In the "charge-per-click" model, once a user clicks on an ad, she will be redirected to the advertiser's site, possibly via the ad network, where billing takes place. In the "charge-per-view" model, the ad network records the number of views of the delivered ads. At the end of a billing cycle, the advertiser is billed by the ad network according to the total views of ads and the cost per view.

Billing in the "charge-per-click" model remains unchanged in our system, as we do not hide the ad click history from the ad network (see Section 2.2). This means that the standard billing procedure for the "charge-per-click" model can be applied. Thus, the challenge we have to tackle is to design a billing scheme properly in the "charge-per-view" model, which should satisfy the following requirements: i) Privacy guarantee: the ad network should be prevented from knowing which ad was displayed to the user; ii) Billing correctness: the ad network should be able to record the number of views of each ad accurately, so that it can charge the advertiser correctly.

To support privacy-preserving billing, viable approaches are to resort to a third party who does not collude with the ad network for assistance [13], [15]. In [15], an intermediate third party called dealer is introduced between users and the ad network. The dealer is required to stay online and anonymize all the ad view reports from the users. However, ideally the third party should only be contacted infrequently in practice. In [13], a secure billing scheme is proposed, in which the third-party is only contacted at the end of a billing cycle. Specifically, their scheme is based on homomorphic encryption, in which the ad network maintains encrypted counters for the views of ads throughout a billing cycle, and relies on a third party to decrypt the counters at the end of a billing cycle. This scheme is preferable in practice because of the infrequent contacting with the third party. However, it leaks the billing information to the third party, which may not be expected by the ad network in practice.

Our privacy-preserving billing designs build on the scheme in [13], while circumventing its limitations and achieving stronger security strengths. In addition, the performance of the billing scheme in [13] remains unclear as not experiments are presented therein, while our enhancement designs are thoroughly evaluated in experiments.

In particular, our basic scheme also resorts to a third party that does not collude with the ad network to decrypt the encrypted billing information maintained by the ad network, while preventing the third party from accessing the billing information. Our key idea is to let the ad network add random masks to the billing information in the ciphertext domain, which can be supported by leveraging the property of additively homomorphic encryption. In this way, the third party is prevented from accessing the billing information as it only obtains a masked version after performing decryption. Later, the ad network can remove the random masks to recover the billing information.

Considering that the ad network may not expect to let the third party have full security control in billing, we further propose an enhanced scheme to avoid full reliance on the third party for decryption, by properly leveraging threshold cryptography.

We want to point out that although our billing schemes focus on the "charge-per-view" model, they can also be readily leveraged to protect the user's ad click history under the "charge-per-click" model, given that the advertisers would not share the ad click reports with the ad network. That is, we can simply treat an ad click report as an ad view report, use the counters to measure the clicks for each ad, and apply the same protection mechanisms.

5.2 Basic Scheme

We now present the details of our basic billing scheme. The basic scheme consists of the following phases:

Preparation. The third party generates a key pair (pk, sk) for an additively homomorphic encryption scheme, e.g., the Paillier cryptosystem, and publishes the public key pk. At the beginning of a billing cycle, for each ad ad_i, the ad network encrypts a counter ctr_{ad_i} which is initialized to 0, i.e., $C_{ad_i} = E_{pk}(ctr_{ad_i}) = E_{pk}(0)$.

Ad View Aggregation. Recall that in the procedure of ad delivery in our system, the user gets back an encrypted buffer of matching ads. As the ad network does not know

the IDs of the matching ads, it also sends a vector of the IDs of all searched ads, i.e., $w = \{w_1, w_2, \ldots, w_t\}$. After the client selects an ad to display, it constructs a binary vector, i.e., $\overline{w} = (\overline{w}_1, \overline{w}_2, \ldots, \overline{w}_t)$, which indicates which ad was displayed to the user. In particular, the element corresponding to the displayed ad is set to 1 while others are set to 0.

Next, the client encrypts each element of the vector using the public key pk to produce a vector of ciphertext: $(E_{pk}(\overline{w}_1), E_{pk}(\overline{w}_2), \ldots, E_{pk}(\overline{w}_t))$, which is then sent to the ad network. Upon receiving the vector of ciphertexts, the ad network multiplies each element of the vector by a scalar c_{ad_i} representing the cost of the corresponding ad view: $E_{pk}(\overline{w}_i)^{c_{ad_i}} = E_{pk}(\overline{w}_i \cdot c_{ad_i})$, where $i \in [1, t]$. Note that all these quantities are an encryption of 0 except for one element which is an encryption of c_{ad_i}.

Then the ad network utilizes the additive homomorphic property to add each encrypted value to the corresponding encrypted counter of each ad. That is, the encrypted counter C_{ad_i} is updated via $C_{ad_i} \cdot E_{pk}(\overline{w}_i \cdot c_{ad_i})$. The result of this process is that the ciphertexts of all counters are updated, while only the value underlying the encrypted counter which corresponds to the ad displayed to the user is changed. The security of additively homomorphic encryption ensures that the ad network does not know which counter was changed.

Settlement. At the end of a billing cycle, the ad network needs to charge the advertiser for all the ad views that have been generated. To do so, it needs to decrypt the encrypted counters of all ads of the advertiser. As the decryption key is held by the third party, the ad network needs to send the encrypted counters to the third party for decryption. To prevent the third party from learning the billing information, the ad network selects random masks and adds them to the encrypted counters by utilizing the additive homomorphic property. In particular, the ad network generates a random mask r_{ad_i} for each ad ad_i that will be billed, and encrypts r_{ad_i} using the public key pk, i.e., $E_{pk}(r_{ad_i})$. Then the ad network adds the encrypted random mask to the encrypted counter of each ad ad_i, and produces the ciphertext $E_{pk}(ctr_{ad_i} + r_{ad_i})$. Upon receiving the ciphertexts, the third party decrypts them using its private key sk and produces $ctr_{ad_i} + r_{ad_i}$ which is then returned to the ad network. The ad network removes the random mask from $ctr_{ad_i} + r_{ad_i}$ for each ad ad_i, and obtains the plaintext counters, which are then used to charge the advertiser.

Security Guarantees. Users' ad view reports are protected under the public key of the Paillier cryptosystem, which is semantically secure. As the third party holds the decryption key and does not collude with the ad network, the ad network is not able to obtain the actual view reports of a particular user throughout a billing cycle. On another hand, as the plaintexts underlying the encrypted counters which are sent to the third party for decryption are obscured with random masks, the third party is not able to obtain the billing information.

5.3 Enhanced Scheme

In our basic scheme, the private key for decryption is held by the third party so the security of the billing information has over-reliance on the third party, which may not be expected by the ad network in practice. Once the third party happens to get access to the encrypted counters, e.g., due to the occasional data leakage caused by careless insiders working at the ad network, it is able to obtain the billing information. We now introduce an enhanced scheme which avoids over-reliance on the third party, while still preserving the security guarantee of the basic scheme.

Using Threshold Paillier Cryptosystem:
Third party: //Decryption share generation

- $ctr^2_{ad_i} = (C_{ad_i})^{2\Delta sk_2}$, where $\Delta = 2!$.

Ad network: //Decryption share generation and combination

- $ctr^1_{ad_i} = (C_{ad_i})^{2\Delta sk_1}$.
- $\lambda^S_{0,j} = \Delta \prod_{j' \in S \setminus \{j\}} \frac{-j}{j-j'}$, where $S = \{1, 2\}$.
- $ctr'_{ad_i} = \prod_{j \in S} (ctr^j_{ad_i})^{2\lambda^S_{0,j}}$.
- The counter ctr_{ad_i} can be further computed from ctr'_{ad_i} according to [39].

Fig. 6. Using the threshold Paillier cryptosystem in our enhanced private billing scheme.

Our main idea is to let the ad network and the third party jointly decrypt the encrypted counters. To this end, we resort to the threshold Paillier cryptosystem [38]. In this cryptosystem, the additive homomorphic property of the Paillier cryptosystem is still preserved, while the decryption of a ciphertext requires the participation of a certain number of parties, each of which holds a share of the private key. Specifically, each participating party produces a decryption share for the ciphertext, and the combination of these decryption shares produces the plaintext. In our enhanced scheme, the private key for decryption is divided into two shares, where one is held by the third party and the other by the ad network. At the end of a billing cycle, the ad network sends the encrypted billing information to the third party, who then produces a decryption share of the billing information and sends it back. Then, the ad network also uses its private key to produce a decryption share of the billing information. After combining both decryption shares, the ad network can obtain the billing information.

Fig. 6 illustrates how to leverage the threshold Paillier cryptosystem in our enhanced scheme. In particular, clients use the public key to encrypt ad views. And the private key is divided into two shares: one is held by the ad network, and the other is held by the third party. At the end of a billing cycle, both the ad network and the third party produce a decryption share for the encrypted counters, and finally the ad network combines the decryption shares to produce the counters, i.e., the billing information. In more detail, our enhanced scheme is described below.

Preparation. The ad network and the third party run a one-time setup to produce the public key pk of the threshold Paillier cryptosystem, and each of them also obtains a share of the private key sk. This can be securely achieved by a standard interactive protocol [39] initiated between the ad network and the third party, without introducing additional parties. We denote sk_1 as the private key share of the ad network, and sk_2 as the one held by the third party. As before, at the beginning of a billing cycle, for each ad ad_i, the ad network encrypts a counter ctr_{ad_i} which is initialized to 0, i.e., $C_{ad_i} = E_{pk}(ctr_{ad_i}) = E_{pk}(0)$.

Ad View Aggregation. This phase is the same as in our basic scheme.

Settlement. For each ad ad_i that will be billed, the ad network forwards the corresponding encrypted counter $C_{ad_i} = E_{pk}(ctr_{ad_i})$ to the third party. Meanwhile, the ad network uses its private key sk_1 to decrypt the encrypted counter and produces a decryption share $ctr^1_{ad_i}$. Upon receiving C_{ad_i}, the third party uses its private key sk_2 to decrypt the encrypted counter and produces a decryption share $ctr^2_{ad_i}$, which is then sent to ad network. The ad network combines the received decryption share $ctr^2_{ad_i}$ and its decryption share $ctr^1_{ad_i}$ to obtain the plaintext ctr_{ad_i}, which can be used to charge the advertiser.

Security Guarantees. The security of the threshold Paillier cryptosystem ensures that neither the ad network nor the third party can learn the counters via their own decryption shares. Besides, as now the ad network holds a share of the decryption key, security over-reliance on the third party is avoided. Moreover, throughout a billing cycle, each user's view report is still protected in a semantically secure way, and the ad network is not able to know which ad was displayed to a particular user.

5.4 Discussion on Other Threats

In our billing schemes, we consider that the users and the third party will follow our protocols correctly. That is, the user will produce the vector $\overline{w}$ accurately, and the third party will decrypt the encrypted counters correctly. When considering that the users and the third party might misbehave, our system can integrate related technique for defense. In particular, we can leverage the technique of zero-knowledge proofs [13], [39] as follows. When the user submits the encrypted vector $(E_{pk}(\overline{w}_1), E_{pk}(\overline{w}_2), \ldots, E_{pk}(\overline{w}_t))$, she is also required to provide zero-knowledge proofs of the following facts: i) $\overline{w}_i$ is either 0 or 1; ii) the sum $\sum_{i=1}^{t} \overline{w}_i$ is 1. When the ad network receives the encrypted vector, it will update the corresponding encrypted counters only when the proofs are valid. Regarding the third party, it is required to provide a zero-knowledge proof showing that the decryption was correctly done.

Another potential threat is that the ad network might cheat in the ad views to overcharge the advertiser. A defense mechanism proposed by a very recent work [40] is to change the workflow of ad view report. That is, the ad views are first reported to the advertisers and publishers, who then forward them to the ad network after some kind of processing. However, this is not directly compatible with the architecture of existing mobile advertising service systems. We leave the solution for such threat as future work.

6 Experiments

6.1 Implementation

To evaluate the performance of our privacy-preserving targeted mobile advertising system, we implement the following entities.

- *Mobile Client.* We implement the client on a Samsung Galaxy S4 with android 5.0.1, which is equipped with a four-core 1.6 GHz processor and a four-core 1.2 GHz processor and 2.0 GB RAM. The client constructs the encrypted ad request and reconstructs the encrypted matching ads returned from the ad network. For simplicity, we assume that user preferences are generated in advance, e.g., they can be generated from multiple user information streams on smartphones as in [20].
- *Ad Network.* We implement the ad network on a desktop PC equipped with a two-core 2.7 GHz processor and 8.0 GB RAM. The ad network collects ads from advertisers, builds ads dictionaries and indices, and conducts private ad searching over the ads. In addition, the ad network collects ad views for charging advertisers throughout a billing cycle.
- *Third Party.* We implement the third party server on a desktop PC equipped with a two-core 2.7 GHz processor and 8.0 GB RAM. The third party assists the ad network to decrypt the encrypted billing information at the end of a billing cycle.

TABLE 1
Ad Request Types with Different Levels of User Preferences and Plaintext Broad Preferences

Ad Request	Plaintext Broad Preferences	User Preferences
Type 1	Null	Level-1 categories
Type 2	Null	Level-2 categories
Type 3	Null	Level-3 categories
Type 4	Level-1 categories	Level-2 categories
Type 5	Level-1 categories	Level-3 categories
Type 6	Level-2 categories	Level-3 categories

In our experiment, we adopt public Google Ads Preferences Categories to describe user preferences and ad profiles, as shown in Fig. 3. The Google Ads Preferences scheme adopts a three-level hierarchical categorization mechanism. There are 24 level-1 categories, 243 level-2 categories, and 634 level-3 categories. Meanwhile, we generate 200,000 ads with randomly assigned categories, and the size of each ad is about 200 bytes.

6.2 Performance Evaluation on Private Ad Delivery

In order to demonstrate the efficiency of our proposed private targeted ad delivery, we first evaluate the bandwidth cost of the client, and then we evaluate the computation cost at the client and ad network, respectively.

6.2.1 Bandwidth Performance

Ad Request. We first measure the size of ad requests in different types. We define six types of ad requests as shown in Table 1, based on different combination of user preference categories and non-sensitive broad preference categories. Now, we assume that a user adopts level-i categories to represent her preferences. When the user does not provide any broad categories, the size of an ad request is $d_i \cdot 256$ bytes, where d_i denotes the number of level-i categories, 256 bytes is the size of a Paillier ciphertext in our experiment. When the user provides b broad level-j categories ($j<i$), the size of the ad request is $\sum_{l=1}^{b} d_{i,l} \cdot 256$ bytes, where $d_{i,l}$ denotes the number of level-i categories in the sub hierarchical structure rooted at the lth level-j broad category. The size of different types of ad request is shown in Fig. 7.

We first analyze the case in which the user does not provide broad preference categories, which corresponds to ad requests of Type 1, Type 2 and Type 3. As shown, the size of ad request will increase as the user uses more fine-grained

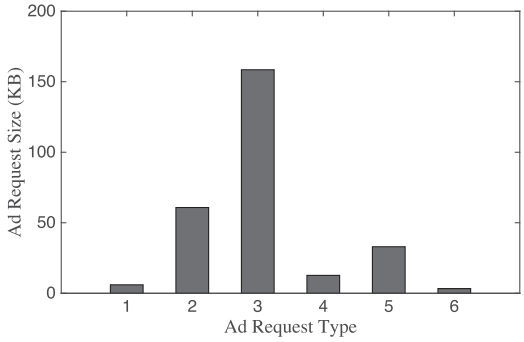


Fig. 7. Bandwidth cost of ad requests in different types.

categories to represent her preferences. In particular, when the adopted user preferences change from level-1 (24 categories) to level-3 (634 categories), the size of the ad request ranges from 6 to 158.5 KB. It is worth noting that there is a trade-off between accuracy and efficiency when different levels are chosen. The more fine-grained level the user adopts, the more accurate the returned ads will be, but at the expense of the ad request with a larger size.

We then analyze the case in which the user provides broad preference categories. As shown by Fig. 7, the size of the ad request is now significantly reduced. For example, in the case that the user adopts level-3 categories as the preferences, if 5 level-1 categories is provided as the broad preferences, the ad request size will reduce from 158.5 to 33 KB. Moreover, when the user provides more fine-grained broad preferences, the ad request will be further reduced. In particular, when the user provides 5 level-2 categories as the broad preferences instead of level-1, the ad request size becomes 3.3 KB. It is worth noting that there is a balance between the efficiency and privacy. The more coarse-grained the user-provided broad preferences are, the less information the ad network will learn, but the ad request size will become larger.

Returned Matching Ads. We now measure the size of returned matching ads. Recall that in our practical system design of private ad delivery, the size of returned matching ads is proportional to the number of user preference categories and the number of returned matching ads per preference category. In our experiments, we assume that the user uses p level-3 categories to represent her preferences, and expects to receive k ads per preference category, and the average size of each ad is s bytes. According to the PSS scheme [24], the size of returned matching ads in our basic system design is $O(m \cdot s)$ bytes, where m is the number of matching ads. In our practical system design, the size of returned matching ads is $O(p \cdot k \cdot s)$ bytes because only the top-k ranked ads of each category are returned to the client. Note that in general $p \cdot k$ is much smaller than m and thus our practical system design is much more efficient in the size of returned matching ads.

To demonstrate the efficiency of our private ad delivery, we further make comparison with two other private ad delivery schemes: client-side targeting and obfuscation based targeting. In the client-side targeting scheme, the client does not provide any information to the ad network and targeting is only performed at the client side. In order to receive accurate targeted ads, the client needs to download ads with a size of $k \cdot s$ bytes from each level-3 category.

TABLE 2
Bandwidth Comparison of Returned Matching Ads in Different Private Ad Delivery Schemes

Scheme	Bandwidth Cost
Naive scheme	5,890 KB
Obfuscation Scheme	1,230 KB
Our Scheme	27 KB

Let d_3 denote the number of level-3 categories, and we have the bandwidth consumption of $d_3 \cdot k \cdot s$ bytes. In the obfuscation based targeting scheme, the client only provides plaintext broad categories to the ad network. To receive the same accurate targeted ads, the client needs to download ads with a size of $k \cdot s$ bytes from each level-3 category that falls in the broad categories. Thus, the bandwidth consumption is $\sum_{l=1}^{b} d_{3,l} \cdot k \cdot s$ bytes, where $d_{3,l}$ denotes the number of level-3 categories that are in the sub hierarchical structure rooted at the lth broad category, and b denotes the number of provided broad categories.

Table 2 shows the evaluation result for the size of returned matching ads in different schemes. We assume that the user uses 5 level-3 categories to construct the ad request, and the ad network returns 15 ads for each user preference category. Therefore, the number of matching ads that the client receives is up to 75. From Table 2, it can be observed that the returned matching ads in our practical system only consumes 27 KB, while the sizes in the client-targeting scheme and obfuscation targeting scheme are 5,890 KB (about $218\times$ larger) and 1,230 KB (about $45\times$ larger), respectively. We note that by using the prefetching and caching mechanism, the bandwidth cost for returned matching ads in our system can be amortized.

6.2.2 Computation Performance

Client Computation Cost. We first measure the computation time at the mobile client in our practical system design of private targeted ad delivery, and the result is shown in Fig. 8. Recall that the client computation consists of two parts: ad request construction and returned matching ads reconstruction. As shown by Fig. 8, when the ad request varies from Type 1 to Type 3, the time of ad request construction ranges from 0.5 to 8.7 s. If the user provides the ad network with broad preference categories, the ad request construction time can be reduced accordingly. And the reduction degree depends on the level of the broad preference categories. Regarding the computation time of

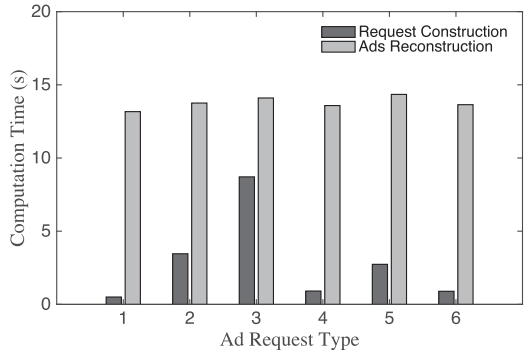


Fig. 8. Computation cost of client in private targeted ad delivery.

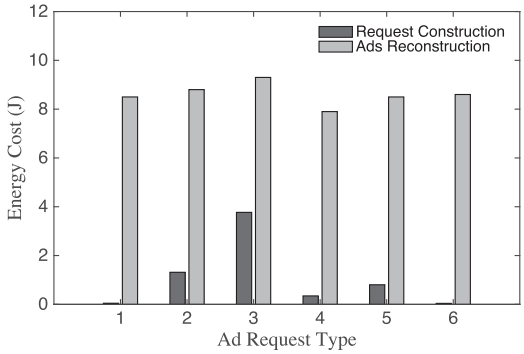


Fig. 9. Energy cost of client in private targeted ad delivery.

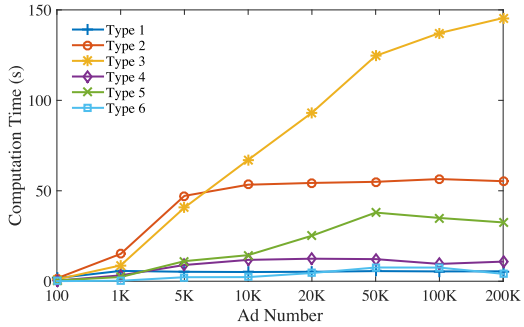


Fig. 10. Search time of ad network in private targeted ad delivery.

matching ads reconstruction, it is not related with the ad request types, and is dependent on the number of user preference categories and the number of returned matching ads per preference category. In our experiments, the time taken by matching ads reconstruction is about 13.8 s when the user uses 5 level-3 categories and the ad network returns 15 ads per preference category.

Client Energy Cost. As battery energy is one of the most precious resources of mobile devices, we also measure the energy cost at the mobile client in our practical system design of private targeted ad delivery. In particular, we use the Power Tutor 2 Pro [41], which is a diagnostic tool for analyzing system and App power usage. Fig. 9 shows the client energy cost of ad request construction and matching ads reconstruction. As shown, in the case that the user does not provide broad preference categories, the energy cost will increase as the user uses fine-grained categories to represent her preferences. In particular, the energy cost of ad request construction ranges from 0.04 J to 3.77 J when the ad request changes from Type 1 to Type 3. When the user provides broad preference categories, the energy cost is significantly reduced. For example, in the case that the user adopts level-3 categories as the preferences, if level-1 categories is provided as the broad preferences, the energy cost will reduce from 3.77 J to 0.8 J. Moreover, when the user provides more fine-grained broad preferences, the energy cost will further diminish.

Regarding the energy cost of ads reconstruction, it is not dependent on the ad request types, and is only related with the number of user preference categories and the number of returned matching ads per preference category. In our experiments, the energy cost of ads reconstruction is about 9 J when the user uses 5 level-3 categories to construct the ad request, and the ad network returns 15 ads for each user preference category.

To help readers understand what the above energy cost implies, we also provide the energy cost of playing a 10-minute audio as a vivid example for comparison. Particularly, playing a 10-minute audio consumes about 10 J energy, while the maximum energy cost at the mobile client in our practical system design of private targeted ad delivery is about 13 J. Therefore, the energy cost of our practical system is affordable for mobile users. Note that such energy cost can be amortized via the prefetching and caching mechanism.

Ad Network Computation Cost. We now measure the computation cost at the ad network in our practical system design of private targeted ad delivery. Fig. 10 shows the computation time of the ad network with regard to different ad request types and different number of ads. As shown, for each type of ad request, the computation time grows linearly when the number of ads is small, and tends to be stable when the average number of ads in each category reaches 15. This is because the private searching will be conducted over all the ads when the number of ads per category is smaller than 15. And when each category has more than 15 ads, the private searching will only be conducted over the ranked top-15 ads. In the latter case, the computation time of the ad network depends on the ad request type. When the ad request types changes from Type 1 to Type 3, i.e., the case where no broad categories are provided, the computation time of the ad network increases. When broad preference categories are provided, the computation time of the ad network can be reduced accordingly.

Ad Network Throughput Measurement. Fig. 11 measures the throughput of the private ad searching on the ad network. In particular, we create multiple threads simultaneously to simulate the concurrent encrypted ad requests from multiple clients. The experimental results show that the ad network can process total 1,350 ads per minute for each of 10 users. Even when the number of users increases to 60, the ad network can still scan over 235 ads per minute for each user. Note that the throughput performance can be further improved if the ad network operates on a more powerful server with more cores.

6.3 Performance Evaluation on Private Billing

In order to demonstrate the efficiency of private billing in our practical system, we evaluate the bandwidth cost of the client, and then the computation cost at the client, the ad network, and the third party, respectively.

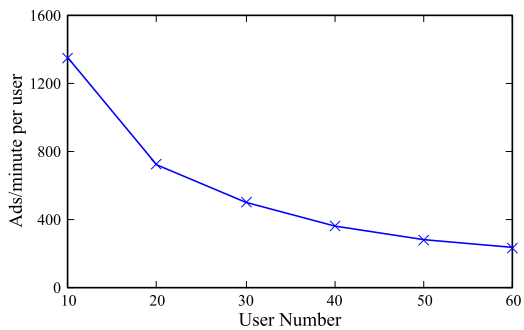


Fig. 11. Throughput measurement on the ad network side.

TABLE 3
Bandwidth and Computation Cost of the Common Ad View Aggregation Phase of Our Private Billing Schemes in Different Ad Request Types

Ad Request	Client Bandwidth	Client Computation	Ad Network Computation
Type-1	90 KB	5.4 s	0.042 s
Type-2	911 KB	54.7 s	0.42 s
Type-3	2,377 KB	142.7 s	1.1 s
Type-4	189.8 KB	11.4 s	0.088 s
Type-5	495.2 KB	29.7 s	0.23 s
Type-6	48.9 KB	2.94 s	0.023 s

6.3.1 Bandwidth Performance

We measure the size of the encrypted ad view report from the mobile client in our private billing designs. Recall that the ad view report is an encrypted binary vector whose number of elements is equal to the number of the searched ads. The number of the searched ads depends on the types of ad request, the number of provided broad categories, and the expected number of returned ads per preference category. Assume that the user adopts level-i categories to represent her preferences and expects to receive k matching ads per category, and provides b broad categories. When the user does not provide any broad categories, the size of an encrypted ad view report is $d_i \cdot k \cdot 256$ bytes, where d_i is the number of the level-i categories. When the user provides b broad level-j categories ($j < i$), the size of an encrypted ad view report is $\sum_{l=1}^{b} d_{i,l} \cdot k \cdot 256$ bytes, where $d_{i,l}$ is the number of level-i categories that are in the sub hierarchical structure rooted at the lth level-j broad category.

Table 3 shows the evaluation result for the size of the encrypted ad view report under different ad request types. We assume that the user uses 5 level-3 categories to construct the ad request, and the ad network returns 15 ads for each user preference category. As shown, for different ad request types, the size of the ad view report ranges from 48.9 to 2377 KB. We remark that such bandwidth cost is acceptable by the mobile client.

6.3.2 Computation Performance

Client Computation Cost. We first measure the computation time at the mobile client in our system designs of private billing. Recall that the client needs to encrypt each element of the constructed binary vector which represents the ad view report. The number of the elements has been analyzed above, so we only need to evaluate the time consumed by one operation of Paillier encryption, the result of which is 15 ms. More precisely, the computation time at the mobile client is shown in Table 3. The minimum computation time is only 2.94 s. Note that the encrypted ad view report is generated after an ad has been selected to display, so the user experience will not be affected by the computation of the mobile client for private billing.

Ad Network Computation Cost. We now measure the computation cost at the ad network in our private billing schemes. Recall that the computation cost of the ad network comes from the ad view aggregation and settlement phases.

In the phase of ad view aggregation, after receiving the encrypted binary vector representing the ad view report, the ad network needs to update the encrypted counters. Recall that for each element of the encrypted vector, the ad network needs to perform one exponentiation operation and one multiplication operation in the Paillier ciphertext domain. The number of the elements has been analyzed above, so we only need to evaluate the total time taken by one exponentiation operation and one multiplication operation, the result of which is 0.116 ms. More precisely, Table 3 shows the evaluation result for the computation time of the ad network in the ad view aggregation phase. The minimum computation time is only 0.023 s for the ad request of Type 6.

Regarding the settlement phase, in our basic scheme, the ad network mainly needs to encrypt a random mask and add it to the encrypted counter for each ad that will be billed. As one operation of Paillier encryption takes 10 ms and one multiplication operation of Paillier ciphertexts takes 0.004 ms, the computation time is only 10.004 ms for each ad in the settlement phase. In our enhanced scheme, for each ad, the ad network needs to calculate a partial decryption share and combines it with the decryption share from the third party. Producing one partial decryption share takes 20 ms and the share combination takes 1.2 ms. Therefore, the computation time of the ad network is only 21.2 ms for each ad in the settlement phase of our enhanced billing scheme.

Third Party Computation Cost. We also measure the computation cost of the third party in the settlement phase of our billing schemes. In our basic billing scheme, the third party needs to decrypt the encrypted masked counter for each ad, and the time taken by one Paillier decryption operation is only 10 ms. In our enhanced billing scheme, the third party needs to produce a partial decryption share for the encrypted counter for each ad, and such operation only takes 20 ms.

7 Related Work

Mobile Advertising. In the literature, mobile advertising has been extensively studied in recent years. In order to target users accurately for ad delivery, the ad network (typically AdMob [6]) usually collects user personal information to perform targeting, such as location, IMEI number, phone number, and call logs [10], [42], [43]. Besides, Nath et al. [30] show that leveraging the keywords in the current mobile app page viewed by the users can also enable targeted ad delivery. From the system perspective, several works [35], [36], [44] have investigated the mobile ad traffic and showed that frequent ad fetching can lead to heavy network signaling and significant energy consumption on the mobile client. To address this issue, the mechanism of prefetching and caching has been adopted by existing works [35], [36], [44] to optimize the energy consumption.

Privacy-Preserving Targeted Advertising. Our work is also akin to the branch of work on privacy-preserving targeted advertising. Toubiana et al. [13] propose Adnostic, which lets the ad network randomly send the user a set of ads, among which the user selects the most relevant one based on her behavioral profile. They also design a private billing scheme based on homomorphic encryption and zero-knowledge proofs, enabling the ad network to correctly charge the advertisers under the "charge-per-view" model, without learning which ad was viewed by a particular user. In their design, a third-party is introduced for decrypting the encrypted ad view reports and thus the billing information is

exposed to it. Similar to [13], Wang et al. [14] also propose to perform targeting on user device, and they focus on incentivizing users to click ads that are interesting yet sensitive to them. The major limitation of the above local targeting approaches is the high communication cost due to that the number of ads sent to the user for local targeting could be very likely large.

Guha et al. [15] propose Privad which introduces a third-party server called dealer to anonymize the messaging flow from the clients, so as to protect user privacy. However, the dealer is required to always stay online, which is not desirable as ideally a third-party should be contacted infrequently in practice. Backes et al. [16] propose ObliviAd which also protects use privacy via anonymizing the communication channels between clients and the ad network, through hardware-based private information retrieval techniques (which are also adopted for private location-based advertising systems [45]). However, these designs rely on an additional trusted hardware residing on the ad network's side, and thus it may be hard to guarantee that that hardware administrated by the ad network would not be rigged. Note that the above anonymization based approaches may be prone to re-identification attacks [17], [18].

There is also a line of works [11], [19], [20] that propose to obfuscate the user profile via profile generalization to achieve user privacy. However, such coarse-grained obfuscation technique can lead to the delivery of a considerable number of irrelevant ads to the clients. There are also some other works supporting privacy-aware targeted advertising either through leveraging the cooperation among users [25] or sending the ad network only the popular bidding keywords for targeting [30].

On the other hand, there are some works studying microtargeted ads and malicious ads in the mobile advertising service systems, which are orthogonal to our work. In [27], Korolova proposes that microtargeted ads manipulated by the advertiser may be exploited to learn information about users' profiles. Son et al. [28] show that malicious ads can access user's data storage and infer sensitive information, and Rastoni et al. [46] study how to detect hidden attacks launched by malicious ads through the mobile app-web interfaces. Besides, some works [47], [48] show that sharing the same permissions between the mobile ad library and the mobile host applications can pose a threat to user privacy. To address this threat, privilege-separated advertising frameworks have been proposed [49], [50], [51], [52].

Private Searching. Private Information Retrieval (PIR) [53] can be used for private ad delivery. However, it is not energy efficient for mobile devices due to the high communication overhead which scales with the total number of ads. To make the communication only dependent on the number of matching results, Ostrovsky et al. [23] propose the private stream searching scheme. Later, the communication cost of PSS is further optimized by Bethencourt et al. [24]. Our system builds on the top of the optimized PSS scheme [24]. However, as shown before, directly applying it does not enable the design of a practical system. Thus, we introduce several effective techniques tailored for targeted mobile advertising services, so as to achieve both security and practicality. To the best of our knowledge, our work initiates the first endeavor in exploring PSS for privacy-aware targeted mobile advertising services.

Differences from Conference Version. Portions of the work presented in this paper have previously appeared in [1]. We have revised the article a lot and improved many technical details as compared to [1]. The primary improvements are as follows. First, we add Section 5 to present the system design of privacy-preserving billing in our system. In particular, we present two private billing schemes in Sections 5.2 and 5.3, respectively. And we also discuss other potential threats in the billing phase in Section 5.4. The performance of the proposed billing schemes is thoroughly evaluated in a new Section 6.3. Second, we redo all the experiments and extend the performance evaluation. Specifically, we extend Section 6.2.2 to evaluate the battery energy cost on the mobile client in privacy-preserving targeted ad delivery, which is one of the most precious resources of mobile devices. This evaluation further validates that our system design of private targeted ad delivery is practical for resource-limited mobile devices. We also newly measure the throughput performance of private ad searching on the ad network side. Finally, the related work has been improved, which now faithfully reflects many recent advancements on private mobile advertising.

8 Conclusion

In this paper, we propose a practical system for privacy-aware targeted mobile advertising services. In order to enable targeted ad delivery while protecting user privacy, we first give a basic construction based on the cryptographic primitive private stream searching and analyze its practicality issues. We then introduce several optimization mechanisms to achieve a practical system design for secure and efficient targeted mobile ad delivery. After that, we present the design of privacy-preserving billing in our practical system, which enables the ad network to correctly charge the advertisers under the "charge-per-view" model, without knowing the ad view history of a user. Extensive experiments demonstrate the practicality of our system.

Acknowledgments

This work was supported in part by the Research Grants Council of Hong Kong under Grant CityU 138513 and Grant CityU 11276816, the National Natural Science Foundation of China under Grant 61572412 and Grant 61472316, the Innovation and Technology Commission of Hong Kong under ITF Project ITS/307/15, the AWS Education Research Grant, and the Science and Technology Project of Shaanxi Province under Grant 2016ZDJC-05. A preliminary version [1] of this paper was presented at the 11th International conference on Mobile Ad-hoc and Sensor Networks (MSN'15).

References

[1] J. Jiang, X. Gui, Z. Shi, X. Yuan, and C. Wang, "Towards secure and practical targeted mobile advertising," in *Proc. 11th Int. Conf. Mobile Ad-Hoc Sensor Netw.*, 2015, pp. 79–88.

[2] J. Hua, Y. Gao, and S. Zhong, "Differentially private publication of general time-serial trajectory data," in *Proc. IEEE Conf. Comput. Commun.*, 2015, pp. 549–557.

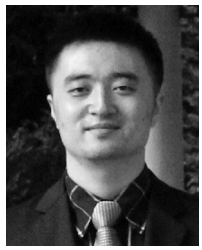
[3] Y. Zhang, Y. Mao, H. Zhang, and S. Zhong, "Privacy preserving market schemes for mobile sensing," in *Proc. 44th Int. Conf. Parallel Process.*, 2015, pp. 909–918.

[4] Danyl Bosomworth, "Mobile Marketing Statistics compilation," [Online]. Available: http://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/
[5] F. Liu, P. Shu, and J. C. Lui, "AppATP: An energy conserving adaptive mobile-cloud transmission protocol," *IEEE Trans. Comput.*, vol. 64, no. 11, pp. 3051–3063, Nov. 2015.
[6] Admob. [Online]. Available: http://www.google.com/admob/
[7] iad. [Online]. Available: http://advertising.apple.com/
[8] Microsoft advertising. [Online]. Available: http://advertising.microsoft.com/en/mobile-advertising
[9] Y. Wang, Y. Chen, F. Ye, J. Yang, and H. Liu, "Towards understanding the advertiser's perspective of smartphone user privacy," in *Proc. IEEE 35th Int. Conf. Distrib. Comput. Syst.*, 2015, pp. 288–297.
[10] S. Nath, "MAdScope: Characterizing mobile in-app targeted ads," in *Proc. 13th Annu. Int. Conf. Mobile Syst. Appl. Services*, 2015, pp. 59–73.
[11] M. Hardt and S. Nath, "Privacy-aware personalization for mobile advertising," in *Proc. ACM Conf. Comput. Commun. Secur.*, 2012, pp. 662–673.
[12] Adrian Chen, "GCreep: Google engineer stalked teens, spied on chats (updated)," [Online]. Available: http://gawker.com/5637234/gcreep-google-engineer-stalked-teens-spied-on-chats
[13] V. Toubiana, A. Narayanan, D. Boneh, H. Nissenbaum, and S. Barocas, "Adnostic: Privacy preserving targeted advertising," in *Proc. Symp. Netw. Distrib. Syst. Secur.*, 2010, pp. 1–16.
[14] W. Wang, L. Yang, Y. Chen, and Q. Zhang, "A privacy-aware framework for targeted advertising," *Comput. Netw.*, vol. 79, pp. 17–29, 2015.
[15] S. Guha, B. Cheng, and P. Francis, "Privad: Practical privacy in online advertising," in *Proc. 8th USENIX Conf. Netw. Syst. Des. Implementation*, 2011, pp. 169–182.
[16] M. Backes, A. Kate, M. Maffei, and K. Pecina, "ObliviAd: Provably secure and practical online behavioral advertising," in *Proc. IEEE Symp. Secur. Privacy*, 2012, pp. 257–271.
[17] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *Proc. IEEE Symp. Secur. Privacy*, 2008, pp. 111–125.
[18] I. Bilogrevic, J. Freudiger, E. D. Cristofaro, and E. Uzun, "What's the gist? privacy-preserving aggregation of user profiles," in *Proc. Eur. Symp. Res. Comput. Secur.*, 2014, pp. 128–145.
[19] M. S. Kodialam, T. V. Lakshman, and S. Mukherjee, "Effective ad targeting with concealed profiles," in *Proc. of IEEE Conf. Comput. Commun.*, 2012, pp. 2237–2245.
[20] D. Davidson, M. Fredrikson, and B. Livshits, "MoRePriv: Mobile OS support for application personalization and privacy," in *Proc. 30th Annu. Comput. Secur. Appl. Conf.*, 2014, pp. 236–245.
[21] P. Shu, et al., "eTime: Energy-efficient transmission between cloud and mobile devices," in *Proc. of IEEE Conf. Comput. Commun.*, 2013, pp. 195–199.
[22] F. Liu, et al., "Gearing resource-poor mobile devices with powerful clouds: Architectures, challenges, and applications," *IEEE Wireless Commun.*, vol. 20, no. 3, pp. 14–22, Jun. 2013.
[23] R. Ostrovsky and W. E. Skeith III, "Private searching on streaming data," in *Proc. Annu. Int. Cryptology Conf.*, 2005, pp. 223–240.
[24] J. Bethencourt, D. Song, and B. Waters, "New techniques for private stream searching," *ACM Trans. Inf. Syst. Secur.*, vol. 12, no. 3, 2009, Art. no. 16.
[25] H. Artail and R. Farhat, "A privacy-preserving framework for managing mobile ad requests and billing information," *IEEE Trans. Mobile Comput.*, vol. 14, no. 8, pp. 1560–1572, Aug. 2015.
[26] A. Reznichenko and P. Francis, "Private-by-design advertising meets the real world," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2014, pp. 116–128.
[27] A. Korolova, "Privacy violations using microtargeted ads: A case study," in *Proc. IEEE Int. Conf. Data Mining Workshops*, 2010, pp. 477–482.
[28] S. Son, D. Kim, and V. Shmatikov, "What mobile ads know about mobile users," in *Proc. Symp. Netw. Distrib. Syst. Secur.*, 2016, pp. 1–14.
[29] J. Bethencourt, D. Song, and B. Waters, "New constructions and practical applications for private stream searching (extended abstract)," in *Proc. IEEE Symp. Secur. Privacy*, 2006, pp. 132–139.
[30] S. Nath, F. X. Lin, L. Ravindranath, and J. Padhye, "SmartAds: Bringing contextual ads to mobile apps," in *Proc. 11th Annu. Int. Conf. Mobile Syst. Appl. Services*, 2013, pp. 111–124.
[31] Google ad settings. [Online]. Available: https://support.google.com/ads/answer/2662856?hl=en
[32] A. Reznichenko, S. Guha, and P. Francis, "Auctions in do-not-track compliant internet advertising," in *Proc. 18th ACM Conf. Comput. Commun. Secur.*, 2011, pp. 667–676.
[33] B. Yan and G. Chen, "AppJoy: Personalized mobile application discovery," in *Proc. 9th Annu. Int. Conf. Mobile Syst. Appl. Services*, 2011, pp. 113–126.
[34] E. Toch, et al., "Empirical models of privacy in location sharing," in *Proc. 12th ACM Int. Conf. Ubiquitous Comput.*, 2010, pp. 129–138.
[35] P. Mohan, S. Nath, and O. Riva, "Prefetching mobile ads: Can advertising systems afford it?" in *Proc. 8th ACM Eur. Conf. Comput. Syst.*, 2013, pp. 267–280.
[36] A. J. Khan, K. Jayarajah, D. Han, A. Misra, R. K. Balan, and S. Seshan, "CAMEO: A middleware for mobile advertisement delivery," in *Proc. 11th Annu. Int. Conf. Mobile Syst. Appl. Services*, 2013, pp. 125–138.
[37] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "L-diversity: Privacy beyond k-anonymity," *ACM Trans. Knowl. Discovery Data*, vol. 1, no. 1, 2007, Art. no. 3.
[38] I. Damgård and M. Jurik, "A generalisation, a simplification and some applications of paillier's probabilistic public-key system," in *Proc. 4th Int. Workshop Practice Theory Public Key Cryptography: Public Key Cryptography*, 2001, pp. 119–136.
[39] T. Nishide and K. Sakurai, "Distributed paillier cryptosystem without trusted dealer," in *Proc. 11th Int. Conf. Inf. Secur. Appl.*, 2010, pp. 44–60.
[40] J. Hua, A. Tang, and S. Zhong, "Advertiser and publisher-centric privacy aware online behavioral advertising," in *Proc. IEEE 35th Int. Conf. Distrib. Comput. Syst.*, 2015, pp. 298–307.
[41] M. Dong and L. Zhong, "Self-constructive high-rate system energy modeling for battery-powered mobile systems," in *Proc. 9th Annu. Int. Conf. Mobile Syst. Appl. Services*, 2011, pp. 335–348.
[42] M. C. Grace, W. Zhou, X. Jiang, and A. Sadeghi, "Unsafe exposure analysis of mobile in-app advertisements," in *Proc. 5th ACM Conf. Secur. Privacy Wireless Mobile Netw.*, 2012, pp. 101–112.
[43] I. Ullah, R. Boreli, M. A. Kâafar, and S. S. Kanhere, "Characterising user targeting for in-app mobile ads," in *Proc. IEEE Conf. Comput. Commun. Workshops*, 2014, pp. 547–552.
[44] N. Vallina-Rodriguez, et al., "Breaking for commercials: Characterizing mobile advertising," in *Proc. Internet Meas. Conf.*, 2012, pp. 343–356.
[45] H. Dang and E. Chang, "PrAd: Enabling privacy-aware location based advertising," in *Proc. 2nd Workshop Privacy Geographic Inf. Collection Anal.*, 2015, Art. no. 1.
[46] V. Rastogi, R. Shao, Y. Chen, X. Pan, S. Zou, and R. Riley, "Are these ads safe: Detecting hidden attacks through the mobile app-Web interfaces," in *Proc. Symp. Netw. Distrib. Syst. Secur.*, 2016, pp. 1–15.
[47] R. Stevens, C. Gibler, J. Crussell, J. Erickson, and H. Chen, "Investigating user privacy in android ad libraries," in *Proc. Workshop Mobile Secur. Technol.*, 2012, pp. 1–10.
[48] T. Book, A. Pridgen, and D. Wallach, "Longitudinal analysis of android ad library permissions," in *Proc. Workshop Mobile Secur. Technol.*, 2013.
[49] P. Pearce, A. P. Felt, G. Nunez, and D. Wagner, "AdDroid: Privilege separation for applications and advertisers in android," in *Proc. 7th ACM Symp. Inf. Comput. Commun. Secur.*, 2012, pp. 71–72.
[50] S. Shekhar, M. Dietz, and D. S. Wallach, "AdSplit: Separating smartphone advertising from applications," in *Proc. USENIX Secur. Symp.*, 2012, pp. 553–567.
[51] X. Zhang, A. Ahlawat, and W. Du, "AFrame: Isolating advertisements from mobile applications in android," in *Proc. 29th Annu. Comput. Secur. Appl. Conf.*, 2013, pp. 9–18.
[52] J. Seo, D. Kim, D. Cho, T. Kim, and I. Shin, "FLEXDROID: Enforcing in-app privilege separation in android," in *Proc. Symp. Netw. Distrib. Syst. Secur.*, 2016, pp. 1–15.
[53] E. Kushilevitz and R. Ostrovsky, "Replication is NOT needed: SINGLE database, computationally-private information retrieval," in *Proc. 38th Annu. Symp. Found. Comput. Sci.*, 1997, Art. no. 364.

Jinghua Jiang received the BE degree from Xi'an Jiaotong University, Xi'an, China, in 2010. He is currently working toward the PhD degree in the School of Electronic and Information Engineering, Xi'an Jiaotong University, and also in the Department of Computer Science, City University of Hong Kong, Hong Kong. His research interests include cloud computing security, multimedia security, mobile security, and targeted advertising.

Yifeng Zheng received the BE degree in information engineering from the South China University of Technology, Guangzhou, China, in 2013. From September to December 2013, he studied with Zhejiang University as a candidate of a master's degree, Hangzhou, China. He is currently working toward the PhD degree in the Department of Computer Science, City University of Hong Kong, Hong Kong. His research interests include cloud computing security, mobile security, and multimedia security.

Zhenkui Shi received the BE degree from the Southwest University of Science and Technology, Mianyang, China, in 2007 and the MS degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2010. He is currently working toward the PhD degree in the Department of Computer Science, City University of Hong Kong, Hong Kong. His research interests include network security, mobile security, and targeted advertising.



Xingliang Yuan received the BE degree from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2008, the MS degree from the Illinois Institute of Technology, Chicago, Illinois, in 2009, and the PhD degree from the City University of Hong Kong, Hong Kong, in 2016. He is currently a research fellow in the Department of Computer Science, City University of Hong Kong. His research interests include cloud computing security and multimedia security.

Xiaolin Gui received the BE, ME, and PhD degrees from Xi'an Jiaotong University, Xi'an, China, in 1988, 1993, and 2001, respectively. He is currently a professor and the deputy dean in the School of Electronic and Information Engineering, Xi'an Jiaotong University. His research interests include secure computation in open network systems, data privacy, and the Internet of Things.

Cong Wang received the BE and ME degrees from Wuhan University, Wuhan, China, in 2004 and 2007, respectively, and the PhD degree from the Illinois Institute of Technology, Chicago, Illinois, in 2012, all in electrical and computer engineering. He is currently an assistant professor in the Department of Computer Science, City University of Hong Kong, Hong Kong. His research interests include cloud security, big data, multimedia security, and mobile security.