Towards Efficient and Accurate Privacy Preserving Web Search

Albin Petit
Université de Lyon, CNRS
INSA-Lyon, LIRIS, UMR5205,
F-69621, France
albin.petit@insa-lyon.fr

Sonia Ben Mokhtar
Université de Lyon, CNRS
INSA-Lyon, LIRIS, UMR5205,
F-69621, France
sonia.benmokhtar@insa-lyon.fr

Lionel Brunie
Université de Lyon, CNRS
INSA-Lyon, LIRIS, UMR5205,
F-69621, France
lionel.brunie@insa-lyon.fr

Harald Kosch
Universität Passau
Innstrasse 43, 94032 Passau,
Germany
harald.kosch@uni-passau.de

ABSTRACT

Querying Web search engines is by far the most frequent activity performed by online users and consequently the one in which they are likely to reveal a significant amount of personal information. Protecting the privacy of Web requesters is thus becoming increasingly important. This is often done by using systems that guarantee *unlinkability* between the requester and her query. The most effective solution to reach this objective is the use of anonymous communication protocols (e.g., onion routing [10]). However, according to [14], anonymity might not resist to machine learning attacks. Thus, an adversary could link a query to her requester's public profile. Other approaches (e.g., [8,17]) guarantee *unidentifiability* of the user interests by generating noise (e.g., creating covert queries or adding extra keywords). However, these solutions overload the network and decrease the accuracy of the results. We present in this paper the first contribution that combines both approaches. It allows a user to perform a private Web search resistant to machine learning attacks while slightly decreasing the relevance of the results. Our three stage architecture contains: (1) a *Privacy Proxy* that relies on two non-colluding servers to hide the requester identity from the search engine ; (2) a *Linkability Assessment* that analyses the risk that a request is reassociated with the identity of the requester; (3) an *Obfuscator* that protects the queries which have been flagged linkable by the linkability assessment.

Categories and Subject Descriptors

K.4.1 [**Computers and society**]: Public Policy Issues—*privacy*

MW4NG '14 December 8-12, 2014, Bordeaux, France

General Terms

Security, Algorithms

Keywords

Privacy, Web search, Unlinkability, Unidentifiability

1. INTRODUCTION

Recent surveillance programs (e.g., PRISM) and online services (e.g., Google, Facebook, Twitter, etc) demonstrate the massive use of personal data without any control or agreement from end-users. In this context, protecting user privacy is becoming increasingly important. Indeed, a recent study shows that 73% of search users do not agree with search engines keeping track of their searches[1].

Literature contains many solutions to protect the privacy of requesters accessing Web search engines. Among these solutions, we distinguish two different directions : hiding the identity of the requester (*unlinkability*) or hiding the content of the query (*unidentifiability*). On the one hand, the most robust approaches to reach unlinkability are anonymous communication protocols (e.g., TOR [7], Dissent [6, 18], RAC [3]). However, in addition to their heavy computational overhead, it was demonstrated [14] that machine learning algorithms break this protection. Consequently, adversaries are able to link a query to its requester's identity. On the other hand, the most common solutions to unidentifiability create covert queries (to drown user queries among fake queries) or add extra keywords to the initial query (to drown user's keywords in these extra keywords). However, these approaches incur a communication overhead and a degradation in the accuracy of the results.

In this paper, we propose a new architecture that guarantees a stronger protection to the users while keeping a high level of accuracy and limiting the communication overhead. First, to overcome the machine learning attacks [14], we combine unidentifiability techniques with a new unlinkability protocol. Further, to keep a high accuracy of the results and limit the communication overhead, we adopt an adap-

[1]Pew Internet & American Life survey - February 2012: http://www.pewinternet.org/2012/03/09/search-engine-use-2012/

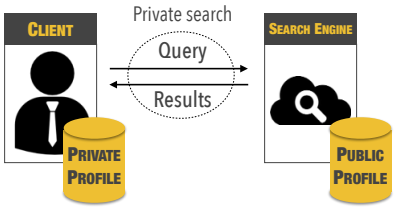

Figure 1: Context overview

	Add fake requests [12, 17]	Add noise to requests [8]	Generalize request [1]
Overhead	++	+	++
Privacy	plausible deniability	k-anonymity	plausible deniability
Accuracy	+++	++	+

Table 1: Comparison of unidentifiability approaches

tive obfuscation mechanism. Specifically, only queries with a high probability to be reassociated to their requester's public profile are obfuscated. This obfuscation step concatenates the initial query with generated fake queries using the logical OR operator. The number of fake queries depends on the risk of reassociating the initial query to its requester's profile. Finally, a post-filtering removes irrelevant answers to retrieve results corresponding to the non-obfuscated query. To reach this objective, our solution relies on three parts: (1) a *Privacy Proxy* that implements a new unlinkability protocol ; (2) a *Linkability Assessment* that assesses the reassociation risk of requests ; (3) an *Obfuscator* that performs the obfuscation in an efficient way.

The remaining of this paper is organized as follows. We define the problem that we want to solve in Section 2. We then present an overview of existing state of the art in Section 3. Further we present our architecture in Section 4 and we discuss aspects related to our assumptions in Section 5. Finally, we present our evaluation plan in Section 6 before concluding the paper in Section 7.

2. PROBLEM STATEMENT

As described in Figure 1, we consider two different entities: a client and a Web search engine. Our objective is to allow a client to query the search engine in a privacy preserving way. This means that the search engine, which receives the query, or any adversary which listens to the network, cannot infer the identity of the requester. Moreover, we assume that the user has been querying directly (i.e., without any privacy preserving protection) the search engine in the past. Consequently, the search engine used all client's previous queries to create on its side a user profile, namely public profile (right side of Figure 1). The client has also saved locally her search history in a private profile (left side of Figure 1).

Finally, we assume that requests sent by the client do not contain quasi-identifiers (e.g., egosurfing) enabling the straightforward re-identification of the user.

3. STATE OF THE ART

There have been a large body of works to protect the privacy of users accessing search engines. These solutions can be classified in three major categories according to the guarantees they offer to users: (1) systems guaranteeing *unlinkability* between the requester and its request ; (2) systems guaranteeing *unidentifiability* of the user's requests and (3) systems following the Private Information Retrieval (PIR) scheme.

3.1 Unlinkability

Unlinkability refers to the ability to anonymously query service providers. The naive solutions in this category of systems rely on the use of a trusted proxy, which handles the sending of a request on behalf of the user [15]. However, this solution only translates the problem to the proxy-side as the latter is able to collect sensitive information about the users. The most robust solutions in this first category of systems are *anonymous communication* protocols (e.g., onion routing [10], TOR [7], Dissent [6, 18], RAC [3]). Nevertheless, these protocols require a large number of cryptographic operations and some of them (e.g., RAC, Dissent), need the use of all to all communication primitives, which engender a large amount of traffic in the network. This category of systems also includes solutions in which users send requests on behalf of each other to protect their identities (e.g., [5], [11]). As such, the service provider is not able to link the request with the identity of the effective requester. However, these approaches are very costly as they rely on group communication between users in order to organize the mixing of their requests.

Moreover, in addition to their costs, it has been proved that unlinkability techniques are not enough to protect the user. Naive machine learning techniques [14] show that it is feasible to identify the requester of an anonymous query. Indeed, the Support Vector Machine (SVM) algorithm, trained on previous non anonymous queries, is able to reassociate a query to its requester's public profile. For instance, if all public profiles contain different topics of interest, this algorithm can easily link an anonymous query to the right profile. Consequently, *unlinkability* approaches do not effectively guarantee the protection of the user.

3.2 Unidentifiability

The second category of approaches aims instead at obfuscating a user profile. As a result, the service provider can hardly distinguish between a user's effective interests (or requests according to the implementation) and fake ones. The main approaches can be classified into three categories as shown in the Table 1.

Unidentifiability is usually achieved by generating additional periodic non relevant requests on behalf of the user (column 1 in the table). For instance, in [17], the system randomly sends fake requests based on RSS feeds (e.g. CNN, New York Times). But the RSS feeds, hardcoded or set up manually by the user, might be irrelevant to cover user's queries and consequently an adversary could distinguish between the real queries and the generated one. A similar approach (e.g., [12]) generates `k` plausibly deniable queries on different topics with the same probability of relevance for the user. These queries are generated using a set of documents as seed. However, these two first solutions (e.g., [17] [12]) overload the network with useless traffic by creating a lot of covert queries.

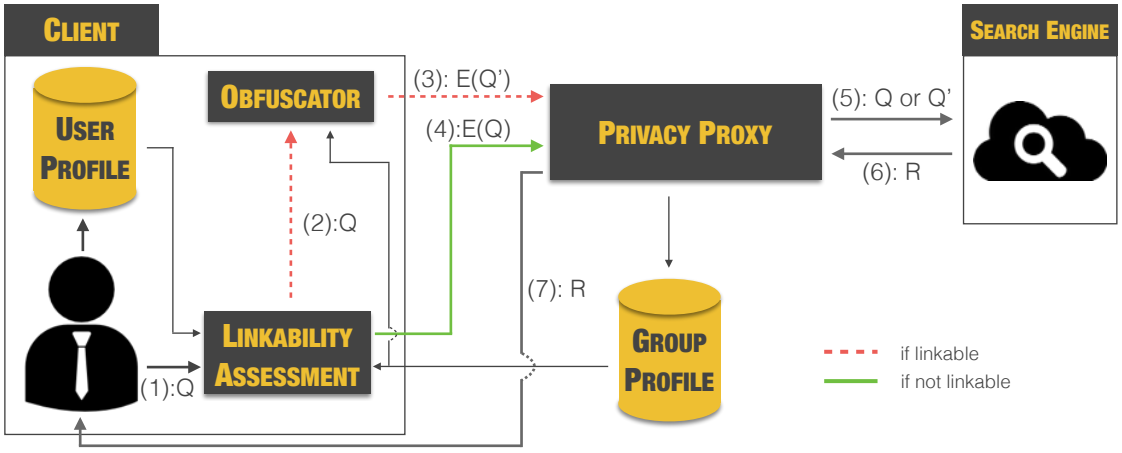

Figure 2: The three stage architecture overview

Another approach is to add noise to existing users' requests (column 2 in the table). For instance, in [8], `k-1` random extra keywords are added to the user's original request. This approach guarantees *k-anonymity* [16] which means that search engines can guess the correct keyword with a probability `1/k`.

A similar solution (e.g., [1]) is to send a set of requests representing more general concepts that the original request, and then to merge and to filter out the result in order to retrieve part of the original results. This method is based on the hypothesis that a subset of the original results is included in the result of the generated queries. Nevertheless, these two former solutions result in a decreased accuracy as the users may not retrieve the original result and receive irrelevant replies due to the added noise. Furthermore, for all these approaches, search engines that use personalization to improve their results will reply more inappropriate results due to the noise deliberately introduced in the user profile.

3.3 Private Information Retrieval

Finally, approaches of Private Information Retrieval (PIR) prevent the Web search engine to know the interest of the user. Indeed, a solution (e.g., [13]) that uses homographic encryption with a mechanism that embellish each user query does not reveal the original query to the search engine. However, this very costly approach (due to the homographic encryption) supposes a modification of the recommendation algorithm on the provider side.

Considering the limitations of state of the art approaches, we aim in this paper to propose a solution that guarantees a higher protection between the requester and her request while providing a lower computational and communication overhead than state of the art approaches.

4. TOWARDS PRIVACY PRESERVING WEB SEARCH

Our solution combines *unidentifiability* techniques with a new *unlinkability* approach to guarantee that the query is not linkable to its requester's public profile. We divide our approach in three parts : the *privacy proxy*, the *linkability assessment* and the *obfuscator*. We first introduce our contribution with a general overview and then present, in the three following subsections, each stage in detail.

4.1 Architecture overview

As shown in Figure 2, when the requester issues a query Q (message (1) in the figure), the linkability assessment analyses Q to determine if the latter can be linked to the user profile. To do so, the linkability assessment relies on the user profile (private profile) and on the group profile (aggregation of public profiles). If the query is flagged linkable (message (2) in the figure), the second stage (i.e., the obfuscator) obfuscates the query to mislead the adversary. In any case, the query is issued through the privacy proxy (message (3) and (4) in the figure). This third stage is responsible for splitting the query into two distinct pieces of information: the requester identity and the query content. As a consequence, the search engine knows the content of the query (message (5) in the figure) but has no clue in the requester's identity. Finally, the search engine replies to the privacy proxy (message (6) in the figure) which retrieves the identity of the requester and forwards the answer to her (message (7) in the figure).

4.2 Privacy Proxy

The goal of the privacy proxy is to guarantee that the query is not linkable with its requester's public profile with a low computational cost.

4.2.1 Protocol overview

To protect the privacy of users, the aim of the privacy proxy is to prevent all distant servers involved in the process of satisfying a user's request to have access to both the query and the identity of the requester (e.g., her IP address). The key idea behind the privacy proxy is thus to split these two pieces of information on two different servers. Specifically, our first server, namely the *receiver*, has access to the identity of the requester without being able to read the request, while our second server, namely the *issuer*, has access to the request without knowing the identity of the user. A key challenge is then to allow these two servers to query the search engine and to return an answer to the requester in a privacy-preserving way. Figure 3 shows how our protocol reaches this objective. In this figure, we use the notations

$E(m)$	RSA encryption of message m with the public key of the issuer
$\{m\}_i$	AES encryption of message m with key K_i
Q_i	i-th query of user U
K_i	AES encryption key associated with query Q_i
A_i	Answer to query Q_i
X	An anonymous identifier

Table 2: Notations

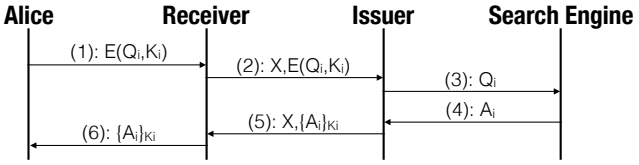

Figure 3: Privacy Proxy protocol

of Table 2. We assume that the client has the public key of the issuer and generates a symmetric encryption key K_i for each request she wants to send to the service provider. We assume that the two non-colluding servers are honest-but-curious: they execute their tasks correctly, but they can collect and reason about knowledge traversing them.

To send a request Q_i to a service provider, the user starts by encrypting Q_i using the issuer's public key and sends it to the receiver along with K_i (message (1) in the figure). As such, the receiver is unable to read the content of the query. Upon receiving a user's request, the receiver stores the identity of the requester and forwards the query to the *issuer* associated with an anonymous unique identifier, i.e., X in the figure (message (2) in the figure). This identifier could simply be the hash of the timestamp at which the receiver received the request. For stronger guarantees we can also use UUIDs[2]. Upon receiving the anonymized query, the issuer decrypts the request and submits it to the search engine (message (3) in the figure), which replies with the corresponding answer (message (4) in the figure). Then, the issuer encrypts the answer with the user's symmetric key, i.e., K_i and forwards the answer to the receiver along with the anonymous identifier, i.e., X (message (5) in the figure). Finally, the receiver retrieves the IP address of the requester and forwards the encrypted reply to her (message (6) in the figure).

For security reasons, the user has to create a new symmetric key K_i each time she wants to send a query. If not, the issuer can link all the queries belonging to a same user together by using this key as a unique identifier and possibly reidentify the user using statistical attacks as it has been done on the AOL dataset [2].

4.2.2 *Privacy-preserving group profile computation*

The machine learning attack [14] on unlinkability solutions is mainly due to not taking into account the popularity of topics searched by all users. Our solution addresses this issue by using a profile that contains information about all users. The privacy proxy will publish periodically an updated version of what we call the group profile. A key requirement of the group profile computation is that it has

[2]UUID standard specification: http://tools.ietf.org/html/rfc4122

Group Profile	#usage	#users
HIV	5	2
risk	4	2
factor	3	1
tennis	9	4
cinema	5	2

User Profile	#usage
HIV	4
risk	3
factor	3
tennis	2
#average	3

Figure 4: Example of a User Profile and a Group Profile

to be done in a privacy preserving way. Consequently, to avoid the leak of personal information, this profile is an aggregation of all user requests. It includes, for a given word, how many users used it and how many times it was used by all users. This profile is used by the *linkability assessment* and the *obfuscator* modules. Figure 4 gives an example of a group profile. In this figure, we can see that the keyword "HIV" has been used 5 times by 2 different users.

4.3 Linkability assessment

The second stage of our mechanism is to evaluate if the query is linkable to the requester's public profile. In fact, the linkability assessment ensures that a non linkable query is resistant to statistical attacks (machine learning algorithms). To reach this objective we defined a linkability metric which quantifies if the query is likely to be linked to her requester public profile. This evaluation is based on two types of information: the popularity of a query (using the group profile) and the proximity of a query to the user profile. Consequently, we define two submetrics:

- **User profile similarity** — The goal of this metric is to evaluate the degree of similarity of a query to the user profile. Specifically, for each keyword belonging to the query, we compare its usage frequency a_i to the average usage frequency of all keywords b_i. More formally, we define the user profile similarity metric M_u as:

$$M_u = \frac{\sum_i \frac{a_i - b_i}{a_i + b_i}}{\#keywords}$$

  The value of this metric varies from -1 (no part of query was already issued) to 1 (the query was already issued multiple times). If we compute this metric for the keyword "HIV" using the user profile of Figure 4 (the usage frequency is 4 and the average usage frequency of all keywords is 3), we deduce that the user profile similarity metric of "HIV" is 0.14.

- **Group profile similarity** — The goal of this metric is to evaluate if a query is popular among the other users in the group. Concretely, this comes down to know, for each keyword belonging to the query, how many times c_i, it was used and how many users d_i used it. More formally, the group profile similarity M_g can be defined as follow:

$$M_g = \frac{\sum_i \frac{a_i - c_i/d_i}{a_i + c_i/d_i}}{\#keywords}$$

  For instance, if we take the keyword "HIV" and the profiles of Figure 4, we obtain a group profile similarity of 0.23.

As shown by Algorithm 1, the linkability is defined as an average between these two above metrics. A low value means that the query is not linkable to a specific public profile whereas a high value means that only one public profile corresponds to the query and consequently the search engine is able to link the anonymous query to its requester.

We consider that all queries with a linkability value over the threshold 0 (i.e., a not popular query close to the user profile is sensitive) need to be protected and sent to the obfuscator. For instance, if we take the keyword "HIV" and the profiles of Figure 4, we have a linkability of 0.18. This value is over 0 and thus the query "HIV" is potentially linkable.

Algorithm 1 Linkability Assessment

Require: The query Q
1: linkability $\leftarrow 0$
2: **for all** keyword $\in$ Q **do**
3: $M_u \leftarrow$ computeUserProfileMetric(keyword)
4: $M_g \leftarrow$ computeGroupProfileMetric(keyword)
5: linkability $\leftarrow$ linkability $+\frac{M_u+M_g}{2}$
6: **end for**
7: linkability $\leftarrow \frac{linkability}{|Q|}$
8: **if** linkability > 0 **then**
9: SendToObfuscator(Q)
10: **else**
11: SendToPrivacyProxy(Q)
12: **end if**

4.4 Obfuscator

In order to protect queries that are tagged as linkable, we need to modify the original query. A major drawback in adding noise to the query is the introduction of irrelevant answers. For example, a naive way of doing that is to replace all words with their synonyms (using Wordnet [9] for instance). Unfortunately, even though this strategy protects quite efficiently the user, the results are not very accurate. That is why, we choose a basic functionality of search engines that allows the use of logical propositions in the query. Our goal is to complete the current query with k other fake queries using logical *OR* propositions. A post-filtering finally decreases the number of irrelevant answers introduced by the fake queries. We describe the implementation of the obfuscator (i.e., the Algorithm 2) in the three following sections.

Algorithm 2 Obfuscator

Require: The set q which contains the query Q
1: **while** evaluateLinkability(q) > 0 **do**
2: $q \leftarrow q \cup$ generateFakeQuery(q)
3: **end while**
4: $Q' \leftarrow$ createObfuscatedQuery(Q)
5: SendToPrivacyProxy(Q')
6: $R \leftarrow$ GetResults( )
7: **for all** result $\in R$ **do**
8: **for all** word $\in$ result **do**
9: **if** word $\in q \setminus \{Q\}$ **then**
10: remove(result)
11: **break**
12: **end if**
13: **end for**
14: **end for**

4.4.1 Fake query generation

The challenge here is to generate fake queries that cannot easily be identified as such by the search engine. Toward this purpose, we aim at generating fake queries that are far from the user profile and close to the profile of other users. The fake queries are generated using information contained in the user profile and in the group profile. It takes words contained in the group profile but not in the user profile. All the fake queries contains words used by other user. Consequently, the obfuscated query, forged by combining the original query and these fake queries (line 4 in Algorithm 2), is potentially closer to another user profile. This strategy misleads the adversary about the real identity of the requester.

4.4.2 Number of fake queries (k)

The number of fake queries is fundamental to guarantee the protection of the user. Indeed, a too small number will not decrease the success rate of the machine learning attack. We consider that an obfuscated query is correctly obfuscated when the average of linkabilities of the original query and the fake queries is above 0 (line 1 in Algorithm 2).

Theoretically, the construction of fake queries gives at least 2-anonymity guarantees. This number is in practice higher especially with a large number of users. In fact, the group profile does not contain enough information to determine if the fake queries are close to one or multiple other user profiles. A solution to have a better approximation of this number consists of adding information to the group profile. However this is not feasible because it discloses publicly too much information about the user.

4.4.3 Post-filtering

The goal of the post-filtering is to remove all irrelevant answers introduced by the fake queries. A basic algorithm (line 7 to 14 in Algorithm 2) will analyze all the results returned by the search engine and remove results that contain words generated during the obfuscation step. These irrelevant results are returned as an answer of the fake queries. Consequently, there are not interesting for the user.

5. DISCUSSION

On the non-collusion assumption of the receiver and the issuer: One of our main assumptions of the privacy proxy is the non-collusion of the receiver and the issuer. Onion routing deals with this issue by enabling the users to use a higher number of relays when they wish to have a higher degree of privacy. To have a higher level of privacy, we prefer not to increase the number of intermediate servers as we aim to provide a practical protocol. Instead, we choose to trust the receiver or the issuer. One could imagine that users and institutions can volunteer with servers acting as receivers or issuers as it is done in the TOR community. As such, the user could choose a receiver and an issuer she would trust (e.g., a server hosted by a non profit organization).

On the data leakage with the publication of the group profile: The group profile contains for each word its usage by all the users. By construction, it does not contain any personal information. Moreover, this aggregated data does not enable any adversary to recover the queries. Of course, we assume that enough users use our system. Furthermore, during the initialization step, the group profile can contain fake information.

6. EVALUATION PLAN

We are currently working on the evaluation of our approach. To do so, we plan to evaluate our solution in terms of:

- **Privacy guarantees** — We have already generated a proof with ProVerif [4] which confirms that the privacy proxy avoids a user and her query to be linked together (if we consider that the search engine does not have any previous information about the user). The full ProVerif description of the privacy proxy is accessible online[3]. We plan to validate the privacy guarantees of the whole architecture by replaying machine learning attacks [14] and by measuring protections offered by our architecture compared to TrackMeNot [17] protections.
- **Computational & communication overhead** — To measure the performance of our architecture, we plan to evaluate the performance of our privacy proxy compared to anonymous protocols like TOR [7].
- **Accuracy of the results** — By comparing the ranking of an non-obfuscated query and of an obfuscated query, we will measure the decrease of the accuracy.

7. CONCLUSION AND FUTURE WORKS

We presented in this paper a new architecture to query Web search engines in a privacy-preserving way. Existing solutions in literature such as anonymous communication protocols or unidentifiability approaches suffer from a huge computational and/or communication overhead and/or poor accuracy of results. Moreover they might not resist to machine learning attacks. To overcome these limitations, our solution relies on three stages to find the perfect trade off between user's privacy and accuracy of results. A key feature of our approach is to detect and protect queries that might be linked to a public profile and thus guaranteeing a real unlinkability between the requester and her request.

Our future work will focus on the evaluation of this solution in terms of security guarantees, computational cost and accuracy of the results.

8. ACKNOWLEDGMENTS

The presented work was supported the EEXCESS project funded by the EU Seventh Framework Program, grant agreement number 600601.

9. REFERENCES

[1] A. Arampatzis, P. S. Efraimidis, and G. Drosatos. A query scrambler for search privacy on the internet. *Information retrieval*, 16(6):657–679, 2013.

[2] M. Barbaro and T. Zeller. A Face Is Exposed for AOL Searcher No. 4417749. [Online] `http://www.nytimes.com/2006/08/09/technology/09aol.html?pagewanted=all&_r=0`, Aug. 2006.

[3] S. Ben Mokhtar, G. Berthou, A. Diarra, V. Quéma, and A. Shoker. Rac: A freerider-resilient, scalable, anonymous communication protocol. In *Proceedings of ICDCS*, 2013.

[4] B. Blanchet. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *Proceedings of the Computer Security Foundations Workshop (CSFW)*, 2001.

[5] J. Castellà-Roca, A. Viejo, and J. Herrera-Joancomartí. Preserving user's privacy in web search engines. *Computer Communications*, 32(13), 2009.

[6] H. Corrigan-Gibbs and B. Ford. Dissent: accountable anonymous group messaging. In *Proceedings of CCS*, 2010.

[7] R. Dingledine, N. Mathewson, and P. Syverson. TOR: The second generation onion router. In *Proceedings of the Usenix Security Symposium*, 2004.

[8] J. Domingo-Ferrer, A. Solanas, and J. Castellà-Roca. h(k)-private information retrieval from privacy-uncooperative queryable databases. *Online Information Review*, 33(4):720–744, 2009.

[9] C. Fellbaum. Wordnet: An electronic lexical database. 1998. *WordNet is available from http://www. cogsci. princeton. edu/wn*, 2010.

[10] D. Goldschlag, M. Reed, and P. Syverson. Onion routing. *Commun. ACM*, 42(2), 1999.

[11] Y. Lindell and E. Waisbard. Private web search with malicious adversaries. In *Privacy Enhancing Technologies*, pages 220–235. Springer, 2010.

[12] M. Murugesan and C. Clifton. *Providing Privacy through Plausibly Deniable Search.*, chapter 65, pages 768–779. 2009.

[13] H. Pang, X. Ding, and X. Xiao. Embellishing text search queries to protect user privacy. *Proceedings of the VLDB Endowment*, 3(1-2):598–607, 2010.

[14] S. T. Peddinti and N. Saxena. On the effectiveness of anonymizing networks for web search privacy. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, pages 483–489. ACM, 2011.

[15] M. Shapiro. Structure and Encapsulation in Distributed Systems: the Proxy Principle. In *Proceedings of ICDCS*, 1986.

[16] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[17] V. Toubiana, L. Subramanian, and H. Nissenbaum. Trackmenot: Enhancing the privacy of web search. *arXiv preprint arXiv:1109.4677*, 2011.

[18] D. I. Wolinsky, H. Corrigan-Gibbs, and B. Ford. Dissent in numbers: Making strong anonymity scale. In *Proceedings of OSDI*, 2012.

[3] Privacy Proxy ProVerif model: http://pastebin.com/FdsYanLz