On the self-adjustment of privacy safeguards for query log streams

David Pàmies-Estrems[a], Joaquin Garcia-Alfaro[b]

[a] *Aneris, Reus, E-43201, Spain*
[b] *SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, Palaiseau, 91120, France*

A R T I C L E   I N F O

Dataset link: https://github.com/dpamies/saps-qls

Keywords:
Internet query service
Query log
Data stream
Anonymization
Privacy
Utility

A B S T R A C T

Internet-based services process and store numerous search queries around the globe. The use of web search engines, such as Bing and Google, as well as personal assistants (e.g., Alexa and Cortana) and task specific systems (e.g., YouTube, Netflix, Amazon) are relevant examples. The queries associated to such services may be stored and sold out for profit. Before doing so, personal and sensitive information must be sanitized, as requested by current regulations. This can be cumbersome for some organizations. We present an automated solution for anonymizing unstructured data, like the one used within query logs. Our solution uses a light-weight probabilistic k-anonymity approach, which allows verifiable real-time privacy protection. It addresses previous limitations and improves performance. We validate the feasibility of the approach, under some evaluation metrics including data utility, privacy and speed.

1. Introduction

Internet query services are the underlying bricks of web search engines (such as Google, Bing, Qwant), as well as personal assistants (Alexa, Cortana, Google Assistant), and task specific systems (YouTube, Netflix, Amazon). They help people to solve a plethora of tasks related to work, everyday chores, leisure activities and even getting decisions. Such services do not just return the list of results. When a search is conducted, they also store the unstructured version of the user queries along with metadata, such as timestamps, location, and so on. The extra information collected during the query, coupled with the search keywords themselves, are referred hereinafter as *query log*. With the intention of being more and more useful to the end user, each Service analyzes several streams of query logs to gain a better understanding of the users. This is useful in order to enhance their experience, i.e., making it easy to find the desired information with fewer keywords. Two main ways to obtain this goal are *personalization* and *usability*. Indeed, terms in a query can have multiple interpretations, which can make it difficult to determine the intended meaning. Using previous queries made by the user can help clarify and resolve ambiguity in future queries (Shen et al., 2007; Xu et al., 2007a). This allows the Internet Service to prioritize relevant results (such as URLs, addresses or products) and display them among the first results. On the other hand, internet query Services use the frequency and chosen results of common queries to enhance their ranking algorithms (Agichtein et al., 2006). Additionally, this data can be utilized to provide alternative query recommendations (Jones et al., 2006). These suggestions can help to correct typos, refine the original query, as well as to offer similar queries that return additional results.

Search data can provide valuable insights into customer intent and other factors (Cameron and Pickersgill, 2014). It can also be used for various purposes by either the Internet Service or a third party, including *Marketing* and *Research*. Query logs can be analyzed to evaluate and improve the results of an advertising campaign. The characteristics of a user, such as gender, age, income, and education, can be determined through their query logs and used to assess the impact of advertisements on the target audience's interests and behavior (Brenes and Gayo-Avello, 2009; Poblete et al., 2007). Query logs can also reveal market trends (Korolova et al., 2009). In terms of research, e.g., when dealing with the field of Information Retrieval, one can focus on testing and studying new IR algorithms (Bar-Ilan, 2007, May 2007), understanding user information needs and query formulation (Korolova et al., 2009), examining language use in queries (Shea, 2010), and exploring other topics (Erola et al., 2011a; Silvestri, 2010).

Privacy concerns may arise when utilizing query logs. Each log contains a user identifier, the search query, time of search, and selected results. This information can reveal the behavior of users, as well as any other interests (e.g., religious believes or sexual orientation). Query keywords can also contain identifiers and quasi-identifiers that can link the queries to real people (Willenborg and De Waal, 2012). This is especially true given the trend of vanity search and ego-surfing, in which people search for their own names online (Soghoian, 2007), or when searching for the home address. Protection of query logs before re-

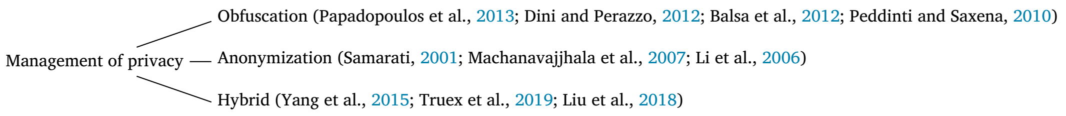

Management of privacy
Obfuscation (Papadopoulos et al., 2013; Dini and Perazzo, 2012; Balsa et al., 2012; Peddinti and Saxena, 2010)
Anonymization (Samarati, 2001; Machanavajjhala et al., 2007; Li et al., 2006)
Hybrid (Yang et al., 2015; Truex et al., 2019; Liu et al., 2018)

Fig. 1. Existing solutions in the literature to handle privacy management.

lease to third parties is important to prevent anonymity issues. Poor protection can result in serious breaches (Jones et al., 2007; Poblete et al., 2007), as demonstrated by the famous AOL case in which 36 million records were publicly released. Despite being anonymized, the AOL case showed that log correlation techniques (Barbaro et al., 2006) were enough to conduct user identification, causing harm to user privacy, as well as damage to corporate reputation and legal actions (Foundation, 2009; Mills, 2006; Hansell, 2006).

Our work tackles the problems related to query log protection and user privacy. We propose an improved anonymization technique to secure query logs on the server side for Internet Services seeking to monetize them while complying with current privacy regulations. Merely hiding user identifiers or replacing them with random data is insufficient (Europe, 2016). A reliable anonymization method, such as Statistical Disclosure Control (Willenborg and De Waal, 2012), must be applied to guarantee limited disclosure risks (Hundepool et al., 2010). A traditional approach to achieve this is by performing a k-anonymity process on the server-side before releasing the query logs. The data release satisfies the k-anonymity privacy property when user data in the query logs cannot be distinguished from at least $k-1$ other users in the release (De Capitani di Vimercati et al., 2011). Our proposed method aims at addressing limitations faced by classical k-anonymity proposals when dealing with unstructured data, particularly in the context of Internet Services that require real-time processing. We overcome this problem by adopting a probabilistic k-anonymity processing of queries. This allows us to limit the degree of disclosure risk associated to personal user information, while efficiently handling unstructured data, in real-time. The idea is as follows. Queries from different users that share similar interests are combined, in a way to ensuring that individual identification does not exceed a given threshold, say $\frac{1}{k}$, where k represents the total number of users with similar interests. This way, users may interchange queries with common interests, reducing the risk of putting in danger their privacy. Moreover, our solution allows internet query services to maintain the original query logs in their raw form, instead of destroying the data, as the anonymized versions are generated in real-time. This provides flexibility in handling data: they can either retain the original logs and the anonymized ones, or choose to keep only the anonymized versions, reducing significantly the risk of information disclosure in the event of a breach. The anonymized logs can be released to third parties without further modifications, with enhanced user privacy while preserving the utility of the data for analysis and other purposes.

To sum up, our main contributions consist of:

- We present an improved anonymity technique to process query logs using probabilistic k-anonymity over unstructured data streams in real time using the Server-side infrastructure.
- Our solution uses a light-weight technique that allows verifiable real-time privacy protection.
- We address previous limitations and improve performance.
- Our system is able to achieve a very wide application range, which adapts to different volumes of data and with a high initial throughput.
- The proposal is able to publish the queries contained in the original logs, without the need to make major modifications.
- The approach is parameterizable, in a way that we can differentiate different levels of utility and privacy, while reducing resource consumption.
- We validate the feasibility of the approach, under some evaluation metrics including data utility, privacy and speed.

The paper is structured as follows. Section 2 surveys related work. Section 3 outlines our proposal. Section 4 explains the architecture and necessary components. Section 5 presents our experimental results. Section 6 closes the paper with some conclusions and perspectives of future work.

2. Related work

Management of privacy is all about safeguarding sensitive information against public access from unauthorized parties. Three main research privacy management branches often listed in the literature are summarized in Fig. 1, with some representative publications per branch. Our work is related to the anonymization category. More specifically, our work is on anonymization solutions for Internet Services, in which we assume services aiming at achieving some degree of monetization, e.g., via the commercialization of protected query logs. Therefore, our objective is to anonymize available data while minimizing information loss before releasing it to third-party organizations.

Majeed et al. (2022) present a comprehensive analysis of anonymization mechanisms recently proposed in the literature, in order to preserve both privacy and utility in data publishing. Different categories are proposed, including (i) information privacy (e.g., privacy-preserving techniques to collect, store, analyze, process and release personal data in a privacy-friendly manner), (ii) communication privacy (e.g., maintenance of privacy during the exchange of sensitive data via digital technologies), and (iii) territorial privacy (e.g., assurance of privacy while respecting socio-cultural policies from different countries).

The aforementioned approaches can be categorized in either theoretical studies (i.e., solutions not yet deployed to real-world scenarios, with very limited experimentation) or practical solutions (i.e., deployed solutions tested over real-world case scenarios, in which evaluations are performed using real-world datasets). Our work enters under the second category, i.e., practical solutions, in which attention is also paid to the use of benchmarking metrics (e.g., privacy and utility degrees). Related work with this goal in mind can be classified based on the type of input they process, such as fixed-length (e.g. block-based) or data-stream inputs, as shown in Fig. 2.

Based on existing classifications in Romero-Tris (2014), Romero-Tris et al. (2015), Erola (2013), Ullah et al. (2023), two main actors can be identified: Users and Services. A first set of proposals, which includes ours, protect users' privacy on the Service side without further user intervention being asynchronous and transparent to the user. A second set includes approaches that protect user privacy without Service's cooperation, and a third set involves approaches that require some level of cooperation between Users and Services. These latter approaches are not considered as server-side, as users must actively participate in the process and quickly detect any lack of Service cooperation.

2.1. *Survey on client side proposals*

WSEs may lack enough motivation to safeguard user privacy, leaving the responsibility of data protection solely on the users themselves. Under this assumption, certain methods of protection can be identified that don't require any collaboration between WSEs and users. These methods fall into two main categories: i) obfuscation techniques and ii) anonymous channels. Obfuscation techniques introduce noise to distort

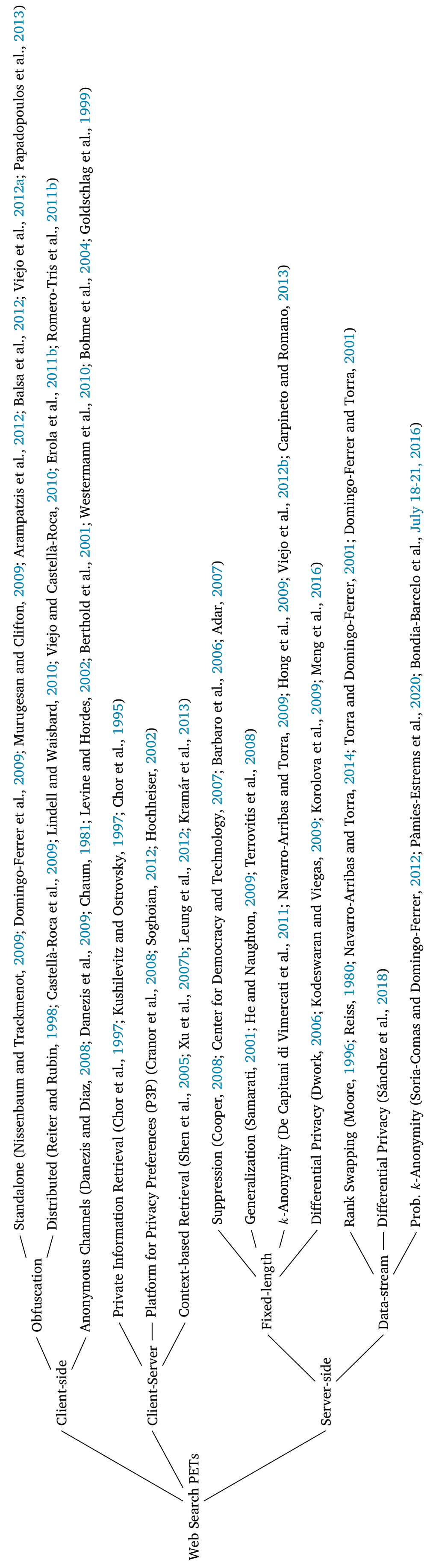

Web Search PETs
Client-side
Obfuscation
Standalone (Nissenbaum and Trackmenot, 2009; Domingo-Ferrer et al., 2009; Murugesan and Clifton, 2009; Arampatzis et al., 2012; Balsa et al., 2012; Viejo et al., 2012a; Papadopoulos et al., 2013)
Distributed (Reiter and Rubin, 1998; Castellà-Roca et al., 2009; Lindell and Waisbard, 2010; Viejo and Castellà-Roca, 2010; Erola et al., 2011b; Romero-Tris et al., 2011b)
Anonymous Channels (Danezis and Diaz, 2008; Danezis et al., 2009; Chaum, 1981; Levine and Hordes, 2002; Berthold et al., 2001; Westermann et al., 2010; Bohme et al., 2004; Goldschlag et al., 1999)
Client-Server
Private Information Retrieval (Chor et al., 1997; Kushilevitz and Ostrovsky, 1997; Chor et al., 1995)
Platform for Privacy Preferences (P3P) (Cranor et al., 2008; Soghoian, 2012; Hochheiser, 2002)
Context-based Retrieval (Shen et al., 2005; Xu et al., 2007b; Leung et al., 2012; Kramár et al., 2013)
Server-side
Fixed-length
Suppression (Cooper, 2008; Center for Democracy and Technology, 2007; Barbaro et al., 2006; Adar, 2007)
Generalization (Samarati, 2001; He and Naughton, 2009; Terrovitis et al., 2008)
k-Anonymity (De Capitani di Vimercati et al., 2011; Navarro-Arribas and Torra, 2009; Hong et al., 2009; Viejo et al., 2012b; Carpineto and Romano, 2013)
Differential Privacy (Dwork, 2006; Kodeswaran and Viegas, 2009; Korolova et al., 2009; Meng et al., 2016)
Data-stream
Rank Swapping (Moore, 1996; Reiss, 1980; Navarro-Arribas and Torra, 2014; Torra and Domingo-Ferrer, 2001; Domingo-Ferrer and Torra, 2001)
Differential Privacy (Sánchez et al., 2018)
Prob. *k*-Anonymity (Soria-Comas and Domingo-Ferrer, 2012; Pàmies-Estrems et al., 2020; Bondia-Barcelo et al., July 18-21, 2016)

Fig. 2. Anonymization solutions, classified based on the type of input they process.

the user's profile maintained by the WSEs, while anonymous channels rely on an infrastructure to manage the profiling of user activities between users and WSEs. However, the utilization of client-side techniques is presumed to generate unrealistic profiles, potentially having a negative impact on the services offered by WSEs.

2.1.1. *Obfuscation techniques*

Initially, techniques relied on the inclusion of random queries, also known as fake queries, to obscure users' profiles. The key requirement for these random queries was to be indistinguishable from genuine queries, a property referred to as unobservability. Representative solutions employing obfuscation techniques can be categorized based on the number of users involved in the protocol. There are standalone solutions where individual users take responsibility for safeguarding their privacy from WSEs independently. On the other hand, distributed solutions involve groups of users collaborating to protect the privacy of each individual in the group. Let's explore some examples for each category.

Standalone Systems — These systems utilize synthetic queries to conceal the actual ones made by users (Nissenbaum and Trackmenot, 2009; Domingo-Ferrer et al., 2009; Murugesan and Clifton, 2009; Arampatzis et al., 2012; Balsa et al., 2012; Viejo et al., 2012a; Papadopoulos et al., 2013; Sánchez et al., 2013; Petit et al., 2015). Alongside real queries, these synthetic queries are submitted, thereby obscuring the user profiles managed by the WSEs. When the synthetic queries bear some semantic relevance to the user's queries, the obfuscated profile remains functional, allowing the WSE to personalize the user's search results effectively. However, when the synthetic queries are semantically unrelated to the user's queries, the resulting profile becomes heterogeneous, leading to less accurate personalization. It is worth noting that neither approach is inherently superior, as users may have different preferences regarding the trade-off between privacy and utility. Certain studies have demonstrated the possibility of differentiating between real and synthetic queries (Chow and Golle, 2009; Peddinti and Saxena, 2010; Al-Rfou' et al., 2012; Balsa et al., 2012). These studies rely on the notion that machine-generated queries possess distinct features compared to human-generated queries.

Distributed Systems — These approaches necessitate the cooperation of a collective of users working together to safeguard their privacy, effectively concealing their actions within the activities of numerous others (Reiter and Rubin, 1998; Castellà-Roca et al., 2009; Lindell and Waisbard, 2010; Viejo and Castellà-Roca, 2010; Erola et al., 2011b; Romero-Tris et al., 2011b, 2011a; Ullah et al., 2016a, 2016b). Typically, these techniques involve placing users into a large group where they submit requests on behalf of other group members, exchanging their queries. Personalization becomes feasible only when the group members share similar interests (Romero-Tris, 2014), however some proposals group members with different interests (Ullah et al., 2022). In certain proposals (Reiter and Rubin, 1998; Castellà-Roca et al., 2009; Lindell and Waisbard, 2010; Ullah et al., 2019, 2021, 2022), a central node is introduced, acting as a bottleneck that could affect the overall system performance. Alternatively, in other instances, a specific path (Reiter and Rubin, 1998; Viejo and Castellà-Roca, 2010; Erola et al., 2011b; Romero-Tris et al., 2011b,a) is created for query submission, or a dedicated group of users must be established (Reiter and Rubin, 1998; Castellà-Roca et al., 2009; Lindell and Waisbard, 2010). In both cases, a notable delay is introduced (Romero-Tris, 2014).

2.1.2. *Anonymous channels*

The proposals falling into this category rely on anonymous infrastructures (Danezis and Diaz, 2008; Danezis et al., 2009) to transmit users' queries to the WSE. This approach aims to hide the users' identities associated with their queries, thereby preventing WSEs from profiling individual users. However, hiding users' identity may have an impact on the quality of service that WSEs can provide to the users.

Chaum's mix networks (Chaum, 1981) exemplify solutions categorized as anonymous channels. In these networks, messages traverse through multiple nodes, and each node uses cryptography to disassociate input messages from output messages (Danezis and Diaz, 2008; Danezis et al., 2009). More advanced techniques involve the utilization of proxies (Levine and Hordes, 2002), which handle connections (such as queries) from users to recipients (e.g., WSEs). The fundamental idea is that proxies deliver messages while keeping the source (e.g., the user's identity) undisclosed. Prominent examples of services utilizing proxy-like infrastructures include DuckDuckGo,[1] Startpage,[2] and Yippy.[3] By adopting these solutions, users transfer their trust from WSEs to the proxies, assuming that the proxies do not monitor or log their traffic.

Web MIXes (Berthold et al., 2001) offer real-time Internet access with anonymity and unobservability features. They integrate an authentication mechanism to thwart flood attacks, and users are provided with a feedback interface to inform them about their current level of protection. Nonetheless, there are certain weaknesses in their authentication process that could potentially enable external attackers to carry out replay attacks (Westermann et al., 2010). Furthermore, the synchronous nature of Web MIXes might lead to difficulties when handling asynchronous TCP/IP networks (Bohme et al., 2004).

In the literature, there have been proposals to utilize onion routing (Goldschlag et al., 1999) for creating anonymous channels in the context of queries and WSEs (Saint-Jean et al., 2007). User-friendly solutions based on the onion routing paradigm include general-purpose plugins and modified web-browsers[4] using the TOR Project (Dingledine et al., 2004). However, several weaknesses have been identified (Syverson, 2011). TOR does not claim to provide security against passive global adversaries (Danezis et al., 2009). Additionally, the Invisible Internet Project (I2P) (Astolfi et al., 2015) constructs an anonymous network layer designed for anonymous communication purposes.

2.2. *Survey on collaborative client-server proposals*

In this category of solutions, it is presumed that users and servers collaborate to safeguard users' privacy. Below, we present this kind of solutions, organized into three primary groups: i) Private Information Retrieval, ii) Platform for Privacy Preferences (P3P), and iii) Context-based Retrieval.

2.2.1. *Private information retrieval*

Private Information Retrieval (PIR) schemes (Chor et al., 1997; Kushilevitz and Ostrovsky, 1997; Chor et al., 1995; Ostrovsky and Skeith-III, 2007; Khan et al., 2019) allow users to retrieve information from a database confidentially, ensuring that the server remains unaware of the specific information accessed. With a PIR scheme, users can search for documents stored in the database and retrieve those that interest them. The problem of submitting a query to a WSE while preserving the user's privacy is equivalent to the PIR problem. Notice that PIR schemes encounter two practical issues that render them unsuitable for WSEs (Castellà-Roca et al., 2009): Firstly, they are not well-suited for handling large databases, and secondly, they assume that users possess the precise locations of the records they want to retrieve. Moreover, PIR schemes have been reported vulnerable to machine learning attacks in recent literature, in which malicious WSE may succeed at correctly associating incoming queries to user profiles (Khan et al., 2020, 2021, 2023). Techniques to handle the problem, common to many other privacy-enhancing techniques, exist (Torra, 2023).

2.2.2. *Platform for privacy preferences (P3P)*

The World Wide Web Consortium (W3C) introduced the Platform for Privacy Preferences (P3P) (Cranor et al., 2006, 2008) with the aim of simplifying the process for users to access information about the privacy policies of websites they visit. P3P serves as a framework that allows users to automate the protection of their privacy by defining their own privacy preferences. If a website does not adhere to these preferences, P3P-enabled browsers can alert the user and even take predefined actions, such as denying access to cookies. An example of a policy-based P3P system is the Do-Not-Track initiative (Soghoian, 2012), where HTTP headers request web applications not to track users. For this to be effective, the web application must comply with P3P standards. Although it has been studied extensively and standardized by the W3C, it is now considered an outdated protocol. P3P-like solutions have faced criticism due to the potential impact of governmental laws on users' privacy (Hochheiser, 2002), the lack of website compliance with privacy-protection mandates in their legal jurisdictions (e.g., challenges in enforcing privacy policies) (Reay et al., 2009), and the limited number of potential adopters (Electronic Privacy Information Center, 2000).

2.2.3. *Context-based retrieval*

Context-based retrieval proposals aim to store user profiles, such as search history, on the client's machine. This information allows for the identification of users' interests, enabling the re-ranking of search results based on these preferences. In this process, both the WSE and users collaborate to obtain the final search results. The WSE receives the query and returns the results, which are then re-ranked at the client-side. The User-Centered Adaptive Information Retrieval (UCAIR) project (Shen et al., 2005) is an example of such a scheme, where available user context from submitted queries and clicked results is collected and utilized. Other similar schemes permit users to determine the content and level of detail of their profiles exposed to the WSE (Xu et al., 2007b; Leung et al., 2012; Kramár et al., 2013). Users have control over the profile content that is disclosed to the WSE when they submit a query, and they can adjust parameters associated with stored profiles to enhance result quality. However, there are potential disadvantages to these proposals. The performance and effectiveness of ranking results at the client-side may be limited compared to ranking results at the server-side (Shen et al., 2005). Additionally, despite these approaches, it is still expected that WSEs can profile users after multiple executions of the approach.

2.3. *Survey on server-side proposals*

Solutions under this category assume that the Service is the only part that is working to protect users' privacy. Solutions under this category are organized in two main groups: i) fixed-length input and ii) data-stream input.

2.3.1. *Fixed-length inputs*

For fixed-length inputs, current anonymization methods use a set of fixed and unchanging data structures. The data to be anonymized is contained within these structures. The protection of the entire dataset involves analyzing all elements in the dataset first and then processing them. This approach is demonstrated by several well-known solutions, which we discuss next.

Suppression — The anonymization process for a fixed-length input involves removing elements that may potentially reveal sensitive information, either individually or in combination. The analysis of the data set employs statistical or semantic methods to determine which elements need to be removed.

Examples of suppression in the context of query log anonymization can be found in related literature (Cooper, 2008). This includes the removal of identifiers such as social security numbers, addresses, bank

[1] https://duckduckgo.com/.

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

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

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

accounts, or any other identification data related to the user (Center for Democracy and Technology, 2007). However, the AOL incident highlights the limitations of this method (Foundation, 2009; Mills, 2006; Hansell, 2006). The presence of quasi-identifiers in the AOL dataset and the difficulty in identifying their combinations were enough to re-identify AOL users through traditional log correlation techniques (Barbaro et al., 2006).

The elimination of uncommon queries is another method used in anonymizing query logs (Adar, 2007). It targets the removal of queries that may contain identifying or quasi-identifying information. To implement this, threshold values must be defined. However, this task can be complex and prone to errors, as many queries only appear a limited number of times (Beitzel et al., 2004). This method can be combined with selecting queries resulting from clicking on common URLs (Korolova et al., 2009) or using graph theory to represent query logs (Poblete et al., 2007). In the latter, nodes represent user queries and are connected if their clicked URL sets intersect. The anonymization process involves removing queries that return less than k documents, with those queries that significantly contribute to the query graph be-ing considered vulnerable and removed.

Generalization — An approach to anonymization is based on the generalization of domain relationships, by analyzing attribute values. Minimal generalization aims to minimize distortion of processed data (Samarati, 2001). Top-down approaches using lexical and semantic databases for general purpose generalizations have been proposed (He and Naughton, 2009; Terrovitis et al., 2008). These convert groups of queries into common abstractions (e.g. *football* and *tennis* to *sports*), to make users indistinguishable. However, the main limitations are building generic dictionaries for words and concepts to anonymize, and possible language adaptations for original datasets.

k-Anonymity — The concept of k-anonymity (De Capitani di Vimercati et al., 2011) reduces the risk of record-linkage by ensuring that each record is indistinct from at least $k - 1$ other records. This means that no individual can be re-identified with a probability greater than $\frac{1}{k}$ only through linking attacks.

Methods of Statistical Disclosure Control (SDC) are used to create anonymous query logs while minimizing query deletion (Navarro-Arribas and Torra, 2009; Hong et al., 2009). Queries are grouped by similarity, then rewritten as a prototype query, making them indistinguishable (Sánchez et al., 2013; Batet et al., 2013; Erola and Castellà-Roca, 2013; Navarro-Arribas et al., 2012; Batet et al., 2014). Users and queries are retained, but transformed to reduce disclosure risk. Fake messages can be generated to mix with legitimate ones (Viejo et al., 2012b), or infrequent queries can be masked using a more general frequent query (Carpineto and Romano, 2013) to achieve privacy comparable to k-anonymity.

Differential Privacy — It was introduced as a way to protect the identities of users participating in a dataset (Dwork, 2006). There are also interactive versions of this approach (Kodeswaran and Viegas, 2009). The original idea of differential privacy was to handle queries that access partial information in a dataset. However, if not managed carefully, these queries can still reveal information about the users. To address this issue, interactive improvements have been proposed to monitor how much information is being disclosed through these queries and to prevent them from returning information when a certain threshold is exceeded. Additionally, to further limit the risk of information disclosure, some proposals aim to restrict the statistics that are returned, such as query suggestions and spelling corrections (Korolova et al., 2009).

A technique is proposed in Meng et al. (2016) that selects high utility samples to be representative records in each cluster, with the goal of retaining more useful information and leaking less privacy. Other proposals (Zhang et al., 2015, 2016) suggest adding Laplacian noise to logs to preserve privacy, but adding noise also decreases data utility.

Table 1
Notation and symbols 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	:	Knowledge tree 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
$Q[v_x^h]$	:	Set of queries for vertex v_x^h
$U[v_x^h]$	:	Set of users for vertex v_x^h
$BQ[v_x^h]$	:	Set of queries for branch starting at vertex v_x^h
$BU[v_x^h]$	:	Set of users for branch starting at vertex v_x^h
γ_x^h	:	Category for vertex v_x^h
$\tau_{\ell,k}^*$	:	τ with a depth ℓ and width k

2.3.2. *Data-stream inputs*

This method enables processing data incrementally and does not require complete data to begin. It allows for partial data processing and results in prompt data output (Gaber et al., 2005). It also enables handling large and potentially infinite datasets, though protecting the privacy of vast data streams remains challenging (Krempl et al., 2014). Next, we review some notable solutions in this category.

Rank Swapping — The technique, initially applied to numerical variables (Moore, 1996), with earlier concepts in other fields (Reiss, 1980), involves exchanging data sorted by attribute value with a randomly selected nearest value in rank (Navarro-Arribas and Torra, 2014). There have also been other similar proposals (Torra and Domingo-Ferrer, 2001; Domingo-Ferrer and Torra, 2001) that only consider structured data.

Differential Privacy — This privacy method can be used to anonymize data streams (Sánchez et al., 2018). Instead of releasing the original query, a synthetic one is produced using semantic similarity. The unstructured nature of query logs, along with new terms not in the semantic database, may pose a challenge. Another challenge in using differential privacy for data streams is maintaining a fixed privacy level, which may limit the amount of data that can be published to preserve user privacy.

Probabilistic k-Anonymity — The probabilistic k-anonymity idea modifies the indistinguishability requirement of k-anonymity by requiring only the re-identification probability to be preserved (Soria-Comas and Domingo-Ferrer, 2012). This allows for better use of the data while still releasing logs with the original queries. However, the generalization of unstructured data elements leads to some imprecision in the generated profiles. The related literature also has limitations in terms of basic classification methods, resulting in low numbers of categories and high data utility loss or low initial throughput (Pàmies-Estrems et al., 2016; Pamies-Estrems et al., 2018; Pàmies-Estrems et al., 2020; Bondia-Barcelo et al., July 18-21, 2016).

3. Our proposal

This section presents our proposal for anonymization, including the notation used (as defined in Table 1), a formal definition of the data to be anonymized and its structure, an analysis of the privacy properties of the proposal, and a detailed explanation of the algorithms utilized in the anonymization process.

3.1. *Data structures*

We name query logs a stream of m registers, represented by:

$$R = \{r_0, .., r_m\} \quad (1)$$

where r_m is the last received query log.

Each query log, r_j, is composed of a user identifier u_i, a query q_j, and a classification c_g:

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

The user who submitted the query q_j to the Service is represented by u_i. The query q_j comprises a set of unstructured terms and is classified using a categorizer (refer to Section 4) to obtain its query classification, c_g. This classification is depicted as a path from a general category, γ_s^1, to a more specific category, γ_{s*}^h, and takes the following form:

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

A knowledge tree,[5] represented by τ, is utilized to construct this path. τ consists of edges $e_f \in E$ and vertices $v_x^h \in V$, where h is the depth and x is the width. Each vertex v_x^h in τ corresponds to a category γ_x^h and is connected to other categories via the edges. The categories (vertices) are more general closer to the roots $\{v_1^1 \dots v_x^1\}$, and become more specific as they are closer to the leaves $\{v_1^h \dots v_x^h\}$. Therefore, each query is assigned to a classification by placing it in one of the tree's vertices. The classification refers to the path between the root and the corresponding vertex, and comprises all the γ categories of the nodes that lie on this path.

The maximum depth of the hierarchy τ is determined by the distance or minimum path between the root and the farthest leaf, which is denoted as ℓ_{max}. The depth of each classification is determined by the number of terms it contains and can be as high as ℓ_{max}, the maximum depth of the hierarchy τ, but for testing purposes, we will use versions of the classification limited to depths of up to ℓ, with ℓ varying from 1 to ℓ_{max}.

Each vertex v_x^h in the tree structure contains two sets of data: a set of users $U[v_x^h]$, and a set of queries $Q[v_x^h]$. The maximum size of a set of users $U[v_x^h]$ is fixed at k as is defined using arity, but the size of the set of queries $Q[v_x^h]$ has no limit imposed as it is defined without using arity. This decision is constrained by the use of additional conditions (cf. Restrictions 3.2).

$$max \mid U[v_x^h] \mid = k \tag{4}$$

We refer to $\tau_{\ell,k}^*$ as the tree τ that has a depth of ℓ and a cardinality of $|U| = k$.

For each vertex v_x^h, the auxiliary sets $BU[v_x^h]$ and $BQ[v_x^h]$ are also defined, which respectively represent the full group of users and queries contained in the branch that starts at the vertex v_x^h. A summary of data structures used for tree construction is shown below:

$$\tau = < V, E > \tag{5}$$

$$V = \{v_1^1, \dots, v_z^\ell\}$$

$$E = \{e_1, \dots, e_g\}$$

$$v_x^h = \{U[v_x^h], BU[v_x^h], Q[v_x^h], BQ[v_x^h], \gamma_x^h\}$$

$$e_f = \{v_x^h, v_{x'}^{h+1}\}$$

3.2. Restrictions

Our proposal is subject to two additional restrictions (cf. Restrictions 1 and 2). Their use is the cause that the sets $U[v_x^h]$ and $Q[v_x^h]$ may have different sizes.

[5] We assume a data model structured as a tree, that allows us accumulating and disseminating knowledge of the real world. In other words, it allows us searching, adding, modifying, or deleting any instances of a given knowledge domain, related to query categories.

Restriction 1. *A query made by a user on the unanonymized log must not be assigned to the same user on the anonymized log.*

Restriction 2. *When creating an anonymized query log, the user must be chosen randomly from at least k different user values.*

Restriction 1 guarantees that the output does not reveal any unanonymized pair of user and query. Meanwhile, Restriction 2 mandates probabilistic k-anonymity, which requires at least k distinct values for users in each set when randomly creating an anonymized log.

3.3. Anonymization process

Our anonymization process refers to the method that produces the stream of logs R', which is probabilistic k-anonymous:

$$R' = \{r'_0, \dots, r'_m\} \tag{6}$$

We assume that each record in the stream of logs R, represented as $r_j = \{u_i, q_j, c_g\}$, is assigned to its corresponding v_x^h based on its categorization c_g. The record r_j is then divided into two parts: user u_i which is assigned to the users' set $U[v_x^h]$, and query q_j which is assigned to queries' set $Q[v_x^h]$. The anonymized stream of logs R' is generated by randomly pairing one user from $U[v_x^h]$ with one query from $Q[v_x^h]$, once $\mid U[v_x^h] \mid$ is equal to k. This generates an anonymized record with the following form:

$$r'_j = u'_i, q_j, c_g \tag{7}$$

where $q_j \in Q[v_x^h]$ is matched with a $u'_i \in U[v_x^h]$ that is different of u_i.

Once the record r'_j is generated, u'_i and q_j are removed from their respective sets $U[v_x^h]$ and $Q[v_x^h]$. If $\mid U[v_x^h] \mid < k$, but the tree contains a $BU[v_x^y] \in \{BU[v_x^1]..BU[v_x^l]\}$ where $\mid BU[v_x^y] \mid \geq k$, then record r'_j is generated by applying a random match between one query of $q_j \in BQ[v_x^y]$ and one user $u'_i \in BU[v_x^y] \neq u_i$. Both u'_i and q_j are also removed from their respective source sets $U[v_{xu}^{hu}]$ for the user u'_i and $Q[v_{xq}^{hq}]$ for the query q_j, once the record r'_j is generated. In this case, to maintain a balance between the source sets it is necessary add an additional step which consist of moving a randomly selected user from $U[v_{xq}^{hq}]$ to $U[v_{xu}^{hu}]$.

The Id function is assumed to correctly identify the original u_i when given r'_j. The function Re is a re-identification function applied to the records in R', which when given r'_j, returns:

$$Re(r'_j) = u_i \in U[v_x^h], u_j \neq u'_i \tag{8}$$

The goal of probabilistic k-anonymity is to limit the probability of successful re-identification to no more than $\frac{1}{k}$ for all u_i in R and all possible values of $Re(r'_j)$, as expressed in the inequality:

$$P(Re(r'_j) = Id(r'_j)) \leq \frac{1}{k} \tag{9}$$

The stream of logs R' is considered to meet the standards of probabilistic k-anonymity if, when given knowledge of R', the likelihood of linking any record r'_j in R' to its corresponding record r_j in R is no more than $\frac{1}{k}$.

Our proposal satisfies Eq. (9) by demonstrating that for each vertex v_x^h in τ, the random selection of an element (per Restriction 2) ensures equal likelihood for all outcomes. Thus, the maximum re-identification probability for r'_j in τ can be expressed as:

$$P(Re(r'_j) = Id(r'_j)) \leq \max_{\forall x,h} \frac{|U[v_x^h] \cap Id(r'_j)|}{|U[v_x^h]|} \tag{10}$$

Since $U[v_x^h]$ sets are constructed using arity, it follows that:

$$\forall x, h, Id(r'_j) \rightarrow |U[v_x^h] \cap Id(r'_j)| \in 0, 1 \tag{11}$$

One may contend that Restriction 1 results in a value of $k-1$. However, this value is set to k by Restriction 2 (which also guarantees $|U[v^h]| \geq k$), making the upper bound of our proposal for $P(Re(r'_j) = Id(r'_j))$ no greater than $\frac{1}{k}$, therefore satisfying probabilistic k-anonymity. A more formal examination of this outcome is presented subsequently.

3.4. Privacy analysis

Given an anonymity parameter k in $\mathbb{Z}^+$, a set of users $\mathcal{U} = u_1, ..., u_n$ where $n \geq k$, and a set of query logs $Q = (u_{i_j}, q_j)_{j=1}^{j}$ up to iteration j, with $q_k \neq q_l \forall k, l \in [j], (k \neq l)$ and $ui_j \in \mathcal{U}$. We allow for repeated user actions (i.e. $u_{i_k} = u_{i_l}$).

Likewise, we can assume a query in R', an adversary $\mathcal{A}$ has at most a $\frac{1}{k}$ probability of determining the user to which the query belongs in R. This is under the assumption that R' and k are known and $\mathcal{A}$ is a PPT (Probabilistic Polynomial-Time) adversary. Let $j_0 \in [j]$ define an experiment $Exp_{Re}(k, R)$ using the notation above, in which:

$$R' \leftarrow Anon(k, R) \tag{12}$$

$$R^* \leftarrow Re(k, R')$$

$$let\ b = \begin{cases} 1, & \text{if } R = R^* \\ 0, & \text{otherwise} \end{cases}$$

$$return\ b$$

Theorem 1. *Anon, as defined by Eq. (12), is considered k-anonymous if for every user set, query log R, and index $j_0 \in [j]$, the advantage of any PPT adversary $\mathcal{A}$ is limited to $\frac{1}{k}$, i.e.,*

$$Adv_{\mathcal{A}}(k, R) = P[Exp_{Re}(k, R)] \leq \frac{1}{k} \tag{13}$$

Proof. Let $R' = (u'_{i_j}, q'_j)_{j=1}^{j'}$ and j be the iteration at which the first log entry from the anonymizer is released after reading (u, q). Also, let $\mathcal{U}_j^{R'} = (u_{i_{j_1}}, ..., u_{i_J})$ be the set of users present in R' at iteration j and $\mathcal{U}_j = (u_{i_1}, ..., u_{i_k})$ be the set of users used internally by the anonymizer at iteration j. It is known that $u \in \mathcal{U}_j$ and $\mathcal{U}_j \in \mathcal{U}_j^{R'}$ and that $\mathcal{U}_j$ contains at least k different users.

$$P(\mathcal{A}(R', q) = u) = \tag{14}$$

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

The output of the anonymizer is not affected if the users U_j and queries Q_j stored after reading query q (with U_j having at least k different users) are permuted from Q_j to R (all within U_j). In other words, this permutation has no impact on the anonymizer output:

$$P(\mathcal{A}(R', q) = u | R = Re(R')) = \tag{15}$$

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

with U_j being the set of users that can appear in step j and u being a member of $\mathcal{U}$. If U_j is fixed and $u \in U_j$, we can construct an R by pairing query q with each user u' in U_j and one of the queries q' such that the entries of u' from U_j are now associated 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, where $R^* = Re(R')$ and $U_j = U$, preserves the original pair (u, q), which is summarized below, in Eq. (16):

$$P(\mathcal{A}(R', q) = u | R = Re(R')) = \tag{16}$$

$$\frac{(j_{u_2} + ... + j_{u_k} + j_u - 1)!}{(j_{u_2} + ... + j_{u_k} + j_u)!} =$$

$$\frac{1}{(j_{u_2} + ... + j_{u_k} + j_u)} \leq \frac{1}{k}. \quad \square$$

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 $\ell$
    $v_x^l \leftarrow \{\gamma_s^1, ..., \gamma_{s^*}^{\ell}\} \in c$;
    // Add user to users' category set
    $U[v_x^l] \leftarrow u$;
    // Add query and categorization to queries' category set
    $Q[v_x^l] \leftarrow \{q, c\}$;
    // Find deepest branch with k distinct users, on current category
    foreach $v \in v_x^0..v_x^l$ do
        if $distinct(BU[v]) > k$ then
            $branch \leftarrow v$
        end
    end
    // If that branch exists
    if $branch$ then
        // If there are more than k distinct users on the set
        if $distinct(U[v]) > k$ then
            // Generate output log from the same set
            Generate($Q[v], U[v]$)
            // If there are still more than k distinct users on the set, generate a
              new output
            if $distinct(U[v]) > k$ then
                Generate($Q[v], U[v]$)
            end
        else
            // Select a random branch's vertex to pick the query and another
              to pick the user
            random $vq \in \{branch..v_x^h\}$
            random $vu \in \{branch..v_x^h\}$
            Generate($Q[vq], U[vu]$)
            // If vertices are not the same
            if $vq \neq vu$ then
                // Move a user from query's to user's vertex
                pop random $u' \in U[vq]$
                $U[vu] \leftarrow u'$
                // Generate a new output from vu if possible
                if $distinct(U[vu]) > k$ then
                    Generate($Q[vu], U[vu]$)
                end
            end
        end
    end
end
function Generate(Q, U):
    // Select and remove a random query and categorization from query's set
    pop random $\{q', c'\} \in Q$
    // Select and remove a random user from user's set, distinct from the
      original user related to the query
    pop random $u' \in U, u' \neq Id(q)$
    // Send to the output the selected user, query and category
    send $u', q', c'$
end
```

3.5. Proposed algorithms

We present three different algorithms in this proposal. First, Algorithm 1 is responsible of the anonymization process. Then two de-anonymization processes are proposed in Algorithm 2 and Algorithm 3. Those two de-anonymization proposals are considered to be used by a PPT adversary. All the algorithms receive three inputs: A stream of hierarchically categorized query logs, and values for k and ℓ.

The proposed algorithm ensures that each time a new anonymized log is generated, the size of the source set from where it is selected (either $U[v_x^h]$ or $BU[v_x^y]$) contains a minimum of k distinct users. Additionally, the algorithm aims to keep the size of the source category set $Q[v_x^h]$ as close to k elements as possible. Probabilistic k-anonymity is guaranteed as the algorithm always selects from a minimum of k distinct users and a minimum of k different queries.

Algorithm 2: Simple 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 $\ell$
    $v_x^l \leftarrow \{\gamma_s^1, ..., \gamma_{s^*}^{\ell}\} \in c$
    // Add current user to users' category set
    $U[v_x^l] \leftarrow u$
    // Add current query and full categorization to queries' category set
    $Q[v_x^l] \leftarrow \{q, c\}$
    // While there are more than $k$ distinct users on the current category
    while $distinct(U[v_x^l]) > k$ do
        Regenerate($Q[v_x^l], U[v_x^l]$)
    end
end
function Regenerate (Q, U):
    // Select and remove a query and categorization from query's set, using the
      record linkage algorithm
    record_linkage $\{q', c'\} \in Q$
    // Select and remove a user from user's set, using one of the record linkage
      algorithms
    record_linkage $u' \in U$
    // Send to the output the selected user, query and category
    send $u', q', c'$
end
```

When there are less than k different users on a category set $U[v_x^y]$, and no anonymized log could be released from that category due to Restriction 2, the *anonymizer* first tries to find a category in the same branch where there are at least k different users on $BU[v_x^y]$. If found, a user is randomly chosen from this set and a query is randomly chosen from $BQ[v_x^y]$ to generate the anonymized log. Then they are removed from their respective category sets $U[v_{x'}^{y'}]$ and $Q[v_{x''}^{y''}]$. If no k different users are found in either $U[v_x^y]$ or $BU[v_x^y]$, the k-value of this category v_x^y is increased by one, each time a new log enters the category and the restriction is not meet.

If a new log with a user already present in a category is added, their frequency (arity) in $U[v_x^h]$ increases and the query is added to $Q[v_x^h]$, causing $|Q[v_x^h]|$ to increase by one while $|U[v_x^h]|$ remains unchanged, therefore $Q[v_x^h]$ size may be bigger than k in this situation. If the condition in Restriction 2 is not satisfied, the log cannot be anonymized and the size of $Q[v_x^h]$ may exceed k.

If Restriction 2 is satisfied and a user has an arity greater than one, Algorithm 1 attempts to anonymize an extra log to decrease the size of Q and the user's arity, while still following Restriction 1. This additional step occurs at most once per input log, resulting in up to two logs can be generated each time a new record is processed, until all users have a frequency of one.

Stable system performance is maintained when set size variations are proportional, according to Pàmies-Estrems et al. (2016). To optimize memory usage, the size of each set is adjusted in incremental unitary steps. Along with the k parameter, the category tree depth ℓ parameter must be specified and both prevail constant throughout the entire process.

4. Implementation of our proposal

An implementation of our proposal is described next. We also explain the architecture and necessary requirements prior the implementation, followed by an evaluation and presentation of experimental results. A companion GitHub repository is available,[6] with the code and evaluation results.

[6] https://github.com/dpamies/saps-qls.

Algorithm 3: Full 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 $\ell$
    $v_x^l \leftarrow \{\gamma_s^1, ..., \gamma_{s^*}^{\ell}\} \in c$;
    // Add user to users' category set
    $U[v_x^l] \leftarrow u$;
    // Add query and categorization to queries' category set
    $Q[v_x^l] \leftarrow \{q, c\}$;
    // Find deepest branch with k distinct users, on current category
    foreach $v \in v_x^0..v_x^l$ do
        if $distinct(BU[v]) > k$ then
            $branch \leftarrow v$
        end
    end
    // If that branch exists
    if $branch$ then
        // If there are more than k distinct users on the set
        if $distinct(U[v]) > k$ then
            // Regenerate log from the same set
            Regenerate($Q[v], U[v]$)
            // If there are still more than k distinct users on the set,
              regenerate a new log output
            if $distinct(U[v]) > k$ then
                Regenerate($Q[v], U[v]$)
            end
        else
            // Select a random branch's vertex to pick the query and another
              to pick the user
            random $vq \in \{branch..v_x^h\}$
            random $vu \in \{branch..v_x^h\}$
            Regenerate($Q[vq], U[vu]$) // If vertices are not the same
            if $vq \neq vu$ then
                // Move a user from query's to user's vertex
                pop random $u' \in U[vq]$
                $U[vu] \leftarrow u'$
                // Regenerate an output from vu if possible
                if $distinct(U[vu]) > k$ then
                    Regenerate(Q[vu],U[vu])
                end
            end
        end
    end
end
function Regenerate($Q, U$):
    // Select and remove a query and categorization from query's set, using the
      record linkage algorithm
    record_linkage $\{q', c'\} \in Q$
    // Select and remove a user from user's set, using one of the record linkage
      algorithms
    record_linkage $u' \in U$
    // Send to the output the selected user, query and category
    send $u', q', c'$
end
```

4.1. *Initial architecture*

Our goal is to create an anonymization method to anonymize query logs in real-time (cf. Fig. 3). This will be done on the server-side. The system will take in a continuous stream of categorized query logs as input and produce a continuous stream of anonymized logs as output. To be successful, our algorithm must meet specific requirements detailed below, and that can be customized to meet different environments.

4.2. *Functional requirements*

Our proposal needs to satisfy a set of functional requirements, in order to be usable in a Internet Service environment. Such requirements are defined next.

Scalability — According to Bondi's characteristics (Bondi, 2000), scalability denotes a system's capability to manage an increasing volume

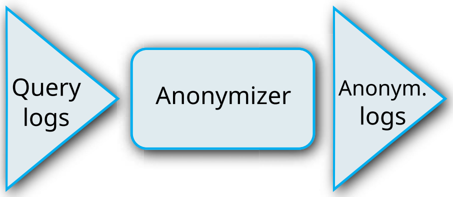


Fig. 3. Outline of the proposed method for anonymizing query logs in real-time. The method uses a stream of query logs as the input for the algorithm, and generates a stream of anonymized logs as output.

of work or its potential to expand to accommodate such growth. A scalable system is one that can continue to function effectively as it grows and handles more work, without the need for frequent upgrades or overhauls. Regarding scalability, there are two main types of scalability (Michael et al., 2007):

- **Horizontal Scalability** involves adding more resources to a system, such as adding more servers to a network, to handle an increased workload. Horizontal scalability allows a system to distribute its workload across multiple resources, which can make it more efficient and better able to handle large amounts of data. This can help to maintain a faster and more reliable performance, but it also demands to efficiently manage and coordinate the multiple resources.
- **Vertical Scalability**, on the other hand, involves adding more power to an existing server, such as by adding more CPU cores or more RAM. However, vertical scalability has its limitations. There is a limit to how much power can be added to a single server, and at some point, it may no longer be cost-effective or practical to continue scaling vertically.

Both vertical and horizontal scalability have their advantages and disadvantages, and can be useful in different situations. Vertical scalability can be a good option for systems that are already heavily optimized and do not have much room for improvement. However, when scaling vertically is no longer cost-effective, horizontal scalability may be a better option. And many systems use a combination of both approaches to achieve the best performance and efficiency.

Resource Consumption — Resource consumption refers to the use of resources, such as memory or storage needed in order to produce a viable output. Resource consumption is an important factor to consider in our case. We want to optimize the system in order to reduce its resource consumption and improve its performance. This enables the deployment of the proposal over architectures with some unused resources, reducing system running costs. Also a small resource usage allows to keep all the necessary data in memory, and therefore allowing more agile executions.

Speed — Speed is a measure of how quickly a system is able to perform a given task, in this case we are interested in data processing speed. A faster system can provide a lower processing delay, and a decreased need of buffers and other intermediate resources to hold the unprocessed data. It also enables the system to handle more complex and demanding tasks, such a real time processing of the received logs.

Efficiency — Efficiency is an important factor to consider as the resulting proposal may be able to handle more complex or demanding tasks. Usually efficiency refers to the ability to perform a given task using a minimum of resources, such as time, memory, or energy. In this case we also want to ensure that the algorithmic efficiency is optimal for our proposal. Taking into account that our input is a continuous stream of data, we aim for a linear algorithmic complexity for our proposal.

Transparency — Transparency refers to the ability of a system to blend into existing environments without the need of modifications over the

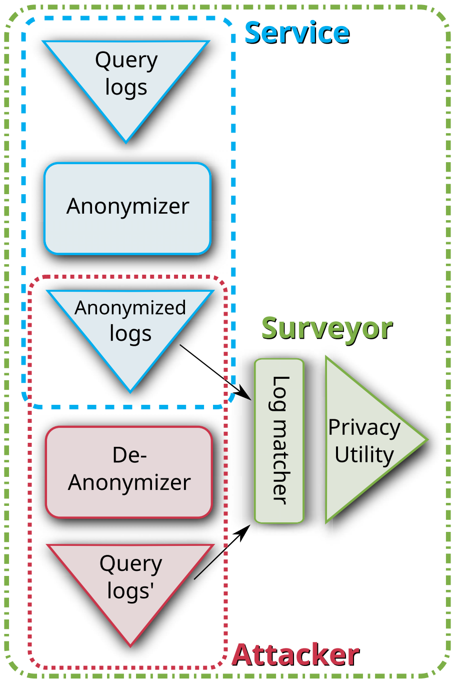


Fig. 4. The full architecture consists of three main components: Anonymizer, De-anonymizer, and Log Matcher. Anonymizer anonymizes a real-time stream of query logs using Algorithm 1. De-anonymizer applies different algorithms (2 3) to simulate record linkage attacks on the anonymized logs, attempting to recreate the original logs. The Log Matcher is responsible for evaluating the anonymization, de-anonymization privacy levels, and also generates a metric for preserved utility.

original environment other than adding the new component. For our proposal, transparency is important as it can help to spread the proposed system into several existing solutions. It can help to address potential privacy concerns, without the need of a great integration effort with different use cases and systems.

Modularity — We want a system that is made up of separate, self-contained components that can be easily developed, tested, and maintained independently. Modularity has several advantages. Since each module can be designed with a specific, well-defined purpose, it can make a system easier to understand and maintain. It can also make a system more flexible and adaptable, since individual modules can be replaced or updated without affecting the rest of the system.

4.3. *Expanded architecture*

Outline shown in Fig. 3 has been broaden to include two new components, the Attacker and the Surveyor, to conduct a comprehensive empirical evaluation, further to the analysis described in Section 3. Proposed design follows a micro-service pattern as shown in Fig. 4. For the purposes of this research, all components are utilized, but in a real-world setting, just components for the Service need to be implemented.

The expanded architecture includes three primary components: the anonymizer, the de-anonymizer and the log matcher. The anonymizer utilizes Algorithm 1 to generate an anonymized real-time stream of query logs. The de-anonymizer applies different algorithms (cf. Algorithms 2 and 3) to simulate record linkage attacks on these anonymized logs, attempting to recreate the original logs. Finally, the log matcher is responsible for evaluating all the processes in the context of the current proposal and will not exist in a real world scenario. It evaluates the performance of the anonymization and de-anonymization algorithms. It also generates metrics related to privacy levels and preserved utility.

4.3.1. Actors

In the current test architecture, we specify the following actors:

- **Service —** is responsible for anonymizing and publishing query logs.
- **Attacker —** attempts to uncover the relationship between the original query and the user who made it.
- **Surveyor —** is who evaluates the effectiveness of the proposal. Can access all the data but is not able to make any changes.

4.3.2. Phases

The following three phases can be identified:

- **Anonymization —** represents the typical operation of the proposal in the Service. It is the responsible of the anonymization of logs and generates an anonymized output stream.
- **De-anonymization —** simulates de-anonymization attacks by attempting to regenerate as many anonymized logs as possible, matching the original user with the original query they made.
- **Survey —** evaluates anonymization, de-anonymization and overall performance, considering all data streams, as well as speed and resource consumption.

4.3.3. Interactions

The main function of the Internet Query Service in a real-world scenario would be to anonymize query logs and provide its clients with the anonymized versions. The Attacker attempts to recreate the original query logs. To achieve its goal it plays the role of a normal client from the Service's perspective, gaining access to the anonymized output from the Service. Processing this stream the Attacker generates another log stream which tries to de-anonymize, in other words reconstruct, the maximum amount of original logs. Only during our analysis, there are additional interactions between the Service, the Attacker and the Surveyor, that takes all the log streams from each step. Additional details will be presented later in the study.

5. Experimental results

Our approach has been practically implemented and its effectiveness in terms of privacy, data utility, and other functional requirements has been validated through experimental results. The evaluation, together with the implementation code, is available online, in a companion GitHub repository.[7] The experiments were carried out on a *Lenovo* computer running *Arch Linux*, with a 1.8 GHz *Intel Core*™ i7-8550U CPU, 16 GB of RAM and a *Intel SSD 600p Series* hard disk, whose performance profile is good for light workloads, but write speeds drop significantly and rather quickly after 16 GB are written. All algorithms were implemented and executed in *Go* (version 1.18.1).

5.1. Implementation

The implementation of Algorithms 1, 2, 3 has been done in the Go programming language. The evaluation uses the set of query logs published by AOL, as this is the largest set of real search logs freely available that we were able to found. In order to achieve a system that meets the previously defined requirement of transparency, and that can therefore be integrated without the need of changes to the existing architectures, we believe it is necessary to preserve the format of the original log file and use it as the input to our system. At the same time, the output of our system will also be a log file, keeping the same format, but with the necessary transformations to increase data privacy. Doing so, the input/output of the system will be compatible with the existing architecture. However, the proposed system can be easily adapted to work with other formats and data transfer protocols.

The original logs as they were published by AOL were not classified in any kind of categories, but this is a requirement to be able to apply our algorithm. To perform this classification, we use the same deterministic classifier that we proposed in our previous works (Pàmies-Estrems et al., 2016, 2020). As it is a deterministic classifier, we are able to obtain the same classification for the same query log. This makes feasible to compare tests carried out as well as the results obtained in our previous works and the current approach. Thanks to the modular approach of the proposed architecture, this classifier could be changed for another more specific one if its required for the application, without the need of making any changes to the anonymization algorithms.

On the analyzed data set, the proposed classifier allows us to obtain a hierarchical classification of each log, consisting of a path of categories, between one and 14 categories. Each element represents a category of different depth, the first categories are the most superficial ones and therefore describing more general concepts. The deeper the levels, the more specific concepts they represent. The categorizer was able to assign these categories to 98% of the logs published by AOL. The remaining 2% of logs contain isolated strings of characters or digits without any specific meaning and therefore cannot be assigned to a specific category path.

Once the logs have been categorized, they will be processed by the anonymizer, generating a set of protected logs. In order to validate the degree of protection achieved in this step and check if the requirements we had set have been met, we propose two algorithms in charge of performing a record linkage on the protected logs. Then a comparison between the output of these algorithms and the original logs will be conducted to determine which degree of protection has been achieved with the current proposal.

5.2. Evaluation methodology

We used all the algorithms defined in Section 5.1 to experimentally evaluate the proposal. Evaluation has been posed taking into account two main aspects, on the one hand we want to validate the feasibility of our current proposal, on the other hand we want to determine the differences between this novel proposal and the ones defined in previous work (Pàmies-Estrems et al., 2016, 2020). One version of the anonymization algorithm and two versions of the record linkage algorithms have been evaluated, to ensure that the defined requirements are met in terms of functionality, privacy and utility.

5.2.1. Used data

In the conducted tests, we utilized plain text files from *AOL's* query logs AOL (2006). These data were released by *AOL* for research purposes, and include 36 389 576 logs from a three-month period of actual searches made by their users. Fig. 5 shows a sample of the logs used.

As described in Section 5.1, Classifier adds an additional field to each log record. The hierarchical classification for each query log is represented as a list with n elements, with each element representing a subcategory of the previous one. During the evaluation, the length of the list ranged from 1 to 13 elements. The categorization process is deterministic and always includes all subcategories that the Classifier can obtain for a certain query, which remains unaffected by the ℓ value used by the anonymizer algorithm, fact that enables to compare the results of different anonymizer algorithms and executions.

5.2.2. Conducted tests

There are only two parameters used to adjust the behavior of the proposed system: k and ℓ. k denotes the desired number of distinct users per category, and ℓ represents the desired depth of categories used per query log. Note that both k and ℓ are desired values, and the system is able to adjust them in order to respond to specific needs and

[7] https://github.com/dpamies/saps-qls.

```
479 family guy              2006-03-01 16:01:20
479 also sprach zarathustra 2006-03-02 14:48:55
479 family guy              2006-03-03 22:37:46 1 http://www.familyguyfiles.com
479 top grossing movies     2006-03-03 22:42:42 1 http://movieweb.com
479 top grossing movies     2006-03-03 22:42:42 2 http://www.imdb.com
479 car decals              2006-03-03 23:20:12 4 http://www.decaljunky.com
479 bose                    2006-03-03 23:30:11 1 http://www.bose.com
479 bose car decal          2006-03-03 23:31:48 1 http://stickers.signprint.com
480 chicago the mix         2006-03-04 22:11:31 1 http://www.wtmx.com
480 chicago the drive       2006-03-04 22:14:51 2 http://www.wdrv.com
480 chicago radio annoucer  2006-03-04 22:16:07
480 chicago radio whip      2006-03-04 22:16:27
480 chicago radio brian     2006-03-04 22:17:00 1 http://www.djheadlines.com
480 emma watson             2006-03-04 23:05:53 1 http://www.imdb.com
480 stanford encyclopedia   2006-03-06 21:57:14 1 http://plato.stanford.edu
```

Fig. 5. This is a sample of the used logs, where each line contains a query log with several fields, from left to right: user id, search text, time stamp, chosen result and destination URL.

try to optimize execution. The effects of these parameters and achieved optimizations were studied through several tests.

Anonymizer —the proposed Anonymizer was used to generate anonymized data by executing it multiple times on all available AOL logs, covering different parameters values. k values ranging from 3 to 200 were used to be able to compare the results with the obtained from Pàmies-Estrems et al. (2016) Pàmies-Estrems et al. (2020). All available values of ℓ, ranging from 1 to 13, were tested, however it was found that values of ℓ greater than 11 did not produce significant differences as logs with a depth of more than 11 categories are rare. Finally each possible combination of those k and ℓ values was evaluated against the privacy, functional, and utility requirements.

Surveyor — Additional tests were specifically performed using the surveyor to measure the amount of lost data utility. Those tests compare distance between assigned categories of each user when using the original logs and the anonymized logs. For those tests, k values ranging from 3 to 90 and ℓ values ranging from 1 to 13 were used. We consider that the greater the distance obtained, the more data utility have been lost.

De-anonymizer — Several attempts to de-anonymize protected logs were conducted using the two proposed algorithms. Those algorithms were executed over all anonymized data, attempting to achieve a *record-linkage* attack, with different approaches:

- **Record-linkage 1 —** This algorithm is the most basic one that was tested. It is the same algorithm that was tested on Pàmies-Estrems et al. (2020) as we want to compare its results against the novel proposed method. It attempts to reverse the anonymization of query logs by using an algorithm with a fixed ℓ value (as described in Algorithm 2 in Section 3). The algorithm also leverages both restrictions 1 and 2 to increase the level of de-anonymization by attempting to reconstruct the original logs through the random pairing of users and queries from the same category.
- **Record-linkage 2 —** This record-linkage algorithm improves upon Record-linkage 1. Instead of randomly pairing users and queries from the same category, we used a similar algorithm to the one employed in the anonymization procedure (outlined in Algorithm 3 in Section 3) to match them. Like the other algorithms, it also respects both restrictions.

5.3. *Privacy study*

During our initial privacy evaluation, we compared the original query logs with the anonymized logs and found that none of the original user/query combinations were present in the anonymized output, as intended by our design. However, this does not guarantee full user privacy as there may still be potential for re-identification through deanonymization attacks on the output data flow. To check strength against those attacks, we applied two record-linkage algorithms (referenced as Algorithm 2 and Algorithm 3) to the anonymized logs and compared the resulting logs to the original ones, determining the percentage of reidentified.

A deterministic classifier was used for generating the categories on the anonymization process. Being deterministic, it can be expected that at some point an attacker could replicate this classification process. As applying the same classification method for the de-anonymization process leads to higher re-identification rates, this privacy study shows results assuming that the attacker has already replicated the classifier, and uses the same categorization as the anonymizer.

We used two different algorithms to simulate a possible de-anonymization attack. The first algorithm is based on our previous proposal (Pàmies-Estrems et al., 2020), since we want to make a comparison between the results obtained. The second proposal is based on Algorithm 1 that we used in the anonymization process. A full explanation of the differences could be found in Section 5.2.2. Both algorithms have the same inputs, outputs and parameters. As input they use the anonymized logs produced by the anonymizer and generate an output using the logs resulting from the de-anonymization process. As parameters both algorithms use k and ℓ. In the second algorithm they are used as explained in Section 3, on the other hand in the first de-anonymization algorithm the value of ℓ is used as an absolute limit in the number of categorization levels to use and this value is not changed in any way throughout the run. Since the highest re-identification of original logs is achieved when the attacker uses the same values of k and ℓ that have been used in the anonymization process, the results we show are those that satisfy this requirement.

In Fig. 6, proportion of matched records is shown. Algorithms were executed using several combinations of k-values (between 3 and 200) and ℓ-values (between 1 and 13). Using a value of $\ell = 1$ the classifier only generates the first level of categories, therefore instead of a tree it uses a flat data structure similar to the one used in our previous proposal (Pàmies-Estrems et al., 2016) which enables the result comparison. The maximum depth that our classifier was able to generate is $\ell = 13$, therefore there is no need to use higher values. We also selected the k values to enable comparison of results with our previous evaluations (Pàmies-Estrems et al., 2016, 2020).

Our results indicate that the probability of re-identification is consistently below the theoretical maximum of $\frac{1}{k}$, as stated in Purdam and Elliot (2007). We used the Kolmogorov-Smirnov test (Kolmogorov, 1933; Smirnov, 1948) to compare these results to the k-anonymity probability. Fig. 7 shows results obtained using the proposed algorithms and a value of $\ell = 1$, which leads to the higher re-identification rates. The D value, which denotes the highest discrepancy between the cumulative distributions, is 0.18, with a p-value of 0.9971. This outcome implies that the probability of re-identification aligns with the null hypothesis, which is accepted at the 5% level of significance.

Obtained results show a significant improvement in privacy when using the anonymization system defined in this article compared with the ones obtained in Pàmies-Estrems et al. (2016, 2020). On the other hand, no significant differences can be seen between the use of the different proposed de-anonymization algorithms.

In all cases, it can be seen that the value of k is directly related to the level of privacy obtained. The higher the value of k, the lower the record linkage achieved. The value of ℓ also affects the results obtained.

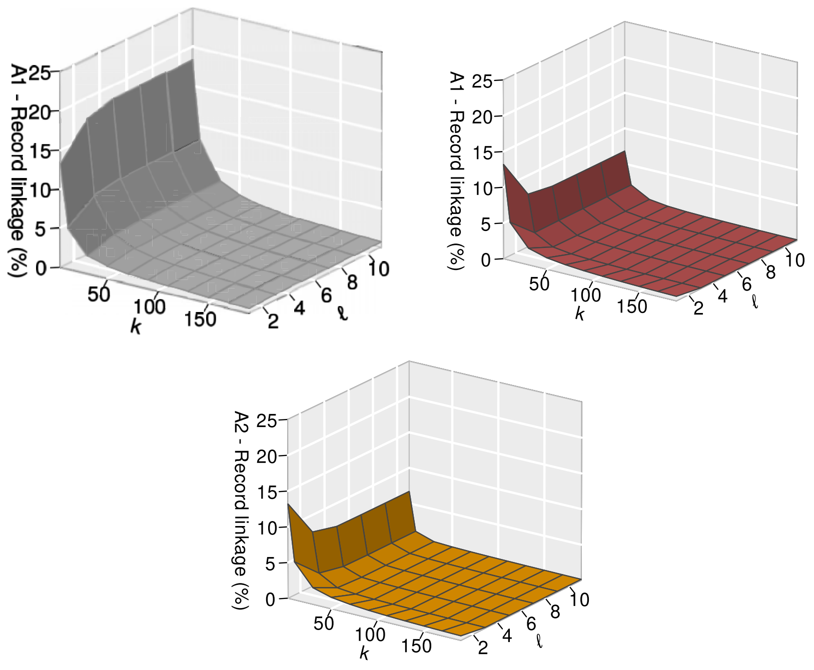


Fig. 6. Percentage of matched records resulting from record linkage, old results (Pàmies-Estrems et al., 2020) (top left) compared with the logs anonymized with the new algorithm and de-anonymized with de-anonymizer 1 (top right) and de-anonymizer 2 (bottom). Values of k between 3 and 200 and values of ℓ between 1 and 11.

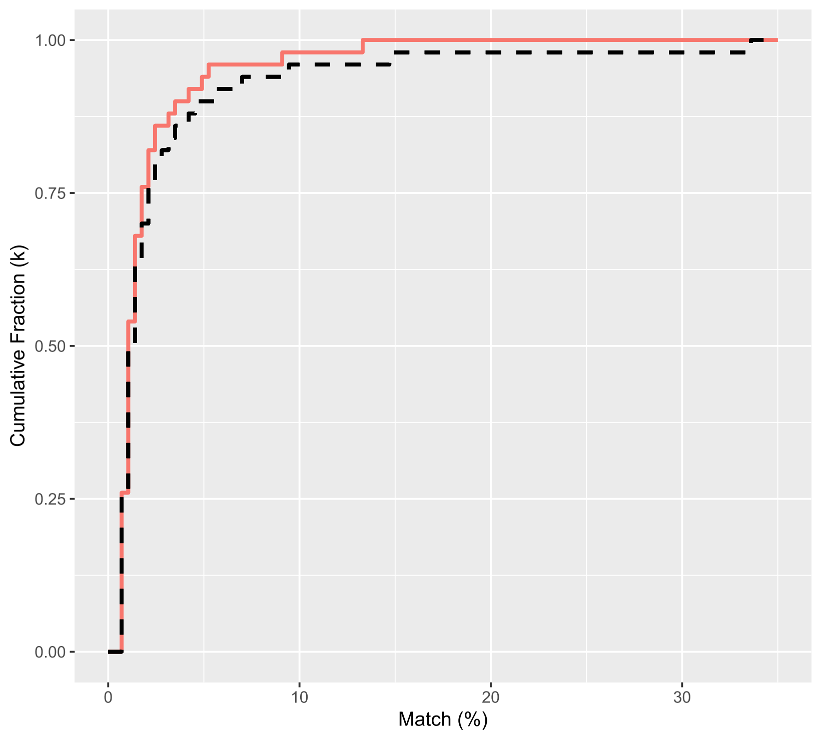


Fig. 7. Illustrates the Kolmogorov-Smirnov test, which utilizes the cumulative fraction functions of the hypothetical k-anonymity (represented by the dashed line) and the actual outcomes (indicated by the solid line). The Kolmogorov-Smirnov test produces a p-value of 0.9971 and D-value of 0.18.

Using our previous proposal (Pàmies-Estrems et al., 2020), by increasing the values of ℓ, more specific categories are generated, through which logs from fewer users pass, and therefore it increases the probability of re-identification of original users. On the other hand, using the current proposal, a higher value of ℓ allows the algorithm to choose between more categories, and therefore the probability of re-identification decreases as the value of ℓ increases.

The results achieved using an ℓ of 1 are similar between previous proposals (Pàmies-Estrems et al., 2016, 2020) and the current one. Where there is a more differentiated behavior is using higher ℓ values, which bring the new data structures and algorithm into play. Due to restrictions 1 and 2, during the execution of the algorithm, temporary augmentation of the size of Q above the target value defined by k is necessary. We see that with the current proposal, when increasing the

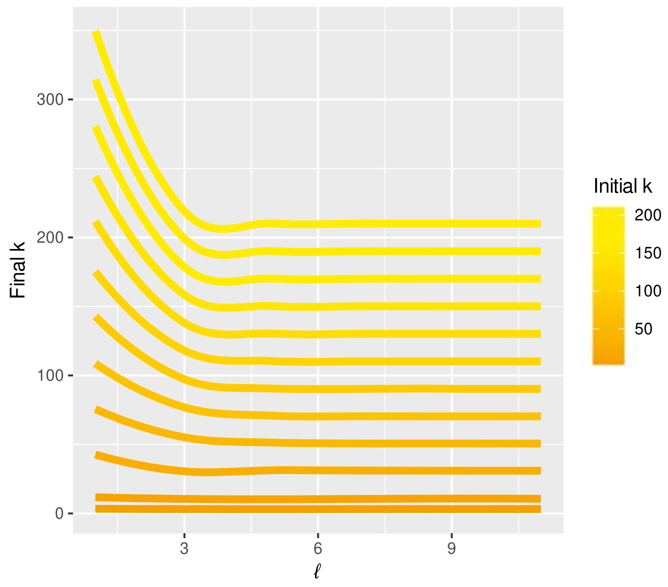


Fig. 8. The final $|Q|$-value represents average size of the query sets. When ℓ is low, the final $|Q|$ tends to be greater, as a result of a larger number of users coincidentally selecting identical categories. As ℓ increases, the final $|Q|$ tends to align with the specified k.

value of ℓ, the algorithm has the opportunity to quickly balance the size of Q. Achieving that the sets size remain closer to k value, with ℓ values of 3 or more.

This effect can be seen in Fig. 8 which shows mean final $|Q|$ values, depending on the values of ℓ and k. When ℓ is low, the average final $|Q|$ values tend to be higher because there are fewer categories and more users coincidentally querying in the same category. As ℓ increases, the final $|Q|$ values generally correspond with the designated k.

The best results for record linkage were achieved with the lowest ℓ and lowest k values. The most favorable de-anonymization effort managed to match 13.19% of records to their original users in this case. The tests were conducted giving to the de-anonymization process full knowledge of the anonymization algorithms, categories, and k and ℓ values used. However, as the initial k and ℓ values increases, the record linkage ratio decreases rapidly and drops below 1% for k values of 30 or more and ℓ values of 3 or more. Therefore, it can be concluded that the level of record linkage improved with the current proposal, and that it can be adjusted effectively by altering the k and ℓ values to reach the desired privacy level. The same rationale applies to any other de-anonymization technique in which the interaction between the Anonymizer and the Attacker reported in Fig. 4 will rapidly drop the linkage of queries from existing profiles, at the expense of decreasing the utility of the resulting queries. This includes as well the evasion of detection techniques via machine learning attacks (Khan et al., 2020, 2021), with the inclusion of noisy parameters at the Anonymizer, in addition to k and ℓ, to decrease precision and recall at the Anonymizer to negligible values.

5.4. Utility study

In this section, we evaluate the usefulness of the anonymizer proposal. The scrutiny is based on two main concepts that we consider relevant to utility in our context. First, maintaining the original user preferences within the anonymized query logs, and second, determining the proportion of logs that can be generated by the system.

We use the Earth Mover's Distance (EMD) metric (Rubner et al., 2000) to determine the similarity among the original user's interests and those obtained from the anonymized user's queries. This is done by categorizing the queries allocated to every user and computing the shortest distance required to link the classifications attributed to the initial and anonymized queries in a tree graph that represents the full categories hierarchy. The total distance for each user is calculated by summing up the distances between individual queries of that user. Therefore, if a query is classified and anonymized under the same category, the path length between categories is zero and no loss of utility is added to the user. The more distance between categories, the more the overall value of EMD will increase. Thus, EMD measures the gap between the initial user preferences and those inferred from the anonymized ones.

Using our previous proposed algorithm (Pàmies-Estrems et al., 2020), when using an ℓ value equal to the maximum depth of categorization, it can be guaranteed that there is no utility loss. But that is not the case with the current proposal, as queries could be assigned to other categories in the same branch. Therefore, all category tree depths and an ℓ values can lead to some degree of utility loss, worsening the overall proposal utility. After evaluating privacy results this could be an expected trade-off, as usually there is a balance between data privacy and utility.

Fig. 9 shows the average and maximum theoretical EMD distances between anonymized and original user logs utilizing the designated categorization. The highest potential distance remains unchanging and does not depend on the values of k and ℓ used. As it can be seen, the average distance is not affected by k, but decreases as ℓ increases. Consequently, the employment of additional tiers during the anonymization process leads to the anonymized queries being more proximate to their initial categories, resulting in better data utility. Fig. 9 shows results of the previous proposal for comparison purposes. As we noted, some utility was lost in exchange of a better privacy. However, utility levels only decreased an average of 2% compared to an average 10% increase of privacy levels. The percentage-based reduction in utility is depicted in Fig. 9. When $\ell = 1$, the typical reduction in utility is greater than 43%. Conversely, when $\ell = 9$, the reduction in utility is nearly negligible, close to 0%. The amount of utility loss that can be assumed varies depending on each application, so once the effects of the ℓ parameter have been demonstrated, we leave the choice of its value open to the final application.

We also aim to evaluate the proportion of system-generated logs in relation to the overall quantity of logs that it receives. The suggested system employs an hierarchical tree of categories containing sets, and each set should contain a minimum of k distinct users before an anonymized log can be released. An issue that we detected in our previous proposal (Pàmies-Estrems et al., 2020) is that some sets may take some time to reach k different users, and in some cases with very specific categories the set may never reach k, causing logs stored in those sets not being outputted.

As shown in Fig. 10 this issue has been solved with the current proposal, for all ℓ values. As explained in Section 3 when the proposal isn't able to generate a viable output using only current set values, it takes into account other sets from the same branch, which enables the system to use values from stagnant sets, increasing in a significative way the amount of anonymized logs that can leave the system, being almost 100% for all the runtime in our tests.

5.5. Functional study

In the sequel, we provide some discussions about the proposed functional requirements achieved in our work.

5.5.1. Modularity

Our proposal utilizes a micro-service architecture to make the system more modular and adaptable to various environments. This design approach allows each service to handle a specific task, resulting in a system that is highly cohesive and loosely coupled. We have provided detailed information about the anonymization service, which can be integrated with additional modules like categorization and profile creation.

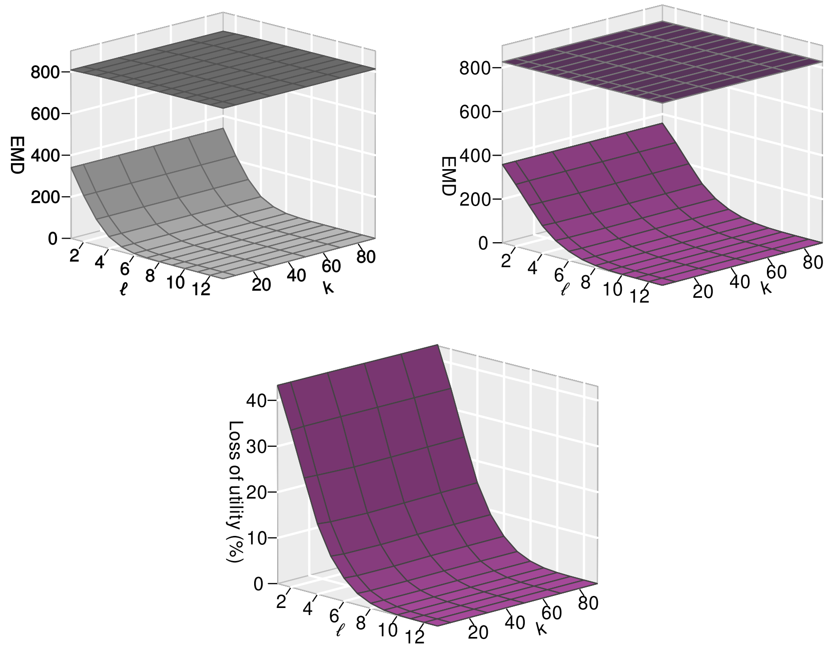


Fig. 9. Previous proposal results (top left). Current method (top right) has a slightly higher constant maximum possible distance between profiles and the average Earth Mover's Distance remain inversely proportional to ℓ. Reduction in data utility as a percentage of the theoretical maximum (bottom). The higher the number of levels used in the anonymization process, the lower the loss of data utility.

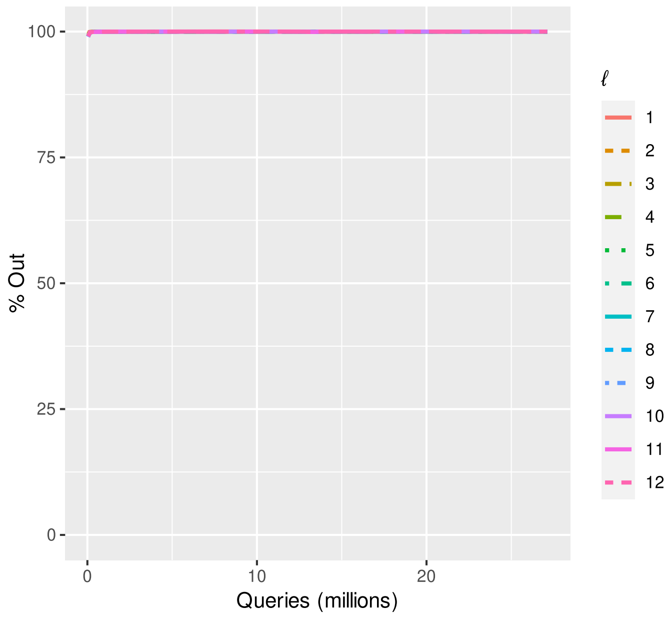


Fig. 10. Output queries vs. total queries (%). All sets are able to sustain a high throughput while generating anonymized queries, regardless of used ℓ value and depth of the category tree as proposed algorithm could take into account other sets from the same branch to reach k different users. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

5.5.2. Scalability

Our proposal has been designed to be scalable, both vertically and horizontally. As the values of both k and ℓ are dynamically modified according to system needs, we see that this allows better scalability by design.

Enabling or disabling several instances concurrently can result in achieving horizontal scalability. The fact of carrying out the implementation using Go allows to easily improve its horizontal scalability, thanks to the use of Goroutines. Using any language, the algorithm allows the data to be distributed and processed throughout several instances of the same algorithm in parallel. On the other hand, vertical scalability can be achieved by adjusting the number of resources allocated to the system, such as memory or CPU cycles.

5.5.3. Speed

The performance was measured for a basic usage. The anonymizer and deanonymizers was evaluated by measuring the time it takes to process a query through a sole execution thread on one core. Results shown are for this configuration. However, it is possible to run all of the suggested algorithms simultaneously both at thread or process level, which improves the overall throughput.

In addition to the calculations necessary to perform the proposed implementation, small code fragments have been added to allow evaluations for the level of privacy obtained, the use of memory and execution time of each algorithm. These additional calculations are not necessary in a production environment and therefore the performance of the algorithms would be slightly increased without them.

The best performance was observed when using $k = 3$ and $\ell = 1$, with an average query processing time of 1.68 μs, which is an improvement of over 11 times respect our last proposal (Pàmies-Estrems et al., 2020). This allows the system to handle an average of 594884 queries per second.

The mean processing time per query, taking into account all the tested values of k and ℓ, was 11.46 μs, or a capacity of 87222 queries per second. This is faster on average than our previous proposal which had a mean processing time of 33.68 μs per query. However, the current proposal has a slightly lower data utility. With all the parameter combinations used, our current proposal is swifter than the prior.

As shown in Fig. 11, the speed of the anonymizer is influenced by the values of k and ℓ. Alterations in ℓ values have minimal influence on the speed, whereas changes in k values have a slightly higher impact. With a $k = 3$, a query is processed in 4.2 μs on average, whereas when $k = 210$, the processing time increases to 20.6 μs.

Using Google load as a reference, which handles 63000 queries per second on average (T.G. Statistics, 2023), as per our test results it is feasible for a single thread of our algorithm to process all queries with

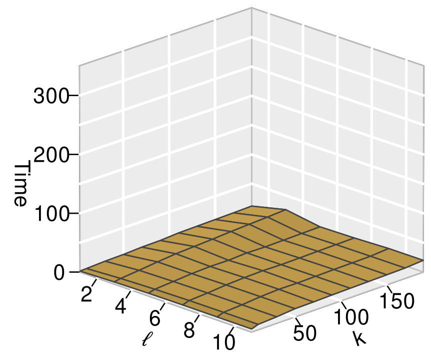


Fig. 11. The anonymizer's mean time per query (µs) was analyzed. The ℓ-value had a minimal impact on the required time, while the k-value had a more significant effect.

k-values up to 90, regardless of the ℓ value in real-time. For higher k-values this could be achieved using lower ℓ values.

The analysis of the mean time per query was also conducted for the two proposed de-anonymization algorithms.

Fig. 12 illustrates the obtained outcomes. On the one hand, the full de-anonymization approach yielded times that are similar to the anonymizer, as expected since the same base algorithm was used in both cases. On the other hand, we could expect a faster speed with the simple de-anonymization method, which involve a more straightforward approach. However, comparable speeds are achieved in all the cases. Since distribution of logs between categories has been carried out by the anonymizer, both de-anonumization algorithms end up doing very similar steps and therefore similar speed results are achieved.

5.5.4. *Delay*

Assessing the average delay of queries, between their entry and exit as anonymized query logs, is another important metric. To eliminate processing speed as a factor, we employed the average count of other queries handled by the system during the time between the entrance and release of a specific query, as a means of abstraction for delay.

Our aim is to compare current result with previous ones, as this is one of the main requirements to improve. As shown in Fig. 13, the delay has improved dramatically. In our previous proposal, the delay increases proportionally to the chosen ℓ-value, but eventually stabilizes at values over 60000 for the biggest ℓ-values. This was expected, as the system needs time to fill all the categories before achieving an stable output. Using the current proposal, the system is able to generate an stable output right from the start, and delay values range between 98 and 134, achieving a much smaller delay for all ℓ-values.

5.5.5. *Resource consumption*

It is worth noting that proposed methods do not require permanent storage, so the sole aspect to evaluate is the memory consumption. Indeed, the variations in the k-value have been identified as the primary parameter that impacts resource consumption. When the k-value is increased, a higher number of records need to be temporary stored to generate an output and therefore the memory consumption also increases.

A higher ℓ-value dynamically creates more categories based on the query's classification. During our tests, a maximum of 194505 categories were used in a tree with a depth of 13. Notice that a different dataset will result in different categories. However, ℓ-value has a lower impact than k-value as our proposal is able to generate anonymized queries, not only with elements from the same category but also with elements from the same branch, and therefore not all the categories need to remain full all the time to be able to generate an output in contrast to our previous proposal.

This effect can be seen on Table 2, which compares the remaining users per tree level between our previous proposal (Pàmies-Estrems et al., 2020) and the current one, once all the sample queries were processed. Using both proposals some users cannot leave the system since a suitable query is not found for them. However, as the current proposal is able to use all the sets from the current branch this number is greatly reduced. Table 2 shows the results corresponding to an initial k-value of 3 and ℓ-value of 11. However, similar results are found on all combinations of tested k and ℓ values.

Table 2
Remaining users per level once all the sample queries were processed. These are the users that cannot leave the system since a suitable query is not found for them. Table shows the results of using an initial k of 3 and ℓ of 11. However, similar results are found using different k and ℓ values.

Level	Base algorithm	Self-adjusting algorithm
1	22	0
2	2 899	5
3	35 004	48
4	155 779	65
5	280 851	80
6	237 149	28
7	237 945	11
8	181 947	17
9	116 852	8
10	54 647	1
11	22 924	0
Total	1 326 019	263

We can observe that the k value has a similar impact on resource consumption, regardless of the value of ℓ used. Results of maximum memory consumption for each tested combination of parameters can be seen in Fig. 14.

All the presented algorithms: *anonymizer* and both *de-anonymizer* exhibit a comparable memory usage pattern. Also the reported memory consumption should remain stable irrespective of the quantity of logs they handle as the system reaches a dynamic balance.

5.5.6. *Efficiency*

As previously discussed, a lightweight method has been developed, which enables fast processing of logs while minimizing resource consumption.

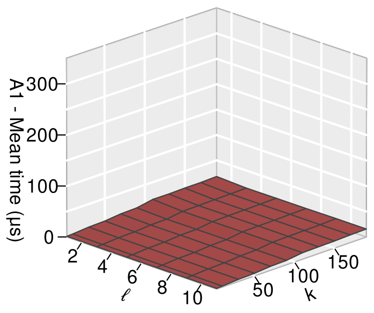
Examining the proposed algorithms, it can be seen that percentage of output queries and delay remain stable throughout the execution of the proposed tests, for each combination of k and ℓ-values.

Additionally, memory consumption and processing speed is dependent mainly on the value of k. It introduces a slight variation on the efficiency, which remains the same for different test with the same k-value. Finally ℓ, also has an effect over memory consumption and processing speed, but it is the least noticeable overall.

Upon examining the suggested algorithm, it becomes apparent that each log undergoes processing only one time, which enables for the efficiency of the algorithm to be compared to that of established singly-linked list traversal algorithms. The time complexity of the algorithm is directly proportional to the input and can be expressed as $\mathcal{O}(n)$, indicating linear growth.

5.5.7. *Transparency*

To operate, the system necessitates a stream of classified query logs, which can be sourced from the Service. If solely unclassified logs are accessible, a classification micro-service, as demonstrated in Ref. Pàmies-Estrems et al. (2016), can be integrated into the Service architecture. In case already classified logs are obtainable, they can be utilized without any additional alteration. The system produces an anonymized log stream that preserves the original format. From the standpoint of a preexisting client, the resulting output will be indiscernible from the original, thus ensuring complete transparency.



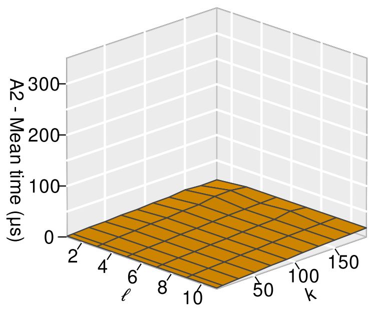
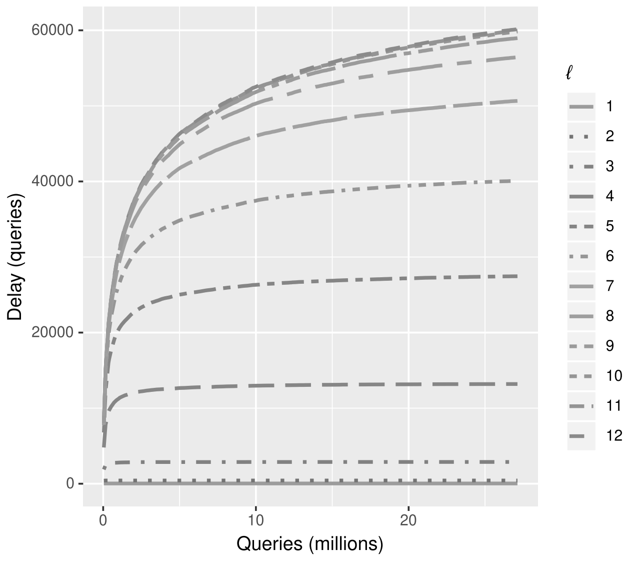


Fig. 12. The mean microseconds per query for the de-anonymizers was also analyzed. Both de-anonymizers yielded speed results similar to those of the anonymizer.



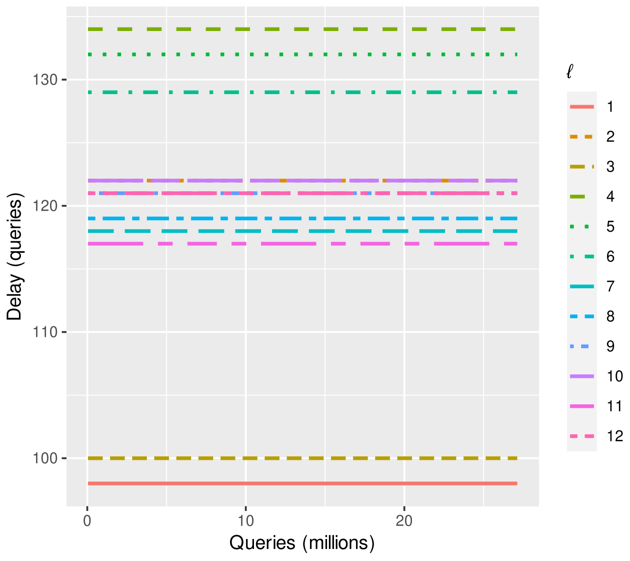
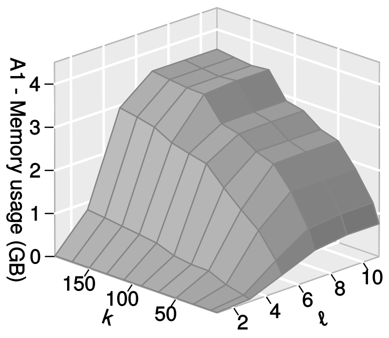
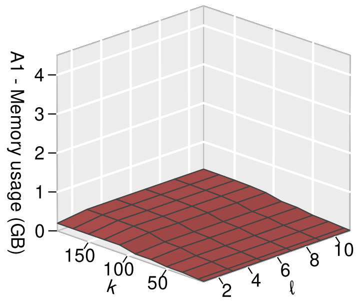


Fig. 13. The delay of queries is represented as the average count of other queries that join the system between the entrance and exit of a particular query. On our previous proposal (left), once the categories are filled, the output becomes stable. On the current one (right) those times remain low and constant from the start. Note the different scale on the y-axes.





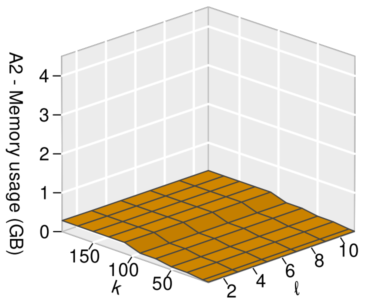


Fig. 14. Maximum memory consumption. Top left figure shows our previous *anonymizer* results. In comparison, our current *anonymizer* (top right) shows a much lower memory usage, which is also similar for the current *de-anonymizer* (bottom). k-value is the primary parameter that impacts memory consumption. Due to the use of branch matching, ℓ has a much lower impact.

6. Conclusion

We have presented an automated proposal for sanitizing query logs. The new construction can enhance a wide variety of applications, in order to process the metadata of queries contained in internet services. Our proposal can be embedded into existing services, upon server-side logs, without the need of modifying legacy systems. It relies on the removal of direct identifiers and their re-assignation to similar categories. Our approach aims as well at maintaining high levels of utility, assur-ing that user privacy is respected while reducing resource consumption at the server-side. Parameterization is assured as well. Two primary parameters, k and ℓ, are proposed, to adjust both privacy and specificity levels. Additional parameters could be envisioned, in order to adjust to satisfy some other constraints.

Two different algorithms have been used to test the de-anonymization of the protected data. The process has been carried out based on the most advantageous scenario for the attacker. This way, we assume the worst case scenario in which an attacker who is aware of our algorithms, and with access to the entire data stream generated, learns and applies the specific k and ℓ parameters. To verify the operation in various scenarios, the tests have been carried out using several values of k and ℓ. In any case, the attacker never has access to the origi-nal logs. The attacker manages to re-identify a maximum of 13.19%of the initial logs while using the lowest values of k and ℓ and the best available attack algorithm. By increasing either of these parameters we found that the re-identification drops rapidly to less than 1% for val-ues of k greater than 30. When using small values of ℓ in combination with small values of k the probability of re-identification is the high-est, although using values of ℓ greater than 3 limits greatly its effect on re-identification. Utility of logs has been assessed after anonymization using Earth Mover's Distance metric to compare differences between the original and anonymized data. Using a single depth level we obtain a utility loss of 43.26% but this amount decreases quickly, reaching less than 1% for ℓ-values of eight or more.

Perspectives for future work include the evaluation of learning attacks, to ensure that under no circumstances an attacker can seriously compromise users' privacy. Learning attacks targeting the classification system used by the anonymizer could be based on artificial intelligence, to allow an improved classification of the logs. It may also be interesting to conduct further experiments focused on improving anonymization using the moment in time when the queries were made. We have not carried out any experiments in this sense and it could be interesting to consider the time variable to achieve an even more optimal pro-tection scheme. Up to now, our tests have been carried out using a sample of query logs corresponding to a web search engine, validating the suitability of the proposal in this environment. In its fastest configuration, the proposal can handle the equivalent of more than nine times *Google*'s average load on a desktop computer. In addition, results show that delay, percentage of output queries and use of memory allow the application of this proposal in other contexts where these other factors are prioritized, i.e. environments with a much lower amount of logs per second. A final stage could also be added to the system, which would be in charge of validating the results prior to their publication, prevent-ing the dissemination of results that puts the desired level of privacy at risk.

Data availability

The evaluation of our work, together with all the implementation code, has also been released online, in a companion GitHub repository at https://github.com/dpamies/saps-qls.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments to improve the quality of the paper. We thank, as well, Jordi Castellà-Roca for fruitful discussions and suggestions on early versions of our work.

References

Adar, E., 2007. User 4xxxxx9: anonymizing query logs. In: Workshop on Query Log Analysis at the 16th World Wide Web Conference.

Agichtein, E., Brill, E., Dumais, S., 2006. Improving web search ranking by incorporating user behavior information. In: Proceedings of the 29th Annual ACM SIGIR Conference. Seattle, Washington, USA. ISBN 1-59593-369-7. ACM, pp. 19–26. http://portal.acm.org/citation.cfm?id=1148170.1148177.

Al-Rfou', R., Jannen, W., Patwardhan, N., 2012. TrackMeNot-so-good-after-all. arXiv:211.0320.

AOL, I., 2006. AOL keyword searches. http://dontdelete.com/default.asp.

Arampatzis, A., Efraimidis, P., Drosatos, G., 2012. A query scrambler for search privacy on the Internet. Inf. Retr. (ISSN 1386-4564), 1–23. https://doi.org/10.1007/s10791-012-9212-1.

Astolfi, F., Kroese, J., Van Oorschot, J., 2015. I2p - the invisible internet project. Web technology report. Media Technology, Leiden University.

Balsa, E., Troncoso, C., Díaz, C., 2012. OB-PWS: obfuscation-based private web search. In: IEEE Symposium on Security and Privacy. SP 2012, 21-23 May 2012, San Francisco, California, USA.

Bar-Ilan, J., 2007, May 2007. Access to query logs — an academic researcher's point of view. In: Amitay, E., Murray, G.C., Teevan, J. (Eds.), Query Log Analysis: Social and Technological Challenges. A Workshop at the 16th International World Wide Web Conference (WWW 2007).

Barbaro, M., Zeller, T., Hansell, S., 2006. A face is exposed for aol searcher no. 4417749. N.Y. Times 9 (2008). 8For, http://mrl.nyu.edu/~dhowe/TrackMeNot/NYTimes_AOL_Exposed.htm.

Batet, M., Erola, A., Sánchez, D., Castellà-Roca, J., 2013. Utility preserving query log anonymization via semantic microaggregation. Inf. Sci. 242, 49–63. https://doi.org/10.1016/j.ins.2013.04.020.

Batet, M., Erola, A., Sánchez, D., Castellà-Roca, J., 2014. Semantic anonymisation of set-valued data. In: ICAART 2014 - Proceedings of the 6th International Conference on Agents and Artificial Intelligence, vol. 1. ESEO, Angers, Loire Valley, France, 6–8 March, 2014, pp. 102–112.

Beitzel, S.M., Jensen, E.C., Chowdhury, A., Grossman, D., Frieder, O., 2004. Hourly analysis of a very large topically categorized web query log. In: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR '04, New York, NY, USA. ISBN 1-58113-881-4. ACM, pp. 321–328. http://doi.acm.org/10.1145/1008992.1009048.

Berthold, O., Federrath, H., Köpsell, S., 2001. Web mixes: a system for anonymous and unobservable Internet access. In: International Workshop on Designing Privacy Enhancing Technologies: Design Issues in Anonymity and Unobservability. New York, NY, USA. Springer-Verlag New York, Inc. ISBN 3-540-41724-9, pp. 115–129. http://dl.acm.org/citation.cfm?id=371931.371983.

Bohme, R., Danezis, G., Diaz, C., Kapsell, S., Pfitzmann, A., 2004. Mix cascades vs. peer-to-peer: is one concept superior? In: Privacy Enhancing Technologies (PET 2004).

Bondi, A.B., 2000. Characteristics of scalability and their impact on performance. In: Proceedings of the 2nd International Workshop on Software and Performance. ACM, pp. 195–203.

Bondia-Barcelo, J., Castellà-Roca, J., Viejo, A., July 18-21, 2016. Building privacy-preserving search engine query logs for data monetization. In: 2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing. Toulouse, France, July 18-21, 2016, pp. 390–397.

Brenes, D.J., Gayo-Avello, D., 2009. Stratified analysis of aol query log. Inf. Sci. (ISSN 0020-0255) 179, 1844–1858. https://doi.org/10.1016/j.ins.2009.01.027. http://portal.acm.org/citation.cfm?id=1523512.1523572.

Carpineto, C., Romano, G., 2013. Semantic search log k-anonymization with generalized k-cores of query concept graph. In: Advances in Information Retrieval - 35th European Conference on IR Research. ECIR 2013, Moscow, Russia, March 24-27, 2013. Proceedings, pp. 110–121.

Castellà-Roca, J., Viejo, A., Herrera-Joancomartí, J., 2009. Preserving user's privacy in web search engines. Comput. Commun. 32 (13–14), 1541–1551. https://doi.org/10.1016/j.comcom.2009.05.009.

Center for Democracy and Technology, 2007. Search privacy practices: a work in progress. https://cdt.org/wp-content/uploads/privacy/20070808searchprivacy.pdf.

Chaum, D.L., 1981. Untraceable electronic mail, return addresses, and digital pseudonyms. Commun. ACM (ISSN 0001-0782) 24 (2), 84–90. https://doi.org/10.1145/358549.358563. http://doi.acm.org/10.1145/358549.358563.

Chor, B., Goldreich, O., Kushilevitz, E., Sudan, M., 1995. Private information retrieval. In: Proceedings of IEEE 36th Annual Foundations of Computer Science, vol. 45(6), pp. 41–50.

Chor, B., Gilboa, N., Naor, M., 1997. Private Information Retrieval by Keywords.

Chow, R., Golle, P., 2009. Faking contextual data for fun, profit, and privacy. In: Proceedings of the 8th ACM Workshop on Privacy in the Electronic Society – WPES'09, pp. 105–108.

Cooper, A., 2008. A survey of query log privacy-enhancing techniques from a policy perspective. ACM Trans. Web (ISSN 1559-1131) 2 (4), 19:1–19:27. https://doi.org/10.1145/1409220.1409222. http://doi.acm.org/10.1145/1409220.1409222.

Cranor, L.F., Guduru, P., Arjula, M., 2006. User interfaces for privacy agents. ACM Trans. Comput.-Hum. Interact. (ISSN 1073-0516) 13 (2), 135–178. https://doi.org/10.1145/1165734.1165735. http://doi.acm.org/10.1145/1165734.1165735.

Cranor, L.F., Egelman, S., Sheng, S., McDonald, A.M., Chowdhury, A., 2008. P3p deployment on websites. Electron. Commer. Res. Appl. 7 (3), 274–293.

Danezis, G., Diaz, C., 2008. A survey of anonymous communication channels. Technical Report MSR-TR-2008-35. Microsoft. https://www.microsoft.com/en-us/research/publication/a-survey-of-anonymous-communication-channels/.

Danezis, G., Diaz, C., Syverson, P., 2009. Systems for anonymous communication. In: CRC Cryptography and Network Security Series. Chapman Hall/CRC.

De Capitani di Vimercati, S., Foresti, S., Livraga, G., Samarati, P., 2011. Foundations of security analysis and design vi. In: Aldini, A., Gorrieri, R. (Eds.), Foundations of Security Analysis and Design VI, chapter Protecting Privacy in Data Release. Springer-Verlag, Berlin, Heidelberg. ISBN 978-3-642-23081-3, pp. 1–34. http://dl.acm.org/citation.cfm?id=2028200.2028202.

Dingledine, R., Mathewson, N., Syverson. Tor, P., 2004. The second-generation onion router. In: Proceedings of the 13th Conference on USENIX Security Symposium, vol. 13. SSYM'04, Berkeley, CA, USA. USENIX Association, p. 21. http://dl.acm.org/citation.cfm?id=1251375.1251396.

Dini, G., Perazzo, P., 2012. Uniform obfuscation for location privacy. In: Data and Applications Security and Privacy XXVI: 26th Annual IFIP WG 11.3 Conference. DBSec 2012, Paris, France, July 11-13, 2012. Proceedings 26. Springer, pp. 90–105. 2012.

Domingo-Ferrer, J., Torra, V., 2001. A Quantitative Comparison of Disclosure Control Methods for Microdata. Elsevier, pp. 111–133.

Domingo-Ferrer, J., Solanas, A., Castellà-Roca, J., 2009. h(k)-private information retrieval from privacy-uncooperative queryable databases. J. Online Inf. Rev. 33 (4), 1468–1527.

Dwork, C., 2006. Differential privacy. In: Automata, Languages and Programming, 33rd International Colloquium. ICALP 2006, Venice, Italy, July 10–14, 2006, Proceedings, Part II, pp. 1–12.

Electronic Privacy Information Center (EPIC), 2000. Pretty poor privacy: an assessment of p3p and internet privacy. http://epic.org/reports/prettypoorprivacy.html.

Erola, A., 2013. Contributions to privacy in Web Search Engines. Universitat Rovira i Virgili. http://hdl.handle.net/10803/130934.

Erola, A., Castellà-Roca, J., 2013. Using search results to microaggregate query logs semantically. In: Data Privacy Management and Autonomous Spontaneous Security - 8th International Workshop, DPM 2013, and 6th International Workshop. SETOP 2013, Egham, UK, September 12-13, 2013, Revised Selected Papers, pp. 148–161.

Erola, A., Castellà-Roca, J., Navarro-Arribas, G., Torra, V., 2011a. Semantic microaggregation for the anonymization of query logs using the open directory project. SORT, 41–58.

Erola, A., Castellà-Roca, J., Viejo, A., Mateo-Sanz, J.M., 2011b. Exploiting social networks to provide privacy in personalized web search. J. Syst. Softw. 84 (9), 1734–1745.

Europe, C., 2016. 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. In: General Data Protection Regulation. January 2016.

Foundation, E.F., 2009. Aol's massive data leak. http://w2.eff.org/Privacy/AOL/.

Gaber, M.M., Zaslavsky, A., Krishnaswamy, S., 2005. Mining data streams: a review. SIGMOD Rec. (ISSN 0163-5808) 34 (2), 18–26. https://doi.org/10.1145/1083784.1083789. http://doi.acm.org/10.1145/1083784.1083789.

Goldschlag, D., Reed, M., Syverson, P., 1999. Onion routing for anonymous and private Internet connections. Commun. ACM 42, 39–41.

Hansell, S., 2006. Increasingly, internet's data trail leads to court.

He, Y., Naughton, J.F., 2009. Anonymization of set-valued data via top-down, local generalization. Proc. VLDB Endow. 2 (1), 934–945. https://doi.org/10.14778/1687627.1687733.

Hochheiser, H., 2002. The platform for privacy preference as a social protocol: an examination within the u.s. policy context. ACM Trans. Internet Technol. (ISSN 1533-5399) 2 (4), 276–306. https://doi.org/10.1145/604596.604598. http://doi.acm.org/10.1145/604596.604598.

Hong, Y., He, X., Vaidya, J., Adam, N., Atluri, V., 2009. Effective anonymization of query logs. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management. CIKM '09, New York, NY, USA. ACM. ISBN 978-1-60558-512-3, pp. 1465–1468. http://doi.acm.org/10.1145/1645953.1646146.

Hundepool, A., Domingo-Ferrer, J., Franconi, L., Giessing, S., Lenz, R., Longhurst, J., Nordholt, E.S., Seri, G., Wolf, P., 2010. Handbook on Statistical Disclosure Control. ESSnet on Statistical Disclosure Control.

Jones, R., Rey, B., Madani, O., Greiner, W., 2006. Generating query substitutions. In: WWW '06: Proceedings of the 15th International Conference on World Wide Web. New York, NY, USA. ACM. ISBN 1-59593-323-9, pp. 387–396. http://portal.acm.org/citation.cfm?id=1135777.1135835.

Jones, R., Kumar, R., Pang, B., Tomkins, A., 2007. "I know what you did last summer": query logs and user privacy. In: CIKM '07: Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management. New York, NY, USA. ACM. ISBN 978-1-59593-803-9, pp. 909–914. http://portal.acm.org/citation.cfm?id=1321440.1321573&coll=GUIDE&dl=GUIDE.

Khan, R., Islam, M.A., Ullah, M., Aleem, M., Iqbal, M.A., 2019. Privacy exposure measure: a privacy-preserving technique for health-related web search. J. Medical Imag. Health Inf. (ISSN 2156-7018) 9 (6), 1196–1204. https://doi.org/10.1166/jmihi.2019.2709. https://www.ingentaconnect.com/content/asp/jmihi/2019/00000009/00000006/art00020.

Khan, R., Ahmad, A., Alsayed, A.O., Binsawad, M., Islam, M.A., Ullah, M., 2020. Qupid attack: machine learning-based privacy quantification mechanism for PIR protocols in health-related web search. Sci. Program. 2020, 8868686. https://doi.org/10.1155/2020/8868686.

Khan, R., Ullah, M., Khan, A., Uddin, M.I., Al-Yahya, M., Aziz, F., 2021. Nn-qupid attack: neural network-based privacy quantification model for private information retrieval protocols. Complexity (ISSN 1076-2787) 2021. https://doi.org/10.1155/2021/6651662.

Khan, R., Ullah, M., Shafi, B., Ihsan, I., 2023. A Survey on Performance Evaluation Mechanisms for Privacy-Aware Web Search Schemes. IGI Global. ISBN 9781668469149, pp. 26–45.

Kodeswaran, P.A., Viegas, E., 2009. Applying differential privacy to search queries in a policy based interactive framework. In: Proceeding of the ACM First International Workshop on Privacy and Anonymity for Very Large Databases. CIKM-PAVLAD 2009, Hong Kong, China, November 6, 2009, pp. 25–32.

Kolmogorov, A., 1933. Sulla determinazione empirica di una legge di distribuzione. G. Ist. Ital. Attuari 4, 83–91.

Korolova, A., Kenthapadi, K., Mishra, N., Ntoulas, A., 2009. Releasing search queries and clicks privately. In: Quemada, J., León, G., Maarek, Y.S., Nejdl, W. (Eds.), WWW. ACM. ISBN 978-1-60558-487-4, pp. 171–180. https://dl.acm.org/citation.cfm?doid=1526709.1526733.

Kramár, T., Barla, M., Bieliková, M., 2013. Personalizing search using socially enhanced interest model, built from the stream of user's activity. J. Web Eng. (ISSN 1540-9589) 12 (1–2), 65–92. http://dl.acm.org/citation.cfm?id=2481562.2481565.

Krempl, G., Žliobaite, I., Brzeziński, D., Hüllermeier, E., Last, M., Lemaire, V., Noack, T., Shaker, A., Sievi, S., Spiliopoulou, M., Stefanowski, J., 2014. Open challenges for data stream mining research. ACM SIGKDD Explor. Newsl. (ISSN 1931-0145) 16 (1), 1–10. https://doi.org/10.1145/2674026.2674028. http://doi.acm.org/10.1145/2674026.2674028.

Kushilevitz, E., Ostrovsky, R., 1997. Replication is not needed: single database, computationally-private information retrieval. In: Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science. IEEE Press, pp. 364–373.

Leung, K.W.-T., Lee, D.L., Ng, W., Fung, H.Y., 2012. A framework for personalizing web search with concept-based user profiles. ACM Trans. Internet Technol. (ISSN 1533-5399) 11 (4), 17:1–17:29. https://doi.org/10.1145/2109211.2109214. http://doi.acm.org/10.1145/2109211.2109214.

Levine, B.N., Hordes, C. Shields, 2002. A multicast based protocol for anonymity. J. Comput. Secur. 10, 213–240.

Li, N., Li, T., t-closeness, S. Venkatasubramanian, 2006. Privacy beyond k-anonymity and l-diversity. In: 2007 IEEE 23rd International Conference on Data Engineering. IEEE, pp. 106–115.

Lindell, Y., Waisbard, E., 2010. Private web search with malicious adversaries. In: Proceedings of the 10th International Conference on Privacy Enhancing Technologies – PETS'10, pp. 220–235.

Liu, X., Deng, R.H., Yang, Y., Tran, H.N., Zhong, S., 2018. Hybrid privacy-preserving clinical decision support system in fog–cloud computing. Future Gener. Comput. Syst. 78, 825–837.

Machanavajjhala, A., Kifer, D., Gehrke, J., L-diversity, M. Venkitasubramaniam, 2007. Privacy beyond k-anonymity. ACM Trans. Knowl. Discov. Data (ISSN 1556-4681) 1 (1). https://doi.org/10.1145/1217299.1217302. http://doi.acm.org/10.1145/1217299.1217302.

Majeed, A., Khan, S., Hwang, S.O., 2022. Towards privacy preservation using clustering based anonymization: recent advances and future research outlook. IEEE Access.

Meng, X., Xu, Z., Chen, B., Zhang, Y., 2016. Privacy-preserving query log sharing based on prior n-word aggregation. In: 2016 IEEE Trustcom/BigDataSE/ISPA. Tianjin, China, August 23–26, 2016, pp. 722–729.

Michael, M., Moreira, J.E., Shiloach, D., Wisniewski, R.W., 2007. Scale-up x scale-out: a case study using nutch/lucene. In: Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International. IEEE, pp. 1–8.

Mills, E., 2006. Aol sued over web search data release. http://news.cnet.com/8301-10784_3-6119218-7.html.

Moore, R., 1996. Controlled data swapping techniques for masking public use microdata sets. (unpublished manuscript).

Murugesan, M., Clifton, C., 2009. Providing privacy through plausibly deniable search. In: Proceedings of the SIAM International Conference on Data Mining, SDM 2009, April 30 - May 2, 2009. Sparks, Nevada, USA, pp. 768–779.

Navarro-Arribas, G., Torra, V., 2009. Tree-based microaggregation for the anonymization of search logs. In: Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology, vol. 03. WI-IAT '09, Washington, DC, USA. IEEE Computer Society. ISBN 978-0-7695-3801-3, pp. 155–158.

Navarro-Arribas, G., Torra, V., 2014. Rank swapping for stream data. In: Torra, V., Narukawa, Y., Endo, Y. (Eds.), Modeling Decisions for Artificial Intelligence. In: Lecture Notes in Computer Science, vol. 8825. Springer International Publishing, Switzerland. ISBN 978-3-319-12053-9, pp. 217–226.

Navarro-Arribas, G., Torra, V., Erola, A., Castellà-Roca, J., 2012. User k-anonymity for privacy preserving data mining of query logs. Inf. Process. Manag. 48 (3), 476–487. https://doi.org/10.1016/j.ipm.2011.01.004.

Nissenbaum, H.F., Trackmenot, H. Daniel, 2009. Resisting Surveillance in Web Search. Lessons from the Identity Trail: Anonymity, Privacy, and Identity in a Networked Society. Oxford University Press, Oxford. https://ssrn.com/abstract=2567412.

Ostrovsky, R., Skeith-III, W.E., 2007. A survey of single-database pir: techniques and applications. In: Lecture Notes in Computer Science, vol. 4450. Springer, Berlin and Heidelberg, pp. 393–411.

Pàmies-Estrems, D., Castellà-Roca, J., Viejo, A., 2016. Working at the web search engine side to generate privacy-preserving user profiles. Expert Syst. Appl. 64 (C), 523–535. https://doi.org/10.1016/j.eswa.2016.08.033.

Pamies-Estrems, D., Kaaniche, N., Laurent, M., Castella-Roca, J., Garcia-Alfaro, J., 2018. Lifelogging protection scheme for Internet-based personal assistants. In: Data Privacy Management, Cryptocurrencies and Blockchain Technology (CBT 2018 and DPM 2018). Springer, pp. 1–10.

Papadopoulos, P., Papadogiannakis, A., Polychronakis, M., Zarras, A., Holz, T., Markatos, E.P., 2013. k-subscription: privacy-preserving microblogging browsing through obfuscation. In: Annual Computer Security Applications Conference. ACSAC '13, New Orleans, LA, USA, December 9–13, 2013, pp. 49–58.

Peddinti, S.T., Saxena, N., 2010. On the privacy of web search based on query obfuscation: a case study of trackmenot. In: Proceedings of the 10th International Conference on Privacy Enhancing Technologies – PETS'10, pp. 19–37.

Petit, A., Cerqueus, T., Mokhtar, S.B., Brunie, L., Kosch, H., 2015. PEAS: private, efficient and accurate web search. In: 2015 IEEE TrustCom/BigDataSE/ISPA, vol. 1. Helsinki, Finland, August 20–22, 2015, pp. 571–580.

Poblete, B., Spiliopoulou, M., Baeza-Yates, R.A., 2007. Website privacy preservation for query log publishing. In: Bonchi, F., Ferrari, E., Malin, B., Saygin, Y. (Eds.), PinKDD. In: Lecture Notes in Computer Science, vol. 4890. Springer, Berlin, Heidelberg. ISBN 978-3-540-78477-7, pp. 80–96. https://link.springer.com/chapter/10.1007%2F978-3-540-78478-4_5.

Purdam, K., Elliot, M., 2007. A case study of the impact of statistical disclosure control on data quality in the individual uk samples of anonymised records. Environ. Plan. A 39 (5), 1101–1118.

Pàmies-Estrems, D., Castellà-Roca, J., Garcia-Alfaro, J., 2020. A real-time query log protection method for web search engines. IEEE Access 8, 87393–87413. https://doi.org/10.1109/ACCESS.2020.2992012. https://ieeexplore.ieee.org/document/9085377.

Reay, I., Dick, S., Miller, J., 2009. A large-scale empirical study of p3p privacy policies: stated actions vs. legal obligations. ACM Trans. Web (ISSN 1559-1131) 3 (2), 6:1–6:34. https://doi.org/10.1145/1513876.1513878. http://doi.acm.org/10.1145/1513876.1513878.

Reiss, S.P., 1980. Practical data-swapping: the first steps. In: Proceedings of the 1980 IEEE Symposium on Security and Privacy. Oakland, California, USA, April 14–16, 1980, pp. 38–45.

Reiter, M., Rubin, A., 1998. Crowds: anonymity for web transactions. ACM Trans. Inf. Syst. Secur. 1 (1), 66–92.

Romero-Tris, C., 2014. Client-side Privacy-enhancing Technologies in Web Search. Universitat Rovira i Virgili. http://hdl.handle.net/10803/284036.

Romero-Tris, C., Castellà-Roca, J., Viejo, A., 2011a. Multi-party private web search with untrusted partners. In: 7th International ICST Conference on Security and Privacy in Communication Networks – SecureComm'11.

Romero-Tris, C., Viejo, A., Castellà-Roca, J., 2011b. Improving query delay in private web search. In: 3PGCIC, pp. 200–206.

Romero-Tris, C., Viejo, A., Castellà-Roca, J., 2015. Multi-party methods for privacy-preserving web search: survey and contributions. In: Advanced Research in Data Privacy. Springer, pp. 367–387.

Rubner, Y., Tomasi, C., Guibas, L.J., 2000. The Earth mover's distance as a metric for image retrieval. Int. J. Comput. Vis. 40 (2), 99–121.

Saint-Jean, F., Johnson, A., Boneh, D., Feigenbaum, J., 2007. Private web search. In: Proceedings of the 2007 ACM Workshop on Privacy in Electronic Society – WPES'07, pp. 84–90.

Samarati, P., 2001. Protecting Respondents' Identities in Microdata Release. IEEE Transactions on Knowledge and Data Engineering 13, 1010–1027. https://doi.org/10.1109/69.971193.

Sánchez, D., Castellà-Roca, J., Viejo, A., 2013. Knowledge-based scheme to create privacy-preserving but semantically-related queries for web search engines. Inf. Sci. 218, 17–30. https://doi.org/10.1016/j.ins.2012.06.025.

Sánchez, D., Batet, M., Viejo, A., Rodriguez-Garcia, M., Castellà-Roca, J., 2018. A semantic-preserving differentially private method for releasing query logs. Inf. Sci. 460–461, 223–237. https://doi.org/10.1016/j.ins.2018.05.046.

Cameron, R.T. Scott, Pickersgill, Andrew, 2014. New ways for turning data into dollars now. https://www.mckinsey.com/business-functions/marketing-and-sales/our-insights/new-ways-for-turning-data-into-dollars-now.

Shea, P., 2010. Book review: 'Click: what millions of people are doing online and why it matters' by Bill Tancer. ELearn Mag. 2010 (1), 8. http://dblp.uni-trier.de/db/journals/elearn/elearn2010.html#Shea10.

Shen, X., Tan, B., Ucair, C. Zhai, 2005. Capturing and exploiting context for personalized search. In: Proceedings of the ACM SIGIR 2005 Workshop on Information Retrieval in Context (IRiX). Salvador, Brazil. Citeseer, p. 45.

Shen, X., Tan, B., Zhai, C., 2007. Privacy protection in personalized search. SIGIR Forum 41 (1), 4–17. https://doi.org/10.1145/1273221.1273222. http://portal.acm.org/citation.cfm?id=1273222&dl=GUIDE&coll=GUIDE&CFID=17786915&CFTOKEN=37653058.

Silvestri, F., 2010. Mining query logs: turning search usage data into knowledge. Found. Trends Inf. Retr. (ISSN 1554-0669) 4 (1&#8212;2), 1–174. https://doi.org/10.1561/1500000013.

Smirnov, N., 1948. Table for estimating the goodness of fit of empirical distributions. Ann. Math. Stat. 19 (2), 279–281.

Soghoian, C., 2007. The problem of anonymous vanity searches. I/S: A. J. Law Policy Inf. Soc. 3, 299.

Soghoian, C., 2012. The history of the do not track header. Slight paranoia.

Soria-Comas, J., Domingo-Ferrer, J., 2012. Probabilistic k-anonymity through microaggregation and data swapping. In: Fuzzy Systems (FUZZ-IEEE), 2012 IEEE International Conference on. IEEE, pp. 1–8.

T.G. Statistics, 2023. Seo and Google statistics. https://www.theglobalstatistics.com/seo-google-mobile-statistics-facts/.

Syverson, P., 2011. Practical vulnerabilities of the tor anonymity network. In: Advances in Cyber Security: Technology, Operation and Experiences. New York City, New York, USA. Fordham University Press.

Terrovitis, M., Mamoulis, N., Kalnis, P., 2008. Privacy-preserving anonymization of set-valued data. Proc. VLDB Endow. 1 (1), 115–125. https://doi.org/10.14778/1453856.1453874.

Torra, V., 2023. Data Privacy for Machine Learning and Statistics. Handbook of Statistics. Elsevier. https://www.sciencedirect.com/science/article/pii/S0169716123000275.

Torra, V., Domingo-Ferrer, J., 2001. Disclosure Control Methods and Information Loss for Microdata. Elsevier, pp. 91–110.

Truex, S., Baracaldo, N., Anwar, A., Steinke, T., Ludwig, H., Zhang, R., Zhou, Y., 2019. A hybrid approach to privacy-preserving federated learning. In: Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security, pp. 1–11.

Ullah, M., Khan, R., Islam, M.A., 2016a. Poshida, a protocol for private information retrieval. In: 2016 Sixth International Conference on Innovative Computing Technology (INTECH), pp. 464–470.

Ullah, M., Khan, R., Islam, M.A., 2016b. Poshida ii, a multi group distributed peer to peer protocol for private web search. In: 2016 International Conference on Frontiers of Information Technology (FIT), pp. 75–80.

Ullah, M., Islam, M.A., Khan, R., Aleem, M., Iqbal, M.A., 2019. Obsecure logging (oslo): A framework to protect and evaluate the web search privacy in health care domain. J. Medical Imag. Health Inf. (ISSN 2156-7018) 9 (6), 1181–1190. https://doi.org/10.1166/jmihi.2019.2708. https://www.ingentaconnect.com/content/asp/jmihi/2019/00000009/00000006/art00018.

Ullah, M., Khan, R., Inam Ul Haq, M., Khan, A., Alosaimi, W., Uddin, M.I., Alharbi, A., 2021. Multi-group obscure logging (mg-oslo) a privacy-preserving protocol for private web search. IEEE Access 9, 79005–79020. https://doi.org/10.1109/ACCESS.2021.3078431.

Ullah, M., Khan, R., Khan, I., Aslam, N., Aljameel, S., Inam Ul Haq, M., Islam, A., 2022. Profile aware obscure logging (paoslo): a web search privacy-preserving protocol to mitigate digital traces. Secur. Commun. Netw. 2022. https://doi.org/10.1155/2022/2109024.

Ullah, M., Abbas, A., Rukh, L., Ullah, K., Inam Ul Haq, M., 2023. State of the Art in Distributed Privacy-Preserving Protocols in Private Web Search. IGI Global. ISBN 9781668469149, pp. 1–25.

Viejo, A., Castellà-Roca, J., 2010. Using social networks to distort users' profiles generated by web search engines. Comput. Netw. 54 (9), 1343–1357.

Viejo, A., Castellà-Roca, J., Bernado, O., Mateo-Sanz, J.M., 2012a. Single-party private web search. In: Proceedings of the 2012 Tenth Annual International Conference on Privacy, Security and Trust (PST), PST '12. Washington, DC, USA. IEEE Computer Society. ISBN 978-1-4673-2323-9, pp. 1–8.

Viejo, A., Sánchez, D., Castellà-Roca, J., 2012b. Preventing automatic user profiling in web 2.0 applications. Knowl.-Based Syst. 36, 191–205. https://doi.org/10.1016/j.knosys.2012.07.001.

Westermann, B., Wendolsky, R., Pimenidis, L., Kesdogan, D., 2010. Cryptographic protocol analysis of an.on. In: Proceedings of the 14th International Conference on Financial Cryptography and Data Security. FC'10, Berlin, Heidelberg. Springer-Verlag. ISBN 3-642-14576-0, pp. 114–128. ISBN 3-642-14576-0, 978-3-642-14576-6.

Willenborg, L., De Waal, T., 2012. Elements of Statistical Disclosure Control, vol. 155. Springer Science & Business Media.

Xu, Y., Wang, K., Zhang, B., Chen, Z., 2007a. Privacy-enhancing personalized web search. In: WWW '07: Proceedings of the 16th International Conference on World Wide Web. New York, NY, USA. ACM. ISBN 978-1-59593-654-7, pp. 591–600. http://portal.acm.org/citation.cfm?id=1242652.

Xu, Y., Zhang, B., Wang, K., 2007b. Privacy-enhancing personalized web search. In: Proceedings of the 16th International Conference on World Wide Web. ACM Press, pp. 591–600.

Yang, J.-J., Li, J.-Q., Niu, Y., 2015. A hybrid solution for privacy preserving medical data sharing in the cloud environment. Future Gener. Comput. Syst. 43, 74–86.

Zhang, S., Yang, H., Singh, L., 2015. Applying epsilon-differential private query log releasing scheme to document retrieval. In: Proceedings of the 2nd International Workshop on Privacy-Preserving Information Retrieval Workshop PIR'15 2015. Santiago, Chile, August 13th, 2015, p. 2015.

Zhang, S., Yang, G.H., Singh, L., 2016. Anonymizing query logs by differential privacy. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR 2016, Pisa, Italy, July 17-21, 2016, pp. 753–756.