IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received March 23, 2020, accepted April 26, 2020, date of publication May 4, 2020, date of current version May 21, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2992012

A Real-Time Query Log Protection Method for Web Search Engines

DAVID PÀMIES-ESTREMS[1], (Student Member, IEEE), JORDI CASTELLÀ-ROCA[1], (Member, IEEE), AND JOAQUIN GARCIA-ALFARO[2], (Senior Member, IEEE)

[1]CRISES Research Group, Department of Computer Engineering and Mathematics, Universitat Rovira i Virgili, E-43007 Tarragona, Spain
[2]SAMOVAR, CNRS UMR 5157, Télécom SudParis—Institut Polytechnique de Paris, 91128 Palaiseau, France

Corresponding author: David Pàmies-Estrems (david.pamies@estudiants.urv.cat)

The authors graciously acknowledge support from CYBERCAT (Center for Cybersecurity Research of Catalonia, Tarragona, Catalonia), the UNESCO Chair in Data Privacy, the Cyber CNI Chair of the Institut Mines-Télécom. The following funding sources are gratefully acknowledged: European Commission (projects H2020-830892 "SPARTA", H2020-871042 "SoBigData++"), Government of Catalonia (grant 2017 SGR 705) and Spanish Government (project RTI2018-095094-B-C21 and network RED2018-102321-T). The views in this article are the authors' own and do not necessarily reflect the views of UNESCO or any of the funders.

ABSTRACT Web search engines (e.g., Google, Bing, Qwant, and DuckDuckGo) may process a myriad of search queries per second. According to Internet Live Stats, Google handles more than two hundred million queries per hour, i.e., about two trillion queries per year. For monetization purposes, the queries can be stored and complemented with additional data, referred to as query logs. Together, they can correlate valuable information to build accurate user profiles. Before releasing the query logs to third parties (e.g., for profit purposes), the personal information contained in the query logs must be properly protected by the web search engines. Current regulations establish strict control, and require from provable anonymization processing (e.g., in terms of statistical disclosure) of any personally identifiable information. In this paper, we tackle this challenge. We propose a real-time anonymization solution to protect streams of unstructured data at the server side. Our approach is based on the use of a probabilistic k-anonymity technique. It allows probabilistic processing of personally identifiable attributes contained in the query logs, with provable privacy properties. Our solution handles limitations of traditional k-anonymity approaches with respect to unstructured data and real-time processing. We present the implementation of our solution and report experimental evaluation results. The evaluation is conducted in terms of privacy, utility, and scalability achievement. Results validate the feasibility of our proposal.

INDEX TERMS Anonymization, data streams, privacy, query logs, web search engines.

I. INTRODUCTION

People use Web Search Engines (WSEs) for research, shopping, and entertainment [1]. Due to the large number of Websites (over 1.7 billion in 2020, according to Internet Live Stats and NetCraft [2]), it would be inconceivable to conduct such activities manually, without the help of a WSE. The usability of WSEs is, moreover, constantly improving. By simply querying the WSE with a few keywords, one may obtain several URLs with the desired contents. However, WSEs are not simply limited to return a list of URLs. When a search is conducted, a query (unstructured data) is processed and stored by the WSE. Together with the query, the WSE will store a timestamp, the URL selected by the user, and any other potential information collected about the user during the search. All this additional meta data, together with the query, is denoted hereinafter as query log. Streams of query logs are processed and analyzed by the WSEs, in order to build and improve users' profiles. This is expected to improve the service offered to users, as follows:

- **Personalization**. The query terms can have multiple meanings. Identifying the sense required by the user represents a challenge. Previous queries submitted by a user in the past can be used to contextualize and disambiguate terms in the future [3], [4]. This way, the WSE can prioritize relevant results (e.g., URLs) for the user and show them in the initial positions of the search results.
- **Usability**. The frequencies and selected results of the most submitted queries are used by WSEs to improve their ranking algorithms [5]. This can also be used to suggest alternative queries [6]. Such suggestions can show how to correct mistakes when typing, add

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

specificity to the initial query, or provide similar queries with more results.

Search data can also be exploited for other purposes because it reveals powerful insights about customer intent-to-purchase and other factors [7]. This new exploitation can be conducted by the WSE itself or by a third party, for the following purposes:

- **Marketing**. The results of an advertising campaign can be studied and improved by means of the query logs. For example, the user can be characterized by their query logs (gender, age, income, education, etc.) and afterwards verify if the advertisements have had an impact on the intended audience (interests and behavior) [8], [9]. Besides this, it is possible to extract market tendencies [10].
- **Research**. It may be centered on the study and test of new Information Retrieval (IR) algorithms [11], to learn about user's information needs and query formulation approaches [10]. It can also revolve around the use of language in queries [12], among other research topics [13]–[17].

The use of query logs can lead to some problems, related to user's privacy. Each query log can contain a user identifier, a text about what the user is looking for, the time when the search was conducted, and the URLs selected by the user. Any party with access to the query logs can obtain information about a user's behavior, habits, interest and more sensitive information, such as religion or sexual orientation. Even more, some query keywords may contain identifiers and quasi-identifiers [18], which may allow to link queries with real people. This is specially feasible, given current tendencies such as vanity search and egosurfing [19], in which people look for their own names over the Internet.

Query logs can be efficiently protected before being released to third parties. However, faulty or weak protection can lead to serious anonymity issues. The combination of modified data can disclose enough information to re-identify users [9], [20]. There is one well-known case, the AOL case, in which around thirty six million records related to query logs from AOL users were publicly released by AOL. Although the records were previously anonymized, it was later shown that it was still possible to identify some of the AOL users via traditional log correlation techniques [21]. As a result, sensitive information about AOL users was exposed publicly, without their express consent. The case ended up with an important damage to AOL users' privacy and to AOL's reputation, as well as several class action suits and complaints against AOL [22]–[24].

In this paper, we address the aforementioned problems. We present an anonymization technique to protect query logs at the server side. We assume WSEs seeking to monetize query logs by making them available to third parties, while respecting privacy regulations. A valid approach is to anonymize the logs prior releasing them to the third parties. Just concealing the user identifiers, or replacing them by random information, is not enough [25]. A provable anonymization method based on, e.g., Statistical Disclosure Control (SDC) techniques [18], must be conducted to guarantee bounded disclosure risks [26]. Traditional approaches can solve this situation by conducting a k-anonymity process at the server-side, before releasing the query logs. The release of data will satisfy the k-anonymity privacy property whenever user data contained within the query logs cannot be distinguished from at least $k-1$ other users — whose data also appear in the release [27].

An important issue of traditional k-anonymity approaches is the difficulty of using unstructured streams of data while satisfying the aforementioned privacy properties. This poses an additional problem to WSEs requiring, moreover, real-time processing. This issue is addressed in our proposal. Our solution relies on the use of probabilistic k-anonymity to bound disclosure risk of personally identifiable user attributes. Our solution can handle unstructured data, allowing real-time processing of query streams. It provides a probabilistic method to blend streams of queries with high similarity to those requiring protection, but coming from different users. More precisely, it ensures that individuals are not identified with a probability exceeding $\frac{1}{k}$, being k the total number of users sharing similar interests to the one meant to be protected (who is also counted in k). By using our solution, a WSE can keep the raw query logs and release the anonymized versions to third party organizations. The WSE can also decide to erase the raw query logs and keep only the anonymized versions. This way, and with low utility loss, the WSE will reduce the risk of information disclosure in case of intrusions.

Paper Organization: Section II presents our proposal. Section III provides architectural components and requirements. Section IV provides experimentation results validating our approach. Section V surveys related work. Section VI concludes the paper.

TABLE 1. Notations used in this paper.

R	:	Stream of query logs
r_j	:	Individual query log
u_i	:	User unique identifier
q_j	:	Individual query text
c_q	:	Full individual query classification
τ	:	Hierarchical ontology of categories
V	:	Set of vertices
v_x^h	:	Vertex at depth h depth and width x
E	:	Set of edges
e_f	:	Edge between two vertices
Q_x^h	:	Set of queries for vertex v_x^h
U_x^h	:	Set of users for vertex v_x^h
γ_x^h	:	Category for vertex v_x^h
$\tau_{\ell,k}^*$	:	τ with a depth ℓ and width k

II. OUR PROPOSAL

We present in this section our anonymization proposal. Table 1 introduces the notation used along this section. Next, we provide a formal definition of the expected data we aim to anonymize, the way how the data is structured, a formal analysis about the privacy properties of the proposal, and the algorithmic version of our anonymization process.

A. DATA STRUCTURES

We assume a stream of query logs, formed by m registers, where r_m corresponds to the last received query log:

$$R = \{r_0, \ldots, r_m\} \tag{1}$$

Each register is of the form:

$$r_j = \{u_i, q_j, c_g\} \tag{2}$$

where u_i is a unique identifier that represents the user who sent the query q_j to the WSE. Each query q_j is composed of a set of unstructured terms, which we previously provided to a categorizer (cf. Section III) to obtain the classification of the query, denoted as c_g. This classification c_g is represented as the path from a general category γ_s^1 to a more specific category γ_{s*}^h, with the form:

$$c_g = \{\gamma_s^1, \gamma_{s'}^2, \ldots, \gamma_{s*}^h\} \tag{3}$$

The path is created according to a hierarchical ontology structure by means of a tree structure τ, which is formed by a set of edges $e_f \in E$ and vertices $v_x^h \in V$, where h is the depth and x the width. Each vertex v_x^h of τ represents a category γ_x^h, and is related to other categories through the edges. The vertices or categories are more generic the closer they are to the roots $\{v_1^1 \ldots v_x^1\}$, and more specific the closer to the leaves. Thus, every query is classified by assigning it to one of the vertices of the tree. As mentioned, the classification is the path between the root and the vertex, and it is composed by all the γ categories of the nodes that are in the path.

$$\begin{aligned} \tau &= < V, E > \\ V &= \{v_1^1, \ldots, v_z^\ell\} \\ E &= \{e_1, \ldots, e_g\} \\ v_x^h &= \{U_x^h, Q_x^h, \gamma_x^h\} \\ e_f &= \{v_x^h, v_{x'}^{h+1}\} \end{aligned} \tag{4}$$

The maximum depth of the hierarchy τ is ℓ_{max}, defined as the distance or minimum path between the root and its farthest leaf. The number of terms or depth for each classification may be ℓ_{max} or lower, but we will use limited versions at depths up to ℓ, where ℓ goes from 1 to ℓ_{max}.

Each vertex v_x^h contains a set of users U_x^h, and a set of queries Q_x^h. The size of U_x^h will be k, but the size of Q_x^h may be larger. This is because U is defined using arity, but Q is defined without the need of using arity

$$max \mid U_x^h \mid = k \tag{5}$$

$$max \mid Q_x^h \mid \geq k \tag{6}$$

Therefore, we call $\tau_{\ell,k}^*$ the tree τ with a depth ℓ and a value of $|U| = k$.

B. RESTRICTIONS

To properly explain why U_x^h and Q_x^h may have different size, we introduce two additional restrictions that we impose to our proposal (cf. Restrictions 1 to 2).

Restriction 1: A given query associated to an anonymized log must not be assigned to the same user that issued the query on the unanonymized log.

Restriction 2: When creating an anonymized query log, user must be selected randomly between at least k different user values.

Restriction 1 ensures that outputs do not contain unanonymized pairs of user and query. Restriction 2 imposes probabilistic k-anonymity, setting at least k distinct values for users in each category when randomly creating an anonymized log.

C. ANONYMIZATION PROCESS

We define our anonymization process as the method that generates the probabilistic k-anonymous stream of logs R':

$$R' = \{r'_0, \ldots, r'_m\} \tag{7}$$

We assume that each record $r_j = \{u_i, q_j, c_g\}$ in R is assigned to the corresponding v_x^h using its categorization c_g. The record r_j is then separated in two parts: u_i which is assigned to U_x^h, and q_j which is assigned to Q_x^h. Records in R' are obtained by applying a random match between one element of U_x^h and one element of Q_x^h, once $\mid U_x^h \mid = k$:

$$r'_j = \{u'_i, q_j, c_g\} \tag{8}$$

where $q_j \in Q_x^h$ is matched with a $u'_i \in U_x^h \neq u_i$.

The Id function is assumed to be a correct identification function, which given r'_j responds with the original u_i. The function Re is a re-identification function used over the records in R', which given a r'_j responds with:

$$Re(r'_j) = u_i \in U_x^h, \quad u_j \neq u'_i \tag{9}$$

The goal of probabilistic k-anonymity is to limit the probability of performing the right re-identification to at most $\frac{1}{k}$ for all $u_i \in R$ and for all the values of $Re(r'_j)$:

$$P(Re(r'_j) = Id(r'_j)) \leq \frac{1}{k} \tag{10}$$

The stream of logs R' is said to satisfy probabilistic k-anonymity if, by knowing R' and the anonymization process, the probability to link any record $r'_j \in R'$ and its corresponding record $r_j \in R$ is, at most, $\frac{1}{k}$.

We show next that our proposal satisfies the property defined in Eq. (10). For each vertex v_x^h of τ, the random selection of an element (Restriction 2) guarantees that all outcomes are equally likely to be selected. Therefore, we can state maximum probability of re-identification of a r'_j over τ using:

$$P(Re(r'_j) = Id(r'_j)) \leq \max_{\forall x,h} \frac{|U_x^h \cap Id(r'_j)|}{|U_x^h|} \tag{11}$$

As U_x^h sets are defined using arity, we know that:

$$\forall x, h, Id(r_j') \rightarrow |U_x^h \cap Id(r_j')| \in 0, 1 \tag{12}$$

Someone could argue that Restriction 1 leads to a value of $k - 1$. However, since Restriction 2 establishes this value to k (Restriction 2 also assures that $|U_x^h| \geq k$), the upper bound of our proposal for $P(Re(r_j') = Id(r_j'))$ is strictly lower or equal to $\frac{1}{k}$, hence satisfying probabilistic k-anonymity. A more formal analysis about this result is provided next.

D. PRIVACY ANALYSIS

Given k (anonymity parameter) in $\mathbb{Z}^+$, a set of users $\mathcal{U}$ equal to $u_1, \ldots, u_n$ (such that $n \geq k$), a set of query logs $\mathcal{Q}$ equal to $(u_{i_j}, q_j)_{j=1}^{j}$ up to the processing iteration j, where $q_k \neq q_l \forall k, l \in [j], (k \neq l), u_{i_j} \in \mathcal{U}$. We also assume that users repeat (i.e. $u_{i_k} = u_{i_l}$).

We assume that given a query in R', the whole R' and k, an arbitrary PPT (Probabilistic Polynomial-Time) adversary $\mathcal{A}$ has at most $\frac{1}{k}$ chance of guessing the user the given query was attached to in R.

Now, with the notation above, and let $j_0 \in [j]$ define and experiment $Exp_{Re}(k, R)$, in which:

$$\begin{aligned} R' &\leftarrow Anon(k, R) \\ R^* &\leftarrow Re(k, R') \\ let\ b &= \begin{cases} 1, & \text{if } R = R^* \\ 0, & \text{otherwise} \end{cases} \\ &\qquad return\ b \end{aligned} \tag{13}$$

Theorem 1: Anon (cf. Eq.(13)) is probabilistic k-anonymous if, for every user set, for every query log R and every index $j_0 \in [j]$, any PPT adversary $\mathcal{A}$ has a bounded advantage up to $\frac{1}{k}$, i.e.,

$$Adv_{\mathcal{A}}(k, R) = P[Exp_{Re}(k, R)] \leq \frac{1}{k}$$

Proof: Let $R' = (u'_{i_j}, q'_j)_{j=1}^{j'}$ and j the iteration at which the first log entry is released by the anonymizer after (u, q) has been read by itself. Let $\mathcal{U}_j^{R'} = (u_{i_{j_1}}, \ldots, u_{i_J})$ be the users presents at R' at iteration j and $\mathcal{U}_j = (u_{i_1}, \ldots, u_{i_k})$ be the user set used internally in the anonymizer at iteration j (i.e., we know $u \in \mathcal{U}_j \in \mathcal{U}_j^{R'}$ and $\mathcal{U}_j$ has at least k different users).

$$\begin{aligned} &P(\mathcal{A}(R', q) = u) \\ &\quad = \sum_{u' \in \mathcal{U}} P(\mathcal{A}(R', q) = u | (u', q) \in R) \cdot P((u', q) \in R) \end{aligned}$$

If U_j and Q_j are the users and queries stored by the anonymizer after reading query q, where U_j has at least k different users, permute users from the queries of Q_j to R (all in U_j) has no effect on the anonymizer output, i.e.:

$$\begin{aligned} &P(\mathcal{A}(R', q) = u | R = Re(R')) \\ &\quad = \sum_{u \in \mathcal{U}_j} P(\mathcal{A}(R', q) = u | [R = Re(R')] \cap [U_j = U]) \\ &\qquad\qquad \cdot P(U_j = U) \end{aligned}$$

where U_j contains the users that can appear in step j, hence $u \in \mathcal{U}$. If U_j is fixed and $u \in U_j$, we can consider an R where the query q is paired with each of the users u' of U_j, and one of the queries q' whence the entries of u' from U_j are now paired with U.

If we have read j_u times the user u, $\forall i : j_i \geq 1$, we obtain that the ratio of R^*s, being $R^* = Re(R')$ and $U_j = U$, which contain the original pair (u, q) is:

$$\begin{aligned} P(\mathcal{A}(R', q) = u | R = Re(R')) &= \frac{(j_{u_2} + \ldots + j_{u_k} + j_{u-1})!}{(j_{u_2} + \ldots + j_{u_k} + j_u)!} \\ &= \frac{1}{(j_{u_2} + \ldots + j_{u_k} + j_u)} \leq \frac{1}{k} \end{aligned} \tag{14}$$

hence satisfying Theorem 1. □

Algorithm 1 Anonymization Process

```
Input  : R, $k$, $\ell$
Output: R'
foreach $r_j \in R$ do
    // Get current user, query text and full query categorization
    $u, q, c \leftarrow r_j$;
    // Truncate categorization to level $\ell$
    $cat \leftarrow \{\gamma_s^1, \ldots, \gamma_{s^*}^{\ell}\} \in c$;
    // Add current user to users' category set
    $users[cat] \leftarrow u$;
    // Add current query and full categorization to queries' category set
    $query[cat] \leftarrow \{q, c\}$;
    // While there are more than $k$ distinct users on the current category
    while $distinct(users[cat]) > k$ do
        // Select and remove a random query and categorization from the category's set
        pop random $\{q', c'\} \in query[cat]$;
        // Select and remove a random user from the category's set, distinct from the original user related to the query
        pop random $u' \in users[cat], u' \neq Id(q)$;
        // Send to the output the selected user, query and category
        send $u', q', c'$;
    end
end
```

E. ALGORITHMIC VERSION OF OUR PROPOSAL

An algorithmic version of our anonymization process is presented in Algorithm 1. Algorithm 2 presents the anonymization process counterpart, assumed to be implemented by a PPT adversary. Algorithm 1 receives three main inputs: desired k, ℓ values, and R as a stream of hierarchically categorized query logs.

Algorithm 2 De-Anonymization Process

```
Input  : R', $k$, $\ell$
Output: R*
foreach $r'_j \in R'$ do
    // Get current user, query text and full query categorization
    $u, q, c \leftarrow r'_j$;
    // Truncate categorization to level $\ell$
    $cat \leftarrow \{\gamma_s^1, \dots, \gamma_{s*}^{\ell}\} \in c$;
    // Add current user to users' category set
    $users[cat] \leftarrow u$;
    // Add current query and full categorization to queries' category set
    $query[cat] \leftarrow \{q, c\}$;
    // While there are more than $k$ distinct users on the current category
    while $distinct(users[cat]) > k$ do
        // Select and remove a query and categorization from the category's set, using one of the record linkage algorithms
        record_linkage $\{q', c'\} \in query[cat]$;
        // Select and remove a user from the category's set, using one of the record linkage algorithms
        record_linkage $u \in users[cat]$;
        // Send to the output the selected user, query and category
        send $u, q, c$;
    end
end
```

Even if all the sets are initialized empty, our proposed algorithm guarantees that U_x^h is of size k every time a new anonymized log is generated from that category. It also tries to keep the Q_x^h size as close as possible to the k value. As it always chooses between k different users and at least k different queries, probabilistic k-anonymity is guaranteed.

Q_x^h size may be bigger than k in the following situation: each time a new log enters a category and the log's user was already present on that category, user's arity is increased by one in U_x^h and the query is added to Q_x^h. Therefore, $|U_x^h|$ stays the same but $|Q_x^h|$ is increased by one. If Restriction 2 is not met, there is no anonymized log release (i.e., the size of Q_x^h can be bigger than k).

If Restriction 2 is met, and some user's arity is greater than one, then Algorithm 1 releases an additional log to reduce the size of Q and user arity, also enforcing Restriction 1. This extra step is only done once per log, therefore at most two logs are generated each time a new record enters the category, until all users' arities are equal to one.

System performance remains stable whenever variations of the set size is proportionally conducted [28]. Hence, we modify the size of each set in incremental unitary steps. This allows the most efficient memory usage. In addition to the k parameter, the depth of categories' tree must be specified

TABLE 2. Applying Algorithm 1 with $k = 2$ and $\ell = 1$.

STEP 1: first query arrives

INPUT: r_1 {u=Alice, q="piano", c=Arts/Music}
users[Arts] = Alice
query[Arts] = "piano"

STEP 2: second query goes to a new category

INPUT: r_2 {u=Bob, q="myspace", c=Computers/Internet}
users[Arts] = Alice
query[Arts] = "piano"
users[Computers] = Bob
query[Computers] = "myspace"

STEP 3, 4: still no distinct users $> k$ in any category

INPUT: r_3 {u=Alice, q="guitar", c=Arts/Music}
INPUT: r_4 {u=Charlie, q="violin", c=Arts/Music}
users[Arts] = Alice, Alice, Charlie
query[Arts] = "piano", "guitar", "violin"
users[Computers] = Bob
query[Computers] = "myspace"

STEP 5: distinct users $> k$ in "Arts", but k after the first output

INPUT: r_5 {u=Bob, q="flute", c=Arts/Music}
users[Arts] = Alice, Alice, Charlie, Bob
query[Arts] = "piano", "guitar", "violin", "flute"
users[Computers] = Bob
query[Computers] = "myspace"
 	
Category full: distinct(users[Arts]) = 3 $> k$
OUTPUT: r'_1 {u=Charlie, q="piano", c=Arts/Music}
users[Arts] = Alice, Alice, Bob
query[Arts] = "guitar", "violin", "flute"
users[Computers] = Bob
query[Computers] = "myspace"

STEP 6: new query, but no disctinct users $> k$ in any category

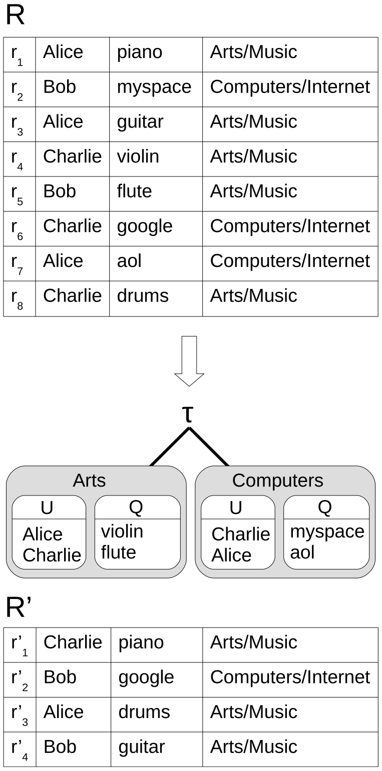
INPUT: r_6 {u=Charlie, q="google", c=Computers/Internet}
users[Arts] = Alice, Alice, Bob
query[Arts] = "guitar", "violin", "flute"
users[Computers] = Bob, Charlie
query[Computers] = "myspace", "google"

STEP 7: distinct users $> k$ in "Computers"

INPUT: r_7 {u=Alice, q="aol", c=Computers/Internet}
users[Arts] = Alice, Alice, Bob
query[Arts] = "guitar", "violin", "flute"
users[Computers] = Bob, Charlie, Alice
query[Computers] = "myspace", "google", "aol"
 	
Category full: distinct(users[Computers]) = 3 $> k$
OUTPUT: r'_2 {u=Bob, q="google", c=Computers/Internet}
users[Arts] = Alice, Alice, Bob
query[Arts] = "guitar", "violin", "flute"
users[Computers] = Charlie, Alice
query[Computers] = "myspace", "aol"

STEP 8: distinct users $> k$ in "Arts"; k after the second output

INPUT: r_8 {u=Charlie, q="drums", c=Arts/Music}
users[Arts] = Alice, Alice, Bob, Charlie
query[Arts] = "guitar", "violin", "flute", "drums"
users[Computers] = Charlie, Alice
query[Computers] = "myspace", "aol"
 	
Category full: distinct(users[Arts]) = 3 $> k$
OUTPUT: r'_3 {u=Alice, q="drums", c=Arts/Music}
Category full: distinct(users[Arts]) = 3 $> k$
OUTPUT: r'_4 {u=Bob, q="guitar", c=Arts/Music}
users[Arts] = Alice, Charlie
query[Arts] = "violin", "flute"
users[Computers] = Charlie, Alice
query[Computers] = "myspace", "aol"



R

r_1	Alice	piano	Arts/Music
r_2	Bob	myspace	Computers/Internet
r_3	Alice	guitar	Arts/Music
r_4	Charlie	violin	Arts/Music
r_5	Bob	flute	Arts/Music
r_6	Charlie	google	Computers/Internet
r_7	Alice	aol	Computers/Internet
r_8	Charlie	drums	Arts/Music

R'

r'_1	Charlie	piano	Arts/Music
r'_2	Bob	google	Computers/Internet
r'_3	Alice	drums	Arts/Music
r'_4	Bob	guitar	Arts/Music

FIGURE 1. Contents of *R*, τ and *R*′ in the example provided in Table 2.

using the ℓ parameter. Both k and ℓ remain fixed to the specified value throughout the entire execution.

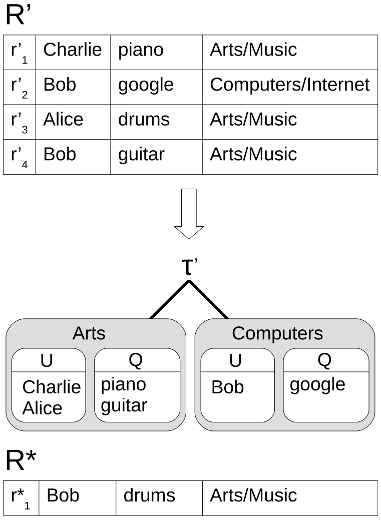
Table 2 provides a full example of our anonymization process, using $k = 2$ and $\ell = 1$ as main values. These values have been chosen to facilitate the understanding of the example, but they are inferior to desirable values in a real application of the algorithm (cf. Section IV). The example starts with an empty system, receiving a stream R of query logs classified in two distinct categories. Figure 1 depicts the used R, and the contents of τ and R' at the end of the aforementioned example. Figure 2 depicts the deanonymization counterpart, leading to faulty re-identification.

III. PRACTICAL IMPLEMENTATION

We present in this section a practical implementation of our proposal. We describe the architecture and requirements, before moving to the presentation of the experimental results.

A. INITIAL ARCHITECTURE

We aim at implementing an anonymization method that can be used by Web Search Engines (WSEs) to anonymize query logs in a streaming environment, and at server-side (cf. Figure 3). The input data of the anonymization algorithm is a continuous stream of categorized query logs. The outputs are a continuous stream of anonymized logs and a database of user profiles. To meet the goals of our proposal, we must ensure that those outputs meet a set of requirements detailed below.



R’

r'_1	Charlie	piano	Arts/Music
r'_2	Bob	google	Computers/Internet
r'_3	Alice	drums	Arts/Music
r'_4	Bob	guitar	Arts/Music

R*

r^*_1	Bob	drums	Arts/Music

FIGURE 2. Contents of τ' and *R when trying to deanonymize *R*′ from the example provided in Table 2.**

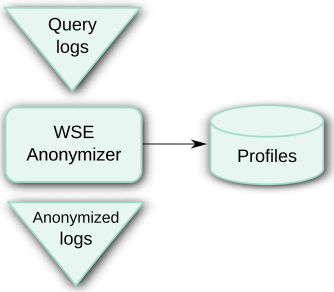


FIGURE 3. Our proposal defines a WSE query logs anonymization method in a streaming environment. The input of the algorithm is a stream of query logs. The outputs are a stream of anonymized logs and a database of user profiles.

B. FUNCTIONAL REQUIREMENTS

In addition to the restrictions and properties already defined in Section II, we report next some functional requirements for the practical implementation of our proposal.

1) SCALABILITY

It refers to the capability of a system to handle a growing amount of work, or its potential to be enlarged in order to accommodate that growth [29]. In our system, the objective is to achieve load scalability, defined as the ability to accommodate heavier or lighter loads. Those methods can be classified in two main categories [30]:

- **Horizontal Scalability** is related to the ability of a system to add more working nodes, such as a new computer. Hundreds of small computers may be configured in a cluster to obtain aggregate computing power. This approach demands an architecture that allows efficient management and maintenance of multiple nodes.
- **Vertical Scalability** is related to the ability of adding resources to a single node in a system, typically

involving the addition of CPUs or memory. Such approach could be interesting in a virtualized environment, as it could provide more resources according to the virtual node needs. This approach demands an architecture that allows efficient management of used processes and memory.

The two models have their own particular benefits and limitations. If necessary, our proposal should use all possible assets. In such a case, the design should be integrated into existing systems on a WSE architecture. Ideally, our system can take advantage of underused resources.

2) RESOURCE CONSUMPTION

In order to take advantage of underused resources on existing architectures, and minimize system deployment costs, we want a minimal resource consumption. If the designed system is able to use a limited amount of resources, all necessary data could be kept and processed in memory, obtaining better execution times.

3) SPEED

We need a fast processing speed to be able to process all received logs in real time. Otherwise, some kind of memory buffer will be necessary to keep incoming logs until processed. That buffer will increment our resource consumption. An additional requirement, in terms of processing speed, must be defined and only use small buffers at specific overload times. Nowadays, a WSE receives millions of user queries each hour. Therefore, our system should handle that load, to be able to integrate it in a existing WSE architecture.

4) EFFICIENCY

Beyond reduced resource consumption and fast processing time, we aim at assuring the algorithmic efficiency of the proposal. We consider that this requirement will be achieved if the algorithmic time complexity of our proposal is linear according to the inputs.

5) TRANSPARENCY

We want a straightforward integration of our approach into an existing architecture. Having a transparent system implies that no component of the existing WSE should be modified. For this purpose, our module is expected to be encapsulated within the WSE. It should also be able to interact to the existing interfaces of the WSE, without forcing any changes. It should also be able to generate anonymized logs, while complying with all the previous requirements.

6) MODULARITY

We want to have low coupling and high cohesion to achieve a fully transparent component. Modularity has the added benefit that modifications to the proposal could be implemented with minimal effort, as well as to carry out tests with different alternatives for the treatment of the data.

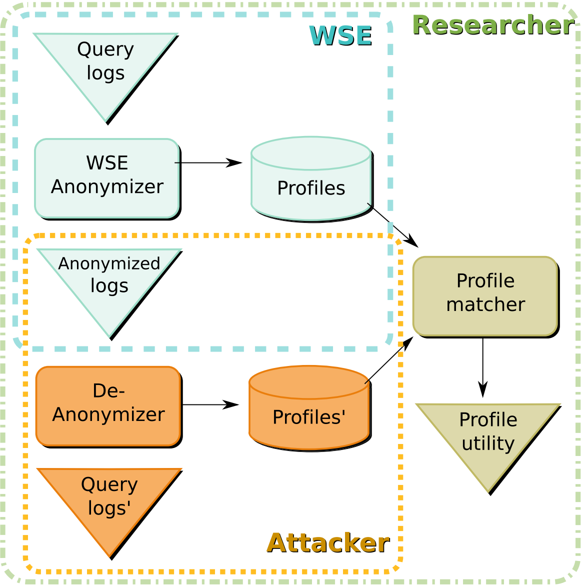


FIGURE 4. Full Architecture: *WSE Anonymizer* takes a stream of query logs and anonymizes them, also generating a database of user profiles. It implements Algorithm 1 (cf. Section II). *De-anonymizer* implements Algorithm 2 and simulates adversarial actions over the anonymized logs. It tries to recreate the original logs and user profiles. *Profile matcher*, responsible of benchmarking anonymization, de-anonymization and performance, also generates a profile utility metric.

C. EXPANDED ARCHITECTURE

The initial proposal depicted in Figure 3 is expanded with two additional parts: Attacker and Researcher. This allows a proper empirical evaluation, in addition to the analysis conducted in Section II. The proposed system is designed using a micro-service architecture pattern as presented in Figure 4. For the current study, all the defined systems are used. In a real WSE environment, only the parts marked as WSE should be deployed.

Within the expanded architecture, we find two main components: anonymizer and profiler. The anonymizer is a component implementing Algorithm 1. The profiler creates protected user profiles, using the categories of each log assigned to that user by the *anonymizer*. Those categories are added to a user profile database in real-time. Each profile on the database contains a frequency distribution of those categories queried by the user. They can be seen as user interests that could be released to third parties, for profit.

1) ACTORS

Three actors are defined in our current test architecture:

- **WSE** — has the responsibility of query logs anonymization and publication.
- **Attacker** — has access to the anonymized stream of logs, tries to recover the original relationship between the log and the user who made the original query.
- **Researcher** — can check all the data, but can not modify anything, to test the validity of the proposal.

2) PHASES

Our study is divided into three main phases:

- **Anonymization and profile creation —** this phase represents the normal execution of the system on the WSE environment. It takes the query logs generated and anonymizes them, also generating a database of user profiles.
- **De-anonymization —** it simulates attacks, trying to link as much of the anonymized logs with the user that originally made the query.
- **Analysis —** it conducts anonymization, de-anonymization and performance benchmarking, taking into account original and generated data, time and resource usage.

3) INTERACTIONS

In a real WSE environment, the WSE will anonymize the query logs and release the anonymized ones to its clients as the main interaction. In our tests, the attacker is acting as a normal client from the WSE point of view. The attacker process the anonymized output of the WSE and generates another log stream, trying to reconstruct original query logs. Only during the tests, secondary interactions occur between those actors and the researcher, who receives original, anonymized and de-anonymized query logs. Some further information about it is presented in the sequel.

IV. EXPERIMENTAL RESULTS

We report in this section a practical implementation of our approach, and report experimental tests and results, to validate our approach in terms of privacy, data utility and other functional requirements.

Experiments were conducted using a *Dell* notebook running *Ubuntu Linux* 16.04 LTS, with a 1.8 GHz *Intel Core*$^{\text{TM}}$i7-4500U CPU and 8GB of RAM. System hard disk was a *Seagate ST1000LM014*, whose performance profile is skewed strongly towards small file I/O, and a below average overall performance. All algorithms were implemented and executed in *Python* 2.7.12.

A. IMPLEMENTATION

Algorithm 1, described in Section II, has been implemented using the *Python* language. Input query logs used to test our system were downloaded from the public available AOL log repository, in form of plain text files. In order to respect our transparency functional requirement, we chose to make this file the main input of our system. However, other methods to feed logs to the system, such as a real time input via sockets, could be used. The same applies to system output and we also decided to store them in plain text files, preserving original logs' format. Additionally, a *No-SQL* database was used to store generated user profiles.

Because AOL's released files do not have any classification, they need to be categorized by an external categorizer before any of the proposed algorithms could be applied. We used a slightly modified version of the deterministic classifier proposed in previous work [28]. The use of a deterministic classifier guarantees that the same query will always provide the same unique category. In case a query triggers multiple categories, the classifier will always take the most probable one. Other families of classifiers can be adapted and integrated in our approach thanks to the proposed micro-service architecture. Classifier modifications allow us to obtain a query categorization organized in several hierarchical levels. Some queries contain letters or symbols without any meaning, and some contain no text at all. Our classifier was not able to resolve those logs, and they were left out of data used to test the proposal. However, some changes made to natural language processing algorithms on the *classifier* lead to categorize 98% of original logs, an improvement of over the 85% categorized in [28]. As it is out of the scope of the current proposal, implementation of the *classifier* will not be evaluated. Priority will be given to allow interoperability between our proposal and different *classifiers*. Usually, classification process needs more specific data, related to WSE environment or desired output categories. Thus, we leave freedom to each WSE to choose the strategy that best suits their needs.

We also validate the possible record linkage of the anonymized stream, implementing three different record linkage algorithms, and evaluate for each algorithm whose requirements are fulfilled. In addition, some other changes that have been made to the initial architecture described in Section III are discussed below.

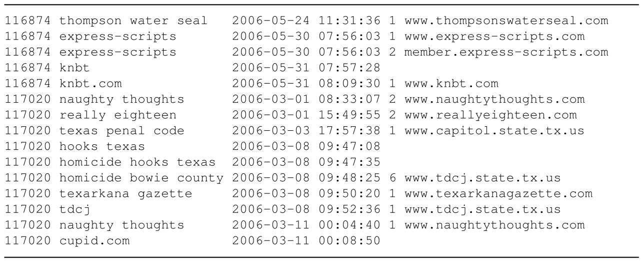
B. EVALUATION METHODOLOGY

The algorithmic solution proposed in Section II, and all the architectural components, requirements and implementation details defined in Sections III and IV-A, have been used to conduct an experimental evaluation and comparison to previous work in [28]. In particular, one version of the anonymizer, and three versions of the de-anonymizer are implemented and evaluated in terms of utility, privacy and functional requirements.

1) EXPERIMENTAL DATASETS

For our experiments, we use plain datasets (i.e., text files), containing query logs released by *AOL* [31]. The released *AOL* data contains up to thirty six million query logs. Such query logs correspond to a three-month period of real web search activity conducted by *AOL* users, and released by *AOL* for research purposes. Figure 5 provides a brief sample of the used logs.

The Classifier (cf. Section IV-A), adds to each log record an additional column with a hierarchical classification in form of a list with n elements. In our case, n was between one and 13, and each element of the list represents a subcategory of the previous element. This classification is generated independently of the anonymizer. Therefore, this list contains all the subcategories which the Classifier is able to generate for



```
116874 thompson water seal   2006-05-24 11:31:36 1 www.thompsonswaterseal.com
116874 express-scripts       2006-05-30 07:56:03 1 www.express-scripts.com
116874 express-scripts       2006-05-30 07:56:03 2 member.express-scripts.com
116874 knbt                  2006-05-31 07:57:28
116874 knbt.com              2006-05-31 08:09:30 1 www.knbt.com
117020 naughty thoughts      2006-03-01 08:33:07 2 www.naughtythoughts.com
117020 really eighteen       2006-03-01 15:49:55 2 www.reallyeighteen.com
117020 texas penal code      2006-03-03 17:57:38 1 www.capitol.state.tx.us
117020 hooks texas           2006-03-08 09:47:08
117020 homicide hooks texas  2006-03-08 09:47:35
117020 homicide bowie county 2006-03-08 09:48:25 6 www.tdcj.state.tx.us
117020 texarkana gazette     2006-03-08 09:50:20 1 www.texarkanagazette.com
117020 tdcj                  2006-03-08 09:52:36 1 www.tdcj.state.tx.us
117020 naughty thoughts      2006-03-11 00:04:40 1 www.naughtythoughts.com
117020 cupid.com             2006-03-11 00:08:50
```

FIGURE 5. **AOL log format. Each row represents a query log. Columns contain, from left to right: user identifier, query submitted, time submitted, result selected and result URL.**

a given query, regardless of the ℓ used by the anonymization process.

2) CONDUCTED TESTS

Proposed system could be configured using two parameters: k and ℓ, being k the desired number of different users on each category and ℓ the maximum depth of categories and subcategories used for each record. Several tests were conducted to determine its effects.

Anonymizer — to generate anonymized data, proposed *anonymizer* was executed on all available AOL logs multiple times, to cover different k and ℓ values. k has taken values between 3 and 200 to be able to compare obtained results with previous ones [28]. To do this, Algorithm 1 needs to be tested at least using $\ell = 1$. We decided to test all available ℓ values, that with our classification correspond to values between one and 13, but we found that from 11 onwards, differences were not significant: few logs have more than 11 categories of depth. Our privacy, functional and utility requirements are checked for every combination of k and ℓ.

Profiler — specific tests were conducted with the *profiler*, to determine the amount of data utility that could be lost with anonymized profiles creation respect to unanonymized profiles. For those tests, we used k values between three and 90 and ℓ values between one and 13.

De-anonymizer — a de-anonymization has been attempted against all anonymized data. All anonymized data was tested against three different *record-linkage* algorithms:

- **Record-linkage 1 —** This is the simplest record-linkage algorithm we tested. It tries to apply an inverse transformation to anonymized query logs by applying a similar algorithm to the one used in the anonymization process (cf. Algorithm 1 in Section II). In short, it tries to recreate original logs by randomly matching users and queries from the same category. Attacker also takes advantage of both restrictions 1, 2 to achieve higher levels of de-anonymization.
- **Record-linkage 2 —** It improves the performance over *Record-linkage 1*. Instead of randomly matching users and queries, it assigns the user that appears more times on a category to the selected query. Just like other algorithms, both restrictions are respected.
- **Record-linkage 3 —** It keeps track of how many times a user issued a query on each category, constantly updating a simplified user profile. When the algorithm needs to assign a user to a query, the user with more issued queries on that category will be chosen. If a user appears more than one time, the result will be multiplied by the number of appearances of that user, balancing the importance between current state of the system and historical values.

C. PRIVACY STUDY

Our privacy test compares original query logs data with the anonymized ones. Results for this base case show that none of the original pairs of user/query appear on the anonymized query log. Notwithstanding, that result did not guarantee full user privacy, since some attacks are possible over the output data flow, and some user logs may be re-identified. Three different record-linkage algorithms were applied to the anonymized query logs (cf. Algorithm 2). Resulting logs were compared to the original ones, counting the percentage of matching records.

Our de-anonymization algorithms proposal is based on Algorithm 2, that is similar to Algorithm 1 used in anonymization. It uses the stream of anonymized logs generated by the WSE as the main input. It also needs k and ℓ parameters (explained in Section II). The smaller the difference between k and ℓ values used in both algorithms, the better the results obtained from de-anonymization. In other words, the attacker will be able to re-identify the original data more easily.

The stream of anonymized logs is classified in the same way as the original one, since we assume that categorization is public and the attacker can use it. Therefore, the de-anonymization process uses the same categorization, which enables this algorithm to obtain the best de-anonymization rate when trying to recover the original logs.

The main difference with the *anonymizer* algorithm is the use of *record_linkage* function, different for each implementation of the *de-anonymizer* algorithms. The most complex de-anonymizers also use additional data structures to improve de-anonymization performance. Differences of each algorithm are fully explained in Section IV-B2.

For analysis purposes, we need to evaluate the amount of memory and time used in each algorithm execution, therefore, previous algorithms were modified to calculate those values. An additional algorithm must be defined to find the number of logs that are identical comparing two log streams.

Figure 6 shows percentage of matching records, executing the three algorithms with values of k between three and 200 and values of ℓ between one and 13. With $\ell = 1$, only one level of the tree structure was used, which results in a data structure equivalent to the one used in our former paper [28]. $\ell = 13$ is the maximum depth that our classifier was able to generate. Thus, there is no need to use higher ℓ values. We also picked out k values to be able to compare results between our current and former evaluation.

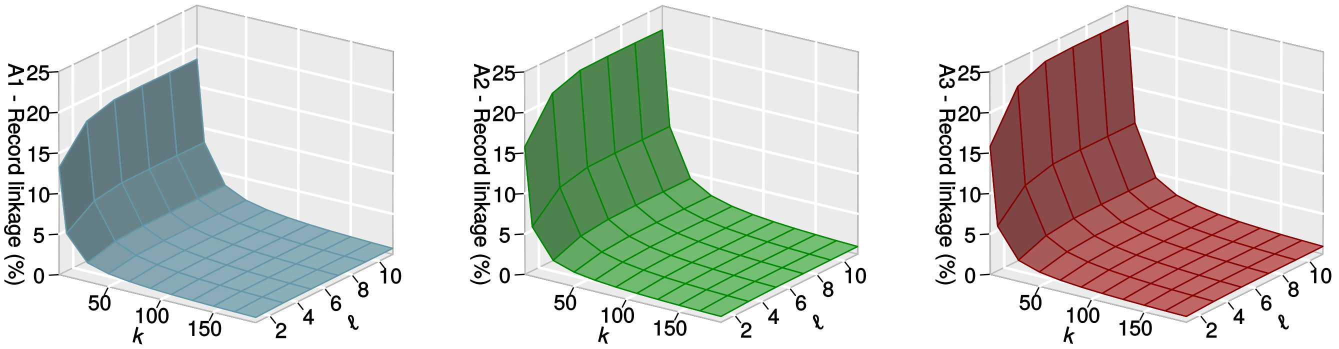


FIGURE 6. Record linkage (%). Percentage of matching records, executing the three de-anonymization algorithms with values of k between three and 200 and values of ℓ between one and 13).

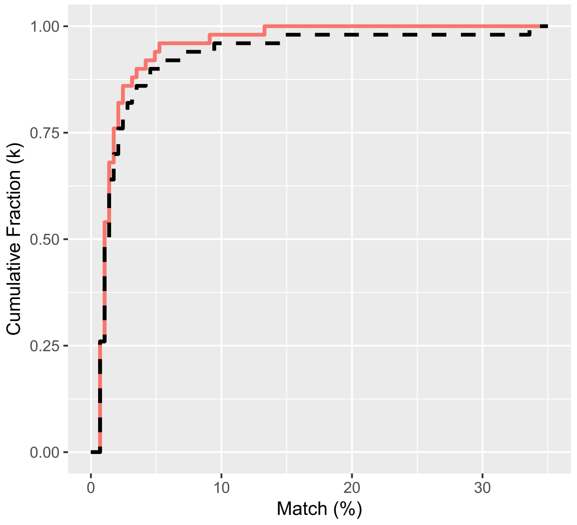


FIGURE 7. Comparison between cumulative fraction functions of the theoretical k-anonymity (dashed) and experimental results (solid). Used for the Kolmogorov-Smirnov test ($D = 0.08$, p-value $= 0.9977$).

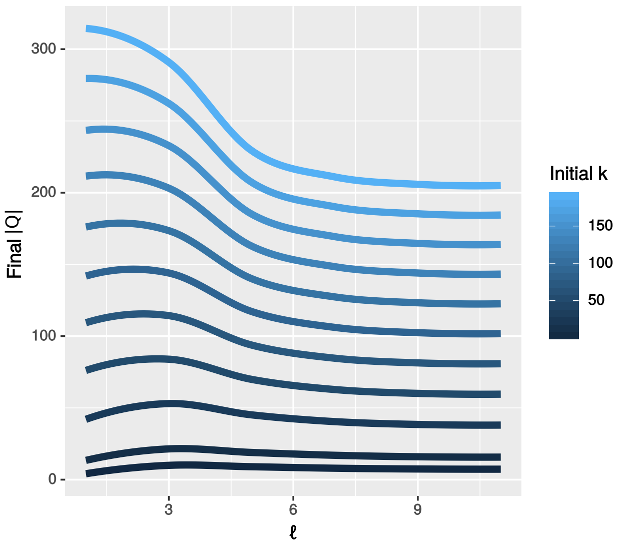


FIGURE 8. Final | Q |-value, as the mean size of queries' sets. For low ℓ values, final | Q | is higher due to more user coincidences on the same category. With higher ℓ values, final | Q | tend to match the specified k.

In all cases, results are under the theoretical maximum probability $\frac{1}{k}$ of being re-identified [32]. We ran the Kolmogorov-Smirnov goodness-of-fit statistical test [33], [34] to compare the k-anonymity probability with the experimental results, Figure 7. The maximum difference between the cumulative distributions, D, is 0.08 with a corresponding p-value of 0.9977. Therefore, the statistical test yields to acceptance of the null hypothesis that our results follow k-anonymity's probability of re-identification (at the 5% level of significance).

Each *record-linkage* version improves re-identification rate, being the third version the one that obtains better results overall. k value was highly correlated with privacy, because when the value of k increases, record linkage decreases. ℓ also affects privacy. With a higher number of levels (high ℓ value) users were matched with more specific queries, therefore, it was also more probable to obtain a correct re-identification of the original user. Here, we face a trade-off between privacy and data utility.

Results obtained this way, are close to the ones obtained in our previous article using the proposed algorithm without restriction, since now the effective size of the category sets are closer to the k value specified as a parameter. However, on average a better anonymization is obtained, since the size of Q must be temporarily increased to meet the restrictions 1 and 2.

Figure 8 shows mean final | Q | values, related to ℓ and initial k value. For low ℓ values, mean final | Q | values are higher because they have less categories results and more user coincidences on the same category. However, with small k and ℓ values, the high number of queries that passes through each category counter this effect. With higher ℓ values, final | Q | values tend to match up with specified k.

The highest record linkage is obtained with highest ℓ and lowest k values. Our best de-anonymizer algorithm was able to link 23.18% records to the original user. De-anonymization tests were conducted knowing exactly all algorithms, categories and variables used for anonymization. This ratio decreases quickly when initial k value is increased, obtaining a record linkage lower than 1% from k values greater than 90. In conclusion, desired record linkage level could be adjusted by modifying the k value, even offsetting the effect of ℓ variations on the record linkage.

D. UTILITY STUDY

We proceed to analyze the utility of the proposed anonymizer. This analysis has been focused on two different aspects:

- Percentage of logs that the system can generate as an output.
- Preservation of original user's interest in anonymized user's profiles.

First, we want to analyze the percentage of logs that can be generated by the system over the total number of logs that it gets. The proposed system uses sets, and each set must have at least k different users before being able to release an anonymized log. A possible drawback to this approach is that some sets do not reach k users and, therefore, the logs contained in this set do not end up leaving the system. As we can see in Figure 9, this effect exists and it is directly proportional to the depth of the category tree. This is consistent, since with more depth, more categories are created and the minimum of k users on these categories is reached more slowly. However, we see that as more queries enter the system, all categories become filled with queries and the percentage of log output increases, tending to a 100% rate for any depth of the tree.

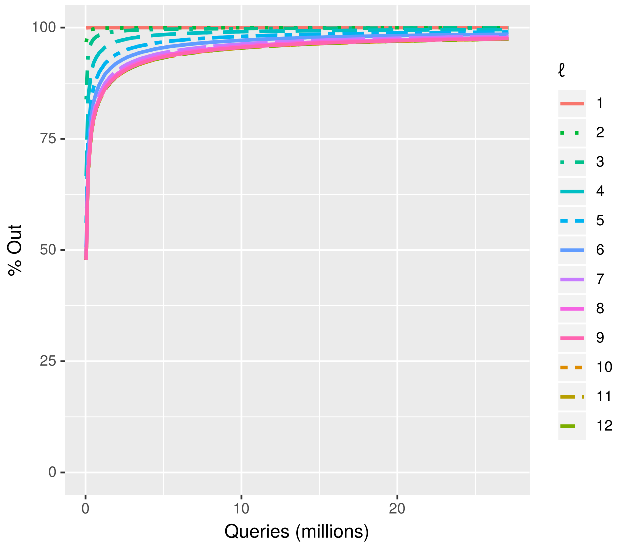


FIGURE 9. Output queries vs. total queries (%). Some sets take a while to fill. This effect is directly proportional to the depth of the category tree as more sets need to get k different users.

Secondly, to measure the preservation of original user's interest in anonymized user's profiles, we will measure the distance between them, using a metric known as Earth Mover's Distance (EMD) [35]. We calculate the distance between the categories of queries assigned to the original profile and the anonymized profile. As our classification of categories is stored in a tree graph, this distance is defined as the minimum length of the path that connects the categories assigned to the original and anonymized query. Once we have calculated the distance between individual queries, we add all the distances of that profile and, thus, we obtain the total distance between profiles.

Notice that if two queries are classified and anonymized with the same category, there is no distance between the two queries and there is no utility loss. This happens to all the queries when the depth of the tree is set to 13. However, other tree depths can lead to utility loss. For instance, in the example of Table 2, "piano" is classified as "Arts/Music" but the anonymizer is just using "Arts", since the value of ℓ is equal to 1. Queries classified as "Arts/Music" and "Arts/Painting" are mixed in "Arts" and assigned to different users. A third party could think that Alice is interested in "Painting", when she is just interested in "Music". i.e., there is a certain degree of utility loss. Since the third party still knows that Alice is interested in "Art", we can see the previous case as an example of partial utility loss. Therefore EMD represents the distance between the original user's interests, and the ones that are deducted from the anonymized queries.

In Figure 10, we can see the average value of the EMD distances, as well as the maximum theoretical distance between profiles using the chosen categorization. This theoretical maximum distance is constant, regardless of which ℓ and k values we use. The real distance we get is not affected by k, but is inversely proportional to ℓ. This means that the more levels we use in our anonymizer, the closer the anonymized queries get to their original category and we obtain a better data utility. In Figure 10, we can see the loss of utility expressed as a percentage. Using this metric, it can be seen that with $\ell = 1$, loss of utility is over 40% on average. With $\ell = 6$, the loss of utility is near to 0%, according to our definition of utility.

E. FUNCTIONAL STUDY

Next, we detail the accomplishment of proposed functional requirements.

1) MODULARITY

To allow a modular system, this has been designed as a set of micro-services. As our proposal uses micro-service architecture, it will be easier to modify and adapt when applied to different environments. In addition, this design helps each service to focus only on a specific process. By doing so, we achieve a system with low coupling and high cohesion. The anonymization service has been thoroughly explained. This service can be connected to other modules such as categorization and profile creation.

2) SCALABILITY

The proposed system can be scaled, both vertically and horizontally. Vertical scalability is achieved by varying the number of resources assigned to the system. These resources can be added either in form of memory or CPU cycles.

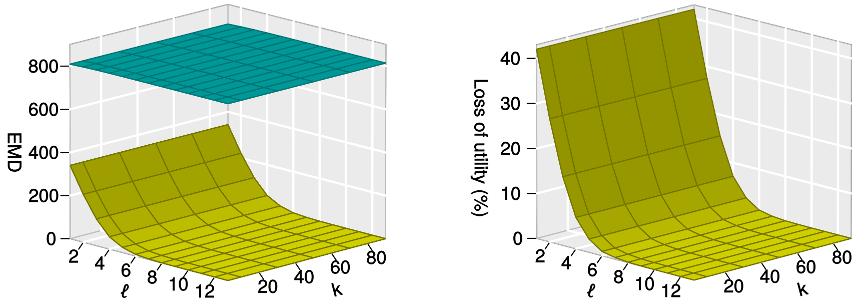


FIGURE 10. **(a) Maximum theoretical distance between profiles, constant, and average EMD distances, inversely proportional to ℓ. (b) Loss of utility (%), the more levels we use in our anonymizer, the better data utility.**

Horizontal scalability can also be achieved by activating or deactivating different instances in parallel. In addition, with the proposed anonymizer, the value of k could be dynamically adjusted, which also allows to improve the scalability of the system using it in a wider range of situations.

3) SPEED

Speed of the anonymizer and deanonymizers was tested. All the results that are shown correspond to the time required to completely treat a query using a single thread of execution on a single core. All the proposed algorithms can be used in parallel, achieving a better system throughput.

The fastest execution was achieved with $k = 3$ and $\ell = 1$, where on average a query was processed in 18.99 μs. Therefore, the system can handle up to 52659 queries per second, on average.

Average processing time per query was 33.68 μs, or 29691 queries per second. It includes executions with all the k and ℓ values we have tested. Compared to our previous proposal where we obtained 22 μs per query, we see that the system is slower on average, but with greater data utility. However, depending on which parameter values are used, the system is faster than our previous proposal, as described below.

Speed of the anonymizer is affected by k and ℓ. If we look at Figure 11, we can see that changes in the value of ℓ have little effect on required time. Contrarily, changes in the value of k have an important effect. For example, for $k = 3$ the system can process a log in about 18.99 μs. This value reaches 49.71 μs with a a value of $k = 190$. Taking into account that Google treats an average of 40000 queries per second (cf. Ref. [36] and citations thereof), a thread of our algorithm could handle all real-time queries, using k-values up to 50 with any value of ℓ, according to our test results.

The same analysis has also been done with proposed de-anonymization algorithms. Results can be seen in Figure 12. The first de-anonymizer approach obtained results comparable to the anonymizer. This was expected since in both cases the same base algorithm was used. Second and third de-anonymizers, which perform more complex operations, are also slower and more affected by increases in k-values. In all cases, we see that variations of ℓ-values are less important.

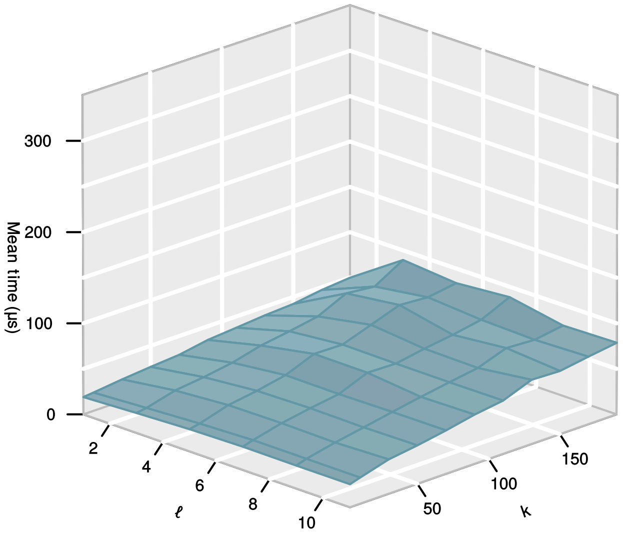


FIGURE 11. **Anonymizer mean time per query (μs). ℓ-value has little effect on required time, k-value has a greater effect.**

4) DELAY

Another factor that we consider important to evaluate is the average delay of queries between entering and leaving the system in form of anonymized query logs. Figure 13 shows this delay as the mean number of other queries that enter the system during the period between the entry and the release of a given query. As we can see, this delay is increased proportionally to the chosen ℓ-value, but it ends up stabilizing. This is reasonable, since the system needs to fill categories initially and once this happens, the output stabilizes.

Taking as reference the 40000 queries per second that Google receives (according to Ref. [36]), we see that our system's output stabilizes in a few minutes for larger values of ℓ. Once the delay is stable, our system takes less than one second for values $\ell \leq 6$, and does not reach two seconds for larger values of ℓ.

5) RESOURCE CONSUMPTION

Notice that our algorithms do not use any disk space, therefore only memory consumption needs to be evaluated.

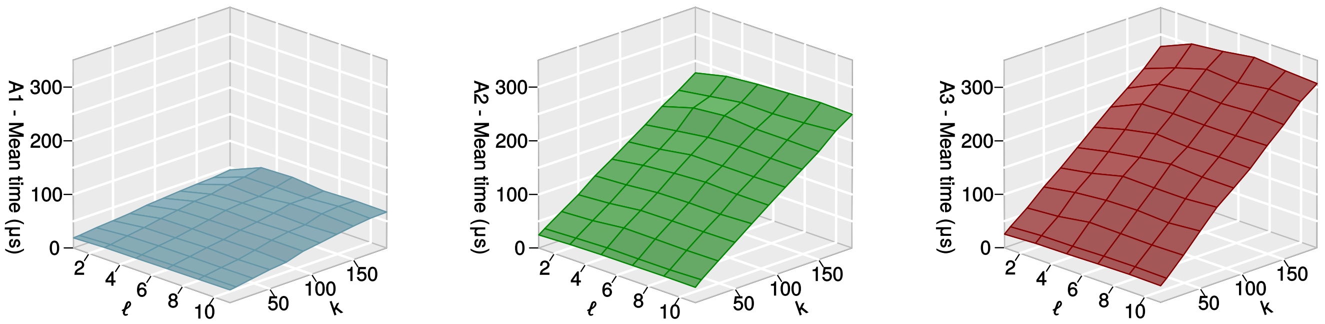


FIGURE 12. De-anonymizer mean time per query (μs). First de-anonymizer obtained comparable results to the anonymizer. Second and third de-anonymizers, which are more complex, are also slower and are more affected by increases in k-value.

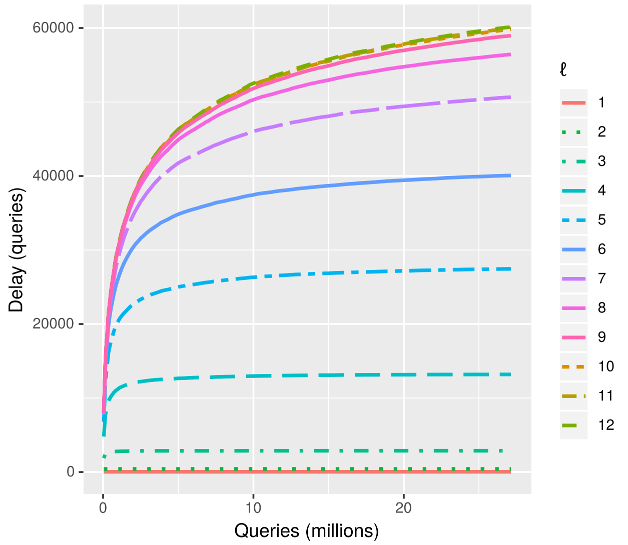


FIGURE 13. Queries delay, as the mean number of other queries that enter the system during the period between the entry and the leave of a given query. Once the categories are full, the output stabilizes.

TABLE 3. Number of categories added with each increase in ℓ-value and total categories of a tree with ℓ depth. Although we found a maximum depth of thirteen, we see that most records are classified at depths between five and seven.

ℓ	Added categories	Total categories
1	16	16
2	537	553
3	5 523	6 076
4	21 786	27 862
5	36 806	64 668
6	35 543	100 211
7	39 998	140 209
8	26 914	167 123
9	16 863	183 986
10	7 863	191 849
11	2 143	193 992
12	441	194 433
13	72	194 505

We have identified the variations in ℓ-value as the main parameter that affects resource consumption. Memory consumption increases when a new level of depth is added to the tree, in proportion to the number of effective categories that are added (cf. Table 3). Categories were created dynamically, depending on query's classification, therefore a different data set will generate different categories. At the end of our tests, we used a maximum of 194505 categories, in a tree with depth thirteen.

With our test data, we see that most records are classified at depths between five and seven, although we found a maximum depth of thirteen. As we increase depth, there are fewer queries that can be classified at the last levels, using the same data and the same classifier. Although we increase the value of ℓ the effective number of categories created is marginally increased from this point. This also causes memory consumption to stabilize. Let us illustrate the previous observation with an example. Given a query classified as "a:b:c:d:e" if we use an ℓ equal to 4, the level 4 vertex "a:b:c:d" is used for anonymization. If we increase ℓ to 5, or a higher value, we use for anonymization the complete category, i.e. level 5 vertex "a:b:c:d:e", even if we use an $\ell = 13$.

On the other hand, we can see that k adds a multiplicative factor in the consumption of resources, depending on the number of existing effective categories. The results in Figure 14, only show the maximum memory consumption.

Regarding different algorithms set forth, both *anonymizer* and *de-anonymizer 1* show the same memory consumption profile. *De-anonymizer 3* is the algorithm with higher memory consumption. This is because that algorithm creates user profiles in memory and therefore is reasonable that it uses more resources. *Anonymizer* and *de-anonymizers 1* and *2* should not use more memory than the reported, regardless of the volume of logs they deal with. However, this is not the case of *deanonymizer 3*, as when it creates new user profiles, it increases the memory consumption.

6) EFFICIENCY

As we have seen in the previous sections, a lightweight method has been defined. It allows the logs to be quickly processed with reduced resource consumption.

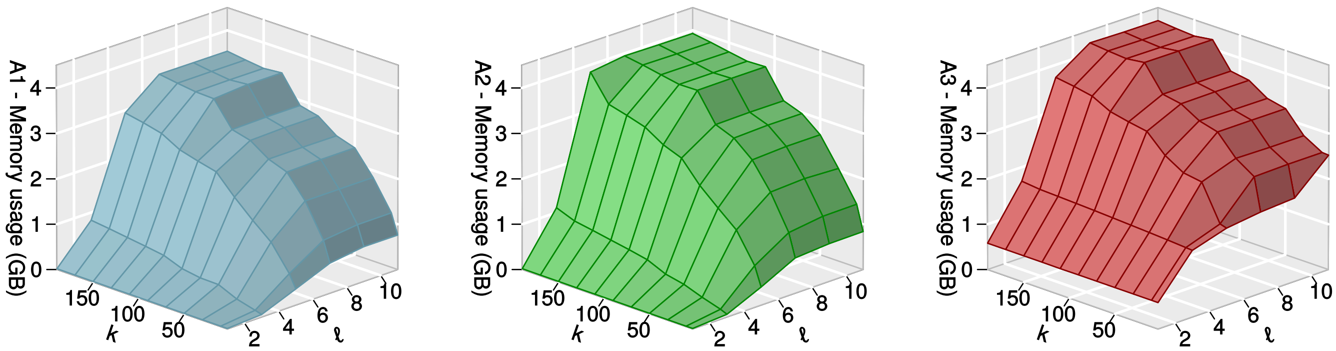


FIGURE 14. The value of ℓ is the main parameter that affects memory consumption. The value of k adds a multiplicative factor. Both the *anonymizer* and *de-anonymizer 1* show the same memory profile. *De-anonymizer 3* is the algorithm with higher memory consumption, because it creates user profiles in memory.

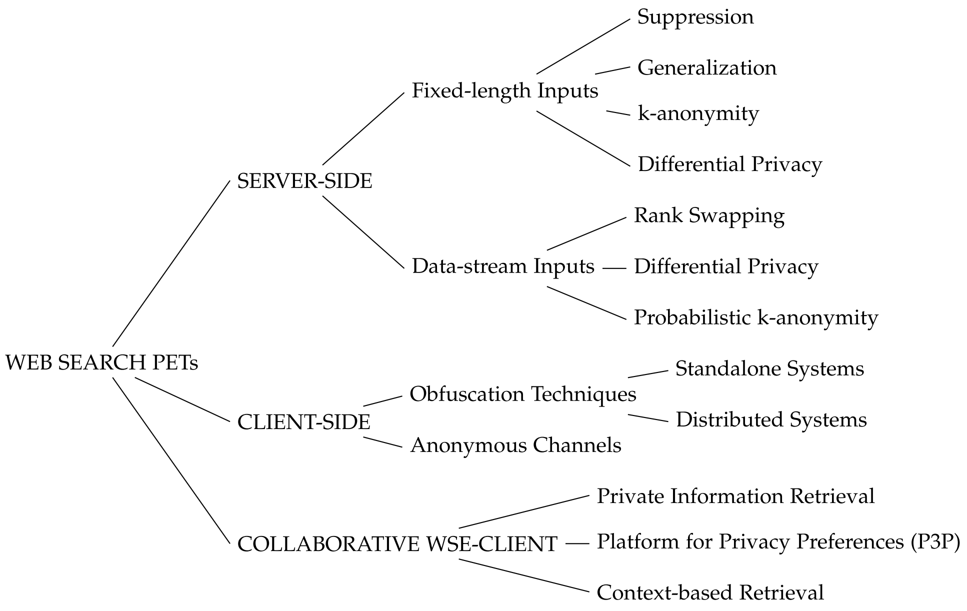


FIGURE 15. Classification of web search Privacy Enhancing Technologies (PETs).

Studying the anonymizer we see that both delay and memory consumption vary initially, because the system starts empty and the sets must be filled. As we have seen, once the sets achieve k elements, these values stabilize. On the other hand, the processing speed of a log depends on the value of k and ℓ, but it remains constant throughout each test set.

Analyzing the proposed algorithm, we can see that each log is only treated once. This allows us to equate its efficiency with well known singly-linked list traversal algorithms. Therefore, the algorithmic time complexity of our proposal is linear regarding to the input and could be established as $\mathcal{O}(n)$.

7) TRANSPARENCY

The input of the system should be a stream of classified query logs that can be obtained from the WSE. In case that only unclassified logs are available, a classification micro-service could be implemented and added to the WSE architecture, as we previously showed in Ref. [28]. In case that classified logs are available, those logs could be used without further modifications. Our system generates an anonymized stream of logs, preserving the existing structure. From the point of view of an existing client, generated output will be completely indistinguishable of the original one. Therefore, total transparency is reached.

V. RELATED WORK

Our work relates to the use of privacy-enhancing technologies (PETs) applied to the web search paradigm. Figure 15 shows and positions a classification of PET proposals designed to protect the users' privacy in front of WSEs — on the basis of previous classifications [37]–[39]. The classification identifies two main actors: users and WSEs. The first group contains proposals that protect users' privacy at the WSE side, without the need for users' participation. They are asynchronous and transparent to the users. Our proposal falls under this first category. The second group includes approaches that protect users' privacy without any help from the WSE, i.e., when users do not require any changes at the server side of the WSE. The third group comprises approaches that require a certain level of cooperation between users and WSEs. The latter are not considered as server-side, since users actively participate in the process — when WSEs do not cooperate, it is assumed that users immediately detect them. In the sequel, we report related work under all three categories.

A. SURVEY ON SERVER SIDE PROPOSALS

WSEs aim at anonymizing data while minimizing information loss, for profit purposes. Our work is focused on this assumption. The goal is to commercialize releases of the protected set of query logs to third-parties. Anonymization solutions to reach such a goal can get classified according to anonymizaiton inputs. Most solutions are either processing fixed-length (e.g., block-based) or data-stream inputs.

1) FIXED-LENGTH INPUTS

In the case of fixed-length inputs, existing proposals consider a set of finite and static data structures. Each set contains all the elements to be anonymized. The protection of the whole dataset is conducted as a two-step process, first analyzing all the dataset elements, then processing them. Some representative solutions under this category are presented next.

a: SUPPRESSION

The anonymization of the dataset is conducted by eliminating those elements which, in isolation or combination, may reveal sensible information. The analysis of the dataset assumes either statistic or semantic methods, to identify which elements require suppression.

Examples of suppression under the context of query logs anonymization exist in the related literature [40]. The deletion of identifiers such as social security numbers, physical addresses, bank accounts or any another identification data related to the user, are traditional examples of suppression in the literature [41]. Nevertheless, the AOL incident reveals the limitations of this approach [22]–[24]. The existence of quasi-identifiers in the AOL dataset, and the complexity of identifying their combinations, were proven enough to re-identify AOL users via traditional log correlation techniques [21].

The suppression of infrequent queries is another approach [13]. It aims at suppressing those queries that are likely to contain identifying or quasi-identifying information. The approach requires the definition and accomplishment of thresholds. Since queries may appear only a limited number of times [14], the elimination of a significant number of non-identifying queries becomes a complex and error-prone task. The approach can be complemented by selecting those queries resulting from clicking on common URLs, i.e., by establishing a correlation between clicking and quasi-identifiers [10]. Another possibility is the representation of query logs using graph theory [9]. Nodes are seen as user queries. A query is connected to other user queries whenever the intersection of their clicked URLs sets is non empty. The anonymization process is done by iteratively suppressing those queries that return less than k documents. Those queries that considerably contribute to the query graph (i.e., queries with partial or full target URLs) are considered vulnerable and suppressed.

b: GENERALIZATION

Another approach used to provide anonymity is based on the generalization of domain relationships, i.e., by analyzing the values that the associated attributes can assume. The concept of minimal generalization seeks to maintain the lowest possible distortion levels of the processed datasets [42]. Top-down approaches, using lexical and semantic databases to conduct general-purpose generalizations have also been proposed [43], [44]. The idea is to transform groups of input queries to common conceptual abstractions (e.g. football and tennis as sports), in order to make users who performed similar queries indistinguishable. The main limitations associated to these approaches rely on the construction of generic dictionaries associated to those words or concepts to anonymize. This may require, moreover, specific adaptations based on the language used on the original datasets.

c: K-ANONYMITY

The property of k-anonymity [27] was proposed to minimize the risk of record-linkage. A k-anonymized dataset has the property that each record is indistinguishable from at least $k - 1$ other records. This way, no individual can be re-identified with probability exceeding $\frac{1}{k}$ through linking attacks.

Current approaches propose methods of Statistical Disclosure Control (SDC) to transform query records into anonymous logs, while reducing the amount of query deletion [45], [46]. Logs of similar queries are used to group users, and later their queries are rewritten by a prototype query. This makes them indistinguishable [47]–[51]. Users and queries are conserved, although queries are transformed to reduce the risk of disclosure. Similar approaches propose the generation of fake messages to mix them with the legitimate ones [52] or masking infrequent queries using a more general frequent query [53] to achieve levels of privacy comparable to k-anonymity.

d: DIFFERENTIAL PRIVACY

Initially described as a solution to manage the risk of identifying users participating in a given dataset [54], interactive scenarios of the same approach do also exist [55]. The initial scenarios associated to differential privacy expect queries accessing partial information of the dataset. However, when intelligently conducted, such queries may end up revealing information from the original users. For that reason, interactive improvements are expected to evaluate how far queries get through, to deny responding whenever a limit is bypassed. Since the protected outputs may still preserve some statistics (e.g., query suggestions and spelling corrections), extended proposals aim at further limiting the risk of information disclosure in such returned statistics [10].

Authors in [56] propose a technique in which samples with high utility are selected to become the representative records in each cluster, i.e., to achieve the objective of leaking less privacy and releasing more useful information. Other proposals [57], [58] pose the addition of Laplacian noise to the logs, to preserve privacy. However, the more noise is added, the more data utility gets reduced.

2) DATA-STREAM INPUTS

This approach allows to treat data partially. The system does not need all the data to start dealing with. It also makes possible a partial treatment of the data. This approach is able to generate data outputs with a minimum delay [59]. In addition, it also opens the doors to deal with very large datasets, even infinite ones. Still, protecting the privacy of very large data streams continues to have some difficulties [60]. Next, we survey some representative solutions under this category.

a: RANK SWAPPING

The method was first described for numerical variables [61], although initial ideas associated to swapping data exist in other previous areas [62]. We can also find other approximations [63], [64]. In all such cases, the proposals only consider structured data. This is because the data is sorted by the value of an attribute and then exchanged with a randomly selected value (the nearest ones in the rank) [65].

b: DIFFERENTIAL PRIVACY

The differential privacy approach can also be applied to anonymize data-streams [66]. In this case, there is no release of the original query, but a synthetic one, obtained using semantic similarity. The lack of structure in query logs, combined with new terms which may not be present into the semantic database, could represent a challenge for this approach. Another limitation using differential privacy in a streaming environment is to maintain a fixed privacy level. It is possible that no more data can be published in order to preserve the privacy of users.

c: PROBABILISTIC K-ANONYMITY

The concept of probabilistic k-anonymity relaxes the indistinguishability requirement of k-anonymity [67]. It only requires that the probability of re-identification is maintained, with regard to the case of k-anonymity. By relaxing the indistinguishability requirement, a better use of the data may be accomplished. Moreover, logs can be released containing the original queries. On the negative side, given the continuous generalization of unstructured dataset elements, a certain imprecision is added to the generated profiles. Existing limitations in the related literature [28], [68] is in terms of classification methods, which are very basic. Hence, the number of resulting categories is low, leading to higher degrees data utility loss.

B. SURVEY ON CLIENT SIDE PROPOSALS

One may argue that WSEs have no motivation to protect the privacy of users. Indeed, users may be seen as the only interested party responsible to protect data privacy. Under this assumption, we find some protection approaches which do not expect any collaboration between WSEs and users. Such approaches can be classified in two main categories: i) obfuscation techniques and ii) anonymous channels. Obfuscation techniques generate noise to distort the user's profile managed by the WSEs. Anonymous channels assume an infrastructure between users and WSEs to handle the profiling of activities. The use of client side techniques are assumed to generate non-realistic profiles that may have an adverse effect on the services provided by WSEs.

1) OBFUSCATION TECHNIQUES

Early techniques assume the introduction of random queries (e.g., fake queries), in order to obscure users' profiles. Random queries must be indistinguishable from the real queries. This property is known as unobservability. Representative solutions based on obfuscation techniques can be classified according to the number of users that participate in the protocol. We have standalone solutions and distributed solutions. Standalone solutions assume individual users handling their own privacy in front of the WSEs. Distributed solutions assume groups of users working together to protect the privacy of each user. Next, we provide some examples for each category.

a: STANDALONE SYSTEMS

These schemes generate synthetic queries that are used to hide the real queries of the users [47], [69]–[76]. Synthetic queries are submitted together with the real queries, obfuscating the profiles that the WSE owns for each user. If the synthetic queries are in some way semantically related to the user's queries, the obfuscated profile will still be usable, i.e., the WSE will be able to personalize the user's results. When the synthetic queries are semantically unrelated to the user's queries, the profile will be heterogenous and the personalization will be less accurate. This does not mean that one alternative is better than the other, since users may have different preferences regarding of the trade-off between privacy and utility. Some works show that it is possible to distinguish real queries from synthetic queries [73], [77]–[79]. These works rely on the idea that machine-generated queries do not have the same features as human-generated queries.

b: DISTRIBUTED SYSTEMS

These schemes require the collaboration of a group of users that work in partnership to protect their privacy, i.e., they hide their actions within the actions of many others [80]–[86]. Typically, these schemes put users into a large group where they submit requests on behalf of other members. Users exchange their queries. Personalization is only possible if the members of the group share the same interests [37]. In some proposals [80]–[82], there is a central node that poses a bottleneck in the overall system performance. In other cases, one type of path [80], [83]–[86] is created to submit the query or a group of users must be created [80]–[82]. In both cases, a significant delay is introduced [37].

2) ANONYMOUS CHANNELS

The proposals under this category use anonymous infrastructures [87], [88] in order to send users' queries to the WSE. By concealing users' identity associated to the queries, WSEs are assumed to be unable to profile users. However, this may affect the quality of the service offered by the WSEs to the users.

Chaum's mix networks [89] are representative cases of solutions under the category of anonymous channels. Messages pass through several nodes. Each node disassociates the input messages from the output messages, by means of cryptography [87], [88]. Evolved techniques assume the use of proxies [90], relying connections (e.g., queries) from users to the recipients (e.g., the WSEs). The key concept is that proxy delivers the messages but does not disclose the source (e.g., the user' identity). DuckDuckGo,[1] Start Page[2] and Yippy[3] are some significant examples using proxy-like infrastructures. By using these solutions, users transfer their trust from WSEs to the proxies (i.e., users must assume that proxies do not monitor or log their traffic).

Web MIXes [91] provides anonymous and unobservable real-time Internet access. It incorporates an authentication mechanism in order to prevent flood attacks. Additionally, it includes a feedback system with an interface that informs users about their current level of protection. However, some flaws in their authentication process may allow external attackers to perform replay attacks [92]. The synchronous nature of Web MIXes may also end in problems when dealing with asynchronous TCP/IP networks [93].

The use of onion routing [94] to establish anonymous channels under the context of queries and WSEs has also been proposed in the literature [95]. General purpose plugins, and modified web-browsers[4] using the Tor Project [96], are user-friendly solutions based on the onion routing paradigm. Nonetheless, several weaknesses have been reported [97]. Tor does not attempt to offer security against passive global adversaries [88]. Similarly, the Invisible Internet Project (I2P) [98] builds an anonymous network layer designed to be used for anonymous communication.

C. *SURVEY ON COLLABORATIVE WSE-CLIENT PROPOSALS*

Solutions under this category assume that users and WSEs work together in order to protect users' privacy. Next, we report solutions under this category in three main groups: i) Private Information Retrieval; ii) Platform for Privacy Preferences (P3P); and iii) Context-based Retrieval.

1) PRIVATE INFORMATION RETRIEVAL

Private Information Retrieval (PIR) schemes [99]–[102] enable users to obtain information from a database privately, i.e., the server cannot know what information was retrieved. Through a PIR scheme, users can search the documents stored in the database and recover those of their interest. The problem of submitting a query to a WSE while preserving the user's privacy is equivalent to the PIR problem. However, PIR schemes suffer from two practical problems that make them not appropriate for WSEs [81]: PIR schemes are not suitable for large databases, and users are assumed to know the precise location of the records to be recovered.

2) PLATFORM FOR PRIVACY PREFERENCES (P3P)

The Platform for Privacy Preferences (P3P) [103], [104] was created by the World Wide Web Consortium (W3C) with the objective of making easier for users to obtain information about the privacy policies of the sites that they visit. P3P is a framework through which users can automate the protection of their privacy. They can define their privacy preferences and, when a website does not conform to these preferences, then P3P-enabled browsers may alert the user and even take pre-established actions (e.g., deny access to cookies). The Do-Not-Track initiative [105] is a policy-based P3P system in which HTTP headers request web applications not to track users. The web application must be P3P-complaint in order to be effective. It has been studied in several works [106]–[108] and standardized by W3C. However, it is considered as an obsolete protocol nowadays. In fact, P3P-like solutions have been criticized due to the impact that governmental laws may have over users [109], the lack of follow-up from websites w.r.t. privacy-protection mandates in their legal jurisdictions (e.g., compliance difficulties of websites to enforce their own privacy policies) [110], and low number of potential adopters [111].

3) CONTEXT-BASED RETRIEVAL

Context-based retrieval proposals aim at storing user profiles (e.g., search history) on the client's machine. This information allows to obtain users' interests and re-rank search results according to them. WSE and users participate together in the searching process in order to obtain the final results, i.e., the WSE receives the query and returns the results. Then, these results are re-ranked at the client-side. The User-Centered Adaptive Information Retrieval (UCAIR) project [112] collects and exploits available user context from submitted queries and clicked results. Similar schemes allows users to choose the content and degree of details of their profiles exposed to the WSE [4], [113], [114]. In the end, users determine the profile content that is revealed to the WSE when a query is submitted. The adjustment of parameters associated to the stored profiles is possible, in order to improve the quality of the results. Potential disadvantages of these proposals

[1] https://duckduckgo.com/

[2] https://www.startpage.com/

[3] https://www.yippy.com/

[4] https://gitweb.torproject.org/tor-browser.git/

relate to performance and effectiveness limitations of results ranked at the client (i.e., much less effective than ranking the results at the server side) [112]. Moreover, it is expected that WSEs can still profile users after several executions of the approach.

VI. CONCLUSION

A formal approach for the anonymization of WSE query logs has been presented. Our proposal allows to publish query logs without any other modification than eliminating direct identifiers and equivalent user re-assignment categories. This contrasts with existing approaches that release heavily modified data, either distorted or generalized, to maintain anonymity. In addition, our proposal allows some degree of configuration, using two main parameters:

- k to adjust the level of diversity on each category.
- ℓ to adjust the amount of available categories.

This parameterization allows to adjust privacy and utility levels of generated logs according to the needs of each application.

Three algorithms have been evaluated performing an attack to the anonymized data, using the most favorable scenario for the attacker, i.e., when the attacker knows the algorithms used by the WSE, all the parameters and the data. The attacker has access to the anonymized log stream, but not to the original logs. Tests with this context and several values of k and ℓ were conducted.

Our best record-linkage attempt re-identified 23.18% of original logs with the lowest k-value, highest ℓ-value and using the most complex record-linkage algorithm, which is also the one that needs more resources. With the same parameters, using the simplest record-linkage algorithm we get an 18.36%. These results are reduced rapidly, recovering less than 1% of original logs when using values of k over 100. Variations in the values of ℓ do not have a representative impact in terms of record linkage, but they do offer a significant improvement in terms of data utility.

Our proposed ideas were tested using the AOL released logs, showing the feasibility of our solution over real environments. The application of our work is sufficient to generate anonymized logs that meet representative criteria, e.g., release of anonymized data to third parties. Our solution can handle the equivalent to Google's average load, using only one execution thread per testing environment. To evaluate log's utility after anonymization, we have measured distances between user profiles using Earth Mover's Distance. We have found that using an ℓ-value of one, a 42.03% of utility was lost. Using ℓ-values of six or more, less than 1% of utility was lost.

There are several avenues for improving our work. Additional categorizers may be proposed, for example using artificial intelligence systems to perform query analysis. Another improvement is to consider dynamic ℓ-values, both globally or for some specific category branches. System performance could also be tested in a distributed node environment, where each node is responsible for processing a part of the queries. A real-time record linkage analysis could be added to ensure that we only publish records that meet a certain threshold of privacy. Finally, some experiments could be conducted with queries' time, both with anonymization and de-anonymization algorithms, to improve their performance.

ACKNOWLEDGMENT

The authors graciously acknowledge support from CYBERCAT (Center for Cybersecurity Research of Catalonia, Tarragona, Catalonia), the UNESCO Chair in Data Privacy, the Cyber CNI chair of the Institut Mines-Télécom. The following funding sources are gratefully acknowledged: European Commission (projects H2020-830892 "SPARTA", H2020-871042 "SoBigData++"), Government of Catalonia (grant 2017 SGR 705) and Spanish Government (project RTI2018-095094-B-C21 and network RED2018-102321-T). The views in this paper are the authors' own and do not necessarily reflect the views of UNESCO or any of the funders.

REFERENCES

[1] Netcraft. (2020). *January 2020Web Server Survey*. [Online]. Available: https://news.netcraft.com/archives/2020/01/21/january-2020-web-server-survey.html

[2] Real Time Statistics Project. (2020). *Internet Live Stats*. [Online]. Available: https://www.internetlivestats.com/

[3] X. Shen, B. Tan, and C. Zhai, "Privacy protection in personalized search," *ACM SIGIR Forum*, vol. 41, no. 1, pp. 4–17, Jun. 2007. http://portal.acm.org/citation.cfm?id=1273222&dl=GUIDE&coll=GUIDE&CFID=17786915&CFTOKEN=37653058

[4] Y. Xu, K. Wang, B. Zhang, and Z. Chen, "Privacy-enhancing personalized Web search," in *Proc. 16th Int. Conf. World Wide Web (WWW)*. New York, NY, USA: ACM, 2007, pp. 591–600. [Online]. Available: http://portal.acm.org/citation.cfm?id=1242652

[5] E. Agichtein, E. Brill, and S. Dumais, "Improving Web search ranking by incorporating user behavior information," in *Proc. 29th Annu. ACM SIGIR Conf.* Seattle, WA, USA: ACM, 2006, pp. 19–26. [Online]. Available: http://portal.acm.org/citation.cfm?id=1148170.1148177

[6] R. Jones, B. Rey, O. Madani, and W. Greiner, "Generating query substitutions," in *Proc. 15th Int. Conf. World Wide Web (WWW)*. New York, NY, USA: ACM, 2006, pp. 387–396. [Online]. Available: http://portal.acm.org/citation.cfm?id=1135777.1135835

[7] R. T. S. Cameron and A. Pickersgill. (2014). *New Ways for Turning Data Into Dollars Now*. [Online]. Available: https://www.mckinsey.com/business-functions/marketing-and-sales/our-insights/new-ways-for-turning-data-into-dollars-now

[8] D. J. Brenes and D. Gayo-Avello, "Stratified analysis of AOL query log," *Inf. Sci.*, vol. 179, no. 12, pp. 1844–1858, May 2009. [Online]. Available: http://portal.acm.org/citation.cfm?id=1523512.1523572

[9] B. Poblete, M. Spiliopoulou, and R. A. Baeza-Yates, "Website privacy preservation for query log publishing," in *Privacy, Security, and Trust in KDD* (Lecture Notes in Computer Science), vol. 4890, F. Bonchi, E. Ferrari, B. Malin, and Y. Saygin, Eds. Berlin, Germany: Springer, 2007, pp. 80–96. [Online]. Available: https://link.springer.com/chapter/10.1007%2F978-3-540-78478-4_5

[10] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas, "Releasing search queries and clicks privately," in *Proc. 18th Int. Conf. World Wide Web (WWW)*, J. Quemada, G. León, Y. S. Maarek, and W. Nejdl, Eds. New York, NY, USA: ACM, 2009, pp. 171–180. [Online]. Available: https://dl.acm.org/citation.cfm?doid=1526709.1526733

[11] J. Bar-Ilan, "Access to query logs—An academic researcher's point of view," in *Proc. Query Log Anal., Social Technol. Challenges. Workshop 16th Int. World Wide Web Conf. (WWW)*, E. Amitay, G. C. Murray, and J. Teevan, Eds., May 2007, pp. 1–4.

[12] P. Shea, *Book Review: 'Click: What Millions of People are Doing Online and Why it Matters' by Bill Tancer*, vol. 2010, no. 1. Santa Rosa, CA, USA: eLearn, Jan. 2010, p. 8. [Online]. Available: http://dblp.uni-trier.de/db/journals/elearn/elearn2010.html#Shea10

[13] E. Adar, "User 4xxxxx9: Anonymizing query logs," in *Proc. Workshop Query Log Anal. 16th World Wide Web Conf.*, 2007, pp. 1–8.

[14] S. M. Beitzel, E. C. Jensen, A. Chowdhury, D. Grossman, and O. Frieder, "Hourly analysis of a very large topically categorized Web query log," in *Proc. 27th Annu. Int. Conf. Res. Develop. Inf. Retr. (SIGIR)*, 2004, pp. 321–328, doi: 10.1145/1008992.1009048.

[15] S. M. Beitzel, E. C. Jensen, A. Chowdhury, O. Frieder, and D. Grossman, "Temporal analysis of a very large topically categorized Web query log," *J. Amer. Soc. Inf. Sci. Technol.*, vol. 58, no. 2, pp. 166–178, Jan. 2007, doi: 10.1002/asi.20464.

[16] A. Erola, J. Castellà-Roca, G. Navarro-Arribas, and V. C. Torra, "Semantic microaggregation for the anonymization of query logs using the open directory project," *Statist. Oper. Res. Trans.*, vol. 35, pp. 41–58, Sep. 2011.

[17] F. Silvestri, "Mining query logs: Turning search usage data into knowledge," *Found. Trends Inf. Retr.*, vol. 4, nos. 1–2, pp. 1–174, Jan. 2010, doi: 10.1561/1500000013.

[18] L. Willenborg and T. De Waal, *Elements of Statistical Disclosure Control*, vol. 155. Springer, 2012.

[19] C. Soghoian, "The problem of anonymous vanity searches," *SSRN Electron. J.*, vol. 3, p. 299, Jan. 2007.

[20] R. Jones, R. Kumar, B. Pang, and A. Tomkins, "'I know what you did last summer': Query logs and user privacy," in *Proc. 16th ACM Conf. Conf. Inf. Knowl. Manage. (CIKM)*. New York, NY, USA: ACM, 2007, pp. 909–914. [Online]. Available: http://portal.acm.org/citation.cfm?id=1321440.1321573&coll=GUIDE&dl=GUIDE

[21] M. Barbaro, T. Zeller, and S. Hansell, "A Face is exposed for AOL searcher no. 4417749," *New York Times*, vol. 9, no. 2008, p. 8, 2006. [Online]. Available: http://mrl.nyu.edu/~dhowe/TrackMeNot/NYTimes_AOL_Exposed.htm

[22] EF Foundation. (2009). *Aol's Massive Data Leak*. [Online]. Available: http://w2.eff.org/Privacy/AOL/

[23] E. Mills. (Sep. 2006). *Aol Sued Over Web Search Data Release*. [Online]. Available: http://news.cnet.com/8301-10784_3-6119218-7.html

[24] S. Hansell, "Increasingly, Internet's data trail leads to court," New York Times, New York, NY, USA, Tech. Rep. 4, 2006.

[25] *Proposal for a Regulation of the European Parliament and of the Council on the Protection of Individuals With Regard to the Processing of Personal Data and on the Free Movement of Such Data (General Data Protection Regulation)*, document 52012PC0011, C Europe, Jan. 2016.

[26] A. Hundepool, J. Domingo-Ferrer, L. Franconi, S. Giessing, R. Lenz, J. Longhurst, E. S. Nordholt, G. Seri, and P. Wolf, "Handbook on statistical disclosure control," in *ESSnet on Statistical Disclosure Control*. Brussels, Belgium: European Commission, 2010.

[27] S. D. C. di Vimercati, S. Foresti, G. Livraga, and P. Samarati, "Protecting privacy in data release," in *Foundations of security Analysis and Design VI: FOSAD Tutorial Lectures*, A. Aldini and R. Gorrieri, Eds. Berlin, Germany: Springer-Verlag, 2011, pp. 1–34. [Online]. Available: http://dl.acm.org/citation.cfm?id=2028200.2028202

[28] D. Pàmies-Estrems, J. Castellà-Roca, and A. Viejo, "Working at the Web search engine side to generate privacy-preserving user profiles," *Expert Syst. Appl.*, vol. 64, no. C, pp. 523–535, Dec. 2016, doi: 10.1016/j.eswa.2016.08.033.

[29] A. B. Bondi, "Characteristics of scalability and their impact on performance," in *Proc. 2nd Int. Workshop Softw. Perform. (WOSP)*. New York, NY, USA: ACM, 2000, pp. 195–203.

[30] M. Michael, J. E. Moreira, D. Shiloach, and R. W. Wisniewski, "Scale-up x scale-out: A case study using nutch/lucene," in *Proc. IEEE Int. Parallel Distrib. Process. Symp.*, 2007, pp. 1–8.

[31] I. AOL. (2006). *AOL Keyword Searches*. [Online]. Available: http://dontdelete.com/default.asp

[32] K. Purdam and M. Elliot, "A case study of the impact of statistical disclosure control on data quality in the individual UK samples of anonymised records," *Environ. Planning A, Economy Space*, vol. 39, no. 5, pp. 1101–1118, May 2007.

[33] A. Kolmogorov, "Sulla determinazione empírica di una legge di distribuzione," *Giorn Dell'inst Ital Degli Att*, vol. 4, pp. 83–91, Jan. 1933.

[34] N. Smirnov, "Table for estimating the goodness of fit of empirical distributions," *Ann. Math. Statist.*, vol. 19, no. 2, pp. 279–281, Jun. 1948.

[35] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *Int. J. Comput. Vis.*, vol. 40, no. 2, pp. 99–121, Nov. 2000.

[36] I. L. Stats. (2017). *Google Search Statistics*. [Online]. Available: http://www.internetlivestats.com/google-search-statistics/

[37] C. Romero-Tris. (2014). Client-Side Privacy-Enhancing Technologies in Web Search. Universitat Rovira I Virgili. [Online]. Available: http://hdl.handle.net/10803/284036

[38] C. Romero-Tris, A. Viejo, and J. Castellà-Roca, "Multi-party methods for privacy-preserving Web search: Survey and contributions," in *Advanced Research in Data Privacy*. 2015, pp. 367–387, doi: 10.1007/978-3-319-09885-2_20.

[39] A. Erola. (2013). Contributions to Privacy Web Search Engines. Universitat Rovira I Virgili. [Online]. Available: http://hdl.handle.net/10803/130934

[40] A. Cooper, "A survey of query log privacy-enhancing techniques from a policy perspective," *ACM Trans. Web*, vol. 2, no. 4, pp. 19:1–19:27, Oct. 2008, doi: 10.1145/1409220.1409222.

[41] Center for Democracy and Technology. (2007). *Search Privacy Practices: A Work in Progress*. [Online]. Available: https://cdt.org/wp-content/uploads/privacy/20070808searchprivacy.pdf

[42] P. Samarati and L. Sweeney, "Protecting privacy when disclosing information: K-anonymity and its enforcement through generalization and suppression," SRI Int., Menlo Park, CA, USA, Tech. Rep. SRI-CSL-98-04, 1998.

[43] Y. He and J. F. Naughton, "Anonymization of set-valued data via top-down, local generalization," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 934–945, Aug. 2009, doi: 10.14778/1687627.1687733.

[44] M. Terrovitis, N. Mamoulis, and P. Kalnis, "Privacy-preserving anonymization of set-valued data," *Proc. VLDB Endowment*, vol. 1, no. 1, pp. 115–125, Aug. 2008, doi: 10.14778/1453856.1453874.

[45] G. Navarro-Arribas and V. Torra, "Tree-based microaggregation for the anonymization of search logs," in *Proc. IEEE/WIC/ACM Int. Joint Conf. Web Intell. Intell. Agent Technol.*, vol. 3. Washington, DC, USA: IEEE Computer Society, 2009, pp. 155–158, doi: 10.1109/WI-IAT.2009.251.

[46] Y. Hong, X. He, J. Vaidya, N. Adam, and V. Atluri, "Effective anonymization of query logs," in *Proc. 18th ACM Conf. Inf. Knowl. Manage. (CIKM)*. New York, NY, USA: ACM, 2009, pp. 1465–1468, doi: 10.1145/1645953.1646146.

[47] D. Sánchez, J. Castellà-Roca, and A. Viejo, "Knowledge-based scheme to create privacy-preserving but semantically-related queries for Web search engines," *Inf. Sci.*, vol. 218, pp. 17–30, Jan. 2013, doi: 10.1016/j.ins.2012.06.025.

[48] M. Batet, A. Erola, D. Sánchez, and J. Castellà-Roca, "Utility preserving query log anonymization via semantic microaggregation," *Inf. Sci.*, vol. 242, pp. 49–63, Sep. 2013, doi: 10.1016/j.ins.2013.04.020.

[49] A. Erola and J. Castellà-Roca, "Using search results to microaggregate query logs semantically," in *Data Privacy Manage. Auto. Spontaneous Secur.-8th Int. Workshop (DPM), 6th Int. Workshop (SETOP)*, Egham, U.K., Sep. 2013, pp. 148–161, doi: 10.1007/978-3-642-54568-9_10.

[50] G. Navarro-Arribas, V. Torra, A. Erola, and J. Castellà-Roca, "User k-anonymity for privacy preserving data mining of query logs," *Inf. Process. Manage.*, vol. 48, no. 3, pp. 476–487, May 2012, doi: 10.1016/j.ipm.2011.01.004.

[51] M. Batet, A. Erola, D. Sánchez, and J. Castellà-Roca, "Semantic anonymisation of set-valued data," in *Proc. 6th Int. Conf. Agents Artif. Intell. (ICAART)*, vol. 1. Loire Valley, France: ESEO Angers, Mar. 2014, pp. 102–112, doi: 10.5220/0004811901020112.

[52] A. Viejo, D. Sánchez, and J. Castellà-Roca, "Preventing automatic user profiling in Web 2.0 applications," *Knowl.-Based Syst.*, vol. 36, pp. 191–205, Dec. 2012, doi: 10.1016/j.knosys.2012.07.001.

[53] C. Carpineto and G. Romano, "Semantic search log k-anonymization with generalized k-cores of query concept graph," in *Proc. 35th Eur. Conf. Inf. Retr. (ECIR)*, Moscow, Russia, Mar. 2013, pp. 110–121, doi: 10.1007/978-3-642-36973-5_10.

[54] C. Dwork, "Differential privacy," in *Proc. Int. Colloq. Automata, Lang., Program. (ICALP)*, Venice, Italy, Jul. 2006, pp. 1–12, doi: 10.1007/11787006_1.

[55] P. Kodeswaran and E. Viegas, "Applying differential privacy to search queries in a policy based interactive framework," in *Proc. ACM 1st Int. Workshop Privacy Anonymity Very Large Databases (PAVLAD)*, Hong Kong, 2009, pp. 25–32, doi: 10.1145/1651449.1651455.

[56] X. Meng, Z. Xu, B. Chen, and Y. Zhang, "Privacy-preserving query log sharing based on prior N-word aggregation," in *Proc. IEEE Trustcom/BigDataSE/ISPA*, Tianjin, China, Aug. 2016, pp. 722–729, doi: 10.1109/TrustCom.2016.0131.

[57] S. Zhang, H. Yang, and L. Singh, "Applying epsilon-differential private query log releasing scheme to document retrieval," in *Proc. 2nd Int. Workshop Privacy-Preserving Inf. Retr. Workshop (PIR)*, Santiago, Chile, Aug. 2015, pp. 1–4.

[58] S. Zhang, H. Yang, and L. Singh, "Anonymizing query logs by differential privacy," in *Proc. 39th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR)*, Pisa, Italy, Jul. 2016, pp. 753–756, doi: 10.1145/2911451.2914732.

[59] M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy, "Mining data streams: A review," *ACM SIGMOD Rec.*, vol. 34, no. 2, pp. 18–26, Jun. 2005, doi: 10.1145/1083784.1083789.

[60] G. Krempl, I. Žliobaite, D. Brzeziński, E. Hüllermeier, M. Last, V. Lemaire, T. Noack, A. Shaker, S. Sievi, M. Spiliopoulou, and J. Stefanowski, "Open challenges for data stream mining research," *ACM SIGKDD Explor. Newslett.*, vol. 16, no. 1, pp. 1–10, Sep. 2014, doi: 10.1145/2674026.2674028.

[61] R. Moore, "Controlled data swapping techniques for masking public use microdata sets," unpublished.

[62] S. P. Reiss, "Practical data-swapping: The first steps," in *Proc. IEEE Symp. Secur. Privacy*, Oakland, CA, USA, Apr. 1980, pp. 38–45, doi: 10.1109/SP.1980.10014.

[63] V. C. Torra and J. Domingo-Ferrer, *Disclosure Control Methods and Information Loss for Microdata*. Amsterdam, The Netherlands: Elsevier, 2001, pp. 91–110.

[64] J. Domingo-Ferrer and V. C. Torra, *A Quantitative Comparison of Disclosure Control Methods for Microdata*. Amsterdam, The Netherlands: Elsevier, 2001, pp. 111–133.

[65] G. Navarro-Arribas and V. Torra, "Rank swapping for stream data," in *Modeling Decisions for Artificial Intelligence* (Lecture Notes in Computer Science), V. Torra, Y. Narukawa, and Y. Endo, Eds, vol. 8825. Cham, Switzerland: Springer, 2014, pp. 217–226, doi: 10.1007/978-3-319-12054-6_19.

[66] D. Sánchez, M. Batet, A. Viejo, M. Rodríguez-García, and J. Castellà-Roca "A semantic-preserving differentially private method for releasing query logs," *Inf. Sci.*, vols. 460–461, pp. 223–237, Sep. 2018, doi: 10.1016/j.ins.2018.05.046.

[67] J. Soria-Comas and J. Domingo-Ferrer, "Probabilistic k-anonymity through microaggregation and data swapping," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Jun. 2012, pp. 1–8.

[68] J. Bondia-Barcelo, J. Castella-Roca, and A. Viejo, "Building privacy-preserving search engine query logs for data monetization," in *Proc. Int. IEEE Conf. Ubiquitous Intell. Comput., Adv. Trusted Comput., Scalable Comput. Commun., Cloud Big Data Comput., Internet People, Smart World Congr. (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, Toulouse, France, Jul. 2016, pp. 390–397, doi: 10.1109/UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld.2016.0074.

[69] H. F. Nissenbaum and H. Daniel, *Trackmenot: Resisting Surveillance in Web Search. Lessons From the Identity Trail: Anonymity, Privacy, and Identity in a Networked Society*. Oxford, U.K.: Oxford Univ. Press, 2009. [Online]. Available: https://ssrn.com/abstract=2567412

[70] J. Domingo-Ferrer, A. Solanas, and J. Castellà-Roca, "h(k)-private information retrieval from privacy-uncooperative queryable databases," *J. Online Inf. Rev.*, vol. 33, no. 4, pp. 1468–1527, 2009.

[71] M. Murugesan and C. Clifton, "Providing privacy through plausibly deniable search," in *Proc. SIAM Int. Conf. Data Mining*, Sparks, NV, USA, Apr./May2009, pp. 768–779, doi: 10.1137/1.9781611972795.66.

[72] A. Arampatzis, P. S. Efraimidis, and G. Drosatos, "A query scrambler for search privacy on the Internet," *Inf. Retr.*, vol. 16, no. 6, pp. 657–679, Dec. 2013, doi: 10.1007/s10791-012-9212-1.

[73] E. Balsa, C. Troncoso, and C. Diaz, "OB-PWS: Obfuscation-based private Web search," in *Proc. IEEE Symp. Secur. Privacy*, San Francisco, CA, USA, May 2012, pp. 491–505, doi: 10.1109/SP.2012.36.

[74] A. Viejo, J. Castella-Roca, O. Bernado, and J. M. Mateo-Sanz, "Single-party private Web search," in *Proc. 10th Annu. Int. Conf. Privacy, Secur. Trust*. Washington, DC, USA: IEEE Computer Society, Jul. 2012, pp. 1–8, doi: 10.1109/PST.2012.6297913.

[75] P. Papadopoulos, A. Papadogiannakis, M. Polychronakis, A. Zarras, T. Holz, and E. P. Markatos, "K-subscription: Privacy-preserving microblogging browsing through obfuscation," in *Proc. 29th Annu. Comput. Secur. Appl. Conf. (ACSAC)*, New Orleans, LA, USA, Dec. 2013, pp. 49–58, doi: 10.1145/2523649.2523671.

[76] A. Petit, T. Cerqueus, S. B. Mokhtar, L. Brunie, and H. Kosch, "PEAS: Private, efficient and accurate Web search," in *Proc. IEEE Trustcom/BigDataSE/ISPA*, Helsinki, Finland, vol. 1, Aug. 2015, pp. 571–580, doi: 10.1109/Trustcom.2015.421.

[77] R. Chow and P. Golle, "Faking contextual data for fun, profit, and privacy," in *Proc. 8th ACM Workshop Privacy Electron. Soc. (WPES)*, 2009, pp. 105–108.

[78] S. T. Peddinti and N. Saxena, "On the privacy of Web search based on query obfuscation: A case study of TrackMeNot," in *Proc. 10th Int. Conf. Privacy Enhancing Technol. (PETS)*, 2010, pp. 19–37.

[79] R. Al-Rfou', W. Jannen, and N. Patwardhan, "TrackMeNot-so-good-after-all," 2012, pp. 1–8, *arXiv:1211.0320*. [Online]. Available: http://arxiv.org/abs/1211.0320

[80] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for Web transactions," *ACM Trans. Inf. Syst. Secur.*, vol. 1, no. 1, pp. 66–92, Nov. 1998.

[81] J. Castellà-Roca, A. Viejo, and J. Herrera-Joancomartí, "Preserving user's privacy in Web search engines," *Comput. Commun.*, vol. 32, nos. 13–14, pp. 1541–1551, Aug. 2009, doi: 10.1016/j.comcom.2009.05.009.

[82] Y. Lindell and E. Waisbard, "Private Web search with malicious adversaries," in *Proc. 10th Int. Conf. Privacy Enhancing Technol. (PETS)*, 2010, pp. 220–235.

[83] A. Viejo and J. Castellà-Roca, "Using social networks to distort users' profiles generated by Web search engines," *Comput. Netw.*, vol. 54, no. 9, pp. 1343–1357, Jun. 2010.

[84] A. Erola, J. Castellà-Roca, A. Viejo, and J. M. Mateo-Sanz, "Exploiting social networks to provide privacy in personalized Web search," *J. Syst. Softw.*, vol. 84, no. 10, pp. 1734–1745, Oct. 2011.

[85] C. Romero-Tris, A. Viejo, and J. Castella-Roca, "Improving query delay in private Web search," in *Proc. Int. Conf. P2P, Parallel, Grid, Cloud Internet Comput.*, Oct. 2011, pp. 200–206.

[86] C. Romero-Tris, J. Castellà-Roca, and A. Viejo, "Multi-party private Web search with untrusted partners," in *Proc. 7th Int. ICST Conf. Secur. Privacy Commun. Netw. (SecureComm)*, 2011, pp. 261–280.

[87] G. Danezis and C. Diaz, "A survey of anonymous communication channels," Microsoft, Redmond, WA, USA, Tech. Rep. MSR-TR-2008-35, Feb. 2008. [Online]. Available: https://www.microsoft.com/en-us/research/publication/a-survey-of-anonymous-communication-channels/

[88] G. Danezis, C. Diaz, and P. Syverson, "Systems for anonymous communication," in *CRC Cryptography and Network Security Series*. London, U.K.: Chapman & Hall, 2009.

[89] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, vol. 24, no. 2, pp. 84–90, Feb. 1981, doi: 10.1145/358549.358563.

[90] B. N. Levine and C. Shields, "Hordes: A multicast based protocol for anonymity1," *J. Comput. Secur.*, vol. 10, no. 3, pp. 213–240, Jul. 2002.

[91] O. Berthold, H. Federrath, and S. Köpsell, "Web MIXes: A system for anonymous and unobservable Internet access," in *Proc. Int. Workshop Designing Privacy Enhancing Technol., Design Issues Anonymity Unobservability*. New York, NY, USA: Springer-Verlag, 2001, pp. 115–129. [Online]. Available: http://dl.acm.org/citation.cfm?id=371931.371983

[92] B. Westermann, R. Wendolsky, L. Pimenidis, and D. Kesdogan, "Cryptographic protocol analysis of AN.ON," in *Proc. 14th Int. Conf. Financial Cryptogr. Data Secur. (FC)*. Berlin, Germany: Springer-Verlag, 2010, pp. 114–128, doi: 10.1007/978-3-642-14577-3_11.

[93] R. Bohme, G. Danezis, C. Diaz, S. Kapsell, and A. Pfitzmann, "Mix cascades vs. peer-to-peer: Is one concept superior?" Privacy Enhancing Technol., Toronto, ON, Canada, Tech. Rep., 2004.

[94] D. Goldschlag, M. Reed, and P. Syverson, "Onion routing for anonymous and private Internet connections," *Commun. ACM*, vol. 42, no. 2, pp. 39–41, 1999.

[95] F. Saint-Jean, A. Johnson, D. Boneh, and J. Feigenbaum, "Private Web search," in *Proc. ACM Workshop Privacy Electron. Soc. (WPES)*, 2007, pp. 84–90.

[96] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *Proc. 13th Conf. USENIX Secur. Symp. (SSYM)*, vol. 13. Berkeley, CA, USA: USENIX Association, 2004, p. 21. [Online]. Available: http://dl.acm.org/citation.cfm?id=1251375.1251396

[97] P. Syverson, "Practical vulnerabilities of the tor anonymity network," in *Advances in Cyber Security: Technology, Operation, and Experiences*. New York, NY, USA: Fordham Univ. Press, 2011.

[98] F. Astolfi, J. Kroese, and J. Van Oorschot, "I2p—The invisible Internet project," Media Technol., Leiden Univ., Leiden, The Netherlands, Web Technol. Rep., 2015.

[99] B. Chor, N. Gilboa, and M. Naor, *Private Information Retrieval By Keywords*. Milwaukee, WI, USA: IEEE, 1997.

[100] E. Kushilevitz and R. Ostrovsky, "Replication is not needed: Single database, computationally-private information retrieval," in *Proc. 38th Annu. Symp. Found. Comput. Sci.* Piscataway, NJ, USA: IEEE Press, 1997, pp. 364–373.

[101] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, ''Private information retrieval,'' *J. ACM*, vol. 45, no. 6, pp. 965–981, 1998.

[102] R. Ostrovsky and W. E. Skeith, III, ''A survey of single-database private information retrieval: Techniques and applications,'' in *Public Key Cryptography—PKC* (Lecture Notes in Computer Science), vol. 4450. Berlin, Germany: Springer, 2007, pp. 393–411.

[103] L. F. Cranor, P. Guduru, and M. Arjula, ''User interfaces for privacy agents,'' *ACM Trans. Comput.-Hum. Interact.*, vol. 13, no. 2, pp. 135–178, Jun. 2006, doi: 10.1145/1165734.1165735.

[104] L. F. Cranor, S. Egelman, S. Sheng, A. M. McDonald, and A. Chowdhury, ''P3P deployment on Websites,'' *Electron. Commerce Res. Appl.*, vol. 7, no. 3, pp. 274–293, Sep. 2008.

[105] C. Soghoian, ''The history of the do not track header,'' Slight Paranoia, Washington, DC, USA, Tech. Rep., Feb. 2012.

[106] J. Mayer, A. Narayanan, and S. Stamm, *Do Not Track: A Universal Third-Party Web Tracking Opt Out*, document draft-mayer-do-not-track-00, IETF Request for Comments, 2011, pp. 1–12.

[107] M. Beck and M. Marhfer, ''Do-not-track techniques for browsers and their implications for consumers,'' in *Privacy and Identity Management for Life* (IFIP Advances in Information and Communication Technology), vol. 375, J. Camenisch, B. Crispo, S. Fischer-Hübner, R. Leenes, and G. Russello, Eds. Berlin, Germany: Springer, 2012, pp. 187–196, doi: 10.1007/978-3-642-31668-5_14.

[108] O. Tene and J. Polenetsky, ''To track or 'do not track': Advancing transparency and individual control in online behavioral advertising,'' *Minnesota J. Law, Sci. Technol.*, vol. 13, no. 1, p. 281, 2012.

[109] H. Hochheiser, ''The platform for privacy preference as a social protocol: An examination within the U.S. Policy context,'' *ACM Trans. Internet Technol.*, vol. 2, no. 4, pp. 276–306, Nov. 2002, doi: 10.1145/604596.604598.

[110] I. Reay, S. Dick, and J. Miller, ''A large-scale empirical study of P3P privacy policies: Stated actions vs. Legal obligations,'' *ACM Trans. Web*, vol. 3, no. 2, pp. 6:1–6:34, Apr. 2009, doi: 10.1145/1513876.1513878.

[111] Electronic Privacy Information Center (EPIC). (Jun. 2000). *Pretty Poor Privacy: An Assessment of P3P and Internet Privacy*. [Online]. Available: http://epic.org/reports/prettypoorprivacy.html

[112] X. Shen, B. Tan, and C. Zhai, ''Ucair: Capturing and exploiting context for personalized search,'' in *Proc. ACM SIGIR Workshop Inf. Retr. Context (IRiX)*. Salvador, Brazil: Citeseer, 2005, p. 45.

[113] K. W.-T. Leung, D. L. Lee, W. Ng, and H. Y. Fung, ''A framework for personalizing Web search with concept-based user profiles,'' *ACM Trans. Internet Technol.*, vol. 11, no. 4, pp. 17:1–17:29, Mar. 2012, doi: 10.1145/2109211.2109214.

[114] T. Kramár, M. Barla, and M. Bieliková, ''Personalizing search using socially enhanced interest model, built from the stream of user's activity,'' *J. Web Eng.*, vol. 12, nos. 1–2, pp. 65–92, Feb. 2013. [Online]. Available: http://dl.acm.org/citation.cfm?id=2481562.2481565

DAVID PÀMIES-ESTREMS (Student Member, IEEE) received the B.E. degree in computer science from the Polytechnic University of Catalonia, in 2001, the M.B.A. degree from Rovira i Virgili University, in 2007, and the M.Sc. degree in information and communication technologies security from the Open University of Catalonia, in 2015. He is currently pursuing the Ph.D. degree with Rovira i Virgili University. He founded three technology companies that have been awarded, worked in technological innovation for the XarxaIT of the Generalitat of Catalonia, and participated in several European Commission funded research projects. His research focuses on the fields of cryptography and privacy.

JORDI CASTELLÀ-ROCA (Member, IEEE) received the degree in computer systems from the University of Lleida, in 1998, the degree in computer science from Rovira i Virgili University, in 2000, and the Ph.D. degree in computer science from the Autonomous University of Barcelona, in 2005. He is currently a tenured Associate Professor with Rovira i Virgili University. He is also a member of the UNESCO Chair in Data Privacy. He has published over 80 works, is coauthor of seven patents, and has participated in 39 research projects (main researcher in 19 of them). His research focuses on the fields of cryptography and privacy.

JOAQUIN GARCIA-ALFARO (Senior Member, IEEE) received the Double Ph.D. Diploma degree in computer science from the Autonomous University of Barcelona and the University of Rennes, and the Habilitation degree from Université Sorbonne VI (Pierre et Marie Curie). He is currently a Full Professor with the Networks and Telecommunication Services Department, Télécom SudParis (Institut Polytechnique de Paris), France, and an Adjunct Research Professor with Carleton University, Ottawa, Canada. He is involved in several research projects at National and European levels, related to ICT security. His research interests include a wide range of information security problems, with an emphasis on the management of security policies, analysis of vulnerabilities, and enforcement of countermeasures.