DNS Query Forgery: A Client-Side Defense Against Mobile App Traffic Profiling

Andrea Jimenez-Berenguel[1], César Gil[2], Carlos Garcia-Rubio[1], Jordi Forné[2], Celeste Campo[1]
[1]Dept. of Telematic Engineering, Universidad Carlos III de Madrid, Leganés (Madrid), Spain
[2]Dept. of Telematic Engineering, Universitat Politècnica de Catalunya, Barcelona, Spain

***Abstract*—Mobile applications continuously generate DNS queries that can reveal sensitive user behavioral patterns even when communications are encrypted. This paper presents a privacy enhancement framework based on query forgery to protect users against profiling attempts that leverage these background communications. We first mathematically model user profiles as probability distributions over interest categories derived from mobile application traffic. We then evaluate three query forgery strategies—uniform sampling, TrackMeNot-based generation, and an optimized approach that minimizes Kullback-Leibler divergence—to quantify their effectiveness in obfuscating user profiles. Then we create a synthetic dataset comprising 1,000 user traces constructed from real mobile application traffic and we extract the user profiles based on DNS traffic. Our evaluation reveals that a 50% privacy improvement is achievable with less than 20% traffic overhead when using our approach, while achieving 100% privacy protection requires approximately 40-60% additional traffic. We further propose a modular system architecture for practical implementation of our protection mechanisms on mobile devices. This work offers a client-side privacy solution that operates without third-party trust requirements, empowering individual users to defend against traffic analysis without compromising application functionality.**

***Index Terms*—DNS traffic, Data Perturbation Techniques, Privacy-Enhancing Technologies, Query Forgery, User Privacy, User Profiling**

I. Introduction

Over the past decade, mobile application usage has grown substantially due to improved device capabilities, increased high-data content, and enhanced network performance [1]. As users interact with these applications (hereafter abbreviated as apps), they explicitly and implicitly disclose personal information. This information is typically utilized by Personal Information System (PIS), such as Location-Based Systems (LBS) and Recommender Systems (RS), which tailor services based on user data. However, the very process of personalization introduces inherent privacy risks. As personal data flows into countless online services, unintended parties—including cybercriminals and network observers—gain more ways to target users and their assets. In response to these privacy concerns, Privacy-Enhancing Technologies (PET) have been in development since the late 20th century, aiming to protect user information in data-driven environments.

The vulnerabilities of user profiling, traditionally associated with foreground interactions, is not limited to direct app-server communications. Background interactions as Domain Name System (DNS) traffic also enable user profiling. Every time a user loads a webpage, or sends a message, DNS is the first checkpoint on the path as it is responsible for translating human-readable domain names into machine-readable IP addresses. The DNS queries expose sensitive data that may be exploited for profiling purposes.

Network observers such as eavesdroppers or DNS resolvers can infer user behavior from the domain patterns. Despite of the development of encrypted DNS protocols such as DNS over TLS (DoT), DNS over HTTPS (DoH), and DNS over QUIC (DoQ) their overall adoption remains relatively low as stated in the study [2]. DNS queries continue to be transmitted in clear text over port 53 (Do53) as demonstrated in the analysis conducted by [3].

Although encrypted DNS communications between clients and resolvers prevents eavesdropping, a malicious or compromised resolver can still violate user privacy. Moreover, most encrypted-DNS traffic is handled by a few major providers such as Google and Cloudflare [1]. Such centralization could lead to data monopolization that ultimately harms clients' privacy. There are emerging DNS Privacy-preserving solutions such as [4, 5, 6, 7] to address this vulnerability.

As shown in our analysis, domain names from DNS queries within a fixed time window readily expose user activity patterns, even with encrypted communications. This fundamental vulnerability exists wherever DNS observers can see queried domains. Our main contribution addresses this privacy risk by applying Data Perturbation Techniques (DPT) to DNS traffic through query forgery. While previous work like [8] has applied these techniques to foreground interactions in location-based systems, we extend them to background DNS communications. Our approach mixes genuine DNS queries with strategically generated false ones, creating an obfuscated view of user behavior. Unlike traditional DPT applications that balance personalization utility against privacy, our DNS implementation only incurs network overhead costs. This overhead represents a reasonable trade-off for the significant privacy protection gained against profiling attempts based on DNS traffic analysis.

In this context, this paper makes the following main contributions. First, we present a mathematical model of user profiles derived from DNS queries and we apply DPT via query forgery to enhance DNS privacy. Second, we evaluate three false-query strategies—uniform sampling, a

Corresponding author: Andrea Jimenez-Berenguel (andrejim@pa.uc3m.es).

[1]https://dnscrypt.info/public-servers/

Profiling Parameter Type	References	Applications
HTTP/HTTPS parameters	[11, 12, 13, 14]	Personality inference, academic classification
DNS parameters	[15, 16]	Behavior profiling
Network metadata parameters	[17, 18]	Activity pattern recognition

TABLE I: User profiling approaches by parameter type, references, and applications.

TrackMeNot-inspired generator [9], and a Kullback-Leibler (KL)-divergence–minimizing optimizer [10]. Third, we build a synthetic dataset by mapping a real dataset traffic onto 1,000 users and we profile these users with our proposed method based on DNS traffic. Fourth, through theoretical analysis and experimental evaluation, we show that query forgery substantially reduces profiling accuracy with minimal performance degradation, and we quantify the trade-off between added network overhead and achieved privacy gains. Finally, we propose a modular system architecture for a potential implementation of our DNS privacy model.

The remainder of this paper is organized as follows. Section II presents a comprehensive review of the state of the art. In Section III, we formally define the problem of DNS-based user profiling. Section IV introduces our proposed DNS query forgery mechanism to enhance user privacy against profiling based on DNS traffic. Section V describes the creation of our synthetic dataset and outlines the methodology for generating user traces and deriving DNS-based user profiles. In Section VI, we present our experimental evaluation, analyzing the trade-off between privacy enhancement and traffic overhead, and discuss the results obtained from applying our method to 1,000 synthetic users. We propose a practical adaptation of query forgery for mobile apps in Section VII. Finally, Section VIII concludes the paper and outlines directions for future research.

II. State of the Art

In this section, we review three key areas of literature. First, we examine studies on user profiling through network traffic analysis. Next, we explore PETs and their real-world applications. Finally, we review existing works that apply PETs to mitigate privacy vulnerabilities in DNS communications. This review frames our contribution within the broader privacy protection landscape.

A. User profiling

User profiling through network traffic analysis involves constructing behavioral patterns of users based on their digital activities. This subsection reviews various methodologies used to extract user profiles from network traffic parameters. Each approach leverages distinct aspects of network communications to reveal behavioral patterns and user interests. We categorize the literature into three main groups: profiling based on HTTP/HTTPS parameters, DNS parameters, and broader network metadata, as summarized in Table I.

Several studies have exploited the characteristics of HTTP/HTTPS traffic to build user profiles. Although HTTPS makes profiling more difficult, it does not eradicate it. Gonzalez et al. [11, 12] demonstrated that user profiling is possible despite encryption. From an eavesdropper's perspective, in [11], they utilized the URL a user visits, which can be obtained from the Server Name Indication (SNI) in the TLS handshake's client_hello message, and web fingerprinting techniques to construct user profiles. From a network observer's perspective, in [12], they analyzed sequences of hostnames visited by users within a predefined period of time. Using a custom Chrome extension to collect browsing data, they applied natural language processing algorithms to infer relevant topics from accessed websites, effectively creating user profiles despite encryption protections.

Building on HTTP/HTTPS-based profiling approaches, Park et al. [13] conducted a comprehensive investigation into four distinct profiling scenarios: (i) profiling based on timestamps; (ii) profiling based on HTTP headers; (iii) profiling based on domain names, assuming interpretable topical categories of URLs; and (iv) profiling based on page content, noting its inapplicability to HTTPS traffic where URLs are encrypted and only domain names remain visible. Their research used a proprietary dataset of mobile network traffic from 61 Spanish participants. By analyzing HTTP(S) traffic patterns of the users, they modeled personality traits, shopping interests, and demographic characteristics.

Gao et al. [14] integrated HTTP(S) parameters with network access records in a campus environment to enhance user profiling capabilities. Their methodology first extracted fundamental identifiers (MAC address, login/logout times, device location) from network access records, then enriched this data by analyzing HTTP(S) packets for additional identifiers like destination IP addresses. Using these parameters, they built a classifier based on Back Propagation Neural Networks (BPNN) to distinguish between different user types and predict academic disciplines.

Several researchers have explored DNS traffic as a rich source of information for user profiling. Shaman et al. [15] proposed a method for user identification and behavior profiling using DNS information. They collected a dataset from 23 users on the Plymouth University network, filtered users based on MAC/IP address mappings, and identified apps using reverse DNS queries. The authors employed a gradient boosting machine learning algorithm to create user profiles based on features such as date and time, destination IP address, and DNS queries.

Lyu et al. [16] conducted an analysis of unencrypted DNS traffic collected over one month from both a university campus and a government research institute. Their research identified distinctive behavioral patterns among various DNS asset types (recursive resolvers, authoritative name servers, and mixed DNS servers). By capturing normal DNS activity patterns and detecting anomalies indicative of security issues, they demonstrated DNS traffic's dual utility for both profiling and security monitoring. Their implementation of an unsupervised machine learning algorithm to classify over 100 DNS assets based on network, functional, and service characteristics further established DNS behavior profiling as a valuable tool for automated security management.

Beyond HTTP/HTTPS and DNS parameters, broader network metadata provides valuable insights for user profiling without requiring access to communication content. Alotibi et al. [17] developed user behavioral profiles based on network metadata parameters such as connection type, duration, number of packets, and packet size. Their research utilized custom-collected network metadata from 27 participants with static IP addresses, providing ground truth for their analysis. After filtering the traffic to focus on user interactions with popular apps (Google, YouTube, Skype, Facebook, etc.), their methodology achieved remarkable identification accuracies. Their results demonstrate network metadata's effectiveness for forensic investigations targeting insider threats.

Li et al. [18] analyzed user activity sequences using real-world data from a Shanghai ISP. Their methodology constructed user profiles from network access records containing user IDs, timestamps, and connection metadata. Their approach used the ISP's systematic classification of apps into categories (games, shopping, education) and observed that each user's records followed a power-law distribution. By segmenting app usage traces into time windows and applying a probabilistic topic model, they successfully inferred users' cyber activities. Their research conclusively demonstrated that digital activity patterns could characterize users' daily life both individually and collectively.

In conclusion, previous studies has shown that a variety of traffic network parameters—including HTTP/HTTPS attributes (timestamps, headers, hostnames), DNS characteristics (query patterns, response types), and generic flow metadata (durations, packet sizes)—can be used to reconstruct user behavior even when traffic is encrypted.

B. Privacy-Enhancing Technology Applications

Numerous PETs have been proposed in the literature and applied in real-world scenarios. According to [19], a possible classification of PETs can be summarized into five groups: (a) basic anti-tracking technologies, (b) approaches based on Trusted Third Parties (TTPs), (c) collaborative mechanisms, (d) methods based on Private Information Retrieval (PIR) cryptography, and (e) DPTs. We concentrate on DPTs which aim to obfuscate the data users share with PIS. In practice, these techniques are specifically designed to hinder the precise profiling of users by third-party privacy attackers. The paradigmatic example of DPT is the transmission of real user data mixed with false data.

Among the five broad PET categories, DPTs operate under a zero-trust model regarding third-party entities. Unlike other approaches that rely on trusted intermediaries, DPTs treat any third party as a potential privacy threat. This approach implements local privacy (user-side privacy), although it can still be combined with collaborative profiling mechanisms when appropriate.

Another important property of DPTs in the context of PISs is the trade-off between cost and benefit. The primary goal of these techniques is to find a balance between system functionality cost (personalization), which depends on data utility, against user privacy protection, which mitigates profiling risk. In DNS traffic between mobile apps and DNS servers—where no personalization occurs—this functionality cost manifests as network overhead rather than reduced service quality.

Data-perturbation techniques	References	Applications
• Forgery	[9, 10, 20, 21, 22, 23, 24, 25, 26]	PWS, PIR, RSs
• Supression	[27, 28, 29]	PWS, RSs
• Both	[30, 31]	RSs
• Generalization	[8]	LBS

TABLE II: DPTs, references, and applications.

While DPTs are typically applied to PISs, their principles can be extended to the scenario of DNS traffic generated by mobile apps. This subsection examines the different approaches to data perturbation of the literature, which we categorize based on their mechanism: forgery, suppression, hybrid approaches combining both, and generalization, as summarized in Table II with their respective references and applications.

It is important to clarify that, in all cases, we refer to deterministic perturbations, as opposed to techniques that rely on random perturbations. Notably, all these techniques have analogous counterparts in the field of Statistical Disclosure Control (SDC), although the object of protection differs. While the studies analyzed in this section focus on protecting a user profile—typically modeled as a Probability Mass Function (PMF)—SDC techniques are designed to safeguard an entire database of records.

Query forgery techniques involve adding false queries to genuine ones. This approach allows users to protect themselves from precise profiling by privacy attackers while avoiding the need to rely on third parties.

Several significant query forgery–based proposals had emerged in the literature. The private web browsing system known as PRAW [20, 21, 22, 23] complicated user profiling by generating false browsing traces when users accessed the web through a shared login session. Similarly, in [24] the authors presented a query injector that generated false queries with probabilities complementary to real ones. That approach assumed that the proportions of real and false queries remained inaccessible to an adversary and were only available on the user side.

A software implementation of query forgery was GooPir [25]. GooPir operated by sending batches of both genuine and false keywords to a web search engine. The selection of false keywords was based on usage frequency similar to that of genuine ones, making profiling attacks more challenging. However, in [32] the authors highlighted that this strategy could be vulnerable to correlation attacks between keywords across different batches.

Similarly, TrackMeNot [9] was a web browsing plugin that implemented query forgery using various strategies. False queries were generated through a continuously updated keyword dictionary sourced from diverse information channels. The transmission strategy of false queries to the server could either mimic human behavior through bursts or be set at predefined time intervals. As with the previous proposal, [33] argued that certain semantic or timing-based attacks on false queries could lead to potential inference of real queries, thus

exposing TrackMeNot users.

A fundamental drawback of adding false queries was the implicit traffic overhead it generated. Addressing this challenge required balancing privacy and overhead, a scenario studied in [10] within the field of PIR. In that work, the authors presented a mathematical model to achieve an optimal trade-off between the rate of falsified queries and user privacy. Later, in [26], researchers investigated and validated tag forgery in real-world scenarios within content-based RSs.

Data suppression was a fully viable and conceptually straightforward DPT, representing the opposite approach to adding activity to a user profile. Suppression had been validated in various scenarios. In [27], applied to the Semantic Web, a limited privacy improvement was achieved through a tag removal process, incurring resource costs that traded off with semantic degradation. The same objective was analytically explored via convex optimization in [28]. Shannon entropy of the perturbed profile and the proportion of tags the user was willing to remove served as the respective privacy and utility metrics for studying the optimal balance. Finally, in [29], parental control and resource recommendation were presented as application scenarios. The evaluation of suppression-based perturbation considered costs resulting from data degradation and the accuracy of predefined parental control policies, offering an insightful perspective.

A combined approach involving both forgery and suppression was also applied to personalized RSs (e.g., Amazon, Spotify, Netflix). Essentially, this strategy, investigated in [30], enabled users to submit false ratings and/or withhold ratings for items of interest. A closed-form solution to the problem of optimal and simultaneous forgery and suppression of real-world ratings was presented in [31].

More recently, in [8] the authors proposed and evaluated a real-world data perturbation strategy based on the generalization of interest categories arranged in a hierarchical taxonomy with varying depth levels. That study proved effective in systems utilizing hierarchical semantic taxonomies or in LBSs, where POI coordinates could be recursively categorized within a properly partitioned area of interest. As in previous studies, and among other properties of the convex optimization problem modeling the privacy–utility trade-off, the authors introduced a critical ratio as a measure of the maximum generalization rate beyond which privacy could not be further improved.

In conclusion, DPTs present various methodologies for enhancing user privacy protection. Query forgery mechanisms, as implemented in systems like PRAW, GooPir, and TrackMeNot, function by strategically mixing false queries with genuine ones to obfuscate user profiles. Data suppression techniques operate on the contrary principle, selectively removing sensitive elements from user profiles to limit information disclosure. Hybrid approaches combine both falsification and suppression strategies to optimize privacy protection, while generalization techniques organize sensitive data into hierarchical taxonomies that balance utility with privacy. These different approaches demonstrate the versatility of DPTs in addressing privacy challenges across various app domains.

C. Applications of PETs to DNS

Beyond DPT, researchers have also explored PIR for DNS privacy. Bhat et al. [4] propose an information-theoretically perfect, single-database PIR scheme for private DNS resolution, and more recently Zhou et al. [34] introduce PIANO, a highly practical, single-server PIR with sublinear server work that scales to 100 GB DNS-sized datasets. However, DNS records are frequently updated, which contradicts a fundamental assumption of PIANO that requires a static database. Moreover, PIR privacy requires also server support so they cannot be deployed purely on the client side. This makes private DNS query particularly challenging to implement with PIR-based approaches.

In the same spirit of DNS privacy, Arana et al. [5] propose Never Query Alone (NQA), a cooperative routing strategy in which users forward their DNS queries through their neighbors, thereby diluting an attacker's ability to link a query to its source. NQA requires cooperation among multiple clients.

Researchers have also explored privacy-enhancing solutions specifically for DNS. Schmitt et al. [6] propose Oblivious DNS (ODNS), which decouples client identity from queries by encrypting requests that recursive resolvers forward to ODNS resolvers. Building on this concept, Singanamalla et al. [7] developed Oblivious DNS over HTTPS (ODoH), adding HTTPS transport to ODNS. Cloudflare has implemented ODoH in their public DNS resolver [2], and the protocol is being standardized through an IETF draft co-authored by Cloudflare and Apple [35].

In conclusion, building upon the aforementioned works, our paper aims to demonstrate that the domain names queried over a fixed time window leaks rich user-level behavior patterns. As long as eavesdroppers or network observers sees those domain names, user activity remains fundamentally exposed. To counter this we apply DPT via query forgery not only because it empowers users to shield themselves without relying on systems that require server-side deployments, but also because it operates under a zero-trust model for any external entity and strikes a practical balance between system-functionality cost and user-privacy protection.

III. Formal Problem Statement

Generally, individual private data generated and transacted between users and PISs over communication networks can be represented as sequences of random variables. In this work, we adopt this representation for DNS traffic, specifically DNS queries or parts thereof (e.g., tuples consisting of the app issuing a query, the queried domain, and the timestamp) generated by user devices through various installed mobile apps. Ultimately, these sequences can assume values in a finite, common, and reduced alphabet of categories, which we define as the set $\mathscr{X} = \{1, \ldots, n\}$ for some integer $n \geq 2$.

Assuming that these random variables are independent and identically distributed, we mathematically model a user's profile using a PMF over the distribution of these variables. We profile each user based on the percentage of DNS queries

[2]https://developers.cloudflare.com/1.1.1.1/encryption/oblivious-dns-over-https/

generated by each app in a predefined interval. The primary advantage of PMF, widely accepted in the privacy literature [10, 36, 37, 38, 39], is its ability to efficiently aggregate large amounts of individual user data and present it as a histogram of relative frequencies across predefined interest categories. Consequently, user profiles based on discrete probability distributions are well-suited for numerical computation and widely applicable in privacy metrics.

Our goal in this formulation is not to identify specific individuals or extract personal details, but rather to formalize how DNS query patterns inherently reveal user behavior signatures. We aim to establish the mathematical foundation for understanding the fundamental privacy vulnerability that exists whenever DNS traffic can be monitored by third parties, regardless of how that information might be exploited.

Considering these aspects, we define q as the distribution of genuine DNS queries from a user, reflecting their interests (e.g., entertainment, culture, etc.) based on the DNS traffic generated by mobile apps installed on their device. Similarly, we define r as the distribution of the user's false queries and p as a reference distribution, which may correspond to the population distribution or the average distribution of a user group. In profiling terms, q, r, and p represent the user's real profile, false profile, and reference profile, respectively.

Finally, as a result of profile mixing, we define t as the user's apparent profile, derived from the combination of the real and false profiles. In this work, $t = (1 - \rho)p + \rho r$, a simple deterministic perturbation strategy based on the convex combination of genuine and false queries, where ρ is the false query rate, also known as the perturbation ratio, with values between 0 and 1.

With the formal problem statement established, the following section presents the key components of our proposal to enhance privacy protection against user profiling based on DNS traffic by employing DNS query forgery. Ultimately, we aim to determine a distribution of false queries r that optimally obfuscates the real user profile q in terms of privacy and utility, making it indistinguishable to a privacy attacker observing the apparent profile t and leveraging DNS traffic for profiling activities.

IV. DNS Query Forgery Against User Profiling

In this section, we present our proposal for enhancing user privacy against profiling based on DNS traffic generated by mobile apps installed on their devices. Fig. 1 illustrates the diagram of the proposed scenario. First, we define the user and adversary models that we assume. Next, we introduce the metrics that will be used to evaluate the obfuscation strategy forming the basis of our privacy-utility trade-off model. Finally, we provide a numerical example to illustrate the proposed privacy model.

A. User Model

The first component to consider in a security analysis is the entity that a privacy attacker will observe, taking into account the parameters that may be compromised. As introduced in Section III, the scenario considered in this work is based on a stream of DNS queries. At a given instant or over a specific time period, processing and aggregating part of the information contained in these queries allows the construction of a user profile in the form of a discrete histogram of relative frequencies, summarizing the user's preferences across a finite set of predefined interest categories. A straightforward definition of these categories can be derived from mapping the mobile apps used by the user on their device, an association inferred from the traces of DNS queries.

This user model, widely studied and employed in the field of PISs, enables us to define and delineate the profiling attack performed by an adversary. It is important to reiterate that user activity profiling is a primary concern in such systems.

In general, profiling is a method used to identify and characterize individuals by generating and applying profiles. However, as discussed in the literature [40, 41], user identification can be understood from two distinct perspectives: as *individuation*, referring to the revelation of an individual's unique attributes, and as *classification*, which involves categorizing an individual as a member of a group.

This duality in profiling usage implies the creation of both individual and group profiles. For instance, PISs are commonly characterized by individual profiling, which personalizes services based on each user's specific interests. Conversely, other systems seek to adapt a group profile to users who may not have directly contributed to that profile.

In this work, we focus on user individuation, adapting the adversary model and privacy metrics to this specific profiling activity. Ultimately, the real and apparent user profiles, denoted as q and t, respectively, will be the targets of the adversary's profiling, whose model we define in the following section.

B. Attacker Model

The level of privacy provided by a PET directly depends on the assumptions we make about the adversary. For this reason, evaluating the effectiveness of a PET requires a proper characterization of the privacy attacker. Clearly, depending on the adversary's properties, a user may implement different techniques, including those reviewed in Section II-B.

Throughout this work, we consider an adversary capable of accessing the DNS traffic generated by the exchange of information between the various mobile apps installed on a user's device and the servers that resolve DNS queries—an essential requirement for the apps to function properly.

Given this access, we assume an adversary who can filter DNS queries and profile users by inferring their interests through the analysis of the information contained in the traces recorded on DNS servers, following approaches such as those described in [3]. The technique we propose assumes that a user aims to conceal their bias toward specific categories of interest by perturbing their traffic with false DNS queries. This ensures that the apparent profile t, as observed by any attacker, approximates either a uniform profile u or the average user profile $\bar{q}$, while deviating as much as possible from the real profile q.

A final but crucial assumption regarding the privacy attacker is their inability to estimate a user's rate of false queries ρ. This

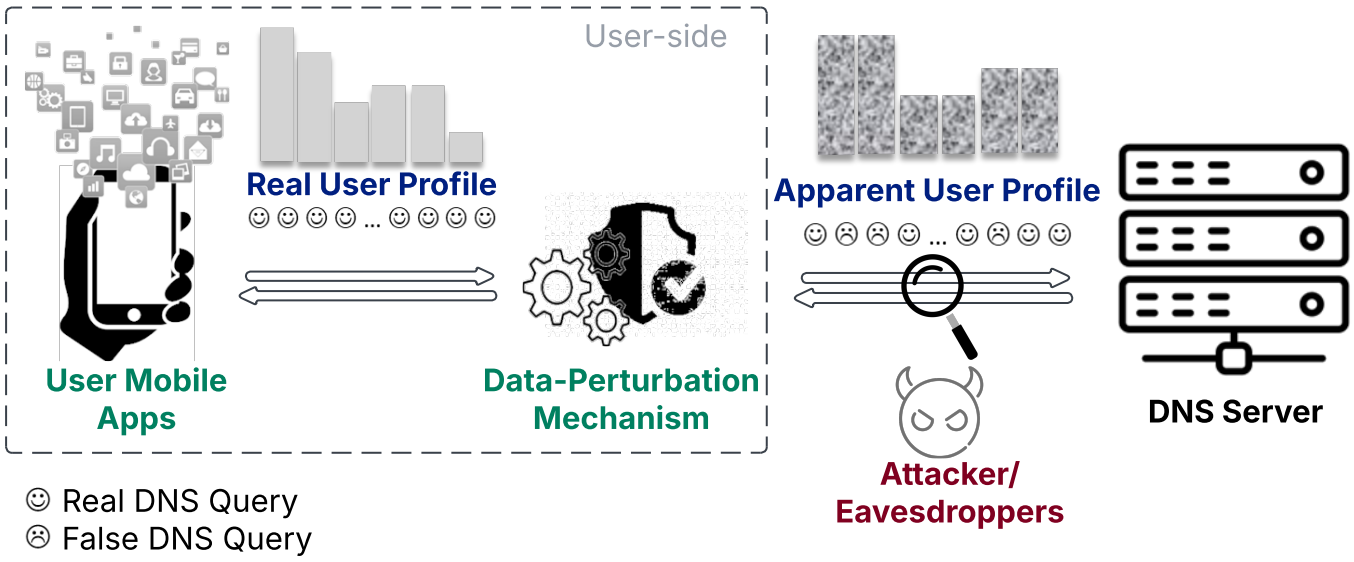


Fig. 1: Representation of the secured scenario with the user model, the data-perturbation mechanism and attacker model.

is based on the premise that the adversary lacks knowledge of whether the user employs the proposed privacy strategy.

C. Metrics

We dedicate this section to justifying and describing the privacy and utility metrics selected for our privacy protection proposal concerning user DNS traffic in mobile apps. For a detailed analysis, these metrics have been extensively studied in [10, 42].

First, we chose to use privacy measures derived from Information Theory (IT). Specifically, we handle two key concepts: Shannon entropy and KL divergence. For readers unfamiliar with this field, we briefly review both measures below.

The Shannon entropy of a discrete random variable with PMF q taking values in the set $\mathscr{X} = 1, \ldots, n$ is a measure of the uncertainty of this random variable and is defined as

$$\mathrm{H}(q) = \sum_i q_i \log_b(q_i). \tag{1}$$

where b [3] is the base of the logarithm used. However, all bases produce equivalent optimization objectives.

Similarly, the KL divergence between two discrete random variables with PMFs q and p is a measure of their divergence, also referred to as relative entropy, as it generalizes the Shannon entropy of one distribution with respect to another. It is defined as

$$\mathrm{D}(q||p) = \sum_i q_i \log_b(\frac{q_i}{p_i}). \tag{2}$$

It is worth noting that Shannon entropy can be considered a special case of KL divergence when the reference distribution p is the uniform distribution u, i.e., $p = u$.

[3]Common values of b are 2, e and 10. In those cases, the units of entropy are bit, nat and dit, respectively

With these notions in place, we define the privacy risk function $\mathcal{R}$ as the divergence between the user's apparent profile t and the reference profile p, that is,

$$\mathcal{R}(\rho) = \mathrm{D}((1-\rho)q + \rho r||p), \rho \in [0, 1]. \tag{3}$$

Recall that the apparent profile t results from applying a simple perturbation strategy based on a convex combination equivalent to mixing the user's real profile q with the false profile r in a proportion ρ, which we refer to as the perturbation rate.

At this point, we address the selection of the user's false profile r as a central element of the false DNS query mechanism investigated in this work. To this end, we consider three variants for shaping this discrete distribution.

First, the simplest option consists of diluting the user's real profile with the uniform distribution u, applying the same number of false DNS queries, $1/n$, to each of its n components, which correspond to categories in our case.

The second option is based on the well-known TrackMeNot mechanism [9]. For practical and simplification purposes, we assume that the false query distribution proposed by this PET is the average of the distribution of a set of users, which we denote as $\bar{q}$.

Finally, we consider an optimized option based on the proposed metrics. In essence, the false query distribution r results from optimizing the convex problem formulated as minimizing KL divergence in the unit simplex Δ_r [10], i.e.,

$$r^* = \arg\min_{r \in \Delta_r} \mathrm{D}((1-\rho)q + \rho r||p). \tag{4}$$

It is important to emphasize that a theoretical analysis of the properties of the function $\mathcal{R}$ is initially based on the assumption that the distributions q and p, understood as probabilities, are strictly positive:

$$q_i, p_i > 0 \quad \text{for all} \quad i = 1, \ldots, n. \tag{5}$$

TABLE III: Summary of the three DNS query forgery strategies we investigated in this paper.

Mechanism	Distribution
Uniform (UNF)	u (uniform distribution)
TMN-based (TMN) [9]	$\bar{q}$ (TMN distribution)
Optimized (OPT) [10]	$r^* = \arg\min_{r\in\Delta} \mathrm{D}(t\|\|p)$

However, in our work, we may refer to continuity arguments to relax this assumption. Moreover, without loss of generality, we assume that

$$\frac{q_i}{p_i} \leq \cdots \leq \frac{q_n}{p_n} \quad \text{for all} \quad i = 1, \ldots, n. \tag{6}$$

We highlight to the reader the initial and final values of the privacy risk function, $\mathcal{R}(0) = \mathrm{D}(q||p)$ and, in the case of the optimized option, $\mathcal{R}(1) = 0$. Analyzing the behavior of $\mathcal{R}$ for intermediate values of ρ when KL divergence is minimized in the unit simplex Δ_r reveals important properties, such as monotonicity and convexity, the existence of a critical perturbation rate ρ_{crit} beyond which privacy is maximized ($\mathcal{R}(\rho) = 0$ for $\rho \geq \rho_{crit}$), and the existence of an optimal and closed-form solution. The theoretical value of this critical rate is expressed as

$$\rho_{crit} = 1 - \frac{q_n}{p_n} \tag{7}$$

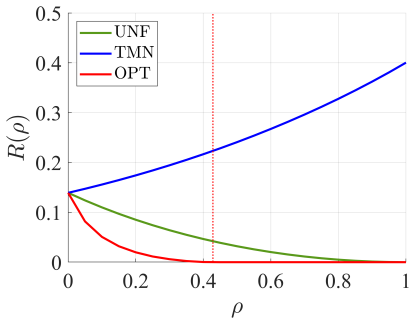
In general, the user's knowledge of the reference distribution p determines whether the appropriate metric is divergence or entropy. Table III summarizes the selected variants for the false DNS query mechanism that forms the core of our research proposal.

Finally, as a utility metric for our proposal, we directly consider the perturbation rate ρ. Intuitively, a higher false query rate leads to greater traffic overhead and a more significant degradation of the user's original profile, making precise profiling— a privacy threat— more challenging. We understand that an increase in DNS traffic reduces the user's quality of experience with their mobile apps.

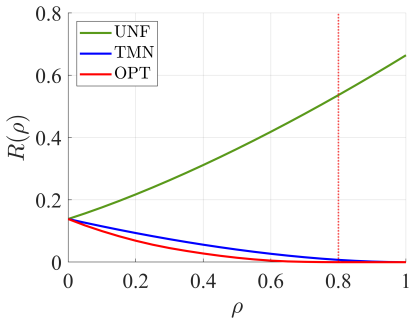
D. Numerical Example

We dedicate this final section to illustrating the privacy model proposed in this work. To this end, we present some results based on an example that allows the reader to become familiar with the introduced concepts. A more in-depth evaluation of our mechanism in a real-world scenario is presented in Section VI.

Let us consider a scenario where, over a given period and using a standard device, a user's mobile apps have sent 100 queries to a reference information system. These queries can be grouped into a set of five categories summarizing the user's interests, denoted alphabetically as $\{a, b, c, d, f\}$. Assuming that the query frequencies are $(5, 15, 20, 25, 35)$, the user's actual profile with $n = 5$ categories is consequently $q = (0.05, 0.15, 0.20, 0.25, 0.35)$. Additionally, we consider the uniform profile $u = (0.20, 0.20, 0.20, 0.20, 0.20)$ and the population profile $\bar{q} = (0.01, 0.05, 0.15, 0.35, 0.44)$.



(a) Entropy-based risk function



(b) Divergence-based risk function

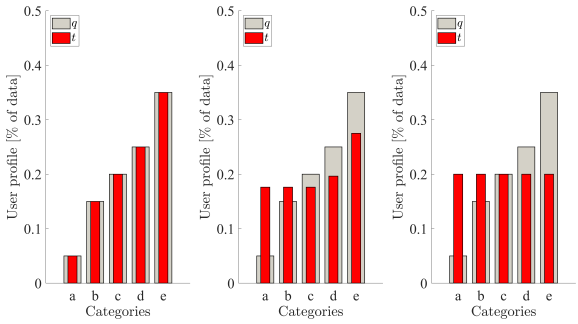
Fig. 2: Privacy risk $\mathcal{R}(\rho)$ according to the perturbation ratio ρ for the numerical example for the three perturbation strategies. Denoted by a red dashed vertical line, the ρ_{crit} value.

Given these parameters, the initial value of the risk function based on KL divergence when $p = u$ (equivalent to Shannon entropy) and when $p = \bar{q}$ is the same, i.e., $\mathcal{R}(0) = 0.1386$. However, the final value varies depending on the perturbation strategy and the metric implemented.

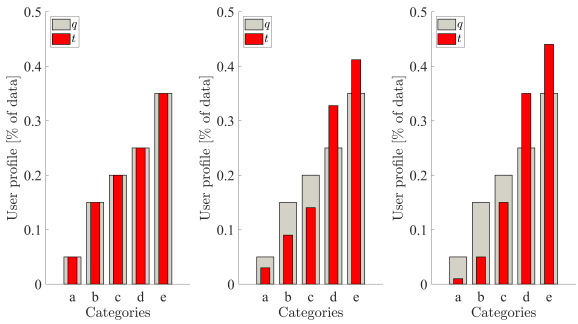
In Fig. 2, we depict the privacy risk function based on entropy and KL divergence for this fictitious user as a function of the perturbation rate ρ. In both cases, we show the curves corresponding to the three data perturbation mechanisms introduced in Section IV-C, namely, uniform (UNF), TrackMeNot-based (TMN), and optimized (OPT). The figure also includes the critical perturbation rate ρ_{crit}, specific to the optimized mechanism, defined as the minimum rate beyond which the user's privacy risk is null. When using entropy as the metric, $\rho_{crit} = 0.6643$, whereas for KL divergence, $\rho_{crit} = 0.8$.

As highlighted in Section IV-C, the privacy risk function $\mathcal{R}$ is monotonic and convex in the case of the optimized mechanism. Undoubtedly, this mechanism is superior to suboptimal mechanisms in terms of privacy risk. Intuitively, and as corroborated by the figures, applying a suboptimal mechanism where the reference distribution coincides with the distribution of false queries contradicts the objective of minimizing privacy risk.

Furthermore, Fig. 3 illustrates how the user's profile evolves as the perturbation rate ρ varies between 0 and 1 under the optimized mechanism. When $\rho = \rho_{crit}$, the apparent profile t converges to the target or reference profile p. This corresponds to the uniform profile when using entropy ($t = u$) and the population profile when using KL divergence ($t = \bar{q}$).



(a) Entropy-based risk function



(b) Relative-entropy-based risk function

Fig. 3: Evolution of the real and apparent profile (q, t) for the optimized strategy (oqf) according to different values of the disturbance ratio $\rho = (0, 1/2\rho_{crit}, \rho_{crit})$.

We implemented the numerical example and the evaluation presented in Section VI using Matlab (Matlab R2021 9.10.0.1602886 64-bit win64) and executed the computations on an Intel CoreTM i3-2370 CPU at 2.4 GHz, with 4 GB of RAM, running a 64-bit Windows 10 operating system.

To conclude this section, we emphasize that our work presents a comprehensive approach to enhancing user privacy against profiling based on DNS traffic. This is achieved by defining user and adversary models, establishing evaluation metrics to balance privacy and utility, and demonstrating their application through a numerical example. Our model focuses on perturbing user profiles by adding false data to genuine ones, effectively obscuring the real interests of users from potential attackers. To validate the impact of our proposal under real-world conditions, the next section introduces the user dataset and obtains the user profiles employed in the final evaluation. Our dataset consists of 1,000 synthetically generated user traces, a number we consider sufficiently significant to yield robust experimental results. Each trace is composed by mobile app traffic and represents the traffic generated by a user over a time interval.

V. Users Dataset

This section describes the selected mobile app traffic dataset used to create our synthetic users, then details the creation of our synthetic dataset composed of mobile app traffic from 1,000 synthetic users. Finally, we explain the process of extracting user profiles based on DNS traffic. These user profiles will subsequently be used to evaluate our proposed privacy model.

We generate a synthetic dataset because there is a notable lack of publicly available datasets that include traffic tied to individual users. Previous studies that analyzed personal traffic [12, 13, 14, 17, 18] do not release their data due to reasons such as ethical concerns regarding user privacy and the risks associated with profiling real users.

The synthetic user generation consist of assigning specific apps and time intervals of traffic of those apps to each user. To create the synthetic user traces we use mobile app traces. Thus, the traffic traces labeled by app provides a controlled environment in which we have a priori knowledge of the active app at any given time, ensuring reliable labeling of traffic per user.

In the literature there are several datasets of mobile apps in Packet Captured (PCAP) format files. The Cross Market dataset [43] consists of network traffic from 229 apps randomly selected from the most popular Android apps in three countries (China, India, and USA) in 2017 [4]. The MAppGraph dataset [44] which was collected in 2021 and available upon request. The original dataset has traces from 101 popular Android apps in Vietnam; however, the version shared with researchers includes 81 apps. Lastly, the dataset by Mankowski et al. [45] from 2023 comprises traces for 90 Android apps from the German market. The traces from the Cross Market and Mankowski datasets include one traces per app with an average duration of 5 minutes. In contrast the MAppGraph dataset has 330 minutes of traffic per app on average, totaling nearly 500 GB of data.

Therefore, we selected the MAppGraph dataset as our source for mobile app traffic traces to generate 1,000 synthetic user traces. This dataset was chosen because it provides data for 81 mobile apps, offering a broad range of app traffic. Each app includes an average of 330 minutes of traffic, which is more extensive compared to other available datasets. Furthermore, its collection in 2021 ensures that the traffic is both recent and relevant. These factors make the MAppGraph dataset ideal for creating diverse and reliable synthetic user traces.

A. MAppGraph Dataset Description and preprocessing

The MAppGraph dataset comprises encrypted traffic captures generated by Android mobile apps. Data we collected at Tan Tao University in Vietnam during multiple sessions. In each session, volunteer students used smartphones provided by the research team to access apps from a predefined list. The primary goal was to record traffic from individual app executions by human users rather than complete user profiles. Despite human users generated the traffic, the dataset contains only records of app executions, not continuous user-specific traces.

Although the MAppGraph dataset primarily contains encrypted traffic, it also includes unencrypted DNS (Do53)

[4]At the time this research was conducted, the Cross Market Dataset was publicly available. However, as of the publication date, the dataset is no longer accessible to the public.

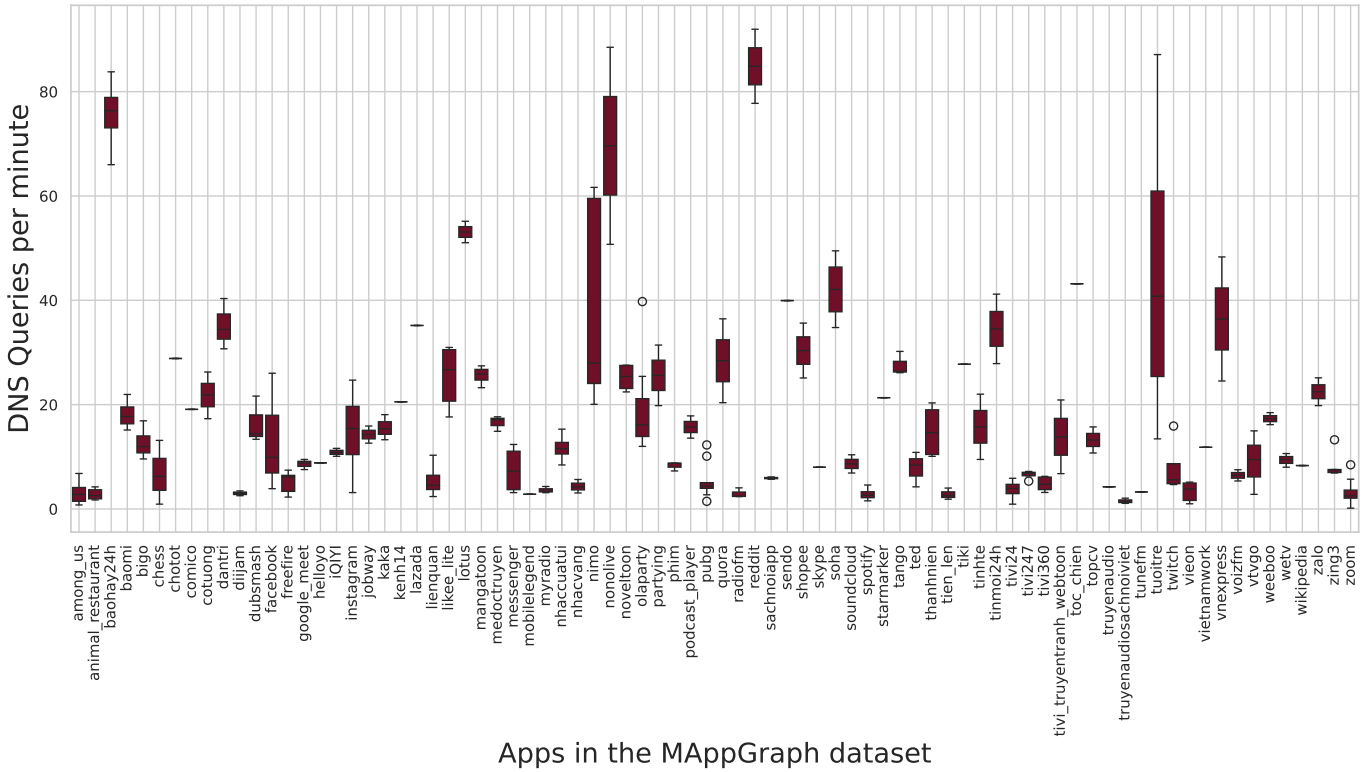


Fig. 4: DNS queries per minute in the MAppGraph dataset.

traffic. In this study we use the DNS traffic to obtain a user profiling, as cleartext DNS queries can be used to infer the app generating the traffic [3]. The preprocessing step involved extracting the timestamp and domain name from the DNS requests of each PCAP file. This extracted parameters are stored in a CSV file per trace; in what follows, we will refer to this CSV files as the traces of the apps or users. In Fig. 4 illustrates the range of DNS queries per minute for each app in the MAppGraph dataset. We can see that the density of DNS queries is different per app.

B. Generation of synthetic user traces

Our approach for generating synthetic user traces involves a twofold process: first, we analyse the mobile app usage behavior in order to obtain the distribution of the installation percentage of the categories and apps. Then, we create the synthetic user traces and we assign traffic to them. The code to create 1000 synthetic user traces is available in GitHub[5].

1) Mobile App Usage Behaviour: Previous studies have shown that mobile app usage follows a power-law distribution: users tend to rely on one main app, and the likelihood of using additional apps drops off according to a power law [46].

To capture this behavior, we analyze the number of installations for each app present exclusively in our dataset. Detailed installation numbers for each app, which can be found in [44], support our analysis and demonstrate that the installation numbers follow a power-law trend. Then, we grouped the apps into 10 categories based on the app-category of Google play.

[5]https://github.com/AndreaJimBerenguel/user_profiling

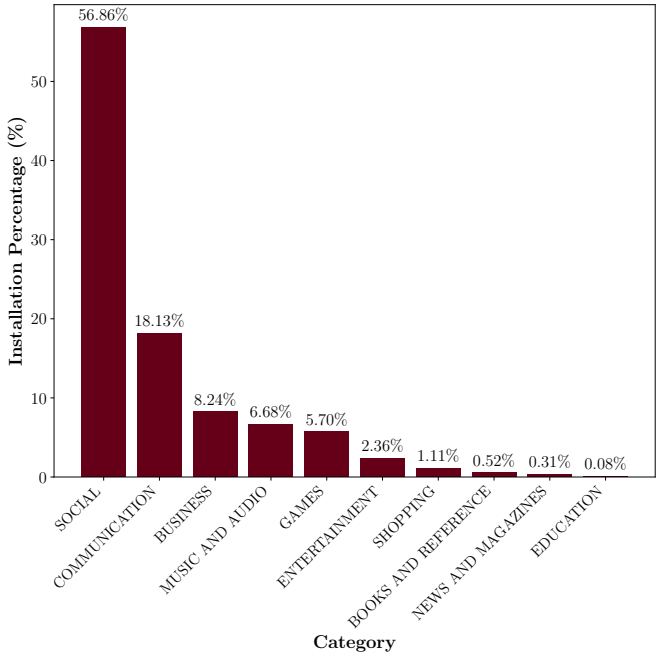
In Table IV are presented the categories and the number of apps per category from the MAppGraph dataset. The numbers of apps sum 80 because we omitted one app due to insufficient traffic trace data.

Category	Number of apps
BOOKS AND REFERENCE	9
BUSINESS	4
COMMUNICATION	4
EDUCATION	2
ENTERTAINMENT	15
GAMES	10
MUSIC AND AUDIO	10
NEWS AND MAGAZINES	11
SHOPPING	5
SOCIAL	10

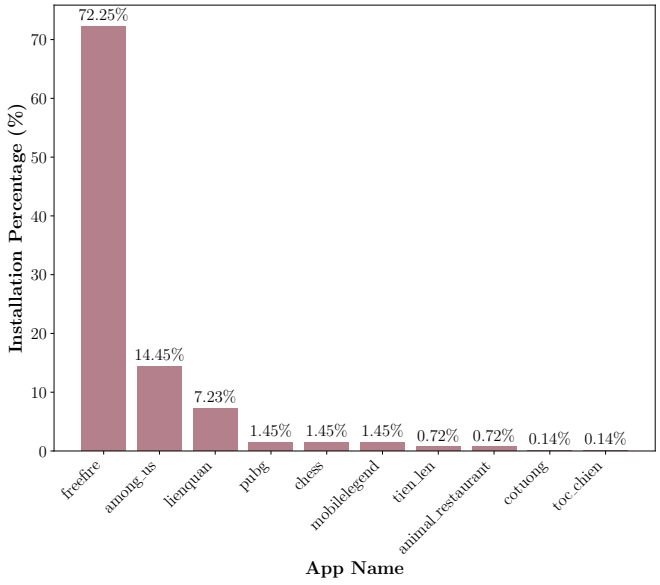
TABLE IV: Categories and number of apps per category.

The Fig. 5 are represented the probabilities of installation: the histogram shown in Fig. 5(a) illustrates that the installation probabilities aggregated by categories (inter-category) follow a power-law distribution. Moreover, in Fig. 5(b) are represented the individual installation probabilities of the apps from the category *games* as an example. We can see that when we analyze the apps within the same category (intra-category), the installation probabilities of individual apps within a category also exhibit a similar power-law trend.

Based on the power-law distribution of the inter-category and intra-category app usage we obtain a set of apps for each user. In our methodology, we assume that the probability of a user selecting a particular app is proportional to its installation probability.



(a) Installation probabilities of the apps aggregated by categories (probabilities inter-categories)



(b) Installation percentage of the apps from *games* (probabilities intra-categories)

Fig. 5: Histograms illustrating the installation probabilities inter and intra-category which follow a power-law distribution.

a) Category and weight assignation: First, we assign to each user the categories and its weight. In this case, the weights are equivalent to usage probabilities. We obtain samples from the power-law distribution of the categories. Each user gets a sample of this distribution. Because of the nature of the power-law distribution sampling, each user ends up with a unique weights vector, some categories may receive a zero weight while others receive a non-zero value. For example, user A obtains this categories and weights [‘social: 68’; ‘communications: 11’; ‘business: 6’; ‘music and audio’: 7; ‘games: 4’;‘entertainment: 4’; ‘shopping: 0’; ‘books and references: 0’; ‘news and magazines: 0’; ‘education: 0’].

b) App per category: Next, for each category that has a non-zero probability, we randomly select one app from that category. This selection is not uniform but weighted according to the power-law distribution observed among the apps within that category. The end result is a vector named *user app usage* for each user that maps each selected category and app to its assigned percentage of usage. For example, we assign to user A one app per category with a non-zero value. In Table V we present the assignation of apps per category to user A.

Category	**App**	**Weight**
Social	Facebook	68
Communications	Messenger	11
Business	Jobway	6
Music and Audio	SoundCloud	7
Games	Freefire	4
Entertainment	Nimo	4
Shopping	–	0
Books and References	–	0
News and Magazines	–	0
Education	–	0

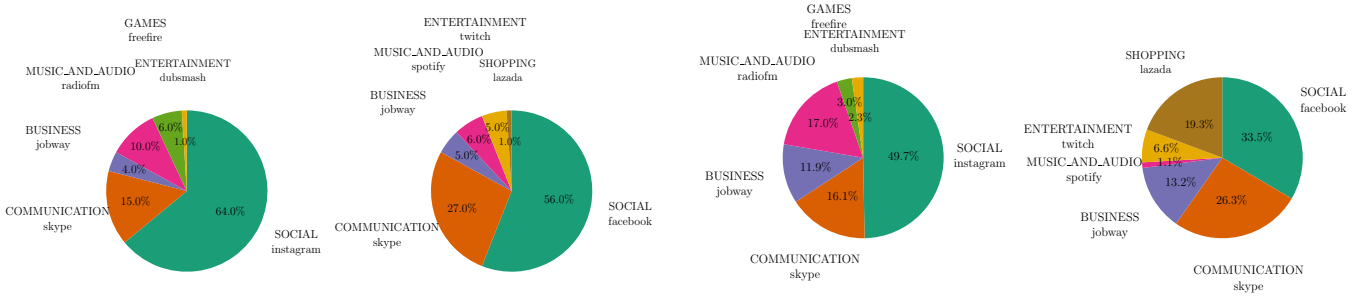
TABLE V: Apps and Weights per Category for User A

2) Synthetic User Trace Generation and Traffic Assignment: We generated traces for 1,000 synthetic users. First, for each user we defined a *user app usage* vector that includes the app categories, selected apps, and the corresponding percentage of usage for each app, as described in the previous subsection. Fig. 6(a) shows an example of the *user app usage* vectors of User A and User B. The diagrams represent their respective categories, apps, and usage percentages derived from the power-law distribution. Using these *user app usage* vectors, we then assigned traffic traces from the corresponding apps to each user, drawing from the traffic available in the MAppGraph dataset.

The synthetic trace for each user consists of 100 minutes of observed traffic. Within this fixed time window, we calculate the traffic interval for each app based on its usage percentage. We then extract the corresponding interval from the available traffic traces in the MAppGraph dataset. Although the power-law sampling naturally produces varied segments, we further enhance variability by selecting different starting points for each extraction. This prevents identical traffic patterns across users. The resulting traffic trace for each user is labeled by app and stored in a CSV file. For example, as shown in Fig. 6(a), user A has a 100-minute trace comprising 68 minutes of Facebook traffic, 11 minutes of Messenger traffic, 6 minutes of Jobway traffic, 7 minutes of SoundCloud traffic, 4 minutes of Freefire traffic, and 4 minute of Nimo traffic.

C. User Profiling

After generating synthetic user traces, we apply the user profiling described in Section III. We profile the users based on the percentage of DNS traffic generated by each app within its respective time interval. This method builds a user profile



(a) Representation of the *user app usage* vector of User A and User B.

(b) Representation of the *user DNS profiling* vector of User A and User B.

Fig. 6: Comparison between *user app usage* vector and *user DNS profiling* vector of User A and User B.

by focusing on the specific DNS traffic produced by each app. Unlike previous approaches as [15] that derive profiles from generic network traffic, our method leverages traffic traces that are directly labeled with the app identity.

Each user trace is stored in a CSV file, we stored each timestamp and domain name of the DNS requests labeled per app. For each user, we calculate the percentage of DNS traffic contributed by each app during the observation period. These percentages are then mapped to their corresponding app categories. The final output for each user is a vector named *user DNS profiling*, equivalent to the vector *user app usage*, where each element represents the percentage of DNS traffic for a specific category.

In Fig. 6(b) shows the *user DNS profiling* for Users A and B, as illustrated in the previous example. Our approach effectively captures the user's behavior based on the DNS traffic generated by the apps. It is important to note that the observed DNS query percentage does not directly reflect actual app usage time; different apps generate DNS queries at different rates. For example, we can appreciate in Fig. 6 between both pie charts, one app might generate many queries in a short period as seen with User A in the category *entertainment*, while another app produces fewer queries over a longer time as it happens in the category *music and audio* in User B.

In addition, we analyze the variability of the user profiles obtained by our method. Fig.7 displays the average percentage of DNS traffic per app for each user. While we anticipated a high percentage of DNS traffic in the *social* category, our results also reveal that the *business*, *entertainment*, and *shopping* categories exhibit high percentages, even though the corresponding time intervals are much shorter. This observation suggests that apps in these categories have a higher rate of DNS queries compared to other categories.

VI. Evaluation

In this section, we analyze the extent to which our proposal can help users protect their privacy when using mobile apps in a real-world scenario. At the same time, we assess the impact of generating fake DNS queries on the quality of the DNS resolution service using the utility metric we define as the forgery rate ρ.

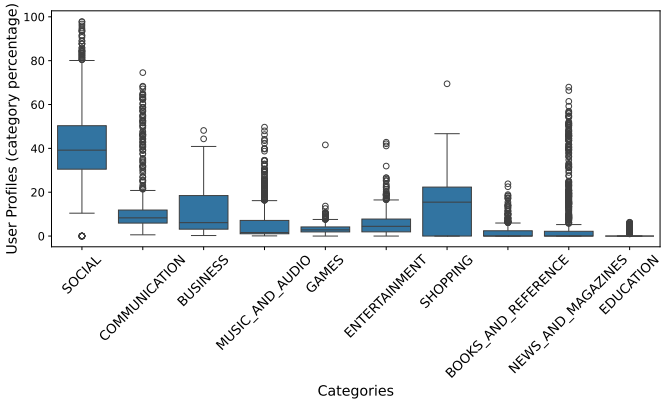
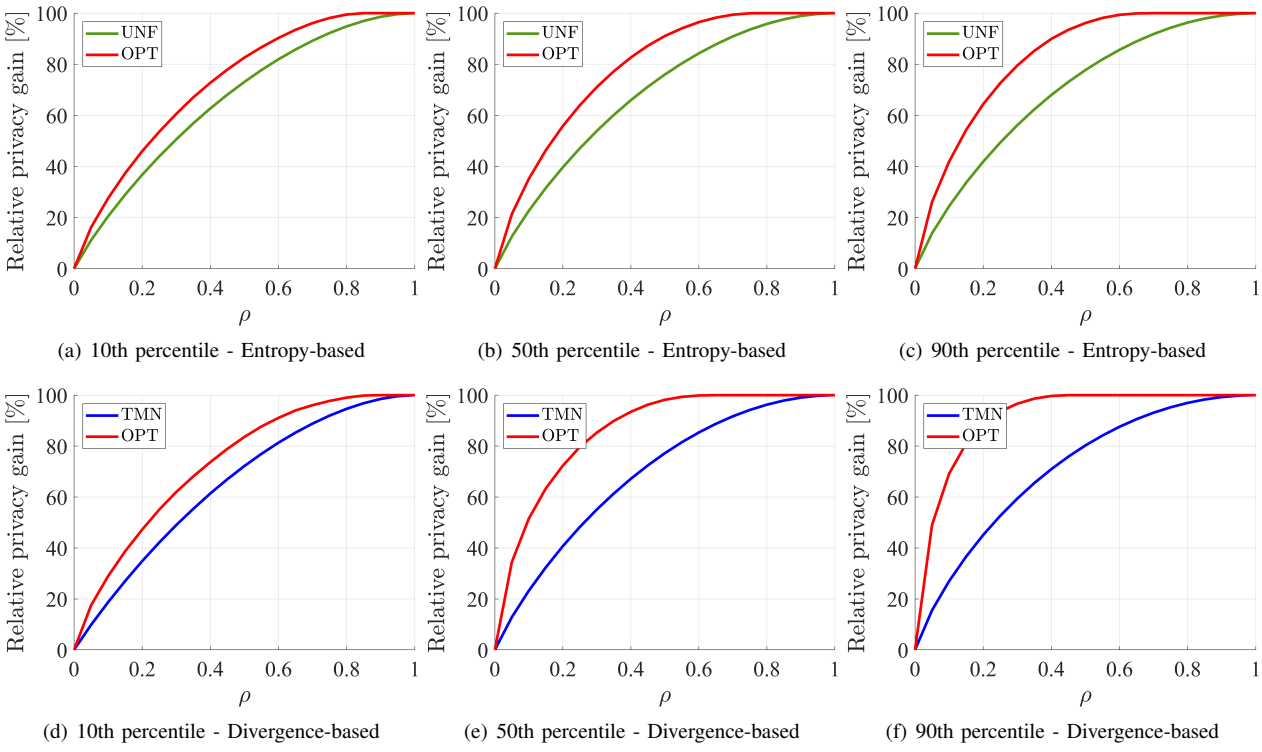


Fig. 7: Representation of the average percentage of DNS traffic per app for each user

A. Results

To evaluate how effectively our proposal enhances user privacy protection, we designed an experiment that measures the individual privacy improvement for each synthetic user in the dataset described in Section V. We compute the privacy risk function for each user based on the perturbation rate ρ for the three DNS query forgery strategies defined in Section IV-C, namely, Uniform, TrackMeNot-based, and Optimized. Ultimately, we compile all collected data and represent the privacy gain as a function of ρ musing the 10%, 50%, and 90% percentiles of the two best-performing strategies.

In Fig. 8, we present the relative privacy gain for the two best forgery strategies as a function of the perturbation rate ρ, conveniently partitioned into 21 values, across the two privacy metrics and three percentiles. Undoubtedly, in all cases, the optimized strategy outperforms the suboptimal ones. With the optimized mechanism, we achieve a 100% privacy improvement for 90% of users with false query rates above 60% when the risk metric is based on entropy (i.e., when no reference profile is available, and the uniform distribution is used). When the risk is based on KL divergence, the same improvement is obtained at perturbation rates above 40%. However, a 50% improvement for most users is only achievable with perturbation rates below 20%. These parameters guide us in understanding the traffic overhead users must assume to achieve reasonable privacy gains with an optimized



(a) 10th percentile - Entropy-based (b) 50th percentile - Entropy-based (c) 90th percentile - Entropy-based

(d) 10th percentile - Divergence-based (e) 50th percentile - Divergence-based (f) 90th percentile - Divergence-based

Fig. 8: The representation of the relative privacy gain for the two best forgery strategies as a function of the perturbation rate ρ, across the entropy and divergence and the 10%, 50%, and 90% percentiles.

mechanism. In contrast, suboptimal mechanisms require a full perturbation rate to attain any 100% improvement.

Furthermore, for the optimized DNS query forgery strategy, Fig. 9 displays the distribution of critical perturbation rate values for the 1,000 users in our dataset. The conclusions drawn from these results further reinforce those obtained from the relative privacy gain analysis.

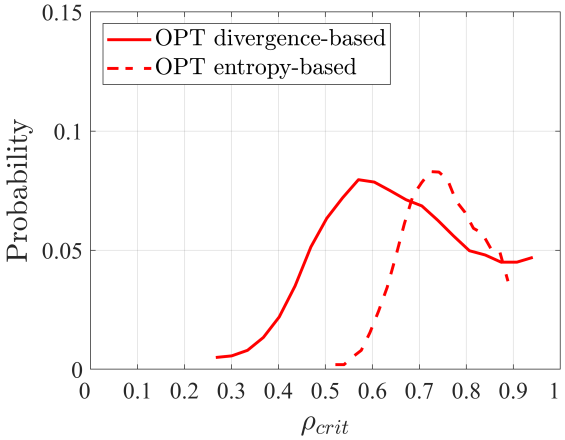


Fig. 9: The distribution of critical perturbation rate values for the 1,000 users in our dataset.

B. discussion

In this section, we discuss the results obtained from our DNS privacy model and compare our approach with state-of-the-art DNS privacy models mentioned in Section II-C.

Our DNS query forgery approach performs best with the Optimized strategy, achieving 100% privacy improvement for 90% of users at false query rates above 60% with entropy-based metrics and above 40% with KL divergence metrics. A 50% privacy improvement requires perturbation rates below 20%. These findings clarify the traffic overhead needed for effective privacy protection, while suboptimal mechanisms require 100% perturbation for similar results.

Despite the overhead our model introduces, it offers significant advantages over other DNS privacy approaches. Unlike PIR-based DNS privacy proposals like those presented in [4, 34], our model can adapt to a dynamically changing DNS environment. PIR-based approaches require static tables and server-side implementation, meaning both client and server must implement the privacy model for it to function properly.

When compared to collaborative DNS privacy models like NQA [5], our approach offers distinct advantages. While NQA can also adapt to DNS dynamism, it depends on creating a trust ecosystem with other users. In contrast, our model follows a zero-trust approach where any third party is considered a potential privacy threat. This provides user-side privacy without requiring trust in external entities.

More recent developments like ODNS [6] or ODoH [35] require implementation of the privacy model both at the user side (stub resolver) and at the DNS server side (dedicated ODNS resolver). They also introduce computational overhead for encrypting each query and forwarding it to a dedicated ODNS server for decryption before the DNS server can respond to the request.

Our model, while adding false queries to the traffic, offers several key benefits: it does not interfere with app functionality, does not require third parties to guarantee user security (following a zero-trust model), and provides user-side privacy. The trade-off is a controlled increase in network traffic, which our results show can be optimized to balance privacy gains and performance impact.

VII. Practical Adaptation of Query Forgery for Mobile Apps

We consider our proposal to be feasible for real-world implementation. With this premise in mind, we dedicate this section to establishing the foundations for the eventual deployment of a system that enables mobile app users to protect their privacy by perturbing the DNS traffic generated during their online activity. To achieve this, we adopt a high-level modular scheme, in which different modules perform specific functions within the system and interact with each other, as well as externally, to achieve the defined objective.

In practical terms, we envision a mobile app or a similar tool installed on the user's device, functioning as a decision-support system. That is, the app generally operates in the background and, upon detecting a privacy threat or compromise, alerts the user and presents possible countermeasures, allowing them to choose the rate of false DNS queries or the perturbation strategy itself. It is important to recall the principle underlying DPT—*hard privacy*—where the user is responsible for their own privacy, without relying on potentially untrustworthy third parties.

Before detailing the main functional components of our design, we must specify how a user's profile could be obtained locally in an app implementing our technique. To this end, we base our approach on three assumptions regarding the user profile.

First, as a common knowledge hypothesis, we assume that both entities—the mobile app and potential privacy attackers—operate over an identical set of interest categories. Consequently, based on their respective categorization algorithms, they derive the same user profile. This assumption holds as long as these categories belong to standardized sets available to both parties.

Second, in terms of profile initialization, we assume that in order to determine whether to add traffic to a specific category, our approach requires an initial user profile. One possible way to address this is by establishing a training phase prior to deployment.

Additionally, we assume the concept of a long-term profile, meaning that the user's profile does not change frequently, in line with [47]. The profile stabilizes after the initialization phase, once the user has shared a significant number of elements. However, we acknowledge that, in practice, user interests may vary significantly over time. Therefore, our implementation should account for this dynamic aspect.

Fig. 10 illustrates a modular architecture for a hypothetical implementation of our methodology as a DNS-query forger. It consists of a series of modules that interact locally and/or with the system, each performing a specific function based on the parameters it receives. From a general perspective, the figure depicts a user interacting with a single, straightforward DNS server. This server provides the user with relevant information, specifically IP addresses for resolving queried domains, generating a communication flow composed of DNS traces that third parties could exploit in undesirable ways, compromising user privacy. The following section provides a functional description of the five modules that comprise the architecture.

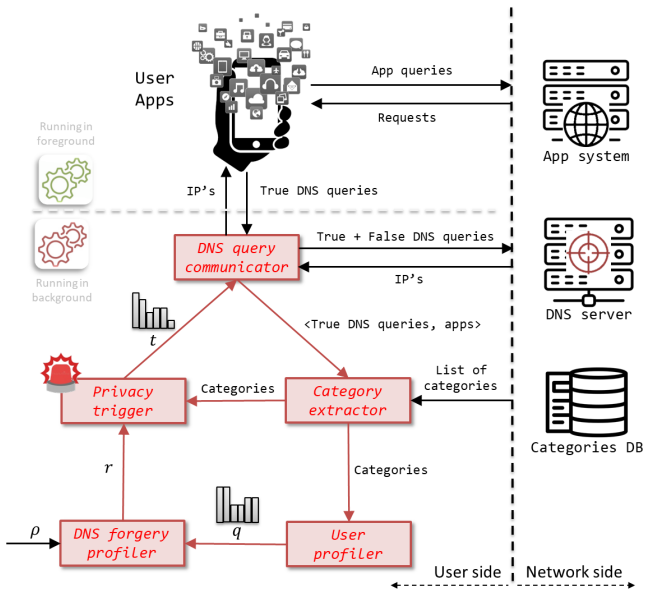


Fig. 10: Software architecture for an implementation of our privacy proposal in the mobile app scenario.

The first module, and the only one interacting externally in our false DNS query app, is the `DNS Query Communicator`. This module is responsible for sending all DNS queries it receives, whether genuine—originating from the user's apps—or false, as dictated by our app's results. Similarly, it receives the IP addresses resolved by the DNS server and returns them to the corresponding apps, provided they originate from a genuine query. Internally, the communicator forwards genuine DNS queries, along with the originating app, to the category extraction module for further processing.

The `Categories Extractor` module plays a crucial role in user profiling. Based on DNS queries and their originating apps, this extractor processes the information primarily by mapping apps to their corresponding categories. By continuously updating its data, which can be modeled as a connection to an external database specializing in this type of information, this module determines the interest category associated with each DNS query—an essential step for constructing the user profile. The extracted category is then transmitted to both the user profiling module and the privacy trigger module.

The `User Profiler` module is responsible for generating and/or updating the user's real profile, denoted as q, based on the categories received from the category extractor module. We assume that the discrete histogram of relative frequencies constructed by this module stabilizes after a certain period. Consequently, profile initialization is a key aspect of our

implementation, and we propose several initialization methods. For example, following [48], the profile can be initialized as $q = (0, \ldots, 0), q \in \mathbb{R}^n$. Another alternative, based on the principle of maximum entropy, involves initializing the real profile as the uniform profile, $q = u$. A further option is to use a self-declared profile provided by the user, which, while not necessarily matching the profile inferred from their online activity, would eventually be replaced after the initial phase. Ultimately, the user's real profile q is transferred to the DNS query forgery strategy module.

The core of our app is the `Strategy Generator` module, which ensures user privacy. This module implements various DNS query forgery strategies that we consider appropriate, such as those detailed in Section IV-C. It generates corresponding distributions of false DNS queries, denoted as r, based on the perturbation percentage ρ defined by the user. The output of this module is then passed to the privacy trigger module.

Finally, the `Privacy Trigger` module is responsible for alerting the user of potential privacy violations based on the distribution of false queries r. With probability r_i, an alert is issued regarding category i, allowing the user to decide whether to perturb their profile. The outcome of these actions is then transmitted to the communication module, which processes them accordingly, mixing false and genuine queries in the defined proportion to ultimately externalize the apparent profile t.

VIII. Conclusions and Future Work

We have proposed DNS query forgery, a data perturbation strategy that minimizes personal information exposure in mobile app DNS traffic while following 'hard privacy' principles. Our approach protects users from profiling by DNS resolvers without requiring trust in third parties.

Our evaluation demonstrates that query forgery effectively reduces profiling accuracy with minimal performance impact. We have quantified the trade-off between network overhead and privacy gains, showing that with optimal parameters, users can achieve significant privacy improvements at reasonable costs. The optimized query forgery strategy, in particular, delivers the best balance of privacy protection and efficiency.

We validated our DNS privacy model using a novel synthetic dataset of 1,000 users created from real mobile app traffic. This methodological innovation enables controlled experimentation on user profiling that would be difficult to achieve with real user data due to ethical and privacy constraints. Additionally, we proposed a modular software architecture that illustrates the feasibility of implementing our approach in real-world applications.

In future work, we consider exploring other user profiling techniques and delving deeper into the context of DNS traffic with targeted attacks on user privacy and new privacy-enhancing strategies.

Acknowledgment

This work was supported by Grant COMPROMISE (PID2020-113795RB-C32 and PID2020-113795RB-C31) funded by MICIU/AEI/10.13039/501100011033, Grant QURSA (TED2021-130369B-C32) funded by MICIU/AEI/10.13039/501100011033 and European Union NextGenerationEU/PRTR, Grant MOBILYTICS (TED2021-129782B-I00) funded by MICIU/AEI/10.13039/501100011033 and European Union NextGenerationEU/PRTR, Grant SISCOM (2021 SGR 01413) funded by Generalitat de Catalunya through Agència de Gestió d'Ajuts Universitaris i de Recerca (AGAUR), Grant DISCOVERY (PID2023-148716OB-C33 and PID2023-148716OB-C32) funded by MICIU/AEI/10.13039/501100011033 and FEDER, UE, and from the I-Shaper Strategic Project (C114/23), due to the collaboration agreement signed between the Instituto Nacional de Ciberseguridad (INCIBE) and the Universidad Carlos III de Madrid, this initiative is being carried out within the framework of the Recovery, Transformation and Resilience Plan funds, funded by the European Union (Next Generation).

References

[1] Ericsson. Ericsson Mobility Report, November 2024. Technical report, Ericsson, November 2024.

[2] Minzhao Lyu, Hassan Habibi Gharakheili, and Vijay Sivaraman. A Survey on DNS Encryption: Current Development, Malware Misuse, and Inference Techniques. *ACM Comput. Surv.*, 55(8), December 2022.

[3] Celeste Campo, Carlos Garcia-Rubio, Andrea Jimenez-Berenguel, Marta Moure-Garrido, Florina Almenares, and Daniel Díaz-Sanchez. Inferring mobile applications usage from DNS traffic. *Ad Hoc Networks*, 163:103601, 2024.

[4] Radhakrishna Bhat and N. R. Sunitha. *A Novel Private Information Retrieval Technique for Private DNS Resolution*, pages 163–171. Springer Singapore, Singapore, 2019.

[5] Oscar Arana, Hector Benítez-Pérez, Javier Gomez, and Miguel Lopez-Guerrero. Never Query Alone: A distributed strategy to protect Internet users from DNS fingerprinting attacks. *Computer Networks*, 199:108445, 2021.

[6] Paul Schmitt, Anne Edmundson, Allison Mankin, and Nick Feamster. Oblivious DNS: practical privacy for DNS queries: published in PoPETS 2019. In *Proceedings of the Applied Networking Research Workshop*, page 17–19. ACM, July 2019.

[7] Sudheesh Singanamalla, Suphanat Chunhapanya, Marek Vavrusa, Tanya Verma, Peter Wu, Marwan Fayed, Kurtis Heimerl, Nick Sullivan, and Christopher A. Wood. Oblivious DNS over HTTPS (odoh): A practical privacy enhancement to DNS. *CoRR*, abs/2011.10121, 2020.

[8] César Gil, Javier Parra-Arnau, and Jordi Forné. Privacy protection against user profiling through optimal data generalization. *Computers & Security*, 148:104178, 2025.

[9] Daniel C. Howe and Helen Nissenbaum. Lessons from the Identity Trail: Privacy, Anonymity and Identity in a Networked Society. NY: Oxford Univ. Press, 2009,

ch. TrackMeNot: Resisting surveillance in Web search. *http://mrl.nyu.edu/ dhowe/trackmenot*, pages 417–436, 2009.

[10] David Rebollo-Monedero and Jordi Forné. Optimized query forgery for private information retrieval. *IEEE Transactions on Information Theory*, 56(9):4631–4642, 2010.

[11] Roberto Gonzalez, Claudio Soriente, and Nikolaos Laoutaris. User Profiling in the Time of HTTPS. In *Proceedings of the 2016 Internet Measurement Conference*, IMC '16, page 373–379, New York, NY, USA, 2016. Association for Computing Machinery.

[12] Roberto Gonzalez, Claudio Soriente, Juan Miguel Carrascosa, Alberto Garcia-Duran, Costas Iordanou, and Mathias Niepert. User profiling by network observers. In *Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies*, CoNEXT '21, page 212–222, New York, NY, USA, 2021. Association for Computing Machinery.

[13] Souneil Park, Aleksandar Matic, Kamini Garg, and Nuria Oliver. When Simpler Data Does Not Imply Less Information: A Study of User Profiling Scenarios With Constrained View of Mobile HTTP(S) Traffic. *ACM Trans. Web*, 12(2), January 2018.

[14] Yang Gao, Jun Tao, Li Zeng, Xiaoming Fang, Qian Fang, and Xiaoyan Li. User Profiling with Campus Wi-Fi Access Trace and Network Traffic. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pages 922–927, July 2019.

[15] Faisal Shaman, Bogdan Ghita, Nathan Clarke, and Abdulrahman Alruban. User Profiling Based on Application-Level Using Network Metadata. In *2019 7th International Symposium on Digital Forensics and Security (ISDFS)*, pages 1–8, June 2019.

[16] Minzhao Lyu, Hassan Habibi Gharakheili, Craig Russell, and Vijay Sivaraman. Enterprise DNS Asset Mapping and Cyber-Health Tracking via Passive Traffic Analysis. *IEEE Transactions on Network and Service Management*, 20(3):3699–3716, Sep. 2023.

[17] Gaseb Alotibi, Nathan Clarke, Fudong Li, and Steven Furnell. User profiling from network traffic via novel application-level interactions. In *2016 11th International Conference for Internet Technology and Secured Transactions (ICITST)*, pages 279–285, Dec 2016.

[18] Tong Li, Yong Li, Mohammad Ashraful Hoque, Tong Xia, Sasu Tarkoma, and Pan Hui. To what extent we repeat ourselves? discovering daily activity patterns across mobile app usage. *IEEE Transactions on Mobile Computing*, 21(4):1492–1507, April 2022.

[19] Javier Parra-Arnau, David Rebollo-Monedero, and Jordi Forné. Privacy-enhancing technologies and metrics in personalized information systems. In *Advanced Research in Data Privacy*, pages 423–442. Springer, 2015.

[20] Yuval Elovici, Bracha Shapira, and Adlai Maschiach. A new privacy model for hiding group interests while accessing the web. In *Proceedings of the 2002 ACM workshop on Privacy in the Electronic Society*, pages 63–70, 2002.

[21] Yuval Elovici, Bracha Shapira, and Adlai Maschiach. A new privacy model for web surfing. In *International Workshop on Next Generation Information Technologies and Systems*, pages 45–57. Springer, 2002.

[22] Yuval Elovici, Chanan Glezer, and Bracha Shapira. Enhancing customer privacy while searching for products and services on the World Wide Web. *Internet Research*, 2005.

[23] Yuval Elovici, Bracha Shapira, and Adlay Meshiach. Cluster-analysis attack against a PRivAte Web solution (PRAW). *Online Information Review*, 2006.

[24] Shaozhi Ye, Felix Wu, Raju Pandey, and Hao Chen. Noise injection for search privacy protection. In *2009 International Conference on Computational Science and Engineering*, volume 3, pages 1–8. IEEE, 2009.

[25] Josep Domingo-Ferrer, Agusti Solanas, and Jordi Castellà-Roca. h(k)-private information retrieval from privacy-uncooperative queryable databases. *Online Information Review*, 2009.

[26] Silvia Puglisi, Javier Parra-Arnau, Jordi Forné, and David Rebollo-Monedero. On content-based recommendation and user privacy in social-tagging systems. *Computer Standards & Interfaces*, 41:17–27, 2015.

[27] Javier Parra-Arnau, David Rebollo-Monedero, and Jordi Forné. A privacy-preserving architecture for the semantic web based on tag suppression. In *International Conference on Trust, Privacy and Security in Digital Business*, pages 58–68. Springer, 2010.

[28] Javier Parra-Arnau, David Rebollo-Monedero, Jordi Forné, Jose L Muñoz, and Oscar Esparza. Optimal tag suppression for privacy protection in the semantic Web. *Data & Knowledge Engineering*, 81:46–66, 2012.

[29] Javier Parra-Arnau, Andrea Perego, Elena Ferrari, Jordi Forne, and David Rebollo-Monedero. Privacy-preserving enhanced collaborative tagging. *IEEE Transactions on Knowledge and Data Engineering*, 26(1):180–193, 2012.

[30] Javier Parra-Arnau, David Rebollo-Monedero, and Jordi Forné. A privacy-protecting architecture for collaborative filtering via forgery and suppression of ratings. In *Data Privacy Management and Autonomous Spontaneus Security*, pages 42–57. Springer, 2011.

[31] Javier Parra-Arnau, David Rebollo-Monedero, and Jordi Forné. Optimal forgery and suppression of ratings for privacy enhancement in recommendation systems. *Entropy*, 16(3):1586–1631, 2014.

[32] Ero Balsa, Carmela Troncoso, and Claudia Diaz. OB-PWS: Obfuscation-based private web search. In *2012 IEEE Symposium on Security and Privacy*, pages 491–505. IEEE, 2012.

[33] Richard Chow and Philippe Golle. Faking contextual data for fun, profit, and privacy. In *Proceedings of the 8th ACM workshop on Privacy in the electronic society*, pages 105–108, 2009.

[34] Mingxun Zhou, Andrew Park, Wenting Zheng, and Elaine Shi. Piano: Extremely Simple, Single-Server PIR with Sublinear Server Computation. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 4296–4314, May 2024.

[35] Eric Kinnear, Patrick McManus, Tommy Pauly, Tanya Verma, and Christopher A. Wood. Oblivious DNS over HTTPS. RFC 9230, June 2022.

[36] Yabo Xu, Ke Wang, Benyu Zhang, and Zheng Chen. Privacy-enhancing personalized web search. In *Proceedings of the 16th international conference on World Wide Web*, pages 591–600, 2007.

[37] Vincent Toubiana, Arvind Narayanan, Dan Boneh, Helen Nissenbaum, and Solon Barocas. Adnostic: Privacy preserving targeted advertising. In *Proceedings Network and Distributed System Symposium*, 2010.

[38] Matthew Fredrikson and Benjamin Livshits. RePriv: Re-envisioning in-browser privacy. In *Proc. IEEE Symp. Security, Privacy (SP)(May 2011)*, 2010.

[39] Javier Parra-Arnau, Jagdish Prasad Achara, and Claude Castelluccia. Myadchoices: Bringing transparency and control to online advertising. *ACM Transactions on the Web (TWEB)*, 11(1):1–47, 2017.

[40] Mireille Hildebrandt, James Backhouse, Vasiliki Andronikou, Emmanuel Benoist, Ana Canhoto, Claudia Diaz, Mark Gasson, Zeno Geradts, Martin Meints, Thierry Nabeth, et al. Descriptive analysis and inventory of profiling practices—deliverable 7.2. *Future Identity Inform. Soc.(FIDIS), Tech. Rep*, 2005.

[41] Mireille Hildebrandt and Serge Gutwirth. *Profiling the European citizen*. Springer, 2008.

[42] Javier Parra-Arnau, David Rebollo-Monedero, and Jordi Forné. Measuring the privacy of user profiles in personalized information systems. *Future Generation Computer Systems*, 33:53–63, 2014.

[43] Jingjing Ren, Daniel J. Dubois, and David Choffnes. An international view of privacy risks for mobile apps. Online, 2019.

[44] Thai-Dien Pham, Thien-Lac Ho, Tram Truong-Huu, Tien-Dung Cao, and Hong-Linh Truong. MAppGraph: Mobile-app classification on encrypted network traffic using deep graph convolution neural networks. In *Proceedings of the 37th Annual Computer Security Applications Conference*, ACSAC '21, page 1025–1038, New York, NY, USA, 2021. Association for Computing Machinery.

[45] Dimitri Mankowski, Thom Wiggers, and Veelasha Moonsamy. TLS → Post-Quantum TLS: Inspecting the TLS landscape for PQC adoption on Android. In *2023 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 526–538, July 2023.

[46] Teun Siebers, Ine Beyens, and Patti Valkenburg. The effects of fragmented and sticky smartphone use on distraction and task delay. *Mobile Media & Communication*, 12, 08 2023.

[47] Susan Gauch, Mirco Speretta, Aravind Chandramouli, and Alessandro Micarelli. User profiles for personalized information access. *The adaptive web*, pages 54–89, 2007.

[48] Alexandre Viejo, David Sánchez, and Jordi Castella-Roca. Using profiling techniques to protect the user's privacy in twitter. In *International Conference on Modeling Decisions for Artificial Intelligence*, pages 161–172. Springer, 2012.

Andrea Jimenez-Berenguel is a PhD Student at the Department of Telematic Engineering of the Universidad Carlos III de Madrid. Her research journey is focused on Android traffic analysis and user privacy. She received her bachelor degree in Telematic Engineering in 2022 and her MS degree in Telecommunications Engineering in 2024, both from the Universidad Carlos III de Madrid.

César Gil received the bachelor's degree in statistics from the Universitat Politècnica de Catalunya (UPC), Barcelona, in 1997. He joined the UPC team in the SLOEGAT Project in 1998. Also received the MS degree in decision systems engineering from the Universidad Rey Juan Carlos (URJC) in 2012 and the MS degree in computational and mathematical engineering from the Universitat Oberta de Catalunya (UOC) and Universitat Rovira i Virgili (URV) in 2018. He is currently a Ph.D. candidate at UPC, where he investigates the optimal trade-off between privacy and data utility in personalized information systems.

Carlos Garcia-Rubio received the Ph.D. degree from the Technical University of Madrid in 2000. He is an associate professor at the Department of Telematic Engineering of the University Carlos III of Madrid. His research focus is centered on mobile and wireless networked computing systems, and on the design and performance evaluation of communication protocols, mainly at the transport and application layers.

Jordi Forné received the M.S. and Ph.D. degrees in telecommunications engineering from Universitat Politècnica de Catalunya (UPC). Currently, he is a Full Professor with the Telecommunications Engineering School, Barcelona— ETSETB. He is also with the Smart Services for Information Systems and Communication Networks (SISCOM) Research Group, leading the research team on data privacy.

Celeste Campo received her Ph.D. degree from the University Carlos III of Madrid in 2004. She is an associate professor at the Department of Telematic Engineering of the University Carlos III of Madrid. Her research interests include design and performance evaluation of communication protocols for ad hoc networks, energy-aware communications, and middleware technologies for pervasive computing.