DisPA: an Intelligent Agent for Private Web Search

Marc Juarez[1] and Vicenç Torra[2]

[1] KU Leuven, Dept. of Electrical Engineering (ESAT), COSIC, iMinds,
Kasteelpark Arenberg, 3000 Leuven, Belgium
marc.juarez@esat.kuleuven.be,
[2] IIIA - Institut d'Investigació en Intel·ligència Artificial
CSIC - Consejo Superior de Investigaciones Científicas
Campus de la UAB, 08193 Bellaterra, Catalonia, Spain.
vtorra@iiia.csic.es

Abstract. Search queries can be used to infer preferences and interests of users. While search engines use this information for, among others, targeted advertising and personalization, these tasks can violate user's privacy. In 2006, after AOL disclosed the search queries of 650,000 users and some of them were re-identified, many Privacy Enhancement Technologies (PETs) have sought to solve this problem. The *Dissociating Privacy Agent* (DisPA), is a browser extension that acts as a proxy between the user and the search engine and semantically dissociates queries on real time. We show that DisPA increases the privacy of the user and hinders re-identification. We also propose an algorithm to measure and evaluate the privacy properties offered by DisPA.

Keywords: Privacy, Web Search, Search Personalization, Query Classification

1 Introduction

Web search has become an elementary task in the Internet and virtually everybody makes use of search engines to find information in a quick and effective way. Providers of search services log data related to their users and track them across the Web. The reason is twofold. Firstly, profiling is profitable for the provider who can exploit business opportunities by means of Marketing Research and Targeted Advertising [1]. Secondly, due to the growth of the Web, profiling is essential to improve ranking algorithms and offer a more efficient search [2–4].

The task of profiling consists in the collection of data about the user's web interaction. These data are stored in files at the server-side called "server logs" or "query logs". By applying data mining techniques on these logs, search providers extract traits of the user such as demographic aspects (e.g., age, gender or nationality) or main areas of interest, that are modelled as categories such as "Cinema" or "Football". Afterwards, the ranking algorithms rearrange the list of results to deliver first those that are more useful for the user according to his preferences [5].

This personalization can be beneficial because saves time to the user who otherwise would have to skim over the large list of results manually. Nevertheless, the indiscriminate logging of data raises privacy concerns with respect to social sorting and discrimination. As it has been shown several times in the past, potentially sensitive information can be inferred from search queries, such as sexual orientation, health status, or political beliefs [6].

A milestone in history of privacy breaches is the AOL search data leak in 2006 [7], when queries of approximately 650,000 users submitted over a 3-month period were disclosed [8]. Despite that AOL claimed to have anonymized the dataset by removing identifiers, journalists of the New York Times managed to link one of the logs to a real identity [9]. This was very remarkable as it proved that queries by themselves can uniquely identify an individual or, at least, reduce the search space considerably.

Several approaches are commonly taken to address this problem. First, the user can use cryptography-based solutions which provide strong privacy guarantees, but require the provider to integrate them in the backend [10]. Second, the user can connect to the service through an anonymous communication system that would provide him a different identity for each session. Finally, he might still be identifiable and obfuscate, either the content of the queries, or his search profile by means of dummy queries [11].

In this paper we describe a Privacy Enhancing Technology (PET) for web search that has been developed through the last two years [12, 13]. It tackles the problem from still another perspective that is characterized by taking into account search personalization. We assume that the user benefits from personalization and, for this reason, we strive for a trade-off between the utility (personalization) and the cost (privacy) of releasing data.

The rest of the paper is organized as follows. Section 2 reviews the state-of-the-art in private web search. In Section 3 we present our threat model and recall the basic operation of the approach we propose: the *Dissociating Privacy Agent* (DisPA for short) [12]. In Section 4 we detail the internal operations of DisPA. In Section 5 we present the different experiments conducted for the evaluation of the agent and show the results obtained. In Section 6 we point out the limitations of our work and bring some discussion points. The paper finishes with the conclusions and lines of future research in Section 7.

2 Related work

There exist several cryptography-based solutions for private search: private information retrieval (PIR) [14, 15], oblivious transfer (OT) protocols [16], and methods based on homomorphic cryptosystems (e.g., Paillier) [17]. These protocols provide strong privacy guarantees such as confidentiality of search terms and results. However, there are some drawbacks for their implementation in a real-world web search engine. For instance, they often assume cooperation by the provider. Search providers however do not have any incentives to implement costly protocols they cannot profit from, and thus the deployment of these solu-

tions is not realistic in practice. Further, given that search terms are encrypted, they render useless for personalization. Another important difficulty that makes them inconvenient is the computational complexity of these methods given the great size of the Web.

For these reasons, the problem is often relaxed towards more applicable schemes. Among these, we find two main different strategies: (i) to obfuscate the user's profile by submitting fake queries together with legitimate ones, and (ii) to hide the identity of the user in front of the search engine, so queries cannot be attributed to him. The former category is often called *obfuscation-based* techniques and it has been thoroughly studied [18–20, 19, 21–26]. We refer the reader to a deeper analysis of obfuscation techniques for more details [27].

Most of low-latency anonymous communications systems fall in the latter category. For instance, the user could employ The Onion Router (Tor) [28] to submit the query. The server would observe the IP of a different Tor exit node for each session, making it harder to link user's queries across sessions. This approach has two important limitations. First, as the AOL case demonstrated, queries by themselves can identify users independently of the communication's metadata. Second, due to the time overhead introduced by Tor, it deteriorates the usability of the service and therefore it is not suitable for a long-term solution.

Besides of the particular shortcomings that each of these approaches have, they have a drawback in common: all of them diminish the quality of server logs for profiling. Obfuscation-based techniques introduce false information about the preferences of the user, and anonymity networks induce the creation of a new server log for each session.

Our research fills this gap by proposing an intelligent agent that helps to protect the user's privacy, while preserving the utility of his profile for personalization. There are different strategies to achieve this goal in the literature [20, 29–31], and we also have found in recent publications very similar approaches to the ones we had presented for the development of DisPA [32, 33]. This shows a common interest of the research community towards the development of protocols that strive for a trade-off between utility and privacy in web search.

3 Threat Model and Fundamentals

We assume that the adversary is the search engine provider or a third party who has access to all server logs. The goal of the adversary is to extract new information about a targeted user out of the logs or, in the worse case from the user's point of view, to discover the real identity of the user.

We can model the adversary as a honest-but-curious adversary. This means a passive adversary who does not alter the functionality of the system but can eavesdrop queries and analyse them. Search providers fit in this model because they are interested in ensuring the availability and good quality of the service.

The adversary might as well have some auxiliary knowledge that enables re-identification. As it has been shown in the past, an adversary can use cross-correlation with multiple databases to uniquely identify an individual (e.g.,

Narayanan and Shmatikov showed it with popular movie databases [34]). We will discuss in more detail this problem in Section 6.

Before diving into the fundamentals of the agent we are going to describe our system model. A query is basically an HTTP(S) request from the user's browser to the search engine's web server. The URL field contains query terms and other user preferences encoded as URL parameters. The cookie field contains, among other domain-specific cookies, a cookie with a user's unique identifier, the so-called "cookie ID". The query terms are stored in a server log along with other connection-related information, such as the IP and the cookie ID.

We make the assumption that search engines only use cookies to identify users. This might be a strong assumption to hold, but the last version of Google's privacy policy and a recent study on log retention policies support it [35, 36]. Also, a recent study on the prevalence of "device fingerprinting", a new tracking technique that leverages information collected about devices for user identification, has not found evidence of the adoption of such technique in most popular search engines [37].

3.1 The Dissociating Privacy Agent

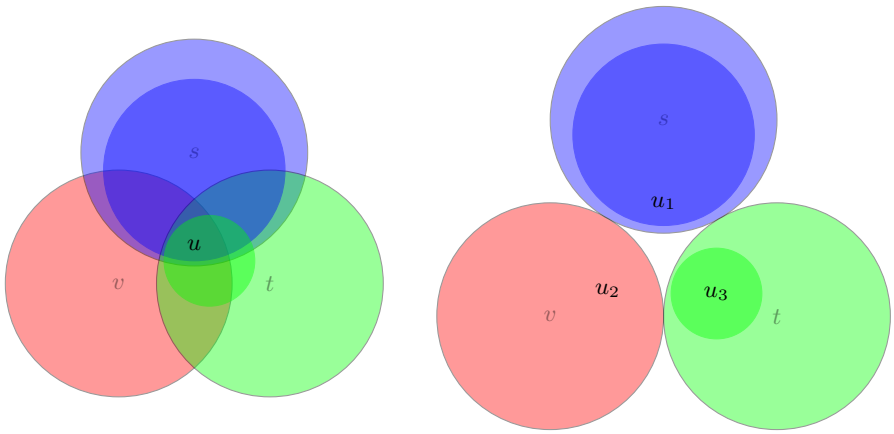
The underlying concept of our approach is "dissociation". Dissociation is based on the observation that users are multifaceted individuals, in the sense they are interested in various areas of knowledge, such as "Science", "Sports" or "Music". Let $C(u) = \{c_1, c_2, \ldots, c_n\} \subset \mathcal{C}$ be the set of categories of interest for a particular user u. Let U_c be the set of users interested in category c. Then, the anonymity set of u is defined as

$$A(u) := U_{c_1} \cap U_{c_2} \cap \ldots \cap U_{c_n}.$$

Our hypothesis is that, for a fine-grained taxonomy $\mathcal{C}$, $A(u)$ is likely to contain only user u. Put differently, the interests of the user define his identity and can be used to uniquely identify him.

The idea of dissociation is to break down the identity of the user into partial identities, each one of them grasping a fraction of his interests. We name these artificial identities as "virtual identities". If we consider each virtual identity as a different user, dissociation increases the probability of users sharing the same interests. It is trivial to prove that $|A(u)| \leq |A(u_i)|$, where u_i are the virtual identities of u after dissociation, for $i = 1, \ldots, n$.

To illustrate this we show in Figure 1 an example of dissociation. Each circle represents a set U_{c_i} and each user is represented with a letter: s, t, u, v. In Figure 1a we see that the anonymity set of u only contains u. We apply dissociation on u by creating a virtual identity for each of the set U_{c_i} for the three main facets: u_1, u_2, u_3. As a result, in Figure 1b we see how the new anonymity sets $A(u_i)$ have size 2.



(a) Before dissociation. (b) After dissociation.

Fig. 1: Venn diagrams showing the dissociation process for user u.

4 Design

In order to implement dissociation, we designed the Dissociating Privacy Agent (DisPA). DisPA is an intelligent agent, i.e., a piece of software that takes decisions on behalf of the user. The agent is implemented as a browser add-on and acts as a proxy between the browser and the web server.

To achieve dissociation, DisPA intercepts HTTP requests to the search engine. Then, the connection is bypassed through a query classifier. DisPA generates new tracking data for each possible classification outcome and replaces them in the HTTP request on real-time. To the eyes of the server, queries classified by DisPA in different categories appear as requests from different users and thus are logged into different files at the server-side (see Figure 2).

To keep consistency across different HTTP sessions, we define a context in the browser formed by: the jar of cookies, history of queries, history of clicked links, lists of results and the user-agent. This is intended to prevent the server from spotting similarities among sessions and link them to the same user.

As a result, the identity of the user is divided and it is harder to achieve re-identification by means of dissociated logs. Furthermore, note that dissociated logs are still useful for profiling as, by construction, they preserve partial but real user interests that search engines can extract and use for personalization.

4.1 Query Classification

A fundamental part of the agent is query classification. The user sends queries through the browser add-on's interface and the agent classifies them before sub-

mitting them to the search engine. DisPA uses the taxonomy of the Open Directory Project (ODP) to build a faceted search engine and classify queries quickly [38].

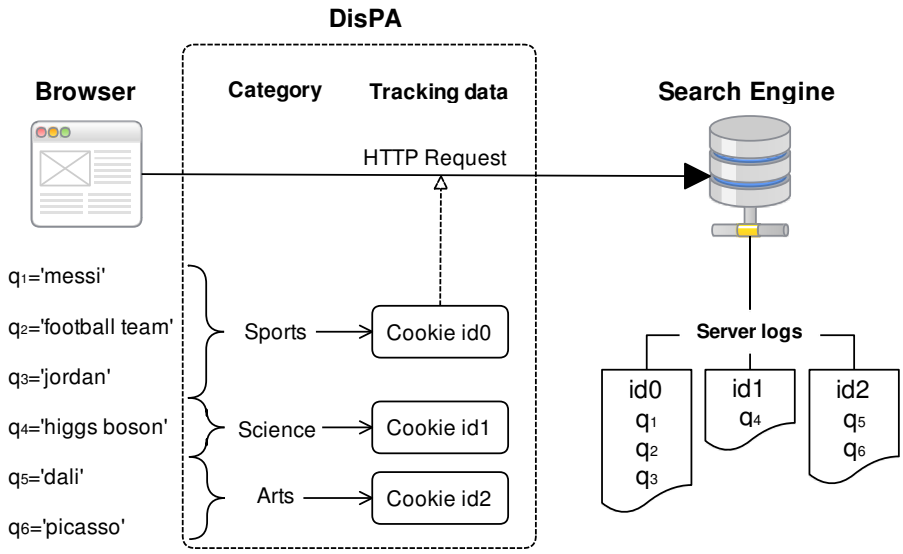


Fig. 2: Representation of the implementation of dissociation in DisPA.

The facets of the user are modelled as categories of the **first level** of the ODP tree which are:

Adult, Arts, Games, Shopping, Business, Health, Society, Computers, Home, News, Reference, Recreation, Sports, Science, Society.

In order to classify a query, we perform a faceted search in a local search engine, which is based on an inverse index of the documents in the ODP corpus. The outcome of the search is a vector with coordinates the number of hits of the query in each of the categories. That is, let q be a query, given the set of categories $C = \{c_1, c_2, \ldots, c_n\}$, the outcome of classification is a vector

$$(h_1, h_2, \ldots, h_n) \tag{1}$$

where h_i is the number of documents indexed in category c_i that are hit by query q, for $i = 1, \ldots, n$.

Given a query q_0, a very basic classifier can be defined as:

$$classify(q_0) := \arg\max_{c_i \in C} p(c_i \mid q_0), \tag{2}$$

where in our approach we estimate $p(c_i \mid q_0)$ by

$$p\left(c_i \mid q_0\right) \approx \frac{h_i(q_0)}{\sum_i h_i(q_0)}.$$

4.2 Personalized Query Classification

Frequently, queries are vague because they lack of context and polysemy of words introduce ambiguity in their interpretation. For example, the query "jordan" might allude to a basketball player, a mathematician, a river, or a country. Personalized Query Classification (PQC) is more challenging than plain QC in that it attempts to resolve ambiguity of queries according to subjective user intents.

In order to take advantage of search engine's personalization, the dissociation process must be consistent with the user's interests. In other words, we need to perform PQC so that a user mainly interested in basketball gets the query "jordan" in the basketball profile and obtains results related to "Michael Jordan".

The probabilistic model used to achieve PQC in DisPA is similar to the one presented by Cao et. al. [39]. We contributed with a novel approach based on inferring user's interests by means of the history of navigation instead of the clicking data. The idea is that DisPA, as a browser add-on can take advantage of its direct access to the browser profile.

Our model can be described in terms of a user u who wants to search for information related to query q. The search engine interprets q as belonging to a set of categories. Independently of this, the user accesses web pages w that might be classified in this set of categories. Thus, w depends on c. In Figure 3 we can see the graphical representation of our model.

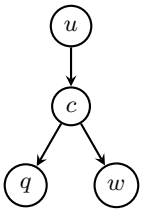


Fig. 3: Graphical model representation for PQC.

The main takeaway is that we keep the classification outcome for the last k web pages to estimate prior probabilities of the user being interested in each category – i.e., $p(u \mid c) \approx p(w_1, \ldots, w_k \mid c)$. Then, these priors are used to weight the final classification for a specific query q.

We skip the mathematical details and refer the interested reader to the original paper for a full specification [12]. The final classifier is

$$classify(q_0) = \arg\max_{c_i \in C} p(c_i \mid q_0)\, p(w_1, \ldots, w_k \mid c_i). \tag{3}$$

And we estimate $p(w_1, \ldots, w_k \mid c_i) \approx \frac{v_i}{|c_i|+v_i}$, where v_i is the number of visited sites classified in the i-th category of the ODP.

4.3 Filters of Queries

Our approach stems from the assumption that the user's identity is defined by his interests. DisPA enlarges anonymity sets and reduces the adversary's inference ability by dissociation. However, it is obvious that this is not sufficient to solve the problem. There are other types of queries that jeopardize user's privacy, for example:

1. Queries that identify the user by their own (e.g., a query containing unique identifiers or emails
2. Queries that contain named entities like names of locations or personal names (e.g., terms like "lilburn" and "arnold")

We characterize these types of queries as the most uncommon queries amongst the world of users in the search engine, as these are the ones that reveal more information. This fits in DisPA's model as people that are fond on rare topics are easier to identify. Even though their profiles are dissociated, the anonymity sets may remain very small because there are no other users interested in the same topics.

We approximate the popularity of a query by the number of results that our ODP-based classifier returns, that is the absolute frequency of the query in the ODP. Following the notation in the previous sections this corresponds to

$$f(q) \approx \sum_i h_i(q).$$

We set a threshold τ for the frequency, which can be initialized by submitting a very uncommon query locally. For subsequent queries, we test $f(q) < \tau$, and filter them out in case it is true. We also used the Stanford's CoreNLP library for Named Entity Recognition (NERQ) to recognize locations and personal names.

In order to filter a query out, the agent generates a new virtual identity exclusively for that particular query. This way we isolate these queries from the rest of profiles and cannot be used to neither extract new information nor to link the other dissociated logs of the user.

4.4 User specializations

One of the main flaws of our first implementation was that classification categories are fixed and did not take into account user's specializations. A user may

have a lot of facets but he might be interested in ones relatively more than in others. For example, compare queries sent from the computer at the work place and the ones that are sent from home.

In some cases, interests of the user are specialized. Then, most of queries are classified in one of DisPA's categories and dissociation makes no difference. As an example to illustrate this, imagine a user very keen on computers. His queries fall mostly in the category *"Computers"* and the other categories are barely used. As a result, the dissociation by means of the first level of the ODP has no effect and the agent fails in its attempt to protect the user.

An improvement on this first implementation consists in breaking down these specialized categories to include more specific categories that describe user's interest more accurately [13]. In the example above, computers would be expanded with the children of its node in the ODP tree: *AI, Algorithms, Games, Hacking, Internet, etc.* This way queries are sparser and dissociation would be effective.

Nevertheless, note the trade-off between privacy and personalization in this process. Categories range from broad topics (upper levels of the tree), to very narrow (lower levels), to the edge case of considering each individual query as a category. The former provides better personalization because it yields more data to the server. On the other hand, the latter obstructs personalization but provides more privacy.

Besides, we have to consider long-term and short-term interests. A user specialization may be temporal and change with time. DisPA copes with that by self-adapting over time and rearranging the set of categories for classification automatically. For instance, in the example above, if the user suddenly becomes more interested in *"Music"*, the system should roll-back to the previous state by retracting the old category *"Computers"* and, afterwards, expand *"Music"*.

Our approach to achieve this, we normalize the vector defined in Expression 1 and consider it as the distribution of probabilities of n random variables X_i representing the event of the query q belonging to the category c_i, for $i = 1, \ldots, n$. Then we measure the dispersion of this distribution by the coefficient of variation defined by $c_v := \frac{\sigma}{\mu}$.

We set a threshold that indicates when the number of queries per category is unbalanced and, thus, we have to expand or retract a category of the tree.

4.5 Self-adaptive classification

Recall from Section 4.1 that $C = \{c_1, \ldots, c_n\}$ is the set of categories used for classification. Rearrangement of C take place when a deviation from the man is detected.

The expansion operation occurs when the deviation is positive. In that case we add all the children of the category to be expanded into C. Note that we do not remove the parent from C because otherwise we would lose a possible outcome of the classification. For instance, imagine a future query that hits documents contained only found in the parent. In case that the deviation is negative, the category has too few queries and it must be dropped. Thus, if this happens with

all the categories that share the same parent category, we aggregate the children into the parent.

Once we have some categories expanded, in order to classify a query, we perform a level-wise classification. We begin at the first level of the tree and find the category of maximum weight. Then, if it has been expanded, we find the child with maximum weight and so on with the lower levels, until the category that minimizes the dispersion at the current level has not been expanded.

During the expansion, we generate a new virtual identity for each child and the virtual identity of the parent stays the same. This way, the log of the parent in the server may contain queries of other subcategories but it does not affect personalization. When retracting a category we simply use the virtual identity of the parent that we preserved in the expansion operation and, if we expand it again, we reuse the old virtual identities for the children.

As a result, we are able to adjust the level of sparseness of the logs in the server and, thereby, adjust the trade-off between privacy and personalization. We refer the reader to the algorithms implemented for this process [13].

5 Empirical results

In order to test the agent and prove that the risk of re-identification is reduced we used the linkage algorithm described in [12]. This algorithm is supposed to be applied by the adversary on the server logs and link those belonging to the same user together.

5.1 Evaluation

The lack of public sets of queries makes difficult the evaluation of the degree of personalization achieved by the agent as well as the effect that the agent has on it. As a preliminary experiment we submitted a set of 803 queries through DisPA from an AOL user and we did not notice any difference with plain search. Then, we submitted a set of 2743 queries of another AOL user several times and there was a difference on the order of two results (from first to second place in the list). Nevertheless, a complete analysis of personalization is out of the scope of this work, we center our evaluation in the disclosure risk.

We developed an attacking algorithm against our own agent. Such an algorithm could be used by the adversary to rebuild the original user's server log out of the partial logs generated by DisPA. The algorithm is based on the observation that there are terms that are more common in the user's queries than in other users'. These terms do not have strong semantic meaning and are classified according to the rest of the terms in the query. Consequently, these terms are spread by the dissociation over all partial logs. The algorithm tries to exploit this by linking the logs that contain these terms.

As mentioned in Section 4.3, an instance of these terms are named entities. This takes inspiration from the AOL case. The NYT journalists identified Thelma Arnold because she was looking for venues in her town ("lilburn") and

names of relatives ("arnold"). Then, they used a telephone directory to narrow the search up to 14 individuals [9].

To detect the terms, the adversary has to represent logs as vectors using a tf-idf scheme. The rationale is that tf-idf reflects the importance of a term in a log offset over its frequency in the collection of all logs. Then, the algorithm clusters the vectorial space using the DBSCAN algorithm with the cosine similarity as distance.

The linkage algorithm initializes with one or more logs known to belong to the user (auxiliary knowledge), that we call "seeds". At the end, clusters containing a seed are joined into one unique cluster that represents the original log.

To evaluate the user's *disclosure risk*, that is the risk that the original log is recovered by the adversary after dissociation, we measure the quality of the clustering provided by the linkage algorithm. To evaluate the clustering we measure the F1-Score of the binary classification defined by the property of a query being part of the final cluster or not.

The F1-Score combines precision and recall. Note that in our case, true positives are queries of the targeted user that fall in the cluster, false positives are queries of other users that fall in, true negatives are queries of other users that fall out and false negatives are queries of the target user that fall out. A higher F1-Score corresponds to a higher success rate of the adversary.

The DBSCAN clustering requires a parameter as an input that defines the neighbourhood of a data point. This parameter is not known a priori by the attacker. We consider the worse-case for the user and find the value that gives the best clustering through experimentation.

5.2 Experiments

We used the AOL released dataset for the experiments. The dataset contains an user ID, the terms of the query and the URLs of the results that were clicked. We performed five experiments described next.

- Experiment 1: We took a sample of logs of 20 different users and submitted their queries through the agent (with query filters disabled). Then, we applied the clustering algorithm to the resulting dissociated logs taking a random seed.
- Experiment 2: We added Arnold's log to the sample of logs. We chose Arnold's log because her log contains named entities like "Arnold" or "Lilburn" in queries that fall in different categories. We repeated the first experiment under the same conditions to see if the clustering algorithm performed better. This time we took as seed the dissociated log corresponding to the class "Arts", one of the largest dissociated logs. The justification is that it is more likely that the attacker finds information related to the user in the largest log.
- Experiment 3: For the third experiment we repeated the second experiment but enabling the filter of uncommon queries and treating queries with named entities as described at Section 4.3.

– Experiment 4: This experiment is intended to evaluate the self-adaptive DisPA. For this experiment we did not use the AOL dataset because logs are very small to be specialized enough. Instead, we developed a generator of queries based on the keywords stored in the ODP that we referred in Section 3. The generator takes a probability distribution for the classification taxonomy as an argument and generates a log of queries accordingly. For this experiment we used the following distribution:

$$\begin{aligned} Adults &\ 0 \\ Arts &\ 0 \\ Games &\ 0.02 \\ Reference &\ 0.02 \\ Shopping &\ 0 \\ Business &\ 0.04 \\ Health &\ 0.02 \\ News &\ 0 \\ Society &\ 0.1 \\ Computers &\ 0.8 \\ Home &\ 0 \\ Science &\ 0 \\ Sports &\ 0 \end{aligned}$$

  We simulated the submission of these queries first using first implementation of DisPA and, then, the self-adaptive version setting the threshold of the coefficient of variation to 80%. We added 20 random users from the AOL released dataset and applied the linkage algorithm.
– Experiment 5: we repeated the fourth experiment but using the self-adaptive version of DisPA.

In order to claim whether the user is protected or not, we set 50% of disclosure risk as a threshold. If F1-Score is below this threshold we say that the user is protected, and not protected otherwise.

5.3 Results

For the first experiment we found that for small values of ε the algorithm reaches maximum precision because the final cluster only contains the seed. All queries in the seed log were queries that fell in the final cluster (true positives) and there were no queries of other users (false positives). In contrast, the recall is zero because all-but-one server logs of the user fell out of the final cluster (false negatives).

This translates to a low F1-Score and hence a low disclosure risk. As we increase ε more and more logs fell into the final cluster. Nevertheless, this server log was well dissociated by DisPA and the algorithm, for the given seed, jumped directly to the situation where the whole collection of server logs fell into the final cluster. This means that user's logs could not be linked using the algorithm with this seed because these logs did not have enough rare terms in common.

For the second experiment we used Thelma Arnold's log as the target log. We saw that for $\varepsilon = 1.39$ we had the optimal clustering. The algorithm had linked most of the dissociated logs and, thus, if offered a disclosure risk close to 90%.

For the third experiment we took the same parameters for the attacking algorithm but this time we used the filter for uncommon queries described in Section 4.3. The disclosure risk was almost constant for the same interval of values taken in the previous experiments. For $\varepsilon = 1.9$ disclosure risk increased. This means that logs could not be linked because uncommon terms had been successfully separated in different logs.

In the fourth experiment, we showed there were some values of the neighbourhood distance for which the user was not protected because the F1-Score was above 50%. We saw that it made no sense to go on evaluating greater values than 2 because the precision was maximum. This means that all targeted logs were falling in the final cluster and the clustering was not going to improve. In fact, we actually saw all logs in the server fell in the final cluster since recall was very low.

Finally, in the fifth experiment we did exactly the same, although the seeds changed because we were considering a different collection of dissociated logs. In fact, two categories were expanded during the simulation: "$Top/Computers$" and "$Top/Computers/Internet$".

For this last experiment the disclosure risk was below 50% for all ε and, therefore, the user was protected. The percentage of disclosure risk reduction from the standard DisPA in the worst case was around 67%.

6 Discussion and Limitations

One of the main limitations of this work is the assumption of search engines exceptionally using cookies to track users. This is a strong assumption to hold today, specially after the revelation of the increasing prevalence of device fingerprinting. However, there is no evidence yet of any search engine using these techniques at the moment. As a very rough countermeasure the agent replaces the user-agent string by a more general one. We extracted this user-agent from a small-scale panopticlick-like survey that we conducted in our circles of acquaintances[3].

We must admit that the adversary considered is not a fully strategic adversary. In the experiments we are facing a particular algorithm of re-identification and it might be that a manual inspection could further reveal other information that the algorithm misses. In favour of the algorithmic approach we must say that given the amount of logs in real-world servers, any manual approach would be infeasible.

A limitation is that dissociated profiles might deviate significantly from the average profile of the population of users. This effect can be detected by an

[3] Results of this study can be found at http://www.iiia.csic.es/~mjuarez/results.html.

adversary who can extract statistics from the server. However, we cannot prove this given that we do not have enough information about the distribution of profiles in real-world servers. As we already noted, there exist real profiles that are very specialized and are not the result of dissociation.

At the same time, we assume that the auxiliary knowledge available to the adversary is limited. It is well known that to achieve perfect privacy against an adversary with unlimited background knowledge is a hard problem [40]. We must clarify that our model does not protect against such an adversary and assumes that the auxiliary information is bounded.

We also assume that the search engine and the agent use the same taxonomy for personalization. This assumption does not hold in most of the cases because search engines' taxonomies are oriented to advertising[4]. It is likely that if we dissociate independently of the search engine's taxonomy, the utility provided by the system will drop. However, since we do not evaluate the utility preserved by the agent, we cannot confirm such effect and leave this evaluation for future work.

Another limitation is that the agent might be considered to break the terms and conditions of the search service. We have implemented DisPA using Google's search engine and we have not found any conflict with their privacy policy. However, since the agent is scraping the URLs of the result list to avoid redirection through Google's servers, it might be argued that the agent is altering the service provided by Google. Furthermore, we do not display Google's advertisements in DisPA results pages.

We note that there is a trade-off between the usability and the privacy offered by the PET for private search. For instance, a search through an anonymous communication system such as Tor would provide stronger privacy guarantees than DisPA. Nevertheless, in terms of overhead, DisPA takes 2.5 seconds to return results in the worse case, when there is no context created (4 times more than a direct search) and 1.5 seconds for the average case, in case the context already exists (2 times overhead). The agent also caches result pages to speed up queries that are submitted multiple times over time and, also, prevent the adversary to extract information from the frequency of these queries. We think it is reasonable for a user to sacrifice a second of his search time for a better privacy.

In order to enforce a reproducible research policy, we have uploaded our code to a public repository[5].

7 Conclusions and Future work

The main contribution of this research is a framework for the development and evaluation of an agent that provides less disclosure risk in search engines with an admissible time of response. However, DisPA has some drawbacks that

[4] https://www.google.com/settings/ads

[5] The source code can be found at https://code.google.com/p/dispa-framework/

future research may deal with. Future work on this line could focus on improving personalized classification. For instance, clicking information could also be incorporated to our model of PQC.

We could also consider vertical searching for personalization. Vertical searching refers to the process of refining consecutive queries to improve search results. The aforementioned PQC should then take into account short-term preferences within a session. One of the issues that arises is how to implement a model for sessions of related queries.

In the line of decreasing disclosure risk, one could consider the generalization of Named Entities using an ontology like WordNet. For instance, if someone searches for "lilburn dentists" the agent could generalize "Lilburn" to "Atlanta" or "Georgia". Information loss would be greater but then it would be possible to measure it by the differences between search results pages.

Along with that, evaluating personalization is still an open problem. There are few studies that aim to measure to what extent search engines personalize search results. However, personalization is a moving target and the authors of these studies often admit that their results are not concluding for not running the experiments for sufficiently long periods of time [41].

Another improvement could be to generalize tracking data used to create virtual identities, from cookies to a more general type of data. Device fingerprinting is still an open problem but there are some promising approaches that could be adopted by DisPA in the future [42].

Besides, the attacking algorithm described in section 5 could be tested with different clustering approaches like sequential clustering described in [43]. The similarity measure could be improved by boosting the tf-idf scheme using a dictionary of terms that differentiates the user from the others.

In addition, other measures of disclosure risk may be defined comparing clusters between clustering of DisPA and non-filtering DisPA. For instance, if dissociated logs in the former fall into several clusters in the latter, disclosure risk is lower than if all fall in the same cluster. The Jaccard index could be used to measure differences between clusters from different classifications.

Finally, we could use more sophisticated privacy measures in our security analysis of the system. For example, we could explore the use of entropy-based measures for this purpose [44, 45].

8 Acknowledgements

The authors wish to acknowledge the anonymous reviewers for their detailed and helpful comments to the manuscript.

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n 262608.

Partial support by the Spanish MEC projects ARES (CONSOLIDER INGENIO 2010 CSD2007-00004) and COPRIVACY (TIN2011-27076-C03-03) is acknowledged.

References

1. S. Hansell, "Increasingly, Internet's Data Trail Leads to Court," Feb. 2006. New York Times.
2. J. Teevan, S. T. Dumais, and E. Horvitz, "Personalizing search via automated analysis of interests and activities," in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 449–456, ACM, 2005.
3. A. Micarelli, F. Gasparetti, F. Sciarrone, and S. Gauch, "Personalized search on the world wide web," in *The adaptive web*, pp. 195–230, Springer, 2007.
4. P. Norvig, "Search Algorithms with Google Director of Research Peter Norvig," Oct. 2011. Stone Temple Consulting.
5. M. Speretta and S. Gauch, "Personalized search based on user search histories," *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, pp. 622–628, 2005.
6. R. Jones, R. Kumar, B. Pang, and A. Tomkins, "I know what you did last summer: query logs and user privacy," in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pp. 909–914, ACM, 2007.
7. EFF, "AOL's Massive Data Leak," 2009.
8. G. Sadetsky, "AOL Data," Aug. 2006.
9. M. Barbaro and T. Zeller, "A Face Is Exposed for AOL Searcher No. 4417749," 2006.
10. B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *Journal of the ACM (JACM)*, vol. 45, no. 6, pp. 965–981, 1998.
11. S. T. Peddinti and N. Saxena, "On the privacy of web search based on query obfuscation: a case study of trackmenot," in *Privacy Enhancing Technologies*, pp. 19–37, Springer, 2010.
12. M. Juárez and V. Torra, "Toward a Privacy Agent for Information Retrieval," *International Journal of Intelligent Systems*, vol. 28, pp. 606–622, June 2013.
13. M. Juárez and V. Torra, "A Self-Adaptive Classification for the Dissociating Privacy Agent," in *PST2013, the eleventh annual conference on Privacy, Security and Trust*, (Tarragona), pp. 44–50, 2013.
14. E. Kushilevitz and R. Ostrovsky, "Replication is not needed: Single database, computationally-private information retrieval," *in proceedings of the 38th Annual Symposium on Foundations of Computer Science*, 1997.
15. S. Yu, T. Thapngam, S. Wei, and W. Zhou, "Efficient web browsing with perfect anonymity using page prefetching," in *Algorithms and Architectures for Parallel Processing* (C.-H. Hsu, L. Yang, J. Park, and S.-S. Yeo, eds.), vol. 6081 of *Lecture Notes in Computer Science*, pp. 1–12, Springer Berlin Heidelberg, 2010.
16. W. Ogata and K. Kurosawa, "Oblivious keyword search," *Journal of complexity*, vol. 20, no. 2, pp. 356–371, 2004.
17. R. Ostrovsky and W. E. Skeith III, "Private searching on streaming data," in *Advances in Cryptology–CRYPTO 2005*, pp. 223–240, Springer, 2005.
18. J. Domingo-Ferrer, A. Solanas, and J. Castellà-Roca, "h(k)-Private Information Retrieval from Privacy-Uncooperative Queryable Databases," 2008.
19. J. Domingo-Ferrer, M. Bras-Amorós, Q. Wu, and J. Manjón, "User-private information retrieval based on a peer-to-peer community," *Data and Knowledge Engineering*, 2009.
20. B. Shapira, Y. Elovici, A. Meshiach, and T. Kuflik, "PRAW: A PRivAcy model for the Web," *Journal of the American Society for Information Science and Technology*, vol. 56, pp. 159–172, Jan. 2005.

21. M. Murugesan and C. Clifton, "Plausibly deniable search," *Workshop on Secure Knowledge Management*, vol. 1, pp. 3–8, 2008.
22. S. Ye, F. Wu, R. Pandey, and H. Chen, "Noise Injection for Search Privacy Protection," *2009 International Conference on Computational Science and Engineering*, pp. 1–8, 2009.
23. D. Howe and H. Nissenbaum, "TrackMeNot: Resisting surveillance in web search," *Lessons from the Identity Trail: Anonymity*, 2009.
24. D. Rebollo-Monedero and J. Forne, "Optimized Query Forgery for Private Information Retrieval," *IEEE Transactions on Information Theory*, vol. 56, pp. 4631–4642, Sept. 2010.
25. H. Pang, X. Xiao, and J. Shen, "Obfuscating the topical intention in enterprise text search," in *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*, pp. 1168–1179, IEEE, 2012.
26. J. E. Jiménez, A. R. Hoyos, J. Parra-Arnau, J. Forné, and D. Rebollo-Monedero, "Medición de la Privacidad de Perfiles de Usuario mediante un Add-on de Navegador," pp. 93–100, October 2013.
27. E. Balsa, C. Troncoso, and C. Diaz, "OB-PWS: Obfuscation-Based Private Web Search," *Security and Privacy (SP), 2012*, 2012.
28. R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," tech. rep., DTIC Document, 2004.
29. A. Viejo and D. Sanchez, "Providing useful and private web search by means of social network profiling," in *Privacy, Security and Trust (PST), 2013 Eleventh Annual International Conference on*, pp. 358–361, July 2013.
30. D. Sánchez, J. Castellà-Roca, and A. Viejo, "Knowledge-based scheme to create privacy-preserving but semantically-related queries for web search engines," *Information Sciences*, vol. 218, pp. 17–30, 2013.
31. A. Arampatzis, P. S. Efraimidis, and G. Drosatos, "A query scrambler for search privacy on the internet," *Information retrieval*, vol. 16, no. 6, pp. 657–679, 2013.
32. A. Erola and J. Castellà-Roca, "Using search results to microaggregate query logs semantically," in *DPM/SETOP*, pp. 148–161, 2013.
33. M. Batet, A. Erola, D. Sánchez, and J. Castellà-Roca, "Utility preserving query log anonymization via semantic microaggregation," *Inf. Sci.*, vol. 242, pp. 49–63, 2013.
34. A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, pp. 111–125, IEEE, 2008.
35. Google, "Key Terms - Policies and Principles," Apr. 2012.
36. V. Toubiana and H. Nissenbaum, "Analysis of Google Logs Retention Policies," *Journal of Privacy and Confidentiality*, vol. 3, no. 1, 2011.
37. G. Acar, M. Juárez, N. Nikiforakis, C. Diaz, S. F. Gürses, F. Piessens, and B. Preneel, "FPDetective: Dusting the web for fingerprinters," in *Proceedings of the 20th ACM Conference on Computer and Communications Security (CCS'13)*, (Berlin), pp. 1129–1140, ACM, 2013.
38. P. Ullegaddi and V. Varma, "A Simple Unsupervised Query Categorizer for Web Search Engines," 2011.
39. B. Cao, J. Sun, E. Xiang, and D. Hu, "PQC: personalized query classification," *Proceedings of the 18th ACM conference on information and knowledge management*, pp. 1217–1225, 2009.
40. P. Ohm, "Broken promises of privacy: Responding to the surprising failure of anonymization," *UCLA Law Review*, 2010.

41. A. Hannak, P. Sapiezynski, A. Molavi Kakhki, B. Krishnamurthy, D. Lazer, A. Mislove, and C. Wilson, "Measuring personalization of web search," in *Proceedings of the 22nd international conference on World Wide Web*, pp. 527–538, International World Wide Web Conferences Steering Committee, 2013.
42. N. Nikiforakis, W. Joosen, and B. Livshits, "Privaricator: Deceiving fingerprinters with little white lies," tech. rep.
43. S. Miyamoto and K. Arai, "Different sequential clustering algorithms and sequential regression models," in *2009 IEEE International Conference on Fuzzy Systems*, pp. 1107–1112, IEEE, IEEE, Aug. 2009.
44. A. Serjantov and G. Danezis, "Towards an information theoretic metric for anonymity," in *Privacy Enhancing Technologies*, pp. 41–53, Springer, 2003.
45. C. Diaz, S. Seys, J. Claessens, and B. Preneel, "Towards measuring anonymity," in *Privacy Enhancing Technologies*, pp. 54–68, Springer, 2003.