IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 3, 2020, accepted August 1, 2020, date of publication August 5, 2020, date of current version August 17, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3014424

Protecting Private Attributes in App Based Mobile User Profiling

IMDAD ULLAH[1], ROKSANA BORELI[2], SALIL S. KANHERE[2], (Senior Member, IEEE), SANJAY CHAWLA[3], TARIQ AHAMED AHANGER[1], AND USMAN TARIQ[1]

[1]College of Computer Engineering and Sciences, Prince Sattam bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia
[2]UNSW Sydney, Sydney, NSW 2052, Australia
[3]QCRI, Hamad Bin Khalifa University, Doha 34110, Qatar

Corresponding author: Imdad Ullah (i.ullah@psau.edu.sa)

This work was supported by the Deanship of Scientific Research at Prince Sattam Bin Abdulaziz University under Grant 2020/01/13203.

ABSTRACT The Analytics companies enable successful targeted advertising via user profiles, derived from the mobile apps installed by specific users, and hence have become an integral part of the mobile advertising industry. This threatens the users' privacy, when profiling is based on apps representing sensitive information, e.g., gambling problems indicated by a game app. In this work, we propose an app-based profile obfuscation mechanism, ProfileGuard, with the objective of eliminating the dominance of private interest categories (i.e. the prevailing private interest categories present in a user profile). We demonstrate, based on wide-range experimental evaluation of Android apps in a nine month test campaign, that the proposed obfuscation mechanism based on *similarity* with user's existing apps (ensuring that selected obfuscating apps belong to non-private categories) can achieve a good trade-off between efforts required by the obfuscating system and the resulting privacy protection. We also show how the *bespoke* (customised to profile obfuscation) and *bespoke*++ (*resource*-aware) strategies can deliver significant improvements in the level of obfuscation and (particularly *bespoke*++) in the use of mobile resources, making the latter a good candidate strategy in resource-constrained scenarios e.g., for fixed data use mobile plans. We also implement a POC ProfileGuard app to demonstrate the feasibility of an automated obfuscation mechanism. Furthermore, we provide insights to Google AdMob profiling rules, such as showing how individual apps map to user's interests within their profile in a deterministic way and that AdMob requires a certain level of activity to build a stable user profile.

INDEX TERMS Privacy, targeted ads, mobile apps, obfuscation, user experience.

I. INTRODUCTION

Online user profiling for the purpose of targeted, personalised advertisements has become an invaluable component of the advertising industry. While in-browser information collection was the initial focus, in the mobile environment the prevalence of apps has enabled further development of the mobile-specific profiling and targeting. Within the mobile advertising environment, companies such as Google Analytics for mobiles[1] or Flurry[2] collect a rich set of information from the apps use [1], to target individuals with ads that will result in sales and increase the revenues for all stakeholders in the advertising ecosystem. However, from the user's point of view, the leakage of their personal information is considered as a threat to privacy and all parties in the ecosystem need to adhere to ethical and legal constraints in regards to user's data [2].

Prior research has addressed the prevalence of collecting individual's information and associated privacy threats [3] including the extent of web tracking [4], information leakages through inference attacks based on monitoring displayed ads in browsing sessions [5], [6], ad library API calls [7], and tracking user's history through browser fingerprinting and cookie syncing [8], [9]. In line with these, there are a number of works on protecting the user's privacy, by using techniques such as obfuscating user data, proposed in different contexts, e.g., for browsing [10] and location based services [11]. Privacy preserving advertising systems that utilise a mix of

The associate editor coordinating the review of this manuscript and approving it for publication was Giuseppe Desolda.

[1]Google Admob https://apps.admob.com
[2]Flurry Analytics https://www.flurry.com/analytics.html

cryptography techniques and obfuscation mechanisms have also been proposed, e.g., [12], [13]. However, we note that the majority of prior works in private advertising systems have focused on browser based ads [14]–[17] whereas only few works address the in-app targeted ads, which is the primary focus of our work [18].

In this paper, our main focus is on protecting individual's privacy within the mobile advertising system from both profiling of specific attributes (selected as private), which the analytics companies may use for targeted advertisements, and ads targeting based on these attributes. We note that the advertising companies usually group mobile applications into thematic categories (e.g., on Google Play store[3] or the Apple store[4]). Furthermore, the various items in a user profile, derived from the use of installed apps (*interests* in Google's terminology), are also gathered in similar categories. Our purpose is to obfuscate a selected *interest* category within a profile along with the corresponding apps category, which may be considered private by the user. For instance, the user may not wish the categories of gaming or porn to be included in their profile, as these would reflect heavy use of corresponding (gaming and porn) apps. This would be of particular relevance in common scenarios such as when a user uses a business mobile device for private purposes. In this paper, we extend our previous work [18] and present a mechanism to deflect the attacks on privacy via app-based profiling and the ad-based inference attacks. We provide the following contributions.

We examine the profiling process used by a major analytics network i.e. Google AdMob, by **investigating the relation between the mobile app characteristics and the resulting user profiles**. We carry out wide-range experiments for a period of over 9 months; based on our extensive experimental results we show that the **profile interest categories can be predicted from the app categories** with a high accuracy of 81.4%, reflecting the dominating interests' categories. We moreover determine the broad profiling rules: that profiles are characterised by an aggregation of interests determined by use of individual mobile apps, that the mapping of apps to sets of interests is deterministic and that it requires a minimum level of activity (up to 72h) to build a stable user profile.

We **propose ProfileGuard, an app-based obfuscation mechanism** that, **for a selected private interest category**, reduces the level of dominance of private interest category in a user profile. We **demonstrate, through a POC implementation, the feasibility of a fully automated obfuscation system** of ProfileGuard app, which installs and runs the selected *obfuscating* apps. Two main obfuscation approaches are considered: (a) based on most *similar* (obfuscating) apps from any non-private app category, similar to the app-based recommender systems; and (b) *bespoke*, which is customised to the user's profile interests (we also take another variant of this strategy, *bespoke++*, that takes into account the *resource* use). In addition, the obfuscation system *randomly* chooses apps from any non-private app category and compares with other approaches. We demonstrate via experiments that the *similarity* based strategy brings the best tradeoff between the app *cost* and the reduction of threat to the privacy by reducing the magnitude of dominance level of private interest category in a user profile. Note that *cost* is measured by the number of newly introduced obfuscating apps, which are required to provide privacy protection for a selected private interest category. For instance, based on *similarity*, a single obfuscating app decreases the dominance of a Comics category by 50% compared to a 33% decrease for *random* obfuscation strategy. On the other hand, the *bespoke* and *bespoke++* strategies provide an average of 48.1% improvement and a low average app *cost*of only 30% required for a non-private category to become dominant in a profile. This compares to 99% and 123% respectively for the *similarity* and *random* based obfuscation strategies. We note, however, that these strategies require the knowledge of mapping of apps profiles to user profiles that presently requires significant pre-processing overhead. We are planning for future work to consider different ways for minimising this overhead by evaluating approximate profile evaluation.

We **evaluate the overhead** introduced by running the obfuscating apps. We show that there is a wide range of the resulting overhead, both in the mobile device *resources* like *battery* power, *CPU* and *memory* use and in the use of communication resources. We show that, in line with the lower number of apps required for obfuscation, the *bespoke* and *bespoke++* strategies have lower *resource* requirements. Additionally, the *resource*-aware *bespoke++* strategy can decrease the bandwidth overhead introduced by *similarity* and *random* strategies by, respectively, 64% and 65% and by 57% compared to *bespoke* when a single app is used for obfuscation. This is increased to 81%, 77% and 66% improvement compared to the same strategies for *complete* obfuscation, demonstrating a significant benefit to the user and a promising direction to further explore.

Finally, we **show the impact of proposed obfuscation strategies on targeted ads**. We demonstrate that user profiles consist of a wider range of interests from different interest categories and will result in a broader range of ads delivered to the corresponding mobile user devices. This will also reduce the level of ads targeting and the likelihood of ads-based profile inference.

We organise this paper as follows: Section 2 presents background on the ads ecosystem. In Section 3, we present system model and describe various obfuscation strategies. Section 4 presents details of the system evaluation and our experimental setup. We present the experimental results using different obfuscation strategies in Section 5. We further discuss the pros and cons of different approaches in Section 6. Prior work is presented in Section 7 and our conclusions in Section 8.

[3]Google Play store: https://play.google.com/store

[4]Apple App store: https://www.apple.com/ios/app-store/

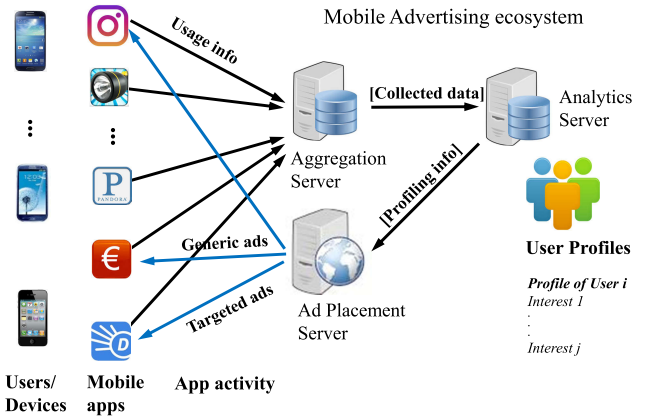


FIGURE 1. The In-App advertising ecosystem and the information flow between different parties.

II. BACKGROUND

This section presents background on the mobile advertising ecosystem. A representative in-app mobile advertising ecosystem is shown in Figure 1, which comprises users, the analytics components and the ad placement system. **User devices** (shown in the left side of Figure 1) have a number of apps installed; these are utilised by mobile users, with specific use frequency and duration. We note that majority of mobile apps include analytics SDKs that instantaneously monitor user's actions and send this information to the Ads system network, for tracking user's activities and for serving ads [19]. The advertising system also comprises the **Aggregation Server**, which aggregates user's information sent by ad analytics libraries, and the **Ads Placement Server** that delivers ads within the mobile applications. An important entity within the advertising ecosystem is the **Analytics Server**, that develops user profiles (associated with specific mobile devices and corresponding users) from the collected data. This data relates to the usage of mobile apps and the success of displayed ads. We note that user profiles comprise a number of *interests*, which specify the use of related apps, e.g. games, business apps, etc. Examples of user profiles derived through use of mobile apps include Google AdMob[5] and Flurry [20] (we note this is only visible to the app developers).

Targeted ads are served to mobile device users according to their individual profiles. We note that other i.e., *generic ads* are also served to diverse profiles according to the ad network's advertising strategy. However, in line with our focus on targeting, in this work our primary interest is in the *targeted ads*.

We consider two attack scenarios. First, the *direct* attack where the Analytics network (in this work, we focus on Google AdMob) legitimately derives user profiles. Second, the *indirect* attack, involves a third party intruder; where the ads traffic (sent in clear text [21] to mobile devices) is being monitored, to infer user profiles based on the served targeted ads. In both attack scenarios, the user is willing to receive relevant ads on selected topics of interest and is not opposed to profiling in general; however, the user does not wish for specific parts of their profile (considered private) (*attributes*) to be known to the Analytics network or any other party, or to be used for targeting.

[5]The inferred Google AdMob profile can be accessed via the Google Settings system app on Android devices.

III. USER PROFILE OBFUSCATION

Our aim is to obfuscate the user profile, so that an adversary (either the analytics companies or an observer listening in to the ad/control traffic) cannot determine the nature of user's specific (private) mobile applications, which produce selected profile interests.

We accomplish the obfuscation using this ProfileGuard system. The ProfileGuard system comprises of a *bespoke* app that performs different tasks, such as, providing user profiling information exchange, recommendations for specific standard (*obfuscating*) apps, and the automated running of obfuscating apps and the server that implements the *obfuscating* app selection. We first explain the system model to describe the obfuscation methodology.

A. SYSTEM MODEL

A mobile app marketplace comprises a set of $\mathcal{A}$ mobile apps, that are usually (e.g., in Apple App store or in Google Play) organised within Φ categories. We represent a mobile app by $a_{i,j}, i = 1, \ldots, A_j$, where A_j is the number of apps that belong to an app category $\Phi_j, j = 1, \ldots, \Phi$, where Φ the number of various apps categories in the marketplace.

Each app $a_{i,j} = \left\{\left\{\kappa_{m,i,j}\right\} : a_{i,j} \in \mathcal{A}\right\}$ can be characterised by keywords $\kappa_{m,i,j}$, derived e.g., from various characteristics associated with an app described in the app marketplace, such as title and description of a apps etc. Here $m = 1, \ldots, M_{i,j}$, where $M_{i,j}$ is the number of keywords for app $a_{i,j}$, $i = 1, \ldots, A_j, j = 1, \ldots, \Phi$.

We characterise users by a combination of mobile apps installed on their mobile device(s), consisting of a subset S_a of app set $\mathcal{A}$ and their associated keywords. An *Apps profile* K_a can therefore be defined as $K_a = \left\{\left\{\left\{\kappa_{m,i,j}\right\}, \Phi_j\right\} : a_{i,j} \in S_a\right\}$.

Analytics companies (e.g., Google or Flurry) characterise and profile users by defining a set of profile *interests* G, i.e., characteristics that may be assigned to individual users. We note that *interests* are commonly grouped into different categories, with an *interest* $g_{k,l}, k = 1, \ldots, G_l$, where G_l is the number of interests that belong to an *interest* category Ψ_l, $l = 1, \ldots, \Psi$. Ψ is various *interest* categories defined by an analytics company.

Profiling characterises the user by the combination of their interests, i.e., by a subset S_g of the full interest set G. An *Interest profile* I_g can therefore be defined as: $I_g = \left\{\left\{g_{k,l}, \Psi_l\right\} : g_{k,l} \in S_g\right\}$.

We focus on the *interests* derived from the use of installed apps although we note that there are various types of information may be used to generate user profiles. We presume that there is a fixed mapping of *Apps profile* to an *Interest profile*, defined by the mapping M, $M : \left\{K_a \rightarrow I_g\right\}$. This

TABLE 1. List of notations.

Symbol	Description
$\mathcal{A}$	Set of apps in a marketplace
Φ	Number of app categories $\Phi_j, j = 1, ..., \Phi$
S_a	Subset of apps installed on a user's mobile device
S_o	Set of obfuscating apps
$a_{i,j}$	An app $a_{i,j} \in A, i = 1, ..., A_j, j = 1, ..., \Phi$, A_j is the number of apps in Φ_j
K_a	App profile consisting of $\kappa_{m,i,j}$ keywords
$\kappa_{m,i,j}$	Keywords of $a_{i,j}, m = 1, ..., M_{i,j}, i, j \in a_{i,j}$
G	Set of interests in Google interests list
Ψ_l	Interest category in $G, l = 1, ..., \Psi$, Ψ is the number of interest categories defined by Google
S_g	Subset of Google interests in G derived by S_a
I_g	Interest profile consisting of $g_{k,l}, g_{k,l} \in S_g$
$g_{k,l}$	An interest in $I_g, k \in G_l, l \in \Psi$
Ψ_p	Private interest category in a user's profile
Φ_d	User's private app category
D	Dominance ratio of a user's profile
U_s	Usability of an app
C	App cost of obfuscation
$R_{i,j}$	App *resource* overhead, when used for obfuscation

also includes the mapping of app categories Φ_j to interest categories Ψ_l.

We define Ψ_p a *private* interest category that a mobile user considers as private and wishes to protect. We assume that there is a corresponding private app category Φ_p, defined by M.

Furthermore, we describe the *dominance ratio* D as the ratio of number of interests in a selected category Ψ_p, $\{|\{g_{k,p}\}|\}$ and the maximum number of interests in any of the other categories Ψ_l present in a user profile. We note that $D > 1$ indicates that a private interest category is dominant in a user profile, i.e., the profile has the largest number of corresponding interests.

$$D = min(\{|\{gk, p\}|\}/\{|\{gk, l\}|\}) : \\ \forall \Psi_l \neq \Psi_p, \quad g_{k,l} \in S_g \tag{1}$$

The aim of the proposed obfuscation mechanism is to generate a new obfuscated profile I'_g by reducing the *dominance ratio* of a selected private category Ψ_p in a user profile I_g. We achieve this by installing and using selected *obfuscating* apps S_o, in addition to the original set of apps (S_a), resulting in an obfuscated app set S'_a. Table 1 summarises the notations used in this paper. We define various strategies for selecting obfuscating apps in the following sub-section.

B. OBFUSCATION STRATEGIES

We describe the following profile obfuscation strategies to select candidate apps. We note that all the obfuscation strategies select the candidate obfuscation apps from categories other than the category Φ_p, which correspond to the profile (private) category that the user is protecting Ψ_p. We state that the user is protecting any number of private interests categories $\Psi_p, p = 1, \cdots, \Omega$ and Ω is the number of different interests categories that are *private* to the user.

1) SIMILARITY BASED STRATEGY

This strategy selects the candidate obfuscating apps based on a similarity metric. The obfuscating set S_o consists of apps with highest similarity to the currently installed apps in S_a. We calculate the set S_o^{sim} for a single obfuscating app and the private app categories Φ_p corresponding to the private interest categories Ψ_p as:

$$S_o^{sim} = \left\{ \begin{array}{l} a^o \leftarrow \left|sim(a_{i,p},\ a_{q,r})\right|_{\max} : \\ \forall r = 1, \Phi;\ \forall p = 1, \Omega;\ a_{i,p} \in S_a; \\ a_{q,r} \notin S_a;\ \Phi_r \neq \Phi_p \end{array} \right\} \tag{2}$$

The *utility* (defined in Section III-C) is the major motivation for the use of *similarity* by taking into consideration that the potential obfuscating apps may be of real interest to the user. We calculate the *similarity* based on app keywords using the $tf-idf$ (*cosine similarity*) metric [22]. Note that this strategy is not metric-specific while other similarity metrics can also be utilised.

We represent $\kappa_{m,i,p}$ and $\kappa_{m,i,r}$ as the set of app keywords respectively in $a_{i,p}$ and $a_{i,r}$. Let t and d be a particular keyword respectively in $\kappa_{m,i,p}$ and $\kappa_{m,i,r}$, then the *term frequency*, i.e. frequency of a particular keyword t in d, is $tf_{t,d}$. The *inverse document frequency* of $t \in \kappa_{m,i,p}$ within d can be calculated as $idf_t = log\frac{N}{df_t}$, where N is the set of apps keywords and df_t is the *document frequency* i.e. the number of apps keywords that contain t. Hence, the *tf-idf* can be calculated as $tf - idf_{t,d} = tf_{t,d} \times idf_t$. Consequently the score for $\kappa_{m,i,p}$ can be calculated as:

$$score\left(\kappa_{m,i,p}, d\right) = \sum_{t \in \kappa_{m,i,p}} tf - idf_{t,d} \tag{3}$$

The similarity score is calculated for the entire set of private interest categories i.e. $p = 1, \cdots, \Omega$.

2) BESPOKE STRATEGY

We assume that the mapping M of *Apps profile* (we take the apps keywords and categories) to an *Interest profile* is available for individual apps. Practically, we conduct a series of tests to derive the user profile interests for specific apps (Section IV presents experimental results for selected apps). Following, we select as the obfuscation candidate apps that generate (known) interests, which belong to categories that are already present in the set of interests from a specific user *Interest profile* S_g. Assuming a single obfuscating app along with the private app categories Φ_p (and the corresponding private interest categories Ψ_p) we calculate obfuscating apps using *bespoke* strategy as:

$$S_o^{bes} = \left\{ \begin{array}{l} a^o \leftarrow a_{q,r} : \\ \forall r = 1, \Phi;\ \forall p = 1, \Omega;\ g_{k,r} \notin \Psi_p; \\ g_{k,r} \in S_g;\ a_{q,r} \notin S_a \end{array} \right\} \tag{4}$$

To achieve the *complete* obfuscation (i.e. have $D < 1$ for the private category), we select multiple obfuscation candidates:

$$S_o^{bes} = \left\{ \begin{array}{l} a^o \leftarrow a_{q,r} : \\ \forall r = 1, \Phi; \forall p = 1, \Omega; g_{k,r} \notin \Psi_p; \\ g_{k,r} \in S_g;\ a_{q,r} \notin S_a; a_{q,r} \notin S_o^{bes} \end{array} \right\} \tag{5}$$

3) BESPOKE++ STRATEGY

Building on the *bespoke* strategy approach, we make a further assumption that the *resource* overhead $R_{i,j}$ of individual apps $a_{i,j}$ is known. I.e., that the pre-processing experiments to generate mapping of apps to interests include characterising the use of resources like bandwidth consumption, *CPU* and *memory* use (we will address *resource* consumption in detail in Section III-C). From the candidate set of *bespoke* apps, as per Equation 5, we select the candidate obfuscating apps that would generate the lowest amount of *resource* overhead:

$$S_o^{bes++} = \left\{ a^o \leftarrow \left| R_{q,r}\left(a_{q,r}\right)\right|_{\min} : a_{q,r} \in S_o^{bes} \right\} \tag{6}$$

4) RANDOM STRATEGY

Finally, for the *random* strategy we randomly select the candidate apps, however still with the awareness of user privacy. This strategy chooses apps from any non-private app category $\Phi_j \neq \Phi_p$.

$$S_o^{rnd} = \begin{cases} a^o \leftarrow a_{q,r} : \\ \forall p = 1, \Omega;\ a_{q,r} \notin S_a;\ \Phi_r \neq \Phi_p \end{cases} \tag{7}$$

C. EVALUATION METRIC

We define the evaluation metrics used to calculate the *utility* and app *cost* of obfuscation in order to compare various obfuscation strategies for selecting obfuscation apps. Additionally, to assist in the refining and implementation of the *bespoke++* strategy, we define the app *resource* overhead, introduced when they are used for obfuscation.

There are few research works related to obfuscation of browser based profiling; the authors in [23] define the *utility* as the success rate for the removal of private query tags. Similarly for rating systems, the proposed *utility* metric is the level of suppression of original user preferences [24]. *Cost* is commonly defined as the ratio of obfuscating to original data [23].

1) UTILITY

We define *utility* based on two components: first, from the effectiveness of privacy protection viewpoint, we use as metric the level of reduction R_p of *dominance ratio* D of a selected private category Ψ_p in a user profile, achieved by obfuscation, with $R_p = D'/D$. Here D' is the new *dominance ratio* resulting from using apps in S'_a.

Next, we introduce the notion of *usability* of the obfuscating app. The *usability* relates to the probability that a user would actually use this app, rather than just install and run it for the purpose of privacy protection. Considering the common use of similarity in recommender systems to suggest [25] e.g., an app that a user is likely to be interested in, we define the *usability* U_s of an app a_o, in regards to a user with a specific *Apps profile*, as the ratio of similarity between this app and any of the apps in the original set of installed apps S_a and the maximum similarity of any other app from $\mathcal{A}$, not present in S_a and apps from the same set:

$$U_s = min\left(\frac{sim\left(a_o, a_{i,p}\right)}{sim\left(a_{q,r}, a_{i,p}\right)}\right) : \\ a_{i,p} \in S_a, \quad a_{q,r} \notin S_a \tag{8}$$

Combining the two, the total *utility* can then be calculated as: $U_T = \alpha \cdot R_p + \beta \cdot U_s$.

The magnitude of weighting factors α and β can be tuned by the obfuscating mechanism (e.g., as part of user preferences). We note that the *similarity* based strategy is targeting high U_s, although it limits the app choices to non-private app categories.

2) COST AND RESOURCE OVERHEAD

Although the *cost* and overhead could be considered as equivalent terms in the context of introducing new activities (a set of new apps) that result in spent time and *resources*, for the sake of clarity we use two separate terms.

We define app *cost* C as a metric that relates to the reduction of the overall user's time available to the original (non-obfuscating) apps. In the basic scenario where, on the average, the usage of all apps is uniformly distributed within a time period, this is equivalent to the ratio of the number of obfuscating apps in S_o, that need to be installed and used to achieve privacy protection, and the size of the original app set S_a. *Cost* is therefore defined as $C = |S_o|/|S_a|$.

We consider the app *resource* overhead $\mathcal{R}$, to be the overall use of resources by the obfuscating apps. For each app $a_{i,j}$, there will be a corresponding overhead $R_{i,j}$, comprising broadly of *communication* $R_{i,j}^{com}$, *computing* $R_{i,j}^{cmp}$, and *battery* consumption overheads $R_{i,j}^{b}$.

$$R_{i,j} = R_{i,j}^{com} + R_{i,j}^{cmp} + R_{i,j}^{b} \tag{9}$$

We experimentally evaluate various components of the overall overhead $\mathcal{R}$ in Section IV-E. To simplify the *bespoke++* strategy implementation, we only consider the *communication* overhead $R_{i,j}^{com}$ of apps, when selecting the obfuscating apps under this strategy.

D. ProfileGuard: THE OBFUSCATING SYSTEM

The ProfileGuard system implements our proposed obfuscation mechanism. Various components are presented in Figure 2. A user installs and runs the ProfileGuard app in order to protect a specific aspect of their profile; this approach is similar to the existing app recommender systems, such as AppBrain [26]. This app acquires the information about existing apps installed on a user's device, to interact with the user in regards to the selection of private attributes to be protected; it also receives the list of candidate obfuscating apps and presents them to user and can be used to install and run the selected obfuscating apps. The Obfuscation engine (i.e. server) calculates the user profile, based on the installed apps and derives the set of obfuscating apps, according to specific obfuscation strategy.

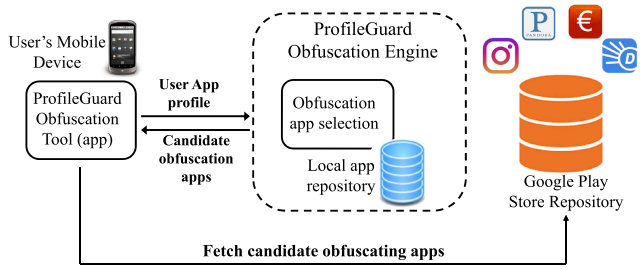


FIGURE 2. ProfileGuard obfuscating system includes the ProfileGuard app client and the obfuscation engine (server) components.

To forward the list of the installed apps to the server (the ProfileGuard Obfuscating Engine), the client starts a `Thread` by opening a `Socket (SERVER_ADDRESS, SERVER_PORT)` with the corresponding server details. Using this socket the client writes this data, using `BufferedWriter`. The Obfuscating Engine then selects the obfuscating apps, according to one of the strategies presented in Section III-B, and forwards the list of apps to the client. Each app is displayed to the user as a link to the Google Play store, by invoking the `startActivity(new Intent(Intent.ACTION_VIEW, Uri.parse("market://details?id=" +appPackageName)))`. The description, i.e., the activity name (the app to be installed) is specified using `appPackageName` function, and is provided using the `Intent` class.

The obfuscating apps are then run using the function `startActivity`, in a loop comprising all (obfuscating) apps, in a singgle `Thread` and for a specified amount of time required to generate new profile interests in the ad system. The Intent is set with the `ComponentName` and the execution of all obfuscating apps is stopped using the same `Thread`.

We envisage that ProfileGuard would be active when the phone is not in use (although the current POC does not implement scheduling, we have verified that this is feasible). The performance can be enhanced by scheduling these activities to only when the phone is being charged (so that to reduce energy costs) and is in Wi-Fi coverage (to help reduce the bandwidth overhead). To improve efficiency, the client may also keep track of the set of originally installed apps, the obfuscating apps suggested by the Obfuscating Engine, the set of installed apps, and the obfuscating apps that were suggested by the Obfuscating Engine but were not installed by the client.

IV. EXPERIMENTAL EVALUATION: SETUP AND OBSERVATIONS

In this section, we outline the methodology used to evaluate the proposed obfuscation strategies, considering both their *utility* and *resource* overhead. We start with an overview of the experimental setup along with the collected datasets. First, we investigate the rules for building user profiles. Next, we demonstrate the degree of private information revealed by the unprotected user profiles. Subsequently, we discuss the *resource* use measurement methods and further provide insights into resource-usage by various apps.

A. EXPERIMENTAL SETUP: PROFILING AND ADS

Among the initial goals of this study is to gather insights about specific rules used by the advertising companies to establish the *Interest profile* entries in addition to the temporal evolution of different profiles. We note that the profiling rules are proprietary to the advertising companies; however, there are previous studies that establish the relevance of app-based profiling (and hence ads targeting, since mobile apps send targeting information to ad networks [27], [28]) to the mobile ads ecosystem. In our present work, we focus on Google AdMob since it is the leading market in mobile user profiling and mobile advertising. However, we note that our methods can be readily applied to other ad or analytics networks. Recall that, our focus is exclusively on profile interests derived from the usage of mobile apps in the *Apps profile*.

For evaluating app based profiling rules and other experimentations, we randomly select 27 different apps categories from Google Play store. We note that the mobile apps in Google Play store are classified into 34 categories (e.g., Entertainment, Business, etc). Frits, we select top 100 free apps from randomly chosen 27 categories and further narrowing down this selection to 10 highest ranked apps from selected categories. We had to make sure that these apps receive ads (we detail the relevance of ads to building of Google profile in Section IV-B).

Google profile interests[6] are also hierarchically grouped under 25 interest categories with 2042 specific interests. We run a series of *profiling experiments* in order to determine temporal *Interest profile* evolution and further demonstrate the mapping of *Apps profile* categories to *Interest profile* categories. For these experiments, we install the full set of 100 apps from a selected category on test phones; we use a subset of 10 apps from this set, one at a time, where each app is run for a period of approximately 2.4 hours within any 24h testing period.

We use an automated app for all the experimentation discussed in this section, which enables all the communication between a PC and connected Android devices through the use of the Android Debug Bridge.[7] Furthermore, all the ads (including the control traffic communicated for tracking/profiling/personalisation purposes) traffic was collected using *tcpdump* [29] and saved to a local database during the entire experimentations. Correspondingly, before starting each experiment we reset the profile[8] to ensure that *Interest profile* is only resulting from the currently installed and used apps. Similarly, in parallel, we also run a set of

[6]Google profile interests are listed in https://adssettings.google.com/authenticated?hl=en, displayed under the 'Ad personalization'.

[7]developer.android.com/tools/help/adb.html

[8]The profile is reset by using the 'Reset advertising ID' option in the Google settings system app.

phones with the same app configuration, but with 'Opt-out of interest-based ads' setting enabled in Google settings system app. This was done in order to have a base reference for both generated user profile and received ads. We manually check the profile on each of the phones in 6h intervals (note this cannot be automated). We run the *profiling experiments* for all the app categories for 24 hours a day and for 5 months; we use 10 smartphones due to practical limitations of the experimental environment.

We note that the collected app traffic is also used to calculate the *resource usage*, as detailed in Section IV-D.

We use a second (*mapping*) experimental setup in which we only select a single mobile app from an app category and run it (in an automated way) for a period of up to 96h. The purpose of these experimentations is to calculate the mapping of specific *App profile* to *Interest profile* so that the contribution of individual app can be determined in an *Interest profile*. During the experiments, we reset each test phone before running any experiment, similarly, we perform manual profile verification every 6h. These experiments resulting in *Interest profile* for the 270 highest ranked apps (that also receive ads) have taken around 3 months to complete. We note that findings from these experiments also help us in selecting obfuscating apps for the *bespoke* obfuscation strategy.

Finally, we run a series of *obfuscation experiments* to verify the extent of profile obfuscation for selected obfuscation strategies. This process results in the evolution of *Interest profile* with the new subset S'_a comprising a combination of the original apps set S_a (used for profiling experiments) and the obfuscating set S_o. In line with the *profiling experiments*, we run each app, in succession, from S'_a for a total of $24h/|S'_a|$ in any 24h period. Recall, as detailed in Section IV-B, that apps must have activity for a minimum time duration in order to generate interests within the *Interest profile*. We make it sure that we run apps from S'_a for a sufficient time period in order to achieve the desired outcome.

We collect the following dataset from the above experiments respectively containing the snapshots of *Interest profiles* taken in 6h intervals for: (a) multiple apps from a single category i.e. we call this dataset a `multiapp-profile`; (b) individual apps from all categories i.e. `singleapp-profile` (c) the set of apps containing a mix of multiple apps from individual categories and a single obfuscating app i.e. `obfuscated-oneapp` and (d) as, previous, but for multiple obfuscating apps i.e. `obfuscated-complete`. Likewise, following datasets contain all the data and control traffic collected from the same experimental configurations: `multiapp-traffic`, `singleapp-traffic`, `obfuscated-oneapp-traffic` and `obfuscated-complete-traffic`. Overall, we run these experiments for a period of 9.5 months and using a total of 2700 apps, which resulted in 140 collected profile interests and 246657 ads.

In addition, we collect data related to apps and Google profile interests, which we use for analysis of user profile protection strategies. The first dataset comprise of the top 100 apps from each category of Google Play store, including all keywords and apps categories. The second dataset includes the 2042 Google profile interests.

It is important to note that we perform all our experiments in a single geographical location. However, we believe that this methodology or the findings of this study are not compromised since the user profiling mechanism is unrelated to locations. On the other hand, the volume and diversity of ads pool are related to geographical areas. However, our focus is to analyse the delivered ads only in regards to profile interests based (i.e. targeted advertising) targeting and we focus on the differences resulting from profile changes.

B. INSIGHTS ON PROFILING RULES AND APP USE THRESHOLD

In this section, we outline our observations about the development of user *Interest profiles*, based on app characteristics and usage.

We note that all the apps in an *Apps profile* (i.e. installed in a specific mobile device) do not necessarily contribute to the profile.[9] This is due to the fact that every mobile app does not come with the AdMob SDK i.e. to receive ads, in order to generate one or more interests. We verify this via testing 1200 apps selected from a subset of 12 randomly chosen apps categories, for duration of 8 days. We note the resulting *Interest profiles* from running these apps on all the test phones indicating ''unknown'' interests. Additionally, we note that the Google analytics generates the *Interest profiles* in a deterministic way, i.e. after a certain period of activity, specific apps will always generate selected (identical) interests. Similarly, we note that the resultant *Interest profile* from the use of multiple apps is an aggregation (union) of individual interests for entire set of apps.

In addition, we note that the profiling process requires a minimum level of activity from one or more apps in order to generate an interest(s) in an *Interest profile*. Our experimental evaluations indicate that the mobile apps need to be active within of 24h, with a minimum activity of about 1.5h for individual apps. We also have observed, in regards to temporal evolution of user profiles, that in some cases additional interests are added to the profile in subsequent 72h period. For this we run a similar experiment with a number of apps for 8 days. However, after 4 days of the profile evolution, it becomes stable and it does not result in any changes with additional app activity.

We now evaluate the mapping M between the app categories in *App profile* and interest categories of *Interest profile*. To do this, as defined by Google, we first unify both types of categories and then experimentally evaluate how this mapping is carried out in practice. Table 2 presents the resulting matches form the original lists of Google Play app's categories to Google interest's categories. We note that 11 apps

[9]The full list of installed apps on an Android based phone is included in the `library.db` file /data/data/com.android.vending/databases/library.db.

TABLE 2. Mapping of Google Play store apps categories to Google interest categories.

Google Play Apps Categories	Google Interest Categories	G.Apps to G.Interests
Books & Reference (BR)	Arts & Entertainment	Books & Literature
Business (BU)	Auto & Vehicles	Business & Industrial
Comics (CO)	Beauty & Fitness	Arts & Entertainment
Communication (CM)	Books & Literature	Internet & Telecom
Education (ED)	Business & Industrial	Jobs & Education
Entertainment (EN)	Computers & Electronics	Arts & Entertainment
Finance (FI)	Finance	Finance
Game (GA)	Food & Drink	Game
Health & Fitness (HF)	Games	Beauty & Fitness
Libraries & Demo (LD)	Hobbies & Leisure	Reference
Lifestyle (LD)	Home & Garden	Hobbies & Leisure
Live Wallpapers (LW)	Internet & Telecom	Online Communities
Media & Video (MV)	Jobs & Education	Arts & Entertainment
Medical (ME)	Law & Government	Science
Music & Audio (MA)	News	Arts & Entertainment
News & Megazines (NM)	Online Communities	News
Personalization (PE)	People & Society	Online Communities
Photography (PH)	Pets & Animals	Arts & Entertainment
Productivity (PR)	Real Estate	Computers & Electronics
Shopping (SH)	Reference	Shopping
Social (SO)	Science	People & Society
Sports (SP)	Shopping	Sports
Tools (TO)	Sports	Computers & Electronics
Transportation (TR)	Travel	Business & Industrial
Travel & Local (TL)	World Localities	Travel
Weather (WE)		News
Widgets (WI)		Computers & Electronics

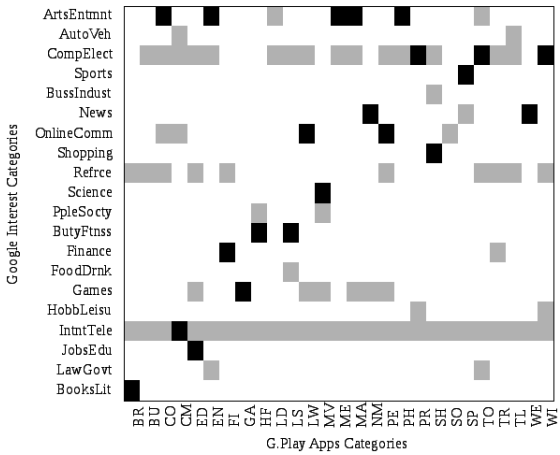


FIGURE 3. Relation between apps characteristics and Google interests: black denotes a category match, grey that an interest from a non-matching category is present in the profile.

categories can be directly matched to top level interest categories by comparing category names (e.g., Business app category with Business & Industrial interest category). Further, 13 apps categories directly matches to a lower category in the interest category hierarchy; an additional 10 categories are listed as second and additional three categories as third hierarchical level of interest categories. We manually map three app categories to interest categories i.e. Medical to Science, Widgets to Computers & Electronics and Lifestyle to Beauty & Fitness. We note that the Medical app category is only challenging category for finding a match, as this is clearly considered as potentially private when defining profiling categories. For this, we consider Science to be the closest match as it includes anatomy - although this only covers a small part of Medical topics, we believe that this mapping minimises the likelihood of false positives.

Figure 3 shows the relationship, results of our *profiling experiments* based on the `multiapp-profile` dataset, between the *Interest profiles* and (generated from) the *Apps profiles*. It can be observed that the interests in *Interest profiles* contain a matching category in majority of cases (81.4%) showing a direct threat to the user privacy. We also observe that only 5 categories do not have direct matches. Therefore, in the remainder of this paper, this mapping is being used to evaluate both the impact of obfuscation and the threats to the privacy, under the assumption that a user wishes to keep a selected interest (and app) category private.

C. PRIVACY EXPOSURE VIA UNPROTECTED PROFILES

In this section we show comprehensive results of *profiling experiments* (using the `multiapp-profile` dataset), demonstrating the dominance of profiling categories that match the selected interest categories in an *Interest profile*. Figure 4(a) shows the number of unique interests and the number of interests in specific categories, for all app interest categories. As an example, it can be noted that the *Interest profile* generated by Games apps category results in seven unique interests, among which, one belongs to Internet & Telecom and 6 to Games interest categories. Overall, during the *profiling experiments*, we capture 140 interests including 78 unique interests belonging to 20 (out of 24) Google interest categories. Furthermore, we observe that Arts & Entertainment and Computer & Electronics interest categories contribute in the majority of the *Interest profile* i.e. respectively in 21.9% and 19.3%. Following, this is followed by Internet & Telecom, Reference, and Games categories being close to 10% while the remainder of interest categories being represented by between 1-4%.

Figure 4(b) presents the *dominance ratio* D of selected interest categories (matching to app category) assuming all categories are chosen private. It can be observed that an average value of $D = 1.35$, which is quite broad across all categories. The critical value of $D > 1$ (which indicates that a private category is dominant in an *Interest profile*) is present in 37% of categories, with a further 31% having $D = 1$. This is consistent with the results presented in Figure 3, which again indicates a high level of threat to user privacy from the presence of private interests in the matching private categories. The Games category has the highest privacy concerns with higher *dominance ratio* and could justifiably be considered as private.

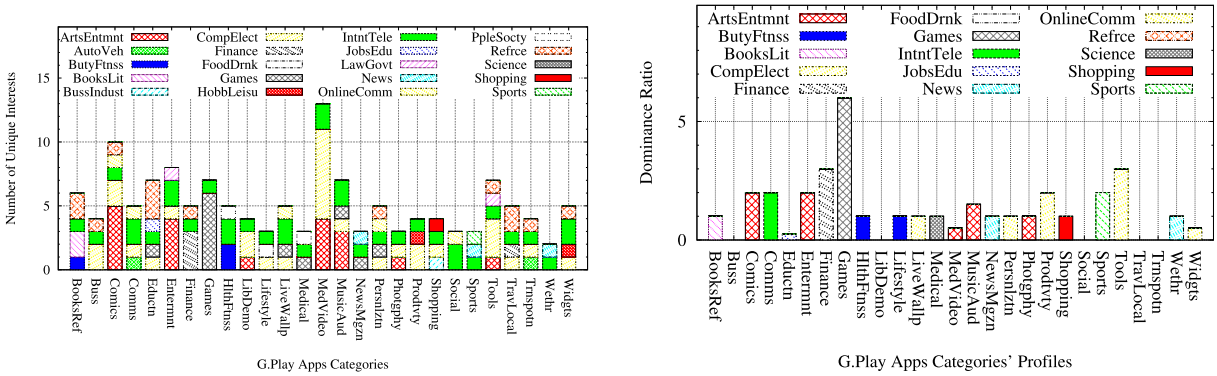
In the following sub-section, we will describe the methodology used for *resource* utilisation experiments.

D. RESOURCE USE EXPERIMENTS

To evaluate the *communication* overhead, we utilise the traffic collected during the experiments as described in Section IV-A.

We rely on the Android SDK[10] utilities to automate our measurements for *computing* and *battery* consumption overheads. E.g., to calculate the current *CPU* usage when any

[10]https://developer.android.com/studio



(a) Unique Interests (b) Dominance Ratio

FIGURE 4. Unique Interests (a) and Dominance Ratio (b) of all Interest Profiles based on their respective apps Profiles.

single app is running, we execute the appropriate `command` (`adb shell top -m 10`) within the `Process p = Runtime.getRuntime().exec("command")`. To evaluate the *computing* overhead, we consider both the *CPU* use and *storage space* consumption.

The *storage space* consumption can be partitioned into *installation* storage space, the size of the internal *data* size i.e. storage used for apps' files, settings, accounts, databases, etc., and the *cache* storage i.e. temporary stored data, such as images, downloaded from the Internet. We use the `PackageStats` package to obtain various static values as defined within this package; we use `codeSize`, `dataSize`, and `cacheSize` respectively for calculating the app's *installation*, *data*, and *cache* storage consumption.

Finally, for *Battery* consumption, we use `adb shell dumpsys battery | grep level` to capture the current battery status. This is displayed by initiating the `startActivity()` of the `Intent` utility of the android SDK. To measure the power consumption for each app, we first charge the mobile battery to full (100%) and run the app for one hour, while accessing the Wi-Fi network. Although, in a real life scenario, users are likely to utilise apps (equally) on a mobile network, our interest is in the overhead of obfuscating apps that we envisage (as per Section III-D) would be used over Wi-Fi to reduce the overhead costs.

E. INSIGHTS ON RESOURCE USE

The *communication* overhead of the obfuscating apps is generated by both the app's core functionality and by the ad related traffic. As reported by prior work [30], and in line with our experimental results, ads are a major contributor to this overhead. Therefore, we approximate the overall *communication* cost by the traffic generated by ads.

From the bandwidth use point of view, ads traffic can be characterised by the *ad refresh rate* (although technically this is the inter-arrival time between two consecutive ads, it is referred to as the *rate* in AdMob[11]), the number of objects contained in an ad and the size of these objects. Table 3 shows a summary of various ad-related objects and messages, derived by analysing the traffic log files. We note that the average size of an ad is 16KBs and that the ad contains, on the average, 8-10 objects (images, javascript files etc). The average number of request/response messages for each ad is between 30-35.

TABLE 3. Various ad-related messages and objects and their average size in bytes.

Message Type	Size (bytes)	Message Type	Size (bytes)
DNS Query Request	68	Ok/(PNG)	1300
DNS Query Response	334	Ok/(JPEG)	1300
GET /pagead/images	578	Ok/(GIF)	1000
GET /simgad	252	Ok/(application/json)	240
GET /mads/gma	685	Ok/(text/javascript)	800
GET /imp	244	Ok/(text/css)	824
GET /geocode	224	no content	396
GET /generate_204	244	TCP Ack	66
GET /csi	595	TCP Syn	74
Ok/(text/html)	1200	TCP reassembled PDU	1434
200 Ok	496	POST	350

It is important to note that the *ad refresh rate* for any selected app is deterministic; this is configured by the developer at the time of registering the app on the app market (i.e. Google Play store). The range of supported values is between 12-120 seconds and our experimental results indicate that the *ad refresh rates* vary between 20-60 seconds, with values of 20, 30, 45 and 60 seconds being adopted by, respectively, 36%, 47%, 15% and 2% of all apps. Practically, as the ad sizes do not vary widely, the *communication* overhead for the *bespoke++* strategy described in Section III-B3 can be minimised by selecting the obfuscating apps that have the maximum overall *ad refresh rate* (while conforming to other *bespoke* criteria).

Figure 5 shows the cumulative distribution of the bandwidth used by apps (the outer graph) along with the probability density function (inner graph). The bandwidth use shown is for a total of 2.5 hours of running each app selected

[11] https://apps.admob.com

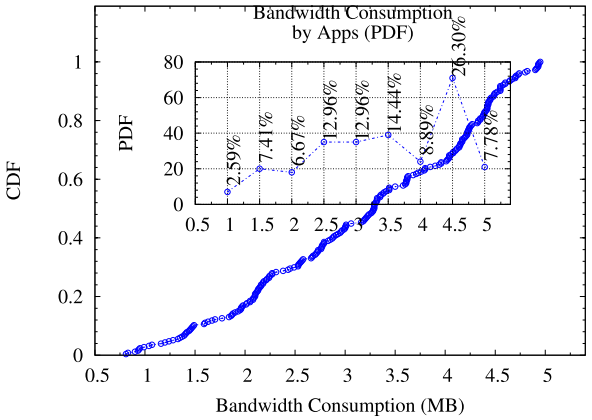


FIGURE 5. Bandwidth consumption (MB) calculated for ads by the apps during experiments.

as per Section IV-A. We note that the apps with lower *ad refresh rates* i.e. 20 and 30 seconds, use between 3MB and 5.5MB bandwidth in the measured time period; these apps represent around 70% of all apps used in our experiments. The remaining 30% of apps (with *ad refresh rates* of 45 and 60 seconds) utilise between 0.5MB and 2.5MB bandwidth. The PDF of network bandwidth consumption (inner graph in Figure 5), shows that the 26.30% of the apps have a high 4–4.5MB bandwidth use.

We now evaluate the *computing* overhead of tested apps. The measured *CPU* use varies, although not widely, across different apps: the CPU-intensive apps such as those from the Games category use between 25% to 30% of the full CPU; less-interactive apps such as *inkpad*[12] use between 15% to 20%.

We note that the app functionality is closely related to the *storage space* requirements: e.g., a *language translation* app will consume less *installation* storage space than the *data* storage space, as it needs to save library files (in *data* storage), so that a user can do an offline translation without accessing the Internet. Similarly, the *Facebook* app would consume more *data* storage space since it has to store different settings, user accounts, group settings etc. On the other hand, a *Google maps* app would use more *cache* storage space due to the requirement to e.g., save searched places. Representative apps with different combinations of storage space requirements are shown in Table 4.

Figures 6(a) through 6(c) show the distribution of different types of *storage space* along with the PDF of apps having different storage space requirements (inner graph in each of the sub-figures of Figure 6). We can observe, from the Figure 6 (a) (inner graph), that about half (54%) of the apps have a low *storage space* use, i.e. are in the bin of 0.5MB to 10MB. On the other hand, only 1.48% of the apps require relatively high storage of 50–60MB. Similar observations can be made about the other two types of *storage* overhead,

[12]https://play.google.com/store/apps/details?id=com.workpail.inkpad.notepad.notes

TABLE 4. Apps with the different combination of storage spaces.

App Names	Storage space consumed		
	Installation	Data	Cache
Google Earth	11.66	26.35	19.41
Talking 3 Headed Dragon	55.35	14.44	9.2
Facebook	27.48	43.70	0.96
File Manager	7.37	15.4	7.34
Jw Language	14.24	77.04	0.02
London City Guide	58.91	78.25	5.15
Google Maps	24.16	7.38	79.8
360 Online	43.56	10.57	0.02
Babel	51.98	14.79	0.02
Subway Surf	44.49	47.89	7.43
Tom Loves Angela	16.87	69.04	1.56
Google Translate	5.53	272	0.34
Fxguru	57.23	27.76	0.06
Amazon Kindle	45.30	23.33	5.76
Talking Angella	56.56	28.98	6.00
Linguee	41.31	92.04	0.02

as per Figures 6 (b) and 6 (c). Overall, we can observe that the majority of apps use a low(er) amount of *storage space*; e.g., 80%, 97%, and 98% of all apps belong to the lowest *storage space* consumption bins, i.e. 0.5MB - 20MB, 0.01MB - 20MB, and 0.01MB - 10MB respectively for *installation*, *data*, and *cache storage spaces*.

The *battery* consumption measured in our experiments shows a relatively low variation between various apps, with between 30% to 40% of the total battery (100%) being used by each app during the measurement period.

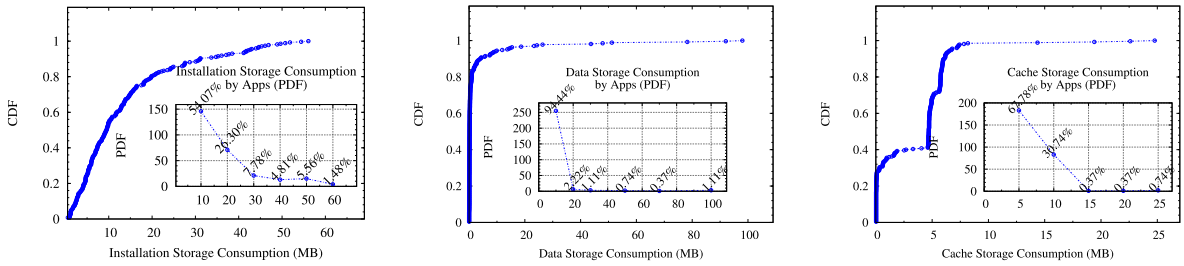
We will further evaluate the consumption of *resources* by selected obfuscation strategies in Section V-D.

V. PERFORMANCE OF OBFUSCATION STRATEGIES

In this section, we provide detailed investigation over the effectiveness of ProfileGuard in obfuscating user profiles through various candidate obfuscating strategies, described in Section III i.e. *similarity*, *bespoke*, *bespoke++*, and *random*. For this, we focus on 12 interest categories (and the corresponding app categories) without loss of generality, which we consider to be *private*: Comics, Games, Entertainment, Health & Fitness, Media & Video, Medical, Music & Audio, Shopping, Photography, Sports, Travel & Local, and Social. For the *obfuscation experiments*, as detailed in Section IV-A, same original set of apps S_a are used from the *profiling experiments* (belonging to a single private category), whereas we choose the obfuscation apps S_o from the remaining 2600 apps in non-private categories.

We note that the private categories of Social and Travel & Local (see Figure 4(b)) have a *dominance ratio* $D = 0$. Therefore, although we perform the experiments for the sake of completeness, we only show the interest categories introduced by obfuscation, rather than the new D values.

We use various metrics, defined in Section III-C, to assess different obfuscating strategies. As discussed the *utility* is the reduction of *dominance ratio* D, R_p, and *usability* U_s, demonstrating the likelihood that the user would be interested in an obfuscating app. The *utility* is vastly dependent on α and β i.e. weight coefficients, therefore we compare its components separately rather than its combined effect. Recall that the *cost* C is the increase in number of apps and is directly related to the number of obfuscating apps. Following, we first



(a) Installation Storage (b) Data Storage (c) Cache Storage

FIGURE 6. Storage space used by the apps for: (a) installation; (b), data (b); and (c) cache.

demonstrate the *minimum cost case* i.e. when only one app is used for obfuscation, subsequent, the *multi-app cost case* i.e. targeting a change in dominating interest category.

A. MINIMUM APP COST CASE: PROFILE OBFUSCATION WITH A SINGLE APPLICATION

Figure 7 shows obfuscated *Interest profiles* achieved by the four obfuscation strategies. In order to compare, we also include the original *Interest profile*, for the 12 selected private categories. The effect of introducing a single obfuscating app can be easily observed, it changes majority of categories in newly constructed *Interest profiles*. As discussed earlier, the objective of *bespoke* strategy is to introduce interests in the already existing interest categories in original profiles, excluding the private category. An additional benefit of the *bespoke++* strategy is to lower the overhead introduced by the obfuscating apps. Figure 7 shows a similar performance of the *bespoke* and *bespoke++* strategies. For example, when either of the strategies is applied to the Comics profile, it increases the number of interests in Computers & Electronics (which is the existing category in the original profile) by one and two, respectively with the *bespoke* and *bespoke++* strategies (we note that minor differences in the performance of these strategies are due to the overlap in the profile interests of a number of apps). We note similar change in newly constructed *Interest profiles* due to single obfuscating app chosen by the *random* and *similarity* based strategies, e.g., the *similarity* based strategy introduces two new interests from Computers & Electronics interest's category to the Games profile. Finally, note that the entire set of interests from original profile are present in the respective obfuscated profiles. This confirms our earlier findings that introducing a new app can either add a new interest to the profile or leave it unchanged.

Table 5 presents the statistics for *usability* metric, U_s, for four obfuscating strategies. It can be observed that the *similarity* strategy achieves higher score than the other strategies, which indicates that it is highly probable that users would utilise the suggested obfuscating applications. As demonstrated in our earlier experiments, there is a minimum level of activity required for apps to create an entry in an *Interest profile*. Hence, it naturally follows that the *similarity* based strategy will have a greater chance at achieving obfuscation. Another interesting fact is to observe that both the *bespoke* and *bespoke++* strategies have slightly lower *usability* than *random* strategy. It can be due to the fact that the former two strategies focus on choosing obfuscating apps that generate interests belonging to interest categories that are already present in the *Interest profile*, without taking into consideration whether the app may be of interest to the user.

TABLE 5. Average and standard deviation (St. dev.) values for *usability* U_s, for *single-app* and *complete* obfuscation.

Single-App				
	Similariy	Bespoke	Bespoke++	Random
Avg	0.78	0.52	0.50	0.58
St. Dev.	0.09	0.16	0.15	0.16
Complete Obfuscation				
Avg	0.80	0.56	0.56	0.57
St. Dev.	0.06	0.13	0.13	0.14

Figure 8 shows the comparison of *dominance ratio D* for private interest categories present in both obfuscated and original profiles.

It can be observed that all the obfuscating strategies achieve a reduction in *dominance ratio* R_p for all but one of the *Interest profiles* (Music & Audio for the *random* strategy). In some instances, we note a substantial reduction in *D*, such as, all strategies achieve a 50% reduction for Comics. Furthermore, we observe an additional 17% in reduction of *dominance ratio* in two categories i.e. Comics and Music & Audio, using the *bespoke++* strategy as compared to *bespoke* strategy.

In summary, we note that introducing a *single* obfuscation app can potentially achieve a notable change in both the overall structure of *Interest profile* and its *dominance ratio*. However, the level of perturbation achieved with a single app is not sufficient to modify the profile so that new dominant categories emerge, i.e. to achieve *complete* obfuscation.

B. COMPLETE OBFUSCATION

For *complete* obfuscation, we recursively select obfuscating apps according to a specific strategy, as an example for *similarity* based strategy, we select the obfuscating apps in descending order i.e. starting with highest similarity value.

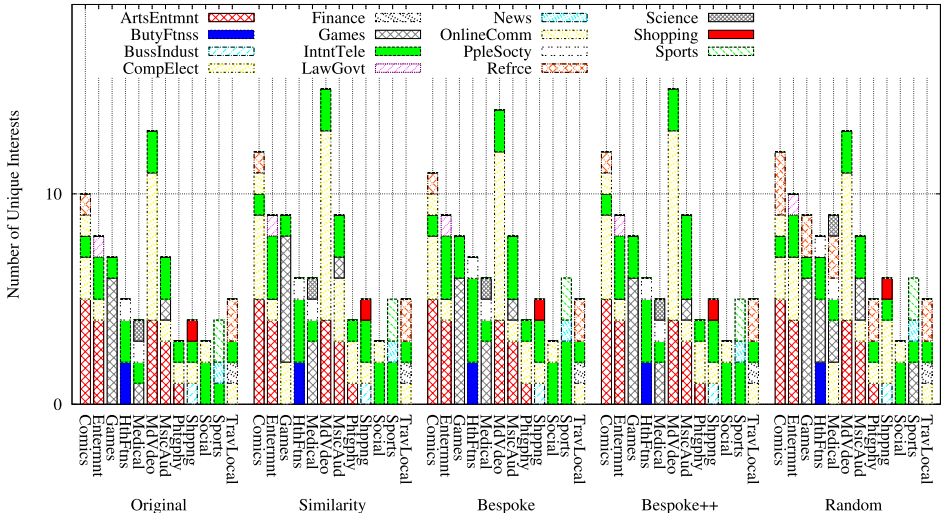


FIGURE 7. Comparison of unique number of interests in the original and obfuscated interest profiles ($|S_O| = 1$).

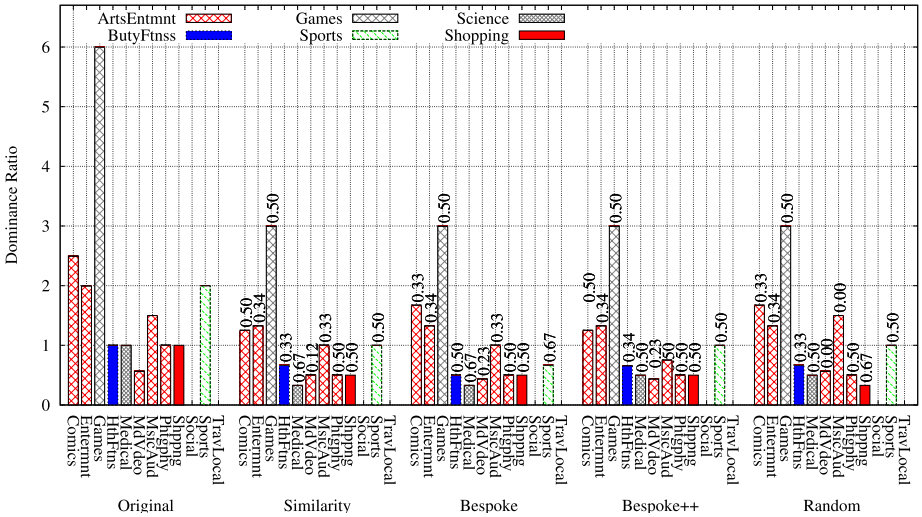


FIGURE 8. Comparison of *Dominance Ratio D* of original and obfuscated interest profiles when $|S_O| = 1$; note the reduction R_p is shown on top of the bars for each category.

We then experimentally derive new *Interest profile* until any non-private category becomes dominant. Figure 9 shows the results of the obfuscated profiles for all strategies including the original profile for comparison purposes, after *complete* obfuscation is achieved. We observe significant differences between the obfuscated and original profiles. We note that after the *complete* obfuscation the originally dominant interest categories are still present, since it adds up to the original profile and profile entries are not removed, however they are no longer dominant. For example, the *similarity* based strategy changes the dominant interest category for Comics profile from Arts & Entertainment to Computers & Electronics, which is an already existed interest category in the original profile. Similarly, the *bespoke* and *bespoke++* strategies only dominates to the already present interest categories, as per their design (see Section III-B), the other two strategies can create interests from new categories. As an example, the *similarity* based strategy introduces interests from two new categories i.e. Computer & Electronics and Arts & Entertainment, in the obfuscated Health & Fitness profile. In some cases, we notice that entirely new interest category becomes dominant, which did not exist in the original profile. For instance, after obfuscation with both *similarity* and *random* strategies, the dominant interest category in the Games profile changes from Games to Computer & Electronics.

Table 5 presents the statistical results for the *usability* metric for both the *single-app* and *complete* obfuscation. Note that the *usability* metric for *similarity* based strategy and for *single-app* obfuscation is considerably higher than for other strategies. It shows that this strategy has the greatest chance to succeed as the obfuscating apps recommended by this strategy would more likely be utilised by the users.

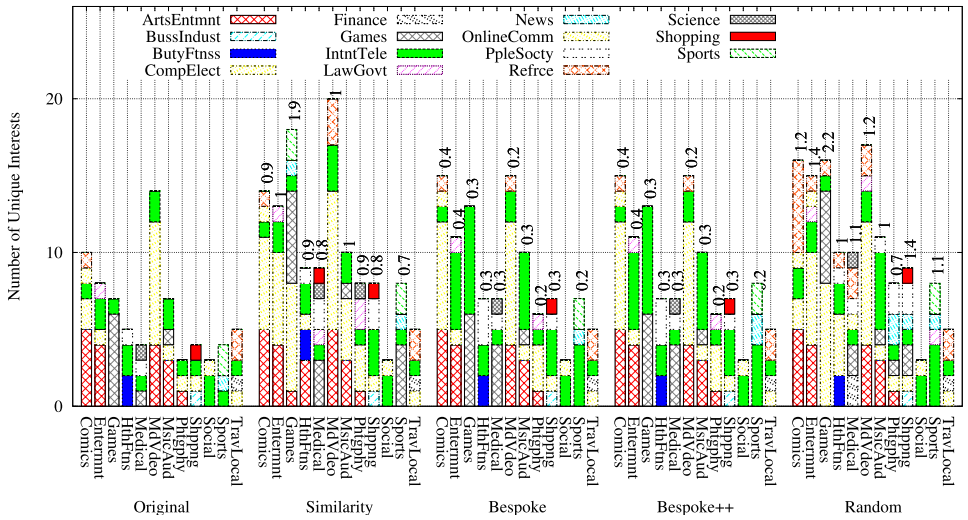


FIGURE 9. Comparison of unique number of interests in the original and obfuscated interest profiles (*complete* obfuscation); note the values of cost *C* are shown on top of the bars for each category.

TABLE 6. Comparison of unique ads received by the original and obfuscated profiles.

Cats	Unique Ads								
		Single-App				Complete			
	O	S	B	B++	R	S	B	B++	R
CO	134	141	121	73	124	324	282	159	343
EN	105	188	93	48	152	302	250	162	331
GA	36	63	122	53	136	268	158	105	291
HF	74	71	84	64	89	286	135	95	264
ME	57	78	82	54	86	196	163	117	219
MV	44	75	77	68	85	296	164	102	306
MA	91	112	86	75	95	308	195	124	318
PH	89	102	113	83	108	237	159	112	261
SH	67	86	92	63	96	219	171	126	273
SP	45	60	58	80	83	196	125	102	237
Cats	Profile Similarity (Jaccard Index)								
		Single-App				Complete			
	O	S	B	B++	R	S	B	B++	R
CO	0.33	0.26	0.14	0.18	0.20	0.13	0.06	0.12	0.08
EN	0.53	0.31	0.21	0.19	0.27	0.08	0.12	0.10	0.14
GA	0.43	0.19	0.16	0.16	0.24	0.06	0.11	0.09	0.08
HF	0.37	0.20	0.22	0.20	0.22	0.13	0.17	0.10	0.14
ME	0.42	0.28	0.24	0.13	0.27	0.15	0.16	0.12	0.13
MV	0.38	0.28	0.17	0.12	0.19	0.11	0.20	0.09	0.13
MA	0.41	0.22	0.21	0.14	0.30	0.12	0.16	0.12	0.16
PH	0.47	0.31	0.26	0.11	0.28	0.09	0.11	0.13	0.15
SH	0.38	0.18	0.21	0.12	0.22	0.14	0.18	0.11	0.13
SP	0.48	0.21	0.14	0.17	0.26	0.13	0.16	0.13	0.11

First Column: Comics (CO), Entertainment (EN), Game (GA), Health Fitness (HF), Medical (ME), Media Video (MV), Music & Audio (MA), Photography (PH), Shopping (SH), Sports (SP)
Third Row: Original (O), Similarity (S), Bespoke (B), Bespoke++ (B++), Random (R)

Figure 9 also includes the values of cost C, on the top of the bars for each category. We can observe that the costs for the *random* and *similarity* based strategies are similar, while the *bespoke* and *bespoke++* strategies consistently incur similar or lower costs, i.e., relative increase in the number of apps.

C. IMPACT ON ADS

The advertising companies utilise the user profiles created by ad analytics for targeted advertising, as is also shown by our work [27]. In this section, the impact of proposed obfuscating strategies is quantified based on received targeted ads using the ad traffic captured in our experiments (see Section IV-A). We calculate this by considering the similarity between unique ads served to the original and obfuscated profile, using the Jaccard Index $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$; where A and B are the sets of unique ads received by respective profiles.

Table 6 presents results for all strategies for the subset of nine private categories in the obfuscation experiments for both the *single-app* and *complete obfuscation*. We observe a consistent increase in number of received unique ads, by obfuscated profiles, for all suggested profiles and obfuscation strategies. The *complete* obfuscation results in substantial increase in number of unique ads i.e. more than 100% increase. It is also consistent with the increased number of interests generated in specific user profiles, generating a richer basis for targeted advertising. For instance, both *bespoke* and *bespoke++* strategy, on average, introduce an additional 9.9 interests, *random* strategy an additional 12 interests and *similarity* strategy 11.7, compared to the average number of original 6 interests. Similarly, we note that there is a proportional increase in the number of unique ads, which is highest for the *random* (339%), followed by 304% for *similarity* based, 165% for the *bespoke* strategy, and 77% for the *bespoke++* strategy. Note that apps introduced with the *bespoke++* strategy have (by strategy design) the lowest number of unique ads (this is also reflected in the overhead, which we will discuss in Section V-D).

D. RESOURCE USE IMPROVEMENTS

For evaluating the *resource* overhead generated by obfuscating apps, we use the experimental setup described in Section IV-A, (2.5 hours of activity is captured for each app).

1) COMMUNICATION OVERHEAD

Table 7 shows the bandwidth utilised by the obfuscating apps under different obfuscating strategies. We can observe

TABLE 7. Bandwidth consumption (MB) by obfuscating apps calculated for various obfuscation strategies; for *single-app* (left) and *complete* (right) Obfuscations.

Apps Categories	Single-App				Complete Obfuscation			
	S	B	B++	R	S	B	B++	R
Comics	4.88	3.50	0.81	3.52	15.74	12.17	7.37	15.73
Entertainment	3.72	2.49	1.33	2.56	13.21	15.53	6.32	18.07
Game	2.29	2.13	0.90	4.35	13.76	10.69	4.28	19.43
Health Fitness	3.50	3.61	1.86	2.64	12.67	9.47	4.48	16.27
Medical	3.13	2.01	0.95	4.50	14.06	8.75	5.31	19.56
Media Video	4.21	2.08	1.33	3.29	12.57	6.29	2.32	20.88
Music Audio	4.12	3.80	2.13	4.33	14.13	10.23	5.40	17.53
Photography	2.24	3.85	1.04	3.83	16.43	8.24	2.33	12.54
Shopping	3.61	4.03	1.07	4.21	13.66	9.78	2.33	16.33
Sports	3.27	2.19	1.16	2.78	12.56	5.56	6.07	12.45
Average	3.50	2.97	1.26	3.60	13.88	9.67	3.25	16.88
St. Dev.	0.82	0.85	0.42	0.75	1.31	2.85	1.81	2.83
S = Similarity, B = Bespoke, B++ = Bespoke++, R = Random								

that for *single-app* obfuscation, the obfuscating apps utilise an average of 3.60MB bandwidth of data (for ad related activity) with the *random* strategy, followed by, respectively, 3.50MB, 2.97MB, and 1.26MB for *similarity*, *bespoke* and *bespoke++* strategies. For *complete* obfuscation, there is a significantly higher difference in the bandwidth use, with, respectively, 16.88MB, 13.88MB, 9.67MB and 3.25MB od data used by obfuscating apps under the *random*, *similarity*, *bespoke* and *bespoke++* strategies. Although it is not surprising that the *bespoke* and *bespoke++* strategies benefit from their (by design) customised app selection mechanisms, it is interesting to note the relevance of *bespoke++* strategy to reduction of *resource* use. Compared to the *similarity* strategy it delivers a 64% improvement and a smaller but still significant improvement of 57% over the *bespoke* strategy.

2) COMPUTING OVERHEAD

We note that the *CPU* usage of the obfuscating apps can be clustered according to high-resource consuming apps, such as game apps, with the usage in 25% to 30% and with the low-resource usage apps with the *CPU* usage in between 15% to 20%. Furthermore, the Table 8 shows the various storage spaces for various obfuscating strategies under the *single-app complete* obfuscations. Recall that these storage spaces depend on the behaviour of apps in terms of storing temporary files i.e. *cache*, the apps settings/databases/accounts etc. *data*, and the actual apps' files i.e. *installation* storage sizes. We can observe that, on average, the obfuscating apps consume 12.15MB, 14.76MB, 13.11MB, and 10.87MB respectively under the *similarity*, *bespoke*, *bespoke++*, and *random* strategies. Similarly, these statistics under the *complete* obfuscation are 125.95MB, 85.18MB, 78.53MB, and 128.85MB of used *installation* storage space. We can also observe that the *installation* storage space is the dominant factor, significantly higher than the cache and data space needed. Note that various statistics under the *complete* obfuscation, given for various app's categories and for different storage types, are the sum of the *storage spaces* consumed by all the obfuscating apps used for the *complete* obfuscation. We farther note, among the three types of *storage spaces*, that the *cache* and *data* take less storage spaces compared to the *installation* storage space.

3) BATTERY CONSUMPTION

We now evaluate the *battery* consumption by the apps under different obfuscation strategies using *single-app* and *complete* obfuscation. Table 9 shows the *battery* consumption for different obfuscation strategies. We observe that the average battery consumption for *single-app* obfuscation ranges from 31% to 33% of the total battery (i.e. 100% of full charge), while this is increased to up to 40% with the complete obfuscation. This is in line with the higher number of apps used for complete obfuscation.

VI. DISCUSSION

The use of privacy apps is very much dependent on the users' motivation towards preserving their privacy. In recent times, both with public and in the regulatory environment, there has been an enormous increase in privacy awareness. This has been motivated by the exposure of mass surveillance activities [31] and by unintentional leaks of datasets with personal records. Hence, there is an increased interest in the adoption of personal (bespoke) privacy tools. The ProfileGuard is considered to be consistent with the growing number of app recommender and personalisation systems, however, with an emphasis on not only suggesting apps of interest to the users, but also enabling privacy protection. The results achieved with the ProfileGuard using *similarity* based strategy, presented in Section V, suggest that it can reduce, on the average, the dominance ratio of a private interest category by 46.3% with a *single-app*. Furthermore, it ensures that the recommended apps are of actual use to the individuals, with a good *usability* of 43.1% lower similarity than the best app from those considered in the Google Play apps set. We plan for more complex recommender system approaches in future work.

The second issue relevant to the user is *resource* use - we demonstrate that this can be significantly reduced by selecting the *bespoke++* strategy. During the total time of our experiments (6 months) we have repeatedly built user profiles based on the app categories (see Section IV and while the time to establish a profile was consistent (24 hours, regardless of the number of tested apps), the time required for an app to register in the user profile was less so. The absolute values for *resource* use, presented in Sections IV and V, for our experiments where each app was utilised for 2.5 hours, indicate a relatively large *resource* (e.g. bandwidth) use. On the average, it ranges between close to 17MB with *random* to around 3MB with *bespoke++* strategy, to achieve *complete* obfuscation. This increases the level of motivation for using the *bespoke++* strategy, particularly for users who may be on a fixed data use mobile plan. We note that AdMob or other advertising companies may change the profiling rules (and often do). E.g., the user profile on Android phones is currently (as of July 2015) derived from both the mobile app and the browsing use, so a combination of app based and browser

TABLE 8. ***Installation, data,*** **and** ***cache*** **storage space taken by the obfuscating apps for** ***single-app*** **(above) and** ***complete*** **(below) obfuscation.**

App Categories	Installation (MB)				Cache (MB)				Data (MB)			
	S	B	B++	R	S	B	B++	R	S	B	B++	R
Single-App Obfuscation												
Comics	4.68	28.70	23.54	7.34	1.64	1.94	0.63	1.68	3.42	2.47	0.15	7.75
Entertainment	11.82	5.12	9.64	8.58	0.85	0.99	1.32	0.38	8.92	5.65	2.92	3.29
Game	8.58	24.06	14.47	31.10	0.38	0.75	5.78	0.38	3.29	1.95	2.54	4.02
Health Fitness	8.89	1.52	13.64	14.25	0.67	0.13	4.66	0.25	0.54	0.51	1.22	4.13
Medical	39.23	9.64	9.07	11.04	1.22	1.45	0.53	1.26	3.64	6.07	0.23	2.67
Media Video	19.86	54.52	1.27	8.92	0.81	0.75	1.09	1.74	5.85	7.39	1.42	7.48
Music Audio	4.45	5.02	9.59	8.29	1.98	1.44	0.42	0.76	2.34	8.20	0.98	7.26
Photography	16.09	8.95	41.12	0.96	0.94	0.16	5.64	0.44	0.36	4.25	0.36	8.17
Shopping	7.47	3.07	6.06	7.47	5.86	1.93	4.66	5.86	49.08	2.10	2.09	49.08
Sports	0.42	6.95	2.65	10.78	0.75	1.58	4.66	1.12	6.44	6.98	0.15	5.49
Average	12.15	14.76	13.11	10.87	1.51	1.11	2.94	1.39	8.39	4.56	1.21	9.93
St. Dev.	11.09	16.61	11.72	7.88	1.60	0.66	2.30	1.66	14.54	2.67	1.03	13.90
Complete Obfuscation												
Comics	104.37	63.79	87.45	128.43	1.64	0.90	1.02	2.64	17.53	14.99	1w8.22	72.75
Entertainment	136.76	83.76	70.58	136.45	1.46	1.32	1.85	1.36	29.70	15.19	14.20	100.03
Game	203.59	59.83	94.05	235.67	2.41	0.78	2.01	3.84	33.42	11.80	21.45	113.06
Health Fitness	98.44	94.71	83.90	103.63	2.62	1.94	1.82	4.35	49.15	10.16	14.57	78.32
Medical	105.30	138.62	62.20	116.09	3.78	1.38	1.41	3.59	73.65	21.12	25.53	55.64
Media Video	127.50	74.92	75.28	126.41	2.47	0.72	2.01	2.19	21.26	20.29	27.98	61.79
Music Audio	149.54	69.88	63.43	116.73	3.32	0.62	1.30	1.37	85.54	23.93	18.34	96.96
Photography	131.58	85.62	84.98	97.50	1.07	1.27	2.18	1.43	23.89	14.67	18.76	55.13
Shopping	105.62	53.85	80.47	121.64	2.19	1.47	1.75	2.25	51.59	15.15	23.87	72.88
Sports	96.82	126.87	82.22	105.93	3.45	0.70	1.50	2.12	17.32	22.49	24.65	62.10
Average	125.95	85.18	78.53	128.85	2.44	1.11	1.69	2.51	40.31	16.98	20.76	76.87
St. Dev.	32.74	28.10	10.28	39.41	0.89	0.43	0.37	1.08	24.04	4.67	4.69	20.15
S = Similarity, B = Bespoke, B++ = Bespoke, R = Random												

TABLE 9. **Battery consumption (%) by obfuscating apps calculated for** ***single-app*** **(left) and** ***complete*** **(right) obfuscation.**

App Categories	Single-App				Complete Obfuscation			
	S	B	B++	R	S	B	B++	R
Comics	34.35	29.30	31.27	30.11	38.44	38.50	37.81	42.08
Entertainment	39.42	32.45	25.62	31.31	39.50	40.25	39.43	41.21
Game	40.46	33.44	27.78	24.12	38.05	39.00	39.37	40.23
Health Fitness	24.41	37.02	31.75	26.29	39.50	40.67	39.08	40.00
Medical	32.01	38.10	26.49	40.26	41.56	39.67	40.33	40.00
Media Video	24.07	30.46	31.64	37.20	40.90	39.50	38.39	40.25
Music Audio	19.19	26.41	33.90	40.33	39.13	39.67	40.19	39.90
Photography	37.23	36.47	25.90	42.15	39.67	42.50	37.04	40.57
Shopping	29.05	27.16	29.69	29.40	40.00	40.00	40.22	43.07
Sports	34.46	25.21	30.16	31.48	39.75	41.00	39.92	42.29
Average	31.47	31.60	29.42	33.26	39.65	40.08	39.18	40.96
St. Dev.	6.75	4.40	2.69	5.98	1.04	1.13	1.06	1.14
S = Similarity, B = Bespoke, B++ = Bespoke++, R = Random								

based obfuscation may be needed to completely obfuscate the full mobile profile (although our methodology would still be applicable to app-based profiling, which is the topic of this paper).

Considering the number of different options, we envisage that the ProfileGuard system app (discussed in Section III-D) would enable users to make an informed choice while selecting obfuscating apps, indicating (with every recommended app) the level of preference for *usability*, *resource* use, and privacy protection along with the their expected levels for all parameters.

On a system level, for a specific selection strategy, we consider the level of effort required by the obfuscation system engine in order to select the candidate obfuscation apps. From a practical point of view, the *similarity* based strategy has a comparable level of implementation effort to the recommender system apps like AppBrain [26], and we note that this is higher than for a *random* strategy, it is arguably manageable. For the *bespoke* and *bespoke++* strategies there is a trade-off, where the level of pre-processing required before these strategies can be effectively applied, based on a rich set of diverse apps, can be balanced with the *resource* use (and other) savings delivered by these strategies.

We note that, to implement the *bespoke* strategy, it requires a significant effort to derive the *Interest profile* for individual apps, if done manually. Additionally, we believe that it would be difficult to collect this information voluntarily while motivating users in a crowdsourcing environment, as it the requirement is to run a selected app in isolation (starting with already reset Google profile) and the profile needs to be manually checked and recorded. However, an approximation of the profile could be derived (and this process automated) based on the app category and keywords, using the interest mapping approach outlined in Section IV-B. We note that this would incur a penalty in profile accuracy, which needs to be evaluated in future work. Obtaining the ad frequency to estimate the bandwidth use for the *bespoke++* strategy would be a relatively simple task, that could be automated on the mobile devices and crowdsourced (e.g., the ProfileGuard app could perform a short test to capture the ad request traffic and subsequently forward this to the back end, for inclusion in the database).

Considering the *utility*, the app *cost* and *resource* use for all strategies, we believe that the *similarity* based strategy has the best overall potential for implementation in a real world obfuscation system. As a second choice, this should be augmented with the *bespoke++* strategy for a *resource* constrained mobile user.

VII. RELATED WORK

Privacy threats resulting from collecting individual's online data have been extensively investigated in the literature, with a number of works [32]–[39] demonstrating how it is possible to infer user's undisclosed private information such as age, gender, relationship status, etc. from their online data. In [34], the authors analysed client-side browsing history of 250K users and were able to infer various personal attributes including age, gender, race, education and income. The extent of data collection and the related privacy treats were first investigated in [3]. Specific information leakage through ad library APIs was presented in [7], while further works including [5] and [6] demonstrate the potential for inference attacks based on (captured) ads from user's browsing sessions.

Researchers have evaluated how users' history can be tracked with browser fingerprinting and cookie syncing [8], [9], the extent of web tracking [4], [40]–[42], user tracking on multiple devices [43], privacy implications using active and passive learning approaches [2], and how third party tracking services leak information [44]. Another work utilizes Oracle's Bluekai registry to investigate user profiling process by simulating web browsing sessions based on the updates posted to Reddit [45]. A second study, based on a large-scale measurement, evaluates the extent of privacy leakage in location-based mobile advertising services [46]. The authors further implement a mechanism for obfuscating the location data in real-time, which disrupts the targeted mobile ads. Studies have shown that the profiling for online advertisements is largely biased [47] and often not correct [48], [49] causing many users feel uncomfortable with being profiled [50] and requesting more control over their data [51].

We note that the majority of prior works on privacy protection in advertising systems have focused on browser based ads [14]–[17], [52], [53] while only a small body of work addresses in-app targeted ads, which is the primary focus of our work [18]. A number of proposals advocate the use of locally (either in the browser of the mobile device) derived user profiles, where user's interests are generalised and/or partially removed (according to user's privacy preferences), before being forwarded to the server or an intermediary that selected the appropriate ads to be forwarded to the clients. The privacy requirements are also, in a number of prior works, considered in parallel with achieving bandwidth efficiency for ad delivery, by using caching mechanisms [12], [54], [55].

The simplest and straightforward privacy protection mechanism is to *anonymise* data by masking or removing specific data fields (direct identifiers) that expose personal information. In the context of targeted advertising, the removal of direct identifiers includes user IDs (replacing them with temporary IDs) or mechanisms to hide used network address (e.g., using TOR [56]). The PBooster [15], an anonymisation scheme for web browsing history, protects user's privacy by anonymising users' browsing history by first inferring relevant links and then adding those links to the browsing history. However, if only the most obvious anonymisation is applied without introducing additional (profiling and targeting oriented) features, the ad networks ecosystem would be effectively disabled.

The second group of privacy mechanisms uses *obfuscation* techniques. A large body of research work focuses on generic noisy techniques, starting with [57] who have proposed the approach of adding random values to data, generated independently of the data itself, from a known e.g., the uniform distribution. Subsequent publications (e.g., [58]) improve the initial technique, however other research work [59] has identified the shortcomings of this approach, where the added noise may be removed by data analysis and the original data (values) recovered.

Generalisation and *noisy* approaches have been recently analysed and compared in [60], [61], demonstrating the advantage of noisy techniques. A novel noisy technique for privacy preserving personalisation of web searches is also recently proposed [61]. In this work, the authors use Bloom cookies that comprise a noisy version of the locally derived profile. This version is generated by using Bloom filters [62], an efficient data structure; they evaluate the privacy versus personalisation trade-off. Another approach generalises potentially identifying user data (e.g., gender, ZIP code), referred to as quasi-identifiers (QIDs), by grouping them into a broader category that includes a specific minimum number of users (e.g., locations into post codes) [63], [64]. This approach is applied in practice in most ad networks by introducing a minimum group size for targeted ads (see e.g. [65]), however, this (traditional) approach has been demonstrated as inadequate by a number of research works (e.g., [66]), where anonymised private information was linked to other publicly available data, resulting in private data exposure.

Differential privacy [67] is a noisy technique that provides an unconditional privacy guarantee, assuming any computational power, or externally available information to the attacker attempting to break privacy and derive information about users. A number of prior works use differential privacy in the context of profiling and advertising. The authors in [68] propose a system for differentially private statistical queries by a data aggregator, over distributed users data. A proxy (assumed to be honest-but-curious) is placed between the analyst (aggregator) and the clients and secure communications including authentication and traffic confidentiality are accomplished using TLS [69].

The SplitX system [70] also provides differential privacy guarantees and relies on intermediate nodes, that forward and process the messages between the client that locally stores their (own) data and the data aggregator. Further examples include works proposing the use of distributed differential privacy [71], [72]. We note that above approaches may render *targeted advertising* to be ineffective as they introduce noise to the user profile. While our approach relies on modifying the user profile, we do so in a calculated manner with the aim of reducing the *dominance* of selected private user profile interest categories.

Additionally, *cryptographic* solutions can be used to provide part of the system functionality. They are commonly used in conjunction with obfuscation, e.g., in [71], [72] or generalisation [13]. [71] combines differential privacy with a homomorphic cryptosystem. Chen *et al.* [68] uses cryptographic mechanism to combine client-provided data (modified in accordance with differential privacy). They utilise a probabilistic Goldwasser-Micali cryptosystem [73]. In their subsequent work [70], the authors use an XOR-based cryptomechanism to provide both anonymity and unlinkability to analysis (queries) of differentially private data distributed on user's devices (clients). Adnostic [13] uses a combination of homomorphic encryption and zero-knowledge proof mechanisms to enable accounting and billing in the advertising system in a (for the user) privacy pre- serving way. Effectively, the user is protected as neither the publisher (website that includes the ads) or the advertisers (that own the ads) have knowledge about which users viewed specific ads. A web-based advertising system based on Private Information retrieval was first proposed by Juels [74], where they use information-theoretic (threshold) PIR in an honest-but-curious multi-server architecture. Finally, a cryptography technique, mixing [75], [76] is also commonly used as part of anonymisation [74], [77], where mix servers are used as intermediaries that permute (and re-encrypt) the input.

We note that the previously proposed in-browser and in-app private advertising systems protect the full user profile and advocate the use of novel mechanisms that necessitate re-designing of some parts or all of the current advertising systems (although some, e.g., Adnostic [13] can operate in parallel with the existing systems). Our proposal focuses on the protection of selected attribute(s) and does not require any changes to the current ad networks; i.e., it enables standard profiling and targeted advertising, however based on attributes that the user does not consider private.

VIII. CONCLUSION

We propose to directly obfuscate the mobile app based profiling by introducing obfuscating apps. We demonstrate that the *similarity* based strategy can achieve a good level of obfuscation of both the profiles and correspondingly received ads, which can be viewed as a version of an app recommender system. We also show how a *resource*-aware strategy can be used to further enhance the performance of the obfuscating system, by reducing the use of bandwidth and other mobile resources. In future work, our plan is to further investigate various factors that could influence the acceptability of an app-based obfuscation system in a real world environment.

REFERENCES

[1] W. Enck, D. Octeau, P. McDaniel, and S. Chaudhuri, "A study of Android application security," in *Proc. USENIX Secur. Symp.*, 2011, p. 2.

[2] L. I. Labrecque, E. Markos, and A. Darmody, "Addressing online behavioral advertising and privacy implications: A comparison of passive versus active learning approaches," *J. Marketing Edu.*, 2019.

[3] B. Krishnamurthy, D. Malandrino, and C. E. Wills, "Measuring privacy loss and the impact of privacy protection in Web browsing," in *Proc. 3rd Symp. Usable Privacy Secur. (SOUPS)*, New York, NY, USA, 2007, pp. 52–63.

[4] A. Karaj, S. Macbeth, R. Berson, and J. M. Pujol, "WhoTracks .Me: Shedding light on the opaque world of online tracking," 2018, *arXiv:1804.08959*. [Online]. Available: http://arxiv.org/abs/1804.08959

[5] A. Chaabane, G. Acs, and M. A. Kaafar, "You are what you like! Information leakage through users' Interests," in *Proc. 19th Netw. Distrib. Syst. Secur. Symp.*, Feb. 2012, pp. 1–15.

[6] C. Castelluccia, M.-A. Kaafar, and M.-D. Tran, "Betrayed by your ads! Reconstructing user profiles from targeted ads," in *Proc. Privacy Enhancing Technol. Symp.* Springer, 2012, pp. 1–17.

[7] T. Book and D. S. Wallach, "A case of collusion: A study of the interface between ad libraries and their apps," in *Proc. 3rd ACM Workshop Secur. Privacy Smartphones Mobile Devices (SPSM)*, 2013, pp. 79–86.

[8] G. Acar, C. Eubank, S. Englehardt, M. Juarez, A. Narayanan, and C. Diaz, "The Web never forgets: Persistent tracking mechanisms in the wild," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2014, pp. 674–689.

[9] M. A. Bashir, S. Arshad, W. Robertson, and C. Wilson, "Tracing information flows between ad exchanges using retargeted ads," in *Proc. 25th USENIX Secur. Symp. (USENIX Secur.)*, 2016, pp. 481–496.

[10] V. Toubiana, L. Subramanian, and H. Nissenbaum, "TrackmeNot: Enhancing the privacy of Web search," *CoRR*, vol. abs/1109.4677, 2011.

[11] C. A. Ardagna, M. Cremonini, E. Damiani, S. D. C. di Vimercati, and P. Samarati, "Location privacy protection through obfuscation-based techniques," in *Proc. 21st IFIP Work. Conf. Data Appl. Secur.*, 2007, pp. 47–60.

[12] S. Guha, B. Cheng, and P. Francis, "Privad: Practical privacy in online advertising," in *Proc. 8th USENIX Conf. Netw. Syst. Design Implement.* Berkeley, CA, USA: USENIX Association, 2011, p. 13.

[13] V. Toubiana, A. Narayanan, D. Boneh, H. Nissenbaum, and S. Barocas, "Adnostic: Privacy preserving targeted advertising," in *Proc. NDSS*. Reston, VA, USA: Internet Society, 2010, p. 23.

[14] K. Kenthapadi, T. T. L. Tran, M. Dietz, T. Greason, and I. V. Koeppe, "Random noise based privacy mechanism," U.S. Patent 15 703 834, Mar. 14, 2019.

[15] G. Beigi, R. Guo, A. Nou, Y. Zhang, and H. Liu, "Protecting user privacy: An approach for untraceable Web browsing history and unambiguous user profiles," in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, Jan. 2019, pp. 213–221.

[16] O. Starov and N. Nikiforakis, "Privacymeter: Designing and developing a privacy-preserving browser extension," in *Proc. Int. Symp. Eng. Secure Softw. Syst.* Springer, 2018, pp. 77–95.

[17] N. Laoutaris and J. Blackburn, "Method, a device and computer program products for protecting privacy of users from Web-trackers," U.S. Patent 10 110 633, Oct. 23, 2018.

[18] I. Ullah, R. Boreli, S. S. Kanhere, and S. Chawla, "ProfileGuard: Privacy preserving obfuscation for mobile user profiles," in *Proc. 13th Workshop Privacy Electron. Soc. (WPES)*, AZ, USA, 2014, p. 10.

[19] S. Han, J. Jung, and D. Wetherall, "A study of third-party tracking by mobile Apps in the wild," Tech. Rep. UW-CSE-12-03-01, 2011.

[20] [Online]. Available: https://www.flurry.com

[21] T. Chen, I. Ullah, M. A. Kaafar, and R. Boreli, "Information leakage through mobile analytics services," in *Proc. 15th Workshop Mobile Comput. Syst. Appl. (HotMobile)*, 2014, pp. 1–6.

[22] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*, vol. 1. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[23] J. Parra-Arnau, D. Rebollo-Monedero, and J. Forné, "A privacy-preserving architecture for the semantic Web based on tag suppression," in *Trust, Privacy and Security in Digital Business*. Springer, 2010, pp. 58–68.

[24] J. Parra-Arnau, D. Rebollo-Monedero, and J. Forné, "A privacy-protecting architecture for collaborative filtering via forgery and suppression of ratings," in *Data Privacy Management and Autonomous Spontaneus Security*. Springer, 2012, pp. 42–57.

[25] X. Xia, X. Wang, X. Zhou, and B. Liu, "Evolving mobile app recommender systems: An incremental multi-objective approach," in *Future Information Technology*. Springer, 2014, pp. 21–27.

[26] [Online]. Available: https://www.appbrain.com

[27] I. Ullah, R. Boreli, M. A. Kaafar, and S. S. Kanhere, "Characterising user targeting for in-App mobile ads," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2014, pp. 547–552.

[28] S. Nath, "MAdScope: Characterizing mobile in-App targeted ads," in *Proc. 13th Annu. Int. Conf. Mobile Syst., Appl., Services (MobiSys)*, 2015, pp. 59–73.
[29] [Online]. Available: https://www.tcpdump.org
[30] N. Vallina-Rodriguez, J. Shah, A. Finamore, Y. Grunenberger, K. Papagiannaki, H. Haddadi, and J. Crowcroft, "Breaking for commercials: Characterizing mobile advertising," in *Proc. ACM Conf. Internet Meas. Conf.*, 2012, pp. 343–356.
[31] [Online]. Available: http://www.theguardian.com/world/the-nsa-files
[32] H. A. Schwartz, J. C. Eichstaedt, M. L. Kern, L. Dziurzynski, S. M. Ramones, M. Agrawal, A. Shah, M. Kosinski, D. Stillwell, M. E. P. Seligman, and L. H. Ungar, "Personality, gender, and age in the language of social media: The open-vocabulary approach," *PLoS ONE*, vol. 8, no. 9, Sep. 2013, Art. no. e73791.
[33] M. Kosinski, D. Stillwell, and T. Graepel, "Private traits and attributes are predictable from digital records of human behavior," *Proc. Nat. Acad. Sci. USA*, vol. 110, no. 15, pp. 5802–5805, Apr. 2013.
[34] S. Goel, J. M. Hofman, and M. I. Sirer, "Who does what on the Web: A large-scale study of browsing behavior," in *Proc. 6th Int. AAAI Conf. Weblogs Social Media*, 2012, pp. 1–8.
[35] J. Otterbacher, "Inferring gender of movie reviewers: Exploiting writing style, content and metadata," in *Proc. 19th ACM Int. Conf. Inf. Knowl. Manage. (CIKM)*, 2010, pp. 369–378.
[36] J. Schler, M. Koppel, S. Argamon, and J. W. Pennebaker, "Effects of age and gender on blogging," in *Proc. AAAI Spring Symp., Comput. Approaches Analyzing Weblogs*, vol. 6, 2006, pp. 199–205.
[37] B. Bi, M. Shokouhi, M. Kosinski, and T. Graepel, "Inferring the demographics of search users: Social data meets search queries," in *Proc. 22nd Int. Conf. World Wide Web (WWW)*, 2013, pp. 131–140.
[38] J. J.-C. Ying, Y.-J. Chang, C.-M. Huang, and V. S. Tseng, "Demographic prediction based on users mobile behaviors," *Mobile Data Challenge*, vol. 2012, pp. 1–4, Jun. 2012.
[39] I. Ullah, B. G. Sarwar, R. Boreli, S. S. Kanhere, S. Katzenbeisser, and M. Hollick, "Enabling privacy preserving mobile advertising via private information retrieval," in *Proc. IEEE 42nd Conf. Local Comput. Netw. (LCN)*, Oct. 2017, pp. 347–355.
[40] F. Roesner, T. Kohno, and D. Wetherall, "Detecting and defending against third-party tracking on the Web," in *Proc. 9th USENIX Conf. Netw. Syst. Design Implement.* Berkeley, CA, USA: USENIX Association, 2012, p. 12.
[41] S. Schelter and J. Kunegis, "On the ubiquity of Web tracking: Insights from a billion-page Web crawl," *J. Web Sci.*, vol. 4, no. 4, pp. 53–66, Mar. 2018.
[42] L. Sweeney, "Discrimination in online ad delivery," 2013, *arXiv:1301.6822*. [Online]. Available: http://arxiv.org/abs/1301.6822
[43] J. Brookman, P. Rouge, A. Alva, and C. Yeung, "Cross-device tracking: Measurement and disclosures," *Proc. Privacy Enhancing Technol.*, vol. 2017, no. 2, pp. 133–148, Apr. 2017.
[44] S. Englehardt, D. Reisman, C. Eubank, P. Zimmerman, J. Mayer, A. Narayanan, and E. W. Felten, "Cookies that give you away: The surveillance implications of Web tracking," in *Proc. 24th Int. Conf. World Wide Web (WWW)*, 2015, pp. 289–299.
[45] M. Degeling and J. Nierhoff, "Tracking and tricking a profiler: Automated measuring and influencing of Bluekai's interest profiling," in *Proc. Workshop Privacy Electron. Soc. (WPES)*, 2018, pp. 1–13.
[46] B. Hu, Q. Yan, and Y. Zheng, "Tracking location privacy leakage of mobile ad networks at scale," in *Proc. IEEE INFOCOM-IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2018, pp. 1–2, doi: 10.1109/INFCOMW.2018.8406986.
[47] A. Datta, M. C. Tschantz, and A. Datta, "Automated experiments on ad privacy settings," *Proc. Privacy Enhancing Technol.*, vol. 2015, no. 1, pp. 92–112, Apr. 2015.
[48] E. Spyromitros-Xioufis, G. Petkos, S. Papadopoulos, R. Heyman, and Y. Kompatsiaris, "Perceived versus actual predictability of personal information in social networks," in *Proc. Int. Conf. Internet Sci.* Springer, 2016, pp. 133–147.
[49] T. Theodoridis, S. Papadopoulos, and Y. Kompatsiaris, "Assessing the reliability of Facebook user profiling," in *Proc. 24th Int. Conf. World Wide Web (WWW Companion)*, 2015, pp. 129–130.
[50] B. Ur, P. G. Leon, L. F. Cranor, R. Shay, and Y. Wang, "Smart, useful, scary, creepy: Perceptions of online behavioral advertising," in *Proc. 8th Symp. Usable Privacy Secur. (SOUPS)*, 2012, p. 4.
[51] W. Melicher, M. Sharif, J. Tan, L. Bauer, M. Christodorescu, and P. G. Leon, "(Do Not) track me sometimes: Users' contextual preferences for Web tracking," *Proc. Privacy Enhancing Technol.*, vol. 2016, no. 2, pp. 135–154, Apr. 2016.
[52] H. Yoo, S. Yao, L. Sun, and X. Du, "Using machine learning to address customer privacy concerns: An application with click-stream data," Tech. Rep. SSRN 3314787, 2019.
[53] D. Sánchez and A. Viejo, "Privacy-preserving and advertising-friendly Web surfing," *Comput. Commun.*, vol. 130, pp. 113–123, Oct. 2018.
[54] A. J. Khan, K. Jayarajah, D. Han, A. Misra, R. 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.
[55] H. Haddadi, P. Hui, and I. Brown, "MobiAd: Private and scalable mobile advertising," in *Proc. 5th ACM Int. Workshop Mobility Evolving Internet Archit. (MobiArch)*, 2010, pp. 33–38.
[56] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," Tech. Rep., 2004.
[57] R. Agrawal and R. Srikant, "Privacy-preserving data mining," *ACM SIGMOD Rec.*, vol. 29, no. 2, pp. 439–450, 2000.
[58] A. Evfimievski, J. Gehrke, and R. Srikant, "Limiting privacy breaches in privacy preserving data mining," in *Proc. 22nd ACM SIGMOD-SIGACT-SIGART Symp. Princ. Database Syst. (PODS)*, 2003, pp. 211–222.
[59] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar, "On the privacy preserving properties of random data perturbation techniques," in *Proc. 3rd IEEE Int. Conf. Data Mining*, Nov. 2003, pp. 99–106.
[60] E. Balsa, C. Troncoso, and C. Diaz, "OB-PWS: Obfuscation-based private Web search," in *Proc. IEEE Symp. Secur. Privacy*, May 2012, pp. 491–505.
[61] N. Mor, O. Riva, S. Nath, and J. Kubiatowicz, "Bloom cookies: Web search personalization without user tracking," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2015, pp. 1–15.
[62] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Commun. ACM*, vol. 13, no. 7, pp. 422–426, Jul. 1970.
[63] P. Samarati and L. Sweeney, "Generalizing data to provide anonymity when disclosing information (abstract)," in *Proc. 17th ACM SIGACT-SIGMOD-SIGART Symp. Princ. Database Syst. (PODS)*, vol. 98, 1998, p. 188.
[64] L. Sweeney, "K-anonymity: A model for protecting privacy," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, Oct. 2002.
[65] T. Chen, A. Chaabane, P.-U. Tournoux, M. Kaafar, and R. Boreli, "How much is too much? Leveraging ads audience estimation to evaluate public profile uniqueness," in *Proc. Privacy Enhancing Technol. Symp. (PETS)*, 2013, pp. 225–244.
[66] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2008, pp. 111–125.
[67] D. Cynthia, "Differential privacy," in *Proc. 33rd Int. Colloq. Automata, Lang. Program. (ICALP)*, 2006, pp. 1–12.
[68] R. Chen, A. Reznichenko, P. Francis, and J. Gehrke, "Towards statistical queries over distributed private user data," in *Proc. NSDI*, vol. 12, 2012, p. 13.
[69] T. Dierks, "The transport layer security (TLS) protocol version 1.2," Tech. Rep., 2008.
[70] R. Chen, I. E. Akkus, and P. Francis, "Splitx: High-performance private analytics," *SIGCOMM Comput. Commun. Rev.*, vol. 43, pp. 315–326, Aug. 2013.
[71] V. Rastogi and S. Nath, "Differentially private aggregation of distributed time-series with transformation and encryption," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2010, pp. 735–746.
[72] E. Shi, T. H. Chan, E. Rieffel, R. Chow, and D. Song, "Privacy-preserving aggregation of time-series data," in *Proc. NDSS*, vol. 2, 2011, pp. 1–17.
[73] O. Goldreich, S. Micali, and A. Wigderson, "How to play ANY mental game," in *Proc. 19th Annu. ACM Conf. Theory Comput. (STOC)*, 1987, pp. 218–229.
[74] A. Juels, "Targeted advertising... and privacy too," in *Topics in Cryptology—CT-RSA 2001*. Springer, 2001, pp. 408–424.
[75] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, vol. 24, no. 2, pp. 84–90, Feb. 1981.
[76] Y. Desmedt and K. Kurosawa, "How to break a practical mix and design a new one," in *Advances in Cryptology—EUROCRYPT 2000*. Springer, 2000, pp. 557–572.
[77] M. Backes, A. Kate, M. Maffei, and K. Pecina, "Obliviad: Provably secure and practical online behavioral advertising," in *Proc. IEEE Symp. Secur. Privacy*, May 2012, pp. 257–271.

IMDAD ULLAH received the Ph.D. degree in computer science and engineering from The University of New South Wales (UNSW) Sydney, Australia. He is currently an Assistant Professor with the College of Computer Engineering and Sciences, PSAU, Saudi Arabia. He has served in various positions of Researcher at UNSW, Research Scholar at National ICT Australia (NICTA)/Data61 CSIRO Australia, NUST, Islamabad, Pakistan, and SEEMOO TU, Darmstadt, Germany, and Research Collaborator at the SLAC National Accelerator Laboratory, Stanford University, USA. He has research and development experience in privacy preserving systems including private advertising and crypto-based billing systems. His primary research interest includes privacy enhancing technologies; he also has interest in the Internet of Things, blockchain, network modeling and design, network measurements, and trusted networking.

ROKSANA BORELI received the Ph.D. degree in communications from the University of Technology, Sydney, Australia. She has over 20 years of experience in communications and networking research and in engineering development, in large telecommunications companies (Telstra Australia, Xantic, NL) and research organizations. She has served in various positions of Engineering Manager, Technology Strategist, Research Leader of the Privacy Area of Networks Research Group, National ICT Australia (NICTA)/CSIRO Data61, and CTO in a NICTA spinoff 7-ip. Her primary research focus is on the privacy enhancing technologies; she also maintains an interest in mobile and wireless communications.

SALIL S. KANHERE (Senior Member, IEEE) received the M.S. and Ph.D. degrees from Drexel University, Philadelphia. He is currently a Professor of computer science and engineering with UNSW Sydney, Australia. He has coauthored a book titled *Blockchain for Cyberphysical Systems* His research interests include the Internet of Things, cyberphysical systems, blockchain, pervasive computing, cybersecurity, and applied machine learning. He is a Senior Member of the ACM, an Humboldt Research Fellow, and an ACM Distinguished Speaker. He serves as the Editor in Chief of the *Ad Hoc Networks* journal and as an Associate Editor of the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, *Computer Communications*, and *Pervasive and Mobile Computing*. He has served on the organizing committee of several IEEE/ACM international conferences.

SANJAY CHAWLA received the Ph.D. degree from the University of Tennessee, Knoxville, USA, in 1995, under Professor Suzanne Lenhart. He is currently a Professor of pattern and data mining with the School of Information Technologies, University of Sydney. He served as the Head of School, from 2008 to 2011. His interests straddle data mining, machine learning, and spatial data management. His research work has appeared in leading data mining journals and conferences including ACM TKDD, Machine Learning, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, DMKD, ACM SIGKDD, IEEE ICDM, SDM, and PAKDD. He is an Associate Editor of the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERINGand serves on the editorial board of *Data Mining and Knowledge Discovery*. He served as a Program Co-Chair of PAKDD 2012.

TARIQ AHAMED AHANGER is currently an Associate Professor with the Department of Information Systems, College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University. He has authored over 40 refereed articles. His interests include the Internet of Things, cybersecurity, and artificial intelligence.

USMAN TARIQ received the Ph.D. degree in information and communication technology in computer science from Ajou University, South Korea. He is a skilled Research Engineer. He has strong background in ad hoc networks and network communications. He experienced in managing and developing projects from conception to completion. He has worked in large international scale and long-term projects with multinational organizations. He is currently attached with Prince Sattam Bin Abdulaziz University as an Associate Professor with the College of Computer Engineering and Science. His research interests span networking and security fields. His current research is focused on several network security problems: botnets, denial-of-service attacks, and IP spoofing. Additionally, he is interested in methodologies for conducting security.