Noise Injection for Search Privacy Protection

Shaozhi Ye, Felix Wu, Raju Pandey, and Hao Chen
sye@ucdavis.edu, {wu,pandey,hchen}@cs.ucdavis.edu
Department of Computer Science
University of California, Davis
Davis, CA 95616

***Abstract*—To protect user privacy in the search engine context, most current approaches, such as private information retrieval and privacy preserving data mining, require a server-side deployment, thus users have little control over their data and privacy. In this paper we propose a user-side solution within the context of keyword based search. We model the search privacy threat as an information inference problem and show how to inject noise into user queries to minimize privacy breaches. The search privacy breach is measured as the mutual information between real user queries and the diluted queries seen by search engines. We give the lower bound for the amount of noise queries required by a perfect privacy protection and provide the optimal protection given the number of noise queries. We verify our results with a special case where the number of noise queries is equal to the number of user queries. The simulation result shows that the noise given by our approach greatly reduces privacy breaches and outperforms random noise. As far as we know, this work presents the first theoretical analysis on user side noise injection for search privacy protection.**

I. Introduction

Privacy concerns have emerged globally as massive user information is collected by search engines. The large body of data mining algorithms, which are potentially employed by search engines while unknown to search users, further increase such concerns. Currently most search engines keep user queries for several months or years, for example, as of Sep. 8, 2008, Google claims to anonymize search log after 9–18 months [1]. Privacy violations may happen within the data retention window. Private information retrieval [2] and privacy preserving data mining [3] have been proposed as responses to the concerns over user profiling methods, while most existing approaches require a server-side deployment in the context of search privacy protection, which relies on the courtesy of search engines and privacy laws/regulations. Vulnerable data anonymization/sanitization designs and improper implementations also result in privacy breaches. For instance, after AOL released a query log of 650k users in 2006, several users were physically identified through their anonymized queries [4]. Moreover, there are chances for a malicious insider to get unprotected data and compromise user privacy.

In this paper, we propose a user-side privacy protection model for search users. Shown as Figure 1, our threat model assumes a malicious search engine (or an attacker who reveals the query log on the network or server side) and a user sending queries to this search engine. The search engine tries to profile this user with the queries it receives and thus further compromises user privacy. In other words, we model the search privacy violation as an information inference problem [5], where the input is user queries and the output is the privacy leaked though these queries. There may exist other information channels which can be employed by the attacker, such as voter registrations or medical records, while in this paper we only consider search queries.

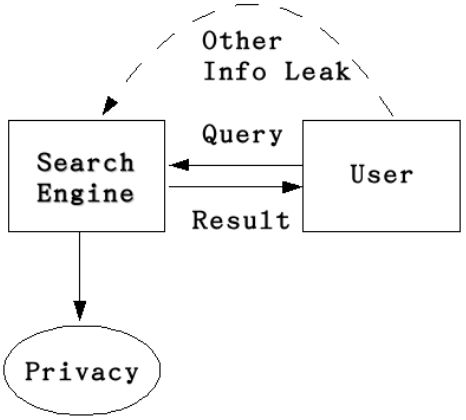

Fig. 1. Search Privacy Inference

To hide the real purpose of search users, it is possible to split a query into several sub-queries and assemble the results of these sub-queries. This approach, however, may not get the same results as the original query. First of all, many search engines only provide the top ranked search results, e.g. at most $1,000$ entries are provided by Google. Hence some results for the original query will be missing if they are not among the top ranked results for all the sub-queries. For example, splitting "Mountain View," a city in California, into "mountain" and "view" probably will not get the desired results for the user. Secondly, the ranking information for the original results is important, while it is hard for the user to recover or re-rank in many cases. In this paper we assume that the user can not change/split queries in order to get the results he/she wants, therefore a natural choice for privacy protection is to inject noise queries.

The next challenge is to formally define privacy under the search engine context. Rather than selecting a set of sensitive queries and protecting ad hoc privacy issues, this paper takes a rigorous approach to bound the possible privacy breaches and derives the optimal noise to protect search users. Moreover, our

model does not target specific privacy issues or user profiling algorithms thus can be applied to a wide range of application scenarios.

We believe that we have made the following contributions.

- We measure the search privacy breaches as the *mutual information* between real user queries and the diluted queries seen by the attacker, and formulate the noise injection problem as a mutual information minimization problem. (Section IV)
- We give the lower bound for the amount of noise queries required by a perfect privacy protection in the information theoretical sense. (Section V)
- We show how to generate optimal noise when the amount of noise is insufficient for perfect protection. (Section VI)
- We compute an approximate solution for the special case where equal number of noise queries are injected and evaluate our result with simulations. (Section VII)

TrackMeNot [6], a web browser extension, is developed to inject noise queries into user queries, while we are not aware of any analytical results published on how to generate noise queries such that privacy breaches are minimized or bounded by certain requirements. We believe that the theoretical analysis presented here complements existing privacy protection research and provides insights for more sophisticated protection tools.

The rest of this paper is organized as follows. After introducing the search privacy problem in Section II, we review related work in Section III. Section IV proposes our noise injection model and the mutual information measure for privacy breaches. Section V gives the lower bound for the expected number of noise queries required by perfect privacy protection. Section VI shows how to generate optimal noise with limited number of noise queries. Section VII computes an approximate solution of the optimal noise for a special case where equal number of noise queries are injected and verifies the results with simulations. We outline our future work in Section VIII and conclude this paper with Section IX.

II. Search Privacy

In this section, we discuss the methods to identify search users and the information sources for users profiling. Then we introduce existing search privacy protection solutions.

A. Search User Identification

The following methods are widely used to identify search users.

- *IP addresses*: IP addresses are probably the most popular user identifier since they are always available to search engines. IP addresses may fail when multiple users share one IP address, such as users behind proxies and NAT (Network Address Translation). DHCP (Dynamic Host Configuration Protocol) also makes the mapping between users and IP addresses unreliable.
- *HTTP cookies*: Supported by almost all modern web browsers, HTTP cookies are a common tool for web applications to identify users. For example, a long term cookie may be kept on the user side and every query will be companioned by this cookie. Such cookies allow search engines to keep track of users better than IP addresses.
- *Client-side search tool*: Client-side search software, such as search toolbars, can generate a unique user ID based on random numbers or information collected from the user side, e.g. user names, operation systems, and hardware configurations. This ID is then embedded in search requests to search engines.

Some tools have been developed to remove such identifiable information, most of which are web browser extensions, such as CustomizeGoogle [7] and PWS [8].

B. Information Sources for User Profiling

There are four major information sources for search engines to profile users.

- *Queries*: User queries are the major source of user information for search engines.
- *Click-through*: Correlating user queries with their corresponding clicked results, click-through data help profiling users with more accurate and detailed information [9].
- *User preferences*: Users may be able to specify their search preferences, such as languages, locations and even interested topics, which are often stored as cookies locally on the user side. Some existing work has incorporated user preferences for better search performance [10].
- *Rich client side*: With the help of search toolbars and desktop search, more user information can be collected from the user side, such as browsing history and even local documents.

C. Protection Solutions

Based on their deployment, existing search privacy protection solutions can be partitioned into the following three categories.

- *Server side*: Privacy preserving data mining allows search engines to profile users without compromising user privacy. We review related work in Section III-B.
- *Network*: Proxies and anonymity networks make it hard for search engines to identify users with IP addresses. Section III-C introduces current network based solutions.
- *User side*: This paper and tools such as TrackMeNot fall into this category.

Some solutions involve more than one party above. For example, most private information retrieval (PIR) approaches require both user and server side deployment.

Based on the target being protected, the protection methods can also be classified as follows.

- *User ID*: We can randomize the user identification and make it hard to keep a complete query log for each user. The possible solutions include:
  - *Mixing multiple users*: For example, it would be hard for search engines to map queries to individual users behind a proxy.

  - *Distributing queries*: For example, a user may distribute his queries to N proxies. When N is large, it would be hard for search engines to correlate these queries.

  In both cases, we assume that there is no other identifiable information sources such as cookies. This family of solutions requires trusted infrastructures such as proxy pools, which may not be available in some cases. Moreover, it is subject to single points of failure when the infrastructure is compromised.
- *Query Semantics*: Additional queries can be injected as noise to change the query semantics and cover the real search goals. For example, as indicated in [11], it is possible to protect user privacy by breaking contextual integrity.

In this paper we focus on how to protect query semantics by limiting the amount of information which search engines can infer based on (diluted) user queries.

III. Related Work

A. Private Information Retrieval

A large body of literature has been devoted to private information retrieval (PIR) [2], [12]–[15] where the user tries to prevent the database operator from knowing his/her interested records. For example, an investor wants to keep his/her interested stocks private from the stock-market database operator.

Chor *et al.* [2] proves that to get a perfect protection, in the information theoretical sense, the user has to query all the entries in the database when dealing with a single server scenario. Following PIR research focuses on a multi-server setting and/or computational restrictions on the server side, which leads to two main families of PIR: information-theoretical [2], [12], [14], [15] and computational [13], [16]. The former prevents any information inference even with unlimited computing resources on the server side, while the latter allows the servers with polynomial-time computational capabilities in most cases.

Multiple replicas of the database are required by many existing PIR, often with non-collusive assumptions, i.e. these replicas can not communicate with each other to compromise user privacy. Moreover, besides the simple query-answer interactions, various server-side computations are employed to reduce the communication cost, for example in [2], *exclusive-or*s are performed on the server side to reduce the size of answers. Such requirements for deployments or cooperations on the server side are infeasible for general search privacy protection. We will discuss the differences between our work and existing PIR in Section IV-C.

B. Privacy Preserving Data Mining

Extensive work has been conducted in the field of privacy preserving data mining [3]. We refer interested readers to [17] for a large collection of related literature.

Evfimievski *et al.* [18] investigates the problem of association rule mining with a randomization operator for privacy protection. An "amplification" method is proposed to limit privacy breaches. The basic idea is to determine how much information will be leaked if a certain mapping of the randomization operator is revealed, which is similar to the mutual information measure in our paper.

Many privacy preserving approaches target a relatively small dataset and a particular family of data mining algorithms. It is challenging for these approaches to protect large scale data sets against unknown user profiling methods [19]. For example, it would be prohibitive to apply k-anonymity [20] to search query logs, due to their large scales and highly skewed distributions.

Within the context of search privacy protection, privacy preserving data mining is a server-side solution. On one hand, users have no control of their data and the privacy associated with these data. On the other hand, the expensive cost incurred by these algorithms discourages search engines to deploy them.

C. Anonymity Browsing and Search

Proxies and anonymity networks have been widely used to protect user browsing privacy [21]. HTTP proxies allow users to hide their IP addresses from search engines. Onion routing [22] and its successor TOR [23] provide more sophisticated network protection, making it hard for attackers to trace users via network traffic analysis. Web Mixes [24] provides a more comprehensive architecture to generate anonymous real-time Internet access by adding an authentication mechanism to prevent flood attacks and a feedback system to inform the user about his/her current level of protection.

To remove other identifiable sources such as HTTP cookies, some web browser extensions have been developed for anonymity search, which can be combined with proxies and anonymity networks. Notable tools are listed as follows.

- CustomizeGoogle provides a large set of privacy protection options for Google users, including randomizing Google user ID, blocking Google Analytics cookies, and removing click-through logging.
- PWS [8] sends user queries through a TOR network and normalizes HTTP headers such that the HTTP requests from all PWS users "look like the same."
- Besides similar protection options as CustomizeGoogle, TrackMeNot claims to automatically generate "logical" noise queries according to the previous search results, while we are not aware of any published result on its details.

Many heuristic and search engine specific features have been used by these tools for search privacy protection. This paper attacks the problem with a general protection model and complements existing solutions with theoretical analysis.

IV. Basic Protection Model

A. Noise Injection

Noise injection has been widely used to protect information privacy [25]. In this section we formalize our noise injection model.

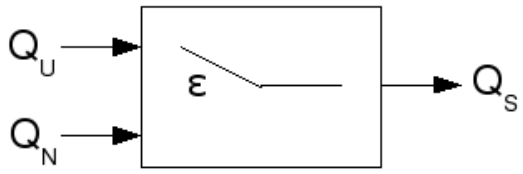

Fig. 2. Noise Injection Model

Shown as Figure 2, our noise injection model works as a black box with a selection switch inside. The black box generates diluted queries (Q_s) by mixing noise queries (Q_n) and user queries (Q_u), then sends Q_s to the malicious search engine. When the black box decides to send a query, with probability ϵ, the switch selects Q_u, and with probability $1-\epsilon$, it selects Q_n. Then a query in the front of the selected queue is sent. Both the status of the switch and ϵ are invisible/unknown to the search engine since they are locally decided at the user side. This process continues until all Q_u are sent. Thus Q_s on the server side follows:

$$\forall i \quad P(Q_s = q_i) = \epsilon P(Q_u = q_i) + (1-\epsilon)P(Q_n = q_i) \quad (1)$$

where $\{q_i\}$ are all the possible unique (privacy sensitive) queries and $\frac{\epsilon}{1-\epsilon}$ can be considered as the signal-to-noise ratio (SNR). The expected number of noise queries sent to the search engine is $\frac{1-\epsilon}{\epsilon}|Q_u|$, where $|Q_u|$ represents the total number of user queries sent to the search engine.

To avoid cumbersome, we denote $P(Q_s = q_i)$ as $P(s = i)$, which also applies to other probabilities when there is no confusion about the random variables.

B. Measure Privacy Breaches

There are two types of privacy breaches through search queries, *explicit* and *implicit*. Explicit privacy information, such as phone numbers, addresses and social security numbers, usually has fixed formats thus is easy to recognize and protect. For example, one may cover a real phone number with multiple randomly generated ones. Implicit privacy information, on the other hand, can not be obtained directly and has to be extracted from user queries via profiling methods or data mining algorithms. For example, it is possible to infer the user's income level via the brands of products he/she often searches [26]. In this paper, we are interested in implicit privacy breaches.

As indicated in [18], the distribution of Q_u may allow the attacker to learn with high confidence some sensitive information. Different Q_u distributions correspond to different user profiles and user profiling methods exploit their characteristics to reveal privacy sensitive information. In our noise injection model, the search engine or attacker can only learn Q_u via Q_s, thus our objective is to prevent or minimize the amount of information about Q_u which can be inferred through Q_s.

To quantify the notion of privacy breaches, we choose the *mutual information* between Q_s and Q_u, i.e. $I(Q_s; Q_u)$, as our measure. Formally, let $\{q_k\}$, $k = 1, 2, \cdots, N_Q$, be all the possible (sensitive) queries, where N_Q is the number of unique user queries. We have:

$$\begin{aligned} I(Q_s; Q_u) &= \textstyle\sum_i \sum_j P(u=i, s=j) \log \frac{P(u=i,s=j)}{P(u=i)P(s=j)} \\ &= \textstyle\sum_i \sum_j P(s=j|u=i)P(u=i) \log \frac{P(s=j|u=i)}{P(s=j)} \end{aligned} \quad (2)$$

More specifically, given Q_u, if $I(Q_s; Q_u) > I(Q'_s; Q_u)$, we believe that Q'_s is better than Q_s for privacy protection [27]. Therefore, our goal is to find a set of Q_n which minimizes $I(Q_s; Q_u)$.

We have the following conditional probability:

$$P(s=j|u=i) = \begin{cases} \epsilon + (1-\epsilon)P(n=j|u=i) & j = i \\ (1-\epsilon)P(n=j|u=i) & j \neq i \end{cases} \quad (3)$$

The model presented in this paper considers all user queries equally sensitive. It has to be noted, however, that the same amount of privacy breach (measured by mutual information) may correspond to different risks or costs. For example, assuming that one bit information is leaked, knowing whether a user has a certain disease is probably more sensitive than knowing whether the user is interested in a certain class of music. Moreover, different users have different privacy concerns, thus we need to investigate the particular user or application scenarios to assess the risks or costs associated with different privacy breaches, which is beyond the scope of this paper.

C. Comparison With PIR

It might seem natural to formulate search privacy protection as a private two-party computation problem within the PIR framework, while the following comparison shows why our model is chosen.

- *Objective*: PIR provides a secure computation protocol to prevent the database from knowing one or multiple items which the user is interested in. The objective of our approach is to protect the probability distribution of Q_u, instead of the query set. Moreover, our model allows a certain portion of information leakage (as shown in Section VI), which is a trade-off towards practical applications in the search scenario, while most existing PIR do not have such flexibility.
- *Server side assumptions*: Most PIR research requires multiple non-collusive database replicas and various computations performed on the server side. Our model assumes a single search engine and its only service for the user is to return the search results according to the queries, which is exactly how current search engines work. We also have no assumptions on the server-side computational capabilities.
- *Measure*: It directly follows the difference of objectives. Most existing PIR models do not allow any information leakage so their most important measure is the communication cost, which considers not only the cost between the user and the servers but also the cost between servers when some servers collude. Our approach measures the

privacy breach with mutual information and only considers the user side cost, i.e. the number of noise queries to inject.

V. Perfect Privacy Protection

Since mutual information can not be negative, the *perfect protection* in our model is $I(Q_s;Q_u)=0$. In this section, we give the lower bound for the amount of noise queries required by this perfect protection. This bound is a function of ϵ and the number of user queries, denoted as $|Q_u|$. In our noise injection model, given ϵ, the more user queries we have, the more noise queries will be injected. $|Q_u|$ is decided by the user/application thus we need the upper bound for ϵ, which is given by the following theorem.

Theorem 1: $I(Q_s;Q_u)=0$ only if $\epsilon \leq 1/N_Q$.

The proof is presented in the appendix. To see that there exists Q_n which satisfies Theorem 1, we show an example here.

Let $\epsilon = \frac{1}{N_Q}$, and

$$P(n=j|u=i)=\begin{cases} 0 & j=i \\ \frac{1}{N_Q-1} & j\neq i \end{cases}$$

In other words, given a user query q_i, the noise is equally likely to be any other query except q_i. With Equation 3, we have

$$P(s=j|u=i)=\begin{cases} \frac{1}{N_Q}+(1-\frac{1}{N_Q})\times 0=\frac{1}{N_Q} & j=i \\ (1-\frac{1}{N_Q})\frac{1}{N_Q-1}=\frac{1}{N_Q} & j\neq i \end{cases}$$

Hence Q_s is uniformly distributed in the entire query space. Plug $P(s=j|u=i)=P(s=j)$ into Equation 2, we get $I(Q_s;Q_u)=0$, i.e. the bound given by Theorem 1 (therefore it is a tight bound).

This solution actually sends every user query with the entire possible query set on average, thus the search engine fails to find the queries which the user is really interested in.

To get a perfect protection, the lower bound for the expected number of noise queries is

$$E(|Q_n|)=\frac{1-\epsilon}{\epsilon}|Q_u| \geq (N_Q-1)|Q_u|$$

This is also a tight bound.

The analysis presented in this section shows that it is expensive for a perfect privacy protection in the information theoretical sense. The fundamental challenge here is that mutual information is a strong requirement. There may be information which is not privacy sensitive although it is over a set of sensitive queries. In our model, however, we eliminate the chance for the search engine/attacker to infer such information as well.

To get a smaller N_Q, we need to limit $\{q_i\}$ to all the privacy sensitive queries, such as those related to sensitive medical information, including queries which are not sensitive individually while may lead to successful inference if being correlated. Such analysis is user/application dependent and out of the scope of this paper.

VI. Limited and Independent Noise

The number of noise queries has to be small or in other words, ϵ has to be larger than $1/N_Q$ in many application scenarios.

- Both the user and the search engine have limited network bandwidth.
- Search engines will deny users who issue a large number of queries within a short period of time to prevent deny-of-service (DoS) attack.
- The user wants his/her real queries to be answered quickly. In our current model, assuming that each query takes the same amount of time, the time for a user query to be served follows a geometric distribution with parameter ϵ, hence the expected waiting time for every real query is $1/\epsilon$. A small ϵ means a long waiting period.

Furthermore, we want Q_u and Q_n to be independent, which simplifies the implementation and makes it much easier for parallelism. With this independence requirement, Equation 3 can be rewritten as

$$P(s=j|u=i)=\begin{cases} \epsilon+(1-\epsilon)P(n=j) & j=i \\ (1-\epsilon)P(n=j) & j\neq i \end{cases} \quad (4)$$

Then Equation 2 becomes

$$\begin{aligned} I(Q_s;Q_u) &= \epsilon\sum_i P(u=i)\log\frac{\epsilon+(1-\epsilon)P(n=i)}{P(s=i)} \\ &+(1-\epsilon)\sum_i P(u=i)[\sum_{j,j\neq i} P(n=j)\log\frac{(1-\epsilon)P(n=j)}{P(s=j)} \\ &+P(n=i)\log\frac{\epsilon+(1-\epsilon)P(n=i)}{P(s=i)}] \end{aligned} \quad (5)$$

Let $u_i=P(u=i)$, $n_i=P(n=i)$, and $\alpha=\epsilon/(1-\epsilon)$, we rearrange Equation 5 as:

$$\begin{aligned} I(Q_s;Q_u) &= \epsilon\sum_i u_i\log\frac{\alpha+n_i}{\alpha u_i+n_i} \\ &+(1-\epsilon)[\sum_i n_i\log\frac{n_i}{\alpha u_i+n_i}+\sum_i u_i n_i\log\frac{\alpha+n_i}{n_i}] \end{aligned}$$

Then our task becomes

$$\arg\min_{\boldsymbol{n}} I(\boldsymbol{n}) \quad \text{w.r.t.} \ \sum_i n_i=1, \quad \forall i \quad n_i\geq 0 \quad (6)$$

where $\boldsymbol{n}=(n_1,n_2,\cdots,n_{N_Q})$.

First we show that I is a convex function of $\boldsymbol{n}$. Since I is a continuous twice differentiable function over its domain, we just need to show that its second derivative is positive. The intermediate steps are omitted here due to the page limit. Interested readers may refer to our technical report [28] for more details.

$$\frac{\partial I}{\partial n_i}=(1-\epsilon)[u_i\log(\alpha+n_i)-\log(\alpha u_i+n_i)+(1-u_i)\log n_i]$$

Therefore

$$\frac{\partial^2 I}{\partial n_i^2} = (1-\epsilon)(\frac{u_i}{\alpha+n_i} - \frac{1}{\alpha u_i + n_i} + \frac{1-u_i}{n_i}) = \frac{(1-\epsilon)(1-u_i)u_i\alpha^2}{(\alpha+n_i)(\alpha u_i+n_i)n_i} > 0$$

Then we can use Lagrange multipliers to solve Equation 6. Use the constraint to define a function $g(\boldsymbol{n})$:

$$g(\boldsymbol{n}) = \sum_i n_i - 1$$

Let

$$\Phi(\boldsymbol{n}, \lambda) = I(\boldsymbol{n}) + \lambda g(\boldsymbol{n})$$

Solve for the critical values of Φ:

$$\forall n_i \quad \frac{\partial \Phi}{\partial n_i} = \frac{\partial I}{\partial n_i} + \lambda = 0$$

$$\lambda = (\epsilon-1)[u_i \log(\alpha+n_i) - \log(\alpha u_i + n_i) + (1-u_i)\log n_i] \quad (7)$$

Decide the desired ϵ and solve the system of Equation 7, we will get the optimal noise Q_n which is independent of Q_u and minimizes privacy breaches.

It would be computationally expensive to solve the nonlinear systems represented by Equation 7 when N_Q is large. In some special scenarios, such as searching medical databases, Q_u may be limited to a small dictionary, while for general web search, N_Q will be huge. We discuss possible ways to reduce N_Q in Section VII-B.

VII. A Special Case: $E(|Q_n|) = |Q_u|$

In this section, we give an approximate solution which can be computed quickly for the case $\epsilon = 0.5$. In other words, equal amount of noise queries are injected into user queries.

A. Approximate Solution

Using Taylor series to replace the logarithm function in Equation 7, we have

$$\ln(1+x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{n+1} x^{n+1} : \quad |x| < 1 \quad (8)$$

We take the following approximation

$$\ln(1+x) \approx x - \frac{x^2}{2} + \frac{x^3}{3} : \quad |x| < 1 \quad (9)$$

Since we assume $\epsilon = 0.5$ in this section, Equation 7 can be rewritten as follows.

$$\begin{aligned}
&\lambda = -0.5[u_i \log(1+n_i) - \log(u_i+n_i) + (1-u_i)\log n_i] \\
&\approx -0.5[u_i(n_i - \frac{n_i^2}{2} + \frac{n_i^3}{3}) - (u_i+n_i-1) \\
&+\frac{(u_i+n_i-1)^2}{2} - \frac{(u_i+n_i-1)^3}{3} \\
&+(1-u_i)((n_i-1) - \frac{(n_i-1)^2}{2} + \frac{(n_i-1)^3}{3})] \\
&= -0.5(\frac{3}{2}u_i^2 - u_i^2 n_i - \frac{u_i^3}{3} + u_i n_i - \frac{7}{6}u_i)
\end{aligned}$$

Thus we have

$$\hat{n}_i = \frac{7-2u_i}{6} + \frac{2\lambda}{u_i^2 - u_i} \quad (10)$$

Considering $\sum_i n_i = 1$, we have

$$\hat{\lambda} = \frac{1 - \sum_i \frac{7-2u_i}{6}}{\sum_i \frac{2}{u_i^2-u_i}} \quad (11)$$

Combining Equation 10 and 11, we can solve for $\hat{n}_i$ given u_i.

One issue here is how accurate this solution is. We list the factors which support this solution as follows.

- The objective function, Equation 6, is convex, which means the smaller $|n_i - \hat{n}_i|$ is, the smaller $|I - \hat{I}|$ we get.
- $\hat{n}_i$ and n_i are asymptotically equivalent. We can always increase the order of the Taylor series as long as we can afford the computation cost.

The counter argument is that I involves a summation of n_i, thus the offset of $|n_i - \hat{n}_i|$ will be accumulated.

B. Simulation Results

Simulations are conducted to evaluate the solution we get in the previous section. Generally speaking, the larger the entropy of Q_u, $H(Q_u)$, is, the larger $I(Q_s; Q_u)$ will be. Hence we consider the following *relative mutual information* to show privacy breaches with less bias from Q_u.

$$\frac{I(Q_s; Q_u)}{H(Q_u)} \quad (12)$$

We assume that the user queries follow a power law distribution, i.e. the number of the ith most popular queries is proportional to $i^{-\alpha}$, which has been observed in many previous studies [29]. In our experiment, α is ranging from 1.0 to 5.5, which covers most power law observations in real world [30]. N_Q is ranging from 100 to 1,000.

We employ Octave [31], a numerical computation software, to solve Equation 10 and 11. Uniformly distributed random noise is chosen as a baseline. Shown as Figure 3, our noise greatly reduces the privacy breach and performs much better than the uniform noise consistently.

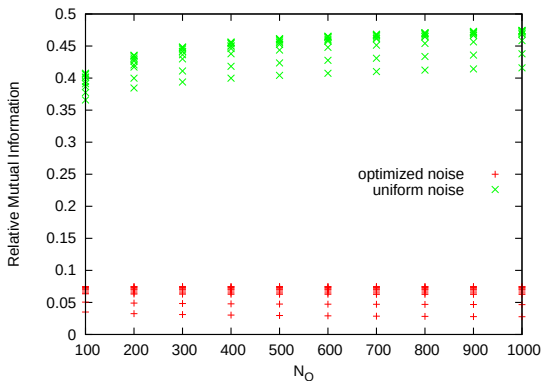

Fig. 3. Privacy Protection: Optimized noise *vs.* uniform noise

Although our simulations are based on several hundreds of user queries, the results are encouraging and promising.

- In many cases, privacy information is restricted within a relatively small set of queries, for example, sensitive medical information, racial or ethnic origins, political or religious beliefs or sexuality.
- As the number of user queries, N_Q, increases, the protection of random noise gets worse as more information is leaked, while our optimized noise does not exhibit such trend, which is a good indicator for its performance when dealing with larger N_Q.
- Combining network solutions with noise injection will help us reduce N_Q. For example we may divide all the queries into N disjoint subsets and send them through N proxies.

VIII. Future Work

In this section we outline the directions to further improve our work.

- As mentioned earlier, there may be some non-sensitive information, which users do not mind to share with search engines, while our current model bounds all the information leakage. We want to relax this constraint by allowing non-sensitive inferences in a restricted way. Some server-side solutions have been proposed, such as the obfuscated database model in [32], while how to achieve it on the user side is still not clear.
- Our threat model assumes that the attacker only knows search queries, while in some cases the attacker may have external knowledge or other information channels. For example, the movie ratings on IMDB have been used to compromise the privacy of Netflix users in a recent study [33]. More specifically, if the attacker has some knowledge or heuristics to distinguish between noise queries and real user queries, the mutual information measure needs to be changed to accommodate such prior knowledge about Q_u, which may lead to more sophisticated protection models.
- Our approach requires that the probability distribution of Q_u is known in advance, while in real world it is hard to predict the future queries for a user. An adaptive noise generator, which actively adjusts Q_n according to the empirical distribution of Q_u, will be an important step towards practical applications.
- We have no assumptions on the user profiling methods employed by the attacker, which makes protection extremely challenging. Having some computational constraints for the attacker would help reducing the number of noise queries. For example, most computational PIR [16] allows servers with polynomial-time computational capabilities only. Moreover, breaking assumptions of user profiling methods, such as query contextual integrity, will be another interesting direction to explore [11].

IX. Conclusion

Privacy issues in the search engine context have become a new threat to Internet/Web users. Currently many search engines claim that obsolete user queries will be removed or anonymized, while their data retention window ranges from months to years. Vulnerable privacy protection methods and practices have caused user privacy leakage. Notable examples include the search log released by AOL in 2006 and the movie rental history released by Netflix [33] in 2006. Existing private information retrieval and privacy preserving data mining approaches require prohibitive server-side deployments which makes them infeasible for search privacy protection. Moreover, server-side solutions are always subject to insider attacks.

This paper proposes a user-side noise injection model, which grants users with the power to protect their privacy from a malicious search engine with no assumptions on the user profiling algorithms. We prove the lower bound for the expected number of noise queries needed by a perfect protection in the information-theoretical sense. Then we show how to compute the optimal noise when the number of noise queries is insufficient and give an approximate solution for the special case when equal number of noise queries are injected into user queries. Our simulation results show that the noise generated by our approach greatly reduces the privacy leakage and provides much better protection than uniform noise.

We believe that the theoretical analysis presented in this paper complements existing research and sheds light on the design and implementation of better privacy protection applications.

Appendix
Proof of Theorem 1

Proof: Assuming a set of noise queries following the probability distribution $P(Q_n)$ which makes $I(Q_s; Q_u) = 0$, hence Q_u and Q_s are independent, i.e.

$$P(Q_s|Q_u) = P(Q_s) \tag{13}$$

Here we require $\forall q_i,\ P(u = i) \neq 0$, otherwise the conditional probability is undefined. We assume that we select all the non-zero q_i for our discussion without losing generality.

On the other hand, according to Equation 1, we have

$$\begin{aligned} P(s = i) &= \epsilon P(u = i) + (1 - \epsilon)P(n = i|u = i)P(u = i) \\ &\quad +(1 - \epsilon)P(n = i|u \neq i)P(u \neq i) \end{aligned}$$

Condition both sides with $u = i$

$$P(s = i|u = i) = \epsilon + (1 - \epsilon)P(n = i|u = i) \tag{14}$$

Combining Equation 1, 13 and 14, we have

$$\epsilon + (1 - \epsilon)P(n = i|u = i) = \epsilon P(u = i) + (1 - \epsilon)P(n = i)$$

Rearranging terms and dividing by $P(u \neq i)$, we get

$$P(n = i|u \neq i) - P(n = i|u = i) = \frac{\epsilon}{1 - \epsilon} \tag{15}$$

Notice that

$$P(n=i|u\neq i)=\sum_{j,j\neq i}[P(n=i|u=j)P(u=j|u\neq i)]$$
$$=\frac{P(n=i)-P(n=i|u=i)P(u=i)}{P(u\neq i)} \quad (16)$$

Plug Equation 16 into Equation 15, we get

$$P(n=i|u=i)+\frac{\epsilon}{1-\epsilon}P(u\neq i)=P(n=i)$$

Considering $\sum_i P(n=i)=1$, we have

$$1=\sum_i[P(n=i|u=i)+\frac{\epsilon}{1-\epsilon}P(u\neq i)]$$
$$1=\sum_i P(n=i|u=i)+\frac{\epsilon}{1-\epsilon}\sum_i(1-P(u=i))$$
$$1=\sum_i P(n=i|u=i)+\frac{\epsilon}{1-\epsilon}(N_Q-1)$$
$$\frac{\epsilon}{1-\epsilon}=\frac{1-\sum_i P(n=i|u=i)}{N_Q-1}$$

Thus

$$\frac{\epsilon}{1-\epsilon} \leq \frac{1}{N_Q-1}$$
$$\epsilon \leq \frac{1}{N_Q} \quad (17)$$

■

ACKNOWLEDGEMENTS

The authors would like to thank Nan Ma, Norm Matloff and Xiaohui Ye for their valuable comments and Bernard Levy for checking our math in Section VI. Equations in Section VII are verified by the Yacas computer algebra system [34]. This research is funded in part by National Science Foundation under CNS-0832202, CNS-0435531 and EIA-0224469, Intel, and a MURI grant from Army Research Office under the ARSENAL project. The opinions contained in this article are those of the authors and do not necessarily reflect those of the funding agencies.

REFERENCES

[1] "Google privacy FAQ," http://www.google.com/privacy_faq.html.
[2] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *Journal of ACM*, vol. 45, no. 6, pp. 965–981, 1998.
[3] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *SIGMOD'00: Proceedings of the ACM international conference on Management of data*, 2000, pp. 439–450.
[4] "AOL's massive data leak," http://w2.eff.org/Privacy/AOL/.
[5] S. K. M. Wong and Y. Y. Yao, "On modeling information retrieval with probabilistic inference," *ACM Transactions on Information Systems*, vol. 13, no. 1, pp. 38–68, 1995.
[6] "TrackMeNot," http://mrl.nyu.edu/~dhowe/trackmenot.
[7] "CustomizeGoogle," http://www.customizegoogle.com.
[8] F. Saint-Jean, A. Johnson, D. Boneh, and J. Feigenbaum, "Private web search," in *WPES'07: Proceedings of the ACM workshop on Privacy in electronic society*, 2007, pp. 84–90.
[9] G.-R. Xue, H.-J. Zeng, Z. Chen, Y. Yu, W.-Y. Ma, W. Xi, and W. Fan, "Optimizing web search using web click-through data," in *CIKM'04: Proceedings of the 13th ACM international conference on Information and knowledge management*, 2004, pp. 118–126.
[10] A. H. Keyhanipour, B. Moshiri, M. Kazemian, M. Piroozmand, and C. Lucas, "Aggregation of web search engines based on users' preferences in webfusion," *Knowledge-Based Systems*, vol. 20, no. 4, pp. 321–328, 2007.
[11] A. Barth, A. Datta, J. C. Mitchell, and H. Nissenbaum, "Privacy and contextual integrity: Framework and applications," in *SP'06: Proceedings of the IEEE Symposium on Security and Privacy*, 2006, pp. 184–198.
[12] A. Beimel, Y. Ishai, and E. Kushilevitz, "General constructions for information-theoretic private information retrieval," *Journal of Computer System Science*, vol. 71, no. 2, pp. 213–247, 2005.
[13] 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*, Oct 1997, pp. 364–373.
[14] A. Beimel, Y. Ishai, E. Kushilevitz, and J.-F. Raymond, "Breaking the $O(n^{1/(2k-1)})$ barrier for information-theoretic private information retrieval," in *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*, 2002, pp. 261–270.
[15] I. Goldberg, "Improving the robustness of private information retrieval," in *SP'07: Proceedings of the IEEE Symposium on Security and Privacy*, May 2007, pp. 131–148.
[16] B. Chor and N. Gilboa, "Computationally private information retrieval (extended abstract)," in *STOC'97: Proceedings of the 29th annual ACM symposium on Theory of computing*, 1997, pp. 304–313.
[17] K. Liu, "Privacy preserving data mining bibliography," 2007, http://www.cs.umbc.edu/~kunliu1/research/privacy_review.html.
[18] A. Evfimievski, J. Gehrke, and R. Srikant, "Limiting privacy breaches in privacy preserving data mining," in *Proceedings of the 21nd ACM SIGMOD-SIGACT-SIGART symposium on Principles of Database Systems(PODS)*, 2003, pp. 211–222.
[19] L. Liu, M. Kantarcioglu, and B. Thuraisingham, "The applicability of the perturbation based privacy preserving data mining for real-world data," *Data Knowledge Engineering*, vol. 65, no. 1, pp. 5–21, 2008.
[20] L. Sweeney, "k-anonymity: a model for protecting privacy," *International Journal on Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, pp. 557–570, 2002.
[21] M. S. Olivier, "Distributed proxies for browsing privacy: a simulation of flocks," in *SAICSIT'05: Proceedings of the annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries*, 2005, pp. 104–112.
[22] D. Goldschlag, M. Reed, and P. Syverson, "Onion routing," *Communications of ACM*, vol. 42, no. 2, pp. 39–41, 1999.
[23] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: the second-generation onion router," in *SSYM'04: Proceedings of the 13th conference on USENIX Security Symposium*, 2004, pp. 21–21.
[24] O. Berthold, H. Federrath, and S. Köpsell, "Web mixes: a system for anonymous and unobservable internet access," in *International workshop on Designing privacy enhancing technologies*, 2001, pp. 115–129.
[25] I. James W. Gray, "On introducing noise into the bus-contention channel," in *Proceedings of the IEEE Symposium on Security and Privacy*, 1993, pp. 90–98.
[26] B. Tancer, *Click: What Millions of People Are Doing Online and Why it Matters*. Hyperion, 2008.
[27] Y. Zhu and L. Liu, "Optimal randomization for privacy preserving data mining," in *KDD'04: Proceedings of the 10th ACM international conference on Knowledge discovery and data mining*, 2004, pp. 761–766.
[28] S. Ye, F. Wu, R. Pandey, and H. Chen, "Noise injection for search privacy protection," Department of Computer Science, University of California, Davis, Tech. Rep. CSE-2008-10, 2008.
[29] R. A. Baeza-Yates, "Applications of web query mining," in *ECIR'05: Proceedings of the 27th European Conference on IR Research*, 2005, pp. 7–22.
[30] R. Albert and A.-L. Barabasi, "Statistical mechanics of complex networks," *Reviews of Modern Physics*, vol. 74, pp. 47–97, 2002.
[31] "GNU Octave," http://www.gnu.org/software/octave/.
[32] A. Narayanan and V. Shmatikov, "Obfuscated databases and group privacy," in *CCS'05: Proceedings of the 12th ACM conference on Computer and communications security*, 2005, pp. 102–111.
[33] ——, "Robust de-anonymization of large datasets," in *Proceedings of the IEEE Symposium on Security and Privacy*, 2008.
[34] "Yacas computer algebra system," http://yacas.sourceforge.net.