On Masking Topical Intent in Keyword Search

Peng Wang and Chinya V. Ravishankar
Department of Computer Science & Engineering
University of California–Riverside, Riverside, CA 92521, USA
{wangpe, ravi}@cs.ucr.edu

***Abstract*—Text-based search queries reveal user intent to the search engine, compromising privacy. Topical Intent Obfuscation (TIO) is a promising new approach to preserving user privacy. TIO masks topical intent by mixing real user queries with dummy queries matching various different topics. Dummy queries are generated using a Dummy Query Generation Algorithm (DGA).**

We demonstrate various shortcomings in current TIO schemes, and show how to correct them. Current schemes assume that DGA details are unknown to the adversary. We argue that this is a flawed assumption, and show how DGA details can be used to construct efficient attacks on TIO schemes, using an iterative DGA as an example. Our extensive experiments on real data sets show that our attacks can flag up to 80% of dummy queries.

We also propose HDGA, a new DGA that we prove to be immune to the attacks based on DGA semantics that we describe.

I. Introduction

Information retrieval based on text or keyword search is now ubiquitous, thanks to its speed and convenience. Users submit text queries to text-search providers (search engines), which maintain massive text databases optimized for search. Search services can be free, since providers derive value by building profiles for users from received queries. Profiles allow search providers to both improve search results through personalization, as well as generate revenue via targeted advertisements.

Users, however, may have a contrary interest. Queries can leak considerable private information to search engines, which routinely record and use it to construct detailed user profiles. If Alice searches for information on certain diseases, say, she reveals something about her interests, perhaps even about her health. Alice loses control over her profile, which may be sold to third parties, and become widely disseminated. As [1] points out, this may even create legal liabilities for users.

Various approaches exist to address this issue, each with its own shortcomings. Those aiming at query indistinguishability negate the two greatest advantages of text-based search, i.e., efficiency and similarity-based retrieval on imprecise keywords. Using Private Information Retrieval [2]–[5] achieves indistinguishability, but incurs debilitating overhead. Searchable encryption approaches [6]–[13] gain efficiency, but make weaker guarantees. Most significantly, they preclude imprecise searches, a bedrock advantage of text-based search.

Keyword-Based Obfuscation (KBO) Schemes [1], [14]–[19] are another alternative. They provide weaker guarantees still, and try merely to obscure user intent. Such methods typically hide the real query in a mass of dummy queries generated using a Dummy Query Generation algorithm (DGA). The real and dummy queries are sent as a single query group to the search engine. The search engine returns results for all queries in the group. Dummy results are subsequently filtered out. Many KBO schemes have the advantage of being purely client-based, requiring no changes to the search engine.

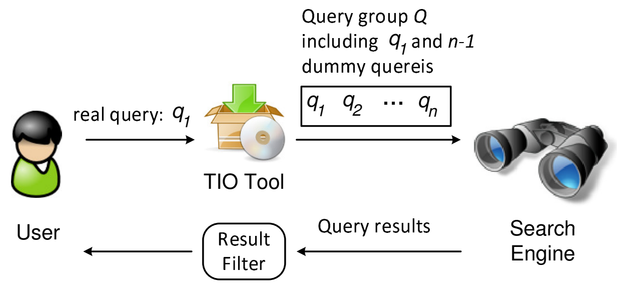


Fig. 1: Scheme model.

For instance, TrackMeNot [17] uses this approach, but does not satisfactorily address the generation of keywords that are semantically consistent with the same topic for each dummy query. It is easy to flag dummy queries by semantic exploration [20]. The work in [14] embellishes each query with decoy terms, but requires search engine modifications. A secure scheme must ensure that the real and dummy queries are indistinguishable. No current KBO scheme is known to be fully secure [20]. Various attacks can filter out dummy queries.

A. Topical Intent Obfuscation (TIO)

The work in [1] represents a novel and promising approach, going beyond masking *keywords* in user queries to obfuscating their *topics of interest*. This TIO scheme, denoted *TIO-1*, uses both a Semantic Classification Algorithm (SCA) and a DGA. Its SCA uses Latent Dirichlet Allocation (LDA) to classify documents into topics. Its DGA first finds topics matching a real user query q, then obfuscates q by creating dummy queries matching other topics. Given its focus on topical intent, it ensures that keywords in each dummy query match the same topic, avoiding the pitfalls of schemes like [17]. Such coherence makes it hard to flag dummy queries by semantic exploration.

To find topical relevance and ensure obfuscation, the scheme uses thresholds ϵ_1, ϵ_2, with $\epsilon_2 < \epsilon_1$. To find the relevance of a topic t to a query q, a prior belief $\Pr[t]$ is first formed on t's relevance based on the general interest patterns of all users. Next, using information latent in q's keywords, this prior is updated to the posterior $\Pr[t|q]$. Topic t is taken as relevant if q enhances belief by at least ϵ_1, that is, if $\Pr[t|q] - \Pr[t] > \epsilon_1$.

Obfuscation is achieved by using dummy queries to reduce this gain to below ϵ_2.

B. Exposing Weaknesses in TIO Schemes

TIO-1 is superior to KBO schemes in many ways. However, it is also claimed in [1] that *TIO-1* effectively obfuscates user intention in queries, and that the (ϵ_1, ϵ_2) scheme creates reasonable doubt in the adversary's mind whether topics relevant to a query constitute the user's true intentions.

We show in this paper that these claims are overly broad, and demonstrate several novel attacks that seriously undermine *TIO-1* and all similar TIO schemes. We construct effective attacks using the semantics of DGAs in current obfuscation schemes, which are secure only under the naive assumption that the adversary does not know the specific DGA used.

1) Attacking TIO Schemes: The assumption that DGAs are secret is unacceptable, given standard practice in the security field. It is contrary to Kerckhoff's principle, a standard security design postulate dating as far as 1883, which dictates that security algorithms be public. Assuming a naive adversary is never a sound strategy.

One must assume that the adversary will know the DGA, one way or another. Now, the following attack suggests itself. Let q_1 be a real query, and let $Q = \{q_1, q_2, \ldots, q_n\}$ be the query group generated by the the DGA $\mathcal{G}$, where $q_2 \ldots, q_n$ are dummy queries. The adversary knows Q. Let $\mathcal{Q}_{\mathcal{G}}(q_i)$ denote the set of all query groups that $\mathcal{G}$ might generate, given a real query q_i. Clearly, $Q \in \mathcal{Q}_{\mathcal{G}}(q_1)$. By the same token, however, for each $q \in Q$ such that $Q \notin \mathcal{Q}_{\mathcal{G}}(q)$, the adversary can be certain that q is a dummy query.

That is, the adversary can filter out many, if not all dummy queries by simply checking if the query group is consistent with each query in the group under the DGA $\mathcal{G}$. We will demonstrate the feasibility of this attack, or variants, against iterative DGAs like the DGA of *TIO-1*. Iterative DGAs are very attractive since they permit high security goals to be achieved efficiently. However, they are susceptible to our attacks.

Another significant flaw in current KBO and TIO schemes is their focus on obfuscating one real query at a time. They do not guard against attacks that combine information across query groups. In many schemes, including *TIO-1* [1], dummy query topics are randomly chosen. Say that a user issues many real queries relating to some specific topic. The adversary can gain information about this user's topical intention by finding topics common across the user's query groups. This attack is discussed in [20], but no solution is proposed. We call this an *aggregation attack*, and show how to address it in this paper.

C. Our Contributions

Our contributions are as follows. First, we present a novel class of attacks on TIOs, as we have outlined above. We will present three instances of such attacks to break *TIO-1* in [1]. We will also show that the (ϵ_1, ϵ_2) scheme for determining and obfuscating topical intention is not sound, and does not perform as claimed in [1]. We will, in fact, demonstrate attacks that perform better when (ϵ_1, ϵ_2) are chosen to maximize their supposed effectiveness.

Second, we propose a new DGA called HDGA, which we show is free from the vulnerabilities of the *TIO-1* DGA [1]. We also show how to deal with aggregation attacks by selecting topics using the Highest Random Weight (HRW) algorithm [21].

We perform extensive experiments on real datasets to show how serious these attacks are. Our results show that up to 80% dummy queries can be flagged by our attacks.

The rest of the paper is organized as follows. Related work appears in Sec. II. Sec. III covers preliminaries. Sec. IV gives an overview of our scheme and the security model. Sec. V analyzes the security of iterative DGAs, and *TIO-1* in [1]. Sec. VI presents our scheme. Sec. VII presents experiments on real-world datasets. Sec. VIII concludes the paper.

II. Related Work

Many features make KBO and TIO schemes attractive. They generally operate on the client site, and require no changes to search engines, preserving their performance and utility. Their communication and computation overhead is small and tunable. If $n-1$ dummy queries accompany a real query, the communication overhead is $O(n)$, far lower than for PIR based schemes [2]–[4], where the communication overhead of a single retrieval is linear in the database size [1].

Queries are not encrypted in KBO and TIO schemes, and plaintext search is very easy, while encryption-based schemes require expensive encryption and decryption operations. Searchable encryption schemes like predicate encryption [6] allow users to send encrypted queries to the search engine. The search engine runs encrypted queries on encrypted databases, and returns query results to users. However, as pointed in [1], these schemes cannot be used in the the vector space model of text retrieval. Moreover, they require changes to the search engine to support searchable encryption schemes.

TrackMeNot [17], a web browser plugin was the first query obfuscation scheme used in practice. TrackMeNot hides real queries within a set of randomly generated dummy queries. However, dummy queries can be filtered out by exploring semantics [20].

In [1], Pang et al. proposed a query obfuscating scheme that uses the LDA model [22] to retrieve topics from a database, and an iterative DGA to generate dummy queries. Since dummy queries use semantically coherent keywords, they cannot be flagged by exploring semantics.

In [14], Pang et al. proposed a query embellishment scheme. In this scheme, decoy terms are added to each user query to hide the user intention. However, this scheme requires that the text search engine be modified accordingly, which is unrealistic.

In [15], Shen et al. tried to protect query privacy by adding a trusted third party which replaces the text search engine to perform the search service. But it makes the text search model more complex. Even if we have a third party with enough power to perform the search service, we still need to consider how to protect query privacy when the third party is not trustworthy.

In [16], Murugesan and Clifton proposed a plausible deniable search scheme. In this scheme, a set of canonical queries is first generated. Each real query is first substituted with a canonical query most similar to the real one, and forms a query group with $n-1$ other canonical queries. However, this scheme requires each real query to be replaced by a canonical query, and only canonical queries can be sent to the search engine. This restricts the utility of this scheme.

In [20], authors gave an excellent survey of current schemes, such as [23]–[26]. We refer interested readers to this paper for other related work not reviewed here.

III. Preliminaries

We briefly review several ideas central to our presentation.

A. The LDA Model For TIO

The Latent Dirichlet Allocation (LDA) model views each document as a mixture of topics distributed according to a Dirichlet prior [22]. Given a text database and the number of topics, we can use this model to classify documents in the database into topics. Let $\mathbf{D} = \{d_1, d_2, \ldots, d_{|\mathbf{D}|}\}$ denote a database of $|\mathbf{D}|$ documents, $\mathbf{T} = \{t_1, t_2, \ldots, t_{|\mathbf{T}|}\}$ denote a topic set, and $\mathbf{K} = \{w_1, w_2, \ldots, w_{|\mathbf{K}|}\}$ denote the keyword set corresponding to $\mathbf{D}$. *TIO-1* uses LDA to classify documents and obtain the following probabilities.

- $\Pr[w|t], \forall w \in \mathbf{K}, \forall t \in \mathbf{T}$. This conditional probability models how a keyword w is related to a given topic t.
- $\Pr[t|d], \forall t \in \mathbf{T}, \forall d \in \mathbf{D}$. This conditional probability models how a topic t is related to a given document d.

As in [1], we assume that each document $d \in \mathbf{D}$ is equally useful, so that $\Pr[d] = \frac{1}{|\mathbf{D}|}$. We further derive the following probabilities used in LDA model based query obfuscation:

- $\Pr[t]$. This is prior belief of a topic $t \in \mathbf{T}$, indicating the coverage of topic t in $\mathbf{D}$.
$$\Pr[t] = \sum_{d\in\mathbf{D}} \Pr[t|d]\Pr[d] = \frac{1}{|\mathbf{D}|}\sum_{d\in\mathbf{D}} \Pr[t|d]$$
- $\Pr[w]$. This is the prior belief of a word $w \in \mathbf{K}$.
$$\Pr[w] = \sum_{t\in\mathbf{T}} \Pr[w|t]\Pr[t]$$
- $\Pr[t|q]$. This is the probability that a user's intention is topic t, given query q with keywords $\{w_{q,1}, \ldots, w_{q,m}\}$.
$$\Pr[t|q] = \sum_{w\in q} \Pr[t|w]\Pr[w]$$
where $\Pr[t|w] = \frac{\Pr[w|t]\Pr[t]}{\Pr[w]}$.
- $\Pr[t|Q]$. This is the probability that a user's intention is t, given the query group $Q = \{q_1, \ldots, q_n\}$.
$$\Pr[t|Q] = \sum_{q\in Q} \Pr[t|q]\Pr[q] = \frac{1}{n}\sum_{q\in Q} \Pr[t|q]$$

B. The Iterative DGA of TIO-1 in [1]

Query group Q enhances the adversary's prior for topic t by the amount $\mathcal{B}(t, Q) = \Pr[t|Q] - \Pr[t]$. This is the security metric used by *TIO-1*'s DGA, which works as follows [1].

1) Obtain probabilities $\Pr[t], \Pr[w|t], \Pr[w]$, and $\Pr[t|d]$ using LDA, and choose thresholds $1 > \epsilon_1 > \epsilon_2 > 0$.
2) Given a real user query q_1 with $|q_1|$ keywords, find a topic set $\mathbf{U} \subseteq \mathbf{T}$ such that $\mathcal{B}(t, q_1) > \epsilon_1$ if and only if $t \in \mathbf{U}$. Build the following sets: $Q = \{q_1\}$ as the initial query group, $\mathbf{X}$ as the topics relevant to dummy queries, and $\mathbf{Y}$ as the topics that are useless.
3) Repeat the following until $\mathcal{B}(t, Q) \le \epsilon_2$, $\forall t \in \mathbf{U}$.
   a) Randomly pick a topic $t_a \in \mathbf{T} \setminus \mathbf{U} \setminus \mathbf{X} \setminus \mathbf{Y}$, and obtain $\Pr[w|t_a]$ for all $w \in \mathbf{K}$. Randomly select $|q_1|$ keywords from t_a to form a tentative dummy query q'. Words with higher $\Pr[w|t_a]$ are selected with higher probability.
   b) Find $\mathcal{B}(t, Q \cup \{q'\}), \forall t \in \mathbf{U}$. Discard t_a as useless if $\max_{t\in\mathbf{U}} \mathcal{B}(t, Q \cup \{q'\}) \ge \max_{t\in\mathbf{U}} \mathcal{B}(t, Q)$, and set $\mathbf{Y} = \mathbf{Y} \cup \{t_a\}$. Otherwise set $Q = Q \cup \{q'\}$, and $\mathbf{X} = \mathbf{X} \cup \{t_a\}$.
4) Shuffle queries in Q, and submit Q to the search engine.

A query is *semantically coherent* [1] if its keywords describe common or related topics. In *TIO-1*, keywords in each dummy query are from the same topic, so dummy queries are semantically coherent. The adversary cannot distinguish the real query from dummy queries in each query group generated by *TIO-1* using attacks exploring semantics of queries.

C. The Highest Random Weight Algorithm

The Highest Random Weight algorithm (HRW) was introduced in [21] to achieve distributed consensus on object-server mappings. It uses a hash function $h : \{0,1\}^* \to \mathbb{Z}_p$ as follows. Given an object name o_i and N servers $\mathbf{S} = \{\alpha_1, \ldots, \alpha_N\}$, HRW first computes $h(o_i\|\alpha_1), \ldots, h(o_i\|\alpha_N)$ where $\|$ means concatenation, and selects $n \le N$ servers $\alpha_{i_1}, \ldots, \alpha_{i_n} \subseteq \mathbf{S}$ having the highest hash values to serve the object.

HRW ensures that each server set is selected to serve a given object with the same probability. Each object is always mapped to the same server set, and this mapping can be computed locally by each client. Most importantly, HRW minimizes disruption in the event of server failure. If a single server is to be chosen, we set $n = 1$.

IV. System and Security Model

Our system consists of two entities: users and a text search engine (see Fig. 1). The search engine provides users with a text search service over a database $\mathbf{D} = \{d_1, d_2, \ldots, d_{|\mathbf{D}|}\}$ containing $|\mathbf{D}|$ documents. These documents contain $|\mathbf{K}|$ keywords $\mathbf{K} = \{w_1, w_2, \ldots, w_{|\mathbf{K}|}\}$. The database covers $|\mathbf{T}|$ topics $\mathbf{T} = \{t_1, t_2, \ldots, t_{|\mathbf{T}|}\}$ in all.

Users submit queries to the search engine, each comprising keywords from $\mathbf{K}$. Privacy is protected by a trusted TIO tool, which generates appropriate dummy queries to hide each real query. The real and dummy queries form a query group Q, which is submitted as a single unit to the search engine. As in [1], we assume that the TIO provider can access $\mathbf{D}$, as well as useful probabilities like $\Pr[w|t]$ and $\Pr[t|d]$ relating to $\mathbf{D}$. The search engine finds and returns documents most relevant to each query in Q. Dummy query results are now filtered out.

A. *Security Model and Goals*

We adopt a "honest but curious" model for our adversary, the search engine. Its goal is to distinguish real queries from dummy queries. It is scrupulous in following the public data retrieval protocol, and always returns the right query results.

The adversary is given no information beyond query groups and the target TIO scheme. Dummy queries are semantically coherent, so the adversary cannot identify dummy queries by exploring query semantics. We will show how even such a weak adversary can filter a large portion of dummy queries using attacks on TIOs. When mounting aggregation attacks, the adversary has access to all query groups sent by a user. We focus on the following attacks.

1) Closure-Based Attacks:

Definition 1: Let $\mathcal{Q}_{\mathcal{G}}(q_i)$ denote the set of all query groups that a DGA $\mathcal{G}$ might generate, given a real query q_i. A query group Q is *closed* under $\mathcal{G}$ iff $Q \in \mathcal{Q}_{\mathcal{G}}(q_i)$ for all $q_i \in Q$.

A query $q_i \in Q$ can be flagged as dummy if $Q \notin \mathcal{Q}_{\mathcal{G}}(q_i)$. This attack fails on closed query groups, but no TIO has been shown to consistently generate closed query groups.

2) Aggregation Attacks: A user interested in a topic t will likely send many real queries relating to t. Aggregation attacks [20] work as follows. Topic t will occur frequently in the user's queries, but dummy queries will reference random topics. If a user interested in birds sends query groups {*pigeon, airport, car*}, {*eagle, computer, pizza*}, {*owl, dictionary, tiger*}, the adversary can infer *bird* as the real topic, since it is common to the user's query groups. We show how to deal with aggregation attacks using HRW [21].

3) Attacks Exploring Semantics: The adversary uses a semantic classification algorithm (SCA) to analyze queries. A dummy query can be filtered out if its keywords are not semantically coherent. Such attacks are well-addressed in [1] by selecting keywords from the same topic for each dummy query. In our scheme, we also adopt LDA for topic modeling, and ensure that dummy queries are all semantically coherent.

Different SCAs may assign different topics to the same query, but as noted in [20], current schemes assume that the adversary only has the same SCA as the target TIO. In practice, we do not know which SCA the adversary may use, so we must consider all possible SCAs, which is unrealistic. Addressing such attacks is still an open problem [20], and we defer this to future work. Whether such attacks are practical is also an open problem. No work exists showing such attacks truly help the adversary filter out dummy queries.

V. Attacks on Iterative DGAs

We now present three attacks on iterative DGAs. $\mathfrak{Clo}$, the first attack is an exhaustive search that elaborates the strategy outlined in Section I-B1 to break any iterative DGAs that do not generate closed query groups. We then present $\mathfrak{Dom}$ and $\mathfrak{Sec}$, two very practical attacks based on distinct heuristics. Our experiments in Section VII show that these attacks are devastating to the security of TIO schemes.

Algorithm 1: Iterative DGAs
```
input : $q_1, \mathbf{P}$    // $\mathbf{P}.\gamma$ is the security goal
output: $Q$
1 $Q = \{q_1\}$
2 while $\sigma(Q)$ does not meet the security goal $\mathbf{P}.\gamma$ do
3     Generate a new dummy query $q'$
4     Calcuate the new security metric $\sigma(Q \cup \{q'\})$
5     if $\sigma(Q \cup \{q'\}) > \sigma(Q)$ then
6         $Q \leftarrow Q \cup \{q'\}$
7     end
8 end
```

A. *The Structure of Iterative DGAs*

Iterative DGAs are attractive since their security goals can be reached efficiently. An iterative DGA $\mathcal{G}$ takes as input a real query q_1 and a parameter set $\mathbf{P}$. It outputs a query group $Q = \mathcal{G}(q_1, \mathbf{P})$, including q_1 and $n-1$ dummy queries $q_2, \ldots, q_n$ used to hide q_1. $\mathbf{P}$ includes the probabilities obtained by the semantic classification algorithm, and the security goal of the DGA.

Algorithm 1 illustrates the structure of iterative DGAs. Q starts out with only the real query q_1. Let $\sigma(Q)$ denote the security metric attained by query group Q. In each iteration, $\mathcal{G}$ generates a new dummy query, adding it to Q if and only if doing so improves the security metric $\sigma(Q)$. This process continues until the security goal is met.

Iterative DGAs can meet very high security goals, but current iterative DGAs like the DGA of *TIO-1* are not guaranteed to generate closed query groups.

B. $\mathfrak{Clo}$*: An Exhaustive Search Using the Closure Heuristic*

We start with $\mathfrak{Clo}$, an exhaustive-search method guaranteed to break iterative DGAs whose query groups are not closed. We present $\mathfrak{Clo}$ mainly as prelude and foundation for more efficient attacks, but our experiments show it is very effective, even when given limited run time. Most important, HDGA, the DGA we propose in Section VI, is immune to this attack.

Let $\langle Q \rangle = \langle q_1, q_2, \ldots, q_n \rangle$ denote the order in which queries are added to Q by Algorithm 1. Let $Q^i = \{q_1, q_2, \ldots, q_i\}, 1 \le i \le n$ be the sets of queries forming prefixes of $\langle Q \rangle$. Since a dummy query q_{i+1} can be added to Q^i to get Q^{i+1} if and only if $\sigma(Q^{i+1}) > \sigma(Q^i)$, Q's evolution history influences whether a given dummy query is acceptable at any step.

$\langle Q \rangle$ is unknown to the adversary. Let $\langle O \rangle = \langle q_{(1)}, \ldots q_{(n)} \rangle$ be any ordering of queries in Q. Again, let O^i be the sets of queries forming prefixes of $\langle O \rangle$. Now, define ordering $\langle O \rangle$ *feasible* if successive O^i cause a monotonic improvement in the security metric, that is, if $\sigma(O^{i+1}) > \sigma(O^i), 1 \le i \le n-1$.

Clearly, a query q is dummy if no order taking q as the first query is feasible. The attack $\mathfrak{Clo}$ is as follows:

1) Pick a $q_i \in Q$ to be tested.
2) Generate all orders starting with q_i, and check for feasibility. If there is no feasible order, q_i is dummy.

In practice, we would not generate each permutation starting with q_i in its entirety. Instead, we would generate permutation prefixes, extending only prefixes found feasible.

C. $\mathfrak{Dom}$: A Heuristic Attack Based on Topical Dominance

$\mathfrak{Dom}$ is an attack based on the following insight. *TIO-1* uses metric $\mathcal{B}(t,Q) = \Pr[t|Q] - \Pr[t]$ to measure how much new information Q yields about the relevance of any topic t.

However, every query q has a *dominant topic* δ_q such that $\Pr[\delta_q|q] - \Pr[\delta_q] \geq \Pr[t|q] - \Pr[t], t \in \mathbf{T} \setminus \{\delta_q\}$. If q is a real query, *TIO-1* tries to hide topics in $\mathbf{U}$, especially δ_q. In this case, *TIO-1* is hardly likely to add dummy queries whose dominant topic is also δ_q, an action that would increase $\Pr[\delta_q|Q]$. However, *TIO-1* does not do the same for dummy queries. There is in fact a strong likelihood of finding pairs (q, q') of dummy queries, both with the same dominant topic.

In lines 2–4, $\mathfrak{Dom}$ (see Algorithm 2) first finds the dominant topic δ_q for each $q \in Q$. In lines 5–11, it checks whether there is another query q' sharing the same dominant topic with q. If such a q' is found, it flags both q, q' as dummies. $\mathfrak{Dom}$ is efficient, with time complexity just $O(n|\mathbf{T}|)$.

Algorithm 2: $\mathfrak{Dom}$

```
input : $Q$, $\mathbf{P}$
output: $\mathcal{C}$        // Set of queries found to be dummy
1  $\mathcal{C} = \emptyset$
2  foreach $q \in Q$ do
3      Obtain $\delta_q$, the dominant topic of $q$
4  end
5  foreach $q \in Q$ do
6      foreach $q' \in Q \setminus \{q\}$ do
7          if $\delta_q = \delta_{q'}$ then
8              $\mathcal{C} \leftarrow \mathcal{C} \cup \{q, q'\}$
9          end
10     end
11 end
12 return $\mathcal{C}$
```

1) Analysis of $\mathfrak{Dom}$: When might two queries have the same dominant topic? For simplicity, consider queries containing only one keyword. *TIO-1* creates a dummy query w by first selecting a topic t and then selecting keyword w. We refer to t as w's *original topic*. The probability that w is selected is proportional to the normalized value $\Pr[w]_t = \frac{\Pr[w|t]}{\sum_{w \in \mathbf{K}} \Pr[w|t]}$. From Bayes Rule, $\Pr[t|w] = \frac{\Pr[w|t]\Pr[t]}{\Pr[w]}$, so we have for topics t and t'

$$\frac{\Pr[t'|w]}{\Pr[t|w]} = \frac{\Pr[w|t']\Pr[t']}{\Pr[w|t]\Pr[t]}$$

The ratio $\frac{\Pr[t'|w]}{\Pr[t|w]}$ is the relative probability of seeing t' or t as the more relevant topic, given query w. This relative probability depends not merely on whether $\Pr[w]_{t'} > \Pr[w]_t$, but really on whether $\Pr[w|t']\Pr[t'] > \Pr[w|t]\Pr[t]$, that is, also on the relative values of $\Pr[t]$ and $\Pr[t']$. Paradoxically, although *TIO-1* chose w to match the original topic t on the basis of a maximized normalized probability $\Pr[w]_t$, another topic t' may be perceived as w's dominant topic.

Definition 1: If $q_i, q_j \in Q$ have the same dominant topic, a *dominance collision* occurs.

We say a query q is *errant* if its original topic is different from its dominant topic. Topics being independent in the LDA model, we can assume that an errant query q's dominant topic will be a random selection from the $|\mathbf{T}| - 1$ topics remaining after we exclude q's original topic. We can also assume that no dummy query has the same dominant topic as the real query, an assumption supported by our experiments in Sec. VII-C.

We analyze dominance collisions among dummy queries. Let query group $Q = \{q_1, q_2, \ldots, q_n\}$ contain n queries. Let q_1 be the real query. Denoting q_i's dominant topic by δ_{q_i}, we can form $\binom{n-1}{2}$ dominant topic pairs $(\delta_{q_i}, \delta_{q_j})$ from $Q \setminus \{q_1\}$. Let these pairs be denoted by $X_1, X_2, \ldots, X_{\binom{n-1}{2}}$. Now, let $Y_a \in \{0,1\}$ be an indicator random variable such that $Y_a = 1$ iff the dominant topics forming pair X_a are identical. Since the pairs X_a are indistinguishable in this formulation, $\Pr[Y_a = 1]$ is identical for all a, so we will abbreviate it to $\Pr[Y]$. Let $\hat{p}$ be the probability that a dummy query in Q is errant. Now,

$$\Pr[Y] = 2\hat{p}(1-\hat{p})\frac{1}{|\mathbf{T}|-2} + \hat{p}^2\frac{|\mathbf{T}|-3}{|\mathbf{T}|-2} \cdot \frac{1}{|\mathbf{T}|-2} \quad (1)$$

To obtain this equation, consider the pair (q_i, q_j). The probability that exactly one of q_i, q_j is errant is $2\hat{p}(1-\hat{p})$, and the probability that q_i, q_j are both errant is $\hat{p}^2$. The probability that $\delta_{q_i} = \delta_{q_j}$ is $\frac{1}{|\mathbf{T}|-2}$ under the first condition, and $\frac{|\mathbf{T}|-3}{|\mathbf{T}|-2} \cdot \frac{1}{|\mathbf{T}|-2}$ under the second condition.

We can model the number of dominance collisions in Q as the number of successes in a series of $\binom{n-1}{2}$ Bernoulli trials over the pairs X_a, each with success probability $\Pr[Y]$. Since $\Pr[Y]$ will be small enough and $\binom{n-1}{2}$ large enough, we can use the Poisson approximation to the Binomial distribution to estimate χ_Q, the number of dominance collisions in Q as

$$\Pr[\chi_Q = z] = \frac{\lambda^z e^{-\lambda}}{z!}, \quad \lambda = \binom{n-1}{2}\Pr[Y] \quad (2)$$

The probability that $\mathfrak{Dom}$ will not find any collisions is thus $e^{-\lambda}$, and the expected number of collisions is λ.

We can estimate the number of queries flagged as dummy by $\mathfrak{Dom}$ as follows. Let $Z_q \in \{0,1\}$ be an indicator random variable such that $Z_q = 1$ iff a dummy query q shares the same dominant topic with at least one another dummy query in Q. Again, symmetry consideration suggest that $\Pr[Z_q = 1]$ is independent of q, so we abbreviate it to $\Pr[Z]$. Now,

$$\Pr[Z] = 1 - \Pr[Z = 0] = 1 - (1 - \Pr[Y])^{n-2} \quad (3)$$

Since $\mathfrak{Dom}$ filters out queries having dominance collisions, the expected Dummy query Filter Rate (DFR) of $\mathfrak{Dom}$ is

$$\mathit{DFR} = \Pr[Z]. \quad (4)$$

The expected number of dummy queries filtered by $\mathfrak{Dom}$ due to collisions is hence $(n-1)\Pr[Z]$.

$\Pr[Y]$ and $\Pr[Z]$ depend on the value of $\hat{p}$, which varies with many factors, including $\mathbf{T}$, $\mathbf{D}$, and $\mathbf{K}$. In Section VII-C, we estimate $\hat{p}$ experimentally and show that our theoretical predictions for DFR agree well with experiments.

D. $\mathfrak{Sec}$: An Attack Exploiting the Security Metric

We now discuss some other vulnerabilities of *TIO-1*, which future DGA designs would do well to avoid. *TIO-1* uses the metric $\mathcal{B}(t,Q) = \Pr[t|Q] - \Pr[t]$ to evaluate the security of query groups, and tunes this metric using thresholds ϵ_1, ϵ_2. *TIO-1* claims that security is higher when ϵ_1, ϵ_2 are smaller. We show this claim to be invalid. Indeed, our attack $\mathfrak{Sec}$ performs better for smaller ϵ_1, ϵ_2 in our experiments.

$\mathfrak{Sec}$ is a heuristic based on the observation that *TIO-1* hides only topics relevant to the real query. For such topics, it tries to reduce $\mathcal{B}(t,Q) = \Pr[t|Q] - \Pr[t]$. Since it does not bother to hide topics relevant to dummy queries, $\mathcal{B}(t,Q)$ may still be very large for topics relevant to dummy queries. If the adversary sorts topics by $\mathcal{B}(t,Q)$, it is more likely that topics with larger values are relevant to dummy queries.

$\mathfrak{Sec}$ takes as input a query group Q, a parameter set $\mathbf{P}$, and an integer $l \in [1, |\mathbf{T}|-1]$. It outputs $\mathcal{C}$ containing dummy queries. The attack is shown in Algorithm 3.

$\mathfrak{Sec}$ collects into set L the l topics with the highest value of $\mathcal{B}(t,Q) = \Pr[t|Q]-\Pr[t]$. This set is deemed to contain topics relevant to dummy queries. The parameter l is used to control the number of topics selected, and may be set to a value such as $\lfloor n/2 \rfloor$ or $\lfloor n/4 \rfloor$. For each topic $t \in L$, $\mathfrak{Sec}$ finds the query $q \in Q$ having the highest $\Pr[t|q]$, and flags q as a dummy query. q is the query most relevant to t. Sec. VII-D shows the performance of $\mathfrak{Sec}$ for different l. $\mathfrak{Sec}$ shows that DGAs must prove that their security metrics are truly security-related.

E. Keeping ϵ_1, ϵ_2 secret

Another vulnerability of *TIO-1* is that it assumes that the two thresholds ϵ_1 and ϵ_2 are secrets, and unknown to the adversary. We argue that parameters that do not control randomness in dummy query generation should not be secrets in DGA design. ϵ_1 and ϵ_2 do not play the same roles as secrets such as keys or passwords. These are usually random values, while ϵ_1 and ϵ_2 are not. The adversary may infer ϵ_1 and ϵ_2 from query groups generated by *TIO-1*.

Let us review the roles of ϵ_1, ϵ_2 in *TIO-1*. ϵ_1 determines topical relevance. Topic t is relevant to the real query iff $\Pr[t|q] - \Pr[t] > \epsilon_1$. Similarly, ϵ_2 determines *TIO-1*'s security goal. *TIO-1* adds dummy queries to ensure that all relevant topics t satisfy $\Pr[t|Q] - \Pr[t] \le \epsilon_2$. If ϵ_1 or ϵ_2 is too big, *TIO-1* may not hide topics relevant to the real query. If they are too low, then no number of dummy queries may suffice.

It is dangerous to assume that ϵ_1 and ϵ_2 are secrets. First, the size of query groups is closely related to these two thresholds. The smaller ϵ_1 and ϵ_2, the larger the size of query groups. Our experiments in Section VII-F show correlations between the size of query groups and ϵ_1, ϵ_2. The adversary may be able to infer ϵ_1 and ϵ_2 from the size of query groups.

Second, ϵ_1 and ϵ_2 are closely related to the performance and security of *TIO-1*, and must be carefully tuned. This is beyond the ability of regular users presenting keyword queries, who will simply use the default thresholds in the TIO tool or plugin. All such users are at high risk of privacy loss.

In Sec. VII-F we expose the correlations between ϵ_1, ϵ_2 and the size of query groups. In practice, adversaries may be able to infer the values of ϵ_1 and ϵ_2 using more sophisticated attacks.

Algorithm 3: Attack $\mathfrak{Sec}$

```
    input : $Q$, $\mathbf{P}$, $l$
    output: $\mathcal{C}$      // Set of queries found to be dummy
1   $\mathcal{C} \leftarrow \emptyset$
2   Let $\mathcal{B}(t,Q) = \Pr[t|Q] - \Pr[t]$
3   $V = \{ \mathcal{B}(t_i,Q), t_i \in \mathbf{T} \}$
4   $L \leftarrow \{ l$ topics $t_1, \ldots, t_l \in \mathbf{T}$ with highest $\mathcal{B}(t,Q) \}$
5   foreach $t \in L$ do
6       find the query $q \in Q$ with the highest $\Pr[t|q]$
        $\mathcal{C} \leftarrow \mathcal{C} \cup \{q\}$
7   end
8   return $\mathcal{C}$
```

F. $\mathfrak{Agg}$: An Aggregation Attack Across Query Groups

As we argued in Section IV-A2, an adversary can infer user intent by aggregating across query groups. Let a user querying for a topic t_a submit query groups $Q_1, Q_2, \cdots, Q_N$. Each Q_i includes a real query whose dominant topic is t_a, and $n-1$ dummies for random topics in $\mathbf{T} \setminus \{t_a\}$, where $n < |\mathbf{T}|$. We assume any two queries in a query group do not have the same dominant topic. To analyze aggregation attacks, form the null hypothesis that dominant topics for queries in Q_i are all randomly chosen. Since $|Q_i| = n$, the probability that t is the dominant topic for one query in any Q_i is $n/|\mathbf{T}|$. Let c_t be the number of Q_i in which t is represented as a dominant topic. Clearly,

$$\Pr[c_t = j] = \binom{N}{j}\left(\frac{n}{|\mathbf{T}|}\right)^j \cdot \left(1-\frac{n}{|\mathbf{T}|}\right)^{(N-j)},$$

with $\mu = \frac{Nn}{|\mathbf{T}|}$ being the expected number of occurrences of t. Now say that t occurs $x > \mu$ times across $Q_1, \ldots, Q_N$. If topics in the query groups were randomly chosen, the probability of t occurring with at least this frequency is

$$\Pr[c_t \ge x] = \sum_{j=x}^{N}\binom{N}{j}\left(\frac{n}{|\mathbf{T}|}\right)^j \cdot \left(1-\frac{n}{|\mathbf{T}|}\right)^{(N-j)}.$$

If we define $\epsilon = \frac{x-\mu}{N}$, it can be shown that [27]

$$\Pr[c_t \ge x] \le e^{-N\left(2\epsilon^2+\frac{4}{9}\epsilon^4+\frac{2}{9}\epsilon^6\right)} \qquad (5)$$

This probability drops rapidly with N and x, so the chances of a topic appearing by chance in a large number of query groups is extremely small. Clearly, a topic whose occurrence is frequent will likely represent user interest.

VI. Our Scheme

We now present our scheme, called HDGA, which overcomes *TIO-1*'s vulnerabilities via systematic improvements.

First, HDGA generates a set of semantically coherent dummy queries all at once, rather than iteratively, as in *TIO-1*. HDGA also ensures that each dummy query matches a different topic, and that its dominant topic is the same as its original topic. Each dummy query is generated completely independently of other dummy queries in the group.

Second, HDGA treats all queries in a query group equally. Unlike *TIO-1*, HDGA does not preferentially hide topics relevant to real queries. From $\mathfrak{Sec}$, we can see that privileging real queries in this manner actually makes them more vulnerable.

Third, HDGA eschews the use of security metrics, other than the number of dummy topics. Even carefully-chosen security metrics can actually decrease security, as our $\mathfrak{Dom}$ and $\mathfrak{Sec}$ attacks against *TIO-1* clearly demonstrate. Unless it can be conclusively proved that a security metric under consideration is secure, it may be better not to use it at all.

Fourth, HDGA addresses aggregation attacks through a novel use of the Highest Random Weight (HRW) algorithm [21] in topic selection.

1) HDGA: HDGA takes as input a real query q_1, an integer $n-1$ indicating the number of dummy query topics, a user secret s, and a parameter set $\mathbf{P}$. It outputs a query group Q that covers n topics. HDGA works as follows:

1) Given a real query q_1 with $|q_1|$ keywords, first determine probabilities $\Pr[t|q_1]$ for each topic $t \in \mathbf{T}$, and find the dominant topic δ_{q_1} of q_1.
2) Use HRW to select dummy topics: Calculate the hash value $e_i = h(\delta_{q_1} \| t \| s)$ for each $t \in \mathbf{T} \setminus \{\delta_{q_1}\}$, and select the $n-1$ topics $\mathbf{T}_D = \{t^1_{q_1}, t^2_{q_1}, \ldots, t^{n-1}_{q_1}\}$ with the largest hash values as dummy query topics.
3) For each topic $t \in \mathbf{T}_D$, randomly select $|q_1|$ keywords for t based on $\Pr[w|t]$, to form a dummy query q'. Words with higher $\Pr[w|t]$ are selected with higher probability. If t is not the dominant topic of q', repeatedly generate a new q' until its dominant topic is t. Add q' to Q.
4) Shuffle queries in Q, and Q is ready to be submitted to the search engine.

In line with Kerckhoff's principle, this algorithm is published. However, we keep the user secret s hidden, as we would do with any password.

A. Resilience of HDGA to Attacks

We analyze the security of HDGA. We prove HDGA is immune to the attacks on TIOs in this paper. We begin by considering the exhaustive search attack $\mathfrak{Clo}$.

Theorem 1: HDGA is immune to attack $\mathfrak{Clo}$.

Proof. Let $\boldsymbol{r(q)}$ denote the event that a query q is a real query, and $\boldsymbol{Q|r(q)}$ denote the event that a query group Q can be generated when q is the real query. Proving Theorem 1 is the same as proving $\Pr[\boldsymbol{Q|r(q)}] > 0$ for any query $q \in Q$.

Let q_1 be the real query of Q, and δ_{q_1} be q_1's dominant topic. Let $\boldsymbol{d(t)}$ denote the event that topic t is selected as a dummy topic. $\Pr[\boldsymbol{d(t)}] = \frac{n-1}{|\mathbf{T}|-1}$ for any topic $t \in \mathbf{T} \setminus \{\delta_{q_1}\}$.

Let $\boldsymbol{q_a {\in} Q}$ denote the event that a query q_a is selected as a dummy query, so that $\Pr[\boldsymbol{q_a {\in} Q | d(\delta_{q_a}), r(q)}]$ denotes the probability that q_a is selected as a dummy query when a query $q \in Q \setminus \{q_a\}$ is the real query and q_a's dominant topic δ_{q_a} is selected as a topic for dummy queries. Let $A(\delta_{q_a}, |q_a|)$ denote the set of all possible queries taking δ_{q_a} as their dominant topic and having the same number of keywords as q_a. We have:

$$\Pr\left[\boldsymbol{q_a {\in} Q | d(\delta_{q_a}), r(q)}\right] = \frac{\Pr[q_a|\delta_{q_a}]}{\sum_{q \in A(\delta_{q_a}, |q_a|)} \Pr[q|\delta_{q_a}]} > 0 \tag{6}$$

Let $\mathbf{T}_Q = \mathbf{T}_D \bigcup \{\delta_{q_1}\}$ denote the set of dominant topics of all queries in Q. $|\mathbf{T}_Q| = n$. For any query $q \in Q$, we have

$$\begin{aligned}\Pr[\boldsymbol{Q|r(q)}] &= \Pr\left[\bigcap_{t \in \mathbf{T}_Q \setminus \{\delta_q\}} \boldsymbol{d(t)}\right] \times \\ &\qquad \prod_{q_a \in Q \setminus \{q\}} \Pr[\boldsymbol{q_a {\in} Q | d(\delta_{q_a}), r(q)}] \\ &= \frac{1}{\binom{|\mathbf{T}|-1}{n-1}} \times \prod_{q_a \in Q \setminus \{q\}} \Pr[\boldsymbol{q_a {\in} Q | d(\delta_{q_a}), r(q)}]\end{aligned}$$

From Equation 6, the product on the right is always positive, so that $\Pr[\boldsymbol{Q|r(q)}] > 0$ for any query $q \in Q$. We have thereby shown that query groups generated by HDGA are always closed, and HDGA is immune to $\mathfrak{Clo}$. ■

We now prove that HDGA is also immune to the $\mathfrak{Dom}$ and $\mathfrak{Sec}$ attacks.

Theorem 2: HDGA is immune to the attack $\mathfrak{Dom}$.

Proof. In each query group generated by HDGA, it is guaranteed no two queries have the same dominant topic. ■

Theorem 3: HDGA is immune to attack $\mathfrak{Sec}$.

Proof. HDGA does not preferentially treat real queries over dummy queries, based on some security metric σ. No metric or other means are available to the adversary for sorting queries as in $\mathfrak{Sec}$. ■

B. HDGA Resilience Against Aggregation Attacks

HDGA is immune to attack $\mathfrak{Agg}$, since it uses HRW [21] to select dummy topics. For a real query q_1, HDGA obtains the dominant topic δ_{q_1}, calculates hash values for the $\mathbf{T} \setminus \{\delta_{q_1}\}$, and picks the $n-1$ topics yielding the highest hash values. *The same topics will be picked for every Q_i.*

Using the notation of Section IV-A2, $\Pr[c_t = j] = 1, \forall j$ if t is the real topic δ_{q_1}, or one of the $n-1$ dummy topics chosen by HRW. $\Pr[c_t = j] = 0$ for all other topics. Equation 5 no longer holds. The adversary can determine the dominant topics in the Q_i, but is utterly unable to distinguish between the real and dummy topics.

Since HRW's generation of hash values $h(\delta_{q_1} \| t \| s)$ for topics $t \in \mathbf{T} \setminus \{\delta_{q_1}\}$ incorporates the secret value s, only the user is able to distinguish the real and dummy topics. A well-chosen hash function preserves randomness in topic selection, as well as consistently choosing the same dummy topics for the same real topic. The adversary is able to guess the dummy topics only with probability $\frac{1}{\binom{|\mathbf{T}|-1}{n-1}}$.

VII. Experiments

We now present experiments showing the performance of $\mathfrak{Clo}$, $\mathfrak{Dom}$, and $\mathfrak{Sec}$ against *TIO-1*. We also show the correlation between the two thresholds ϵ_1, ϵ_2 and the size of query groups generated by *TIO-1*. At last, we show the performance of HDGA. Our code is implemented using Matlab, and all experiments are performed on a Linux server with eight 2GHz cpus and 8GB memory.

A. Datasets Used and Parameter Setting

We used real-world text datasets in our experiments. The two datasets we used are KOS blog entries (KOS) and Enron emails (ERN) retrieved from the UCI machine learning repository [28]. Table I shows the number of documents and the number of keywords of each dataset.

We used the lda-c library [22] for topic modeling on these two datasets, setting the number of topics to 50 and 100. We used two (ϵ_1, ϵ_2) settings for *TIO-1* in the experiments. The first setting is $(\epsilon_1 = 0.1, \epsilon_2 = 0.05)$ and the second setting is $(\epsilon_1 = 0.05, \epsilon_2 = 0.01)$. Setting two is claimed to have better security than setting one in [1].

TABLE I: Datasets

Dataset	Number of documents	Number of keywords
KOS	3430	6906
ERN	39861	28102

B. Effectiveness of the Brute-Force Attack $\mathfrak{Clo}$ on TIO-1

$\mathfrak{Clo}$ is surprisingly effective. Our performance metrics are the Dummy Query Filter Rate (DFR), i.e., the fraction of dummy queries flagged by the attack, and the False Positive Rate (FPR), i.e., the fraction of real queries misclassified as dummies.

Running $\mathfrak{Clo}$ to completion can be expensive, since it entails exhaustive search. We therefore give $\mathfrak{Clo}$ just 5 seconds to determine whether each query q is dummy. After 5 seconds, we flag q as real if q is not flagged as dummy. Hence, the DFRs in Figure 2 are lower bounds on what $\mathfrak{Clo}$ can achieve, given more resources.

Let m denote the number keywords per query. Figures 2a and 2b show the DFRs for $\mathfrak{Clo}$ on the KOS and ERN datasets when queries contain $m = \{1, 3, 5, 10\}$ keywords. Surprisingly, about 40% dummy queries are filtered out when $m = 1$. DFRs drop as m increases, due to the 5-second limit. False positive rates (FPR) for $\mathfrak{Clo}$ are always 0 because there always exists a feasible order for the real query.

Queries containing as many as 20 keywords are used in [1]. However, we do not generate such long queries since 95.21% queries sent by American users contain only 1-5 words [29].

C. Effectiveness of Attack $\mathfrak{Dom}$ Against TIO-1

We first perform experiments to obtain $\mathbb{E}[\hat{p}]$, the expectation that a dummy query is errant. The experiments are performed as follows:

1) Set the number of keywords $m = \{1, 3, 5\}$, and $|\mathbf{T}| = \{50, 100\}$.
2) For each combination of m and $|\mathbf{T}|$, use *TIO-1* to generate 1000 queries. Find N_e, the number of errant queries, and estimate $\mathbb{E}[\hat{p}] \approx N_e \times 10^{-3}$.

Figures 3a and 3b show our results on the KOS and ERN datasets respectively. $\mathbb{E}[\hat{p}]$ decreases as m increases, and increases as $|\mathbf{T}|$ increases. $\mathbb{E}[\hat{p}]$ is about 60% when $m = 1$, which is promising, since 42.10% user queries contain just one keyword [29]. $\mathbb{E}[\hat{p}]$ is about 10% to 20% even when $m = 5$.

We perform further experiments to show the effectiveness of $\mathfrak{Dom}$ on the KOS and ERN datasets. We generate query groups as follows.

1) Set $m=\{1, 3, 5\}$, $|\mathbf{T}|=\{50, 100\}$, $\{(\epsilon_1=0.1, \epsilon_2=0.05), (\epsilon_1=0.05, \epsilon_2=0.01)\}$.
2) For each combination of m and $|\mathbf{T}|$, generate 1000 real queries, uniformly covering all $|\mathbf{T}|$ topics. Each real query q contains m keywords randomly selected for the same dominant topic δ_q, based on $\Pr[w|\delta_q]$.
3) For each combination of a real query q and (ϵ_1, ϵ_2), use *TIO-1* to generate a query group.
4) Use $\mathfrak{Dom}$ to filter out dummy queries.

Figures 4 and 5 show how $\mathfrak{Dom}$ performs on KOS and ERN. About 40% dummy queries are filtered out for $m = 1$. DFR decreases as m increases, reflecting the behavior of $\mathbb{E}[\hat{p}]$, which is as expected. DFR is much higher when $(\epsilon_1 = 0.05, \epsilon_2 = 0.01)$, which means lower ϵ_1 and ϵ_2 cannot provide better security. FPR is nearly 0 for all settings, which means that each real query rarely has the same dominant topic as dummy queries.

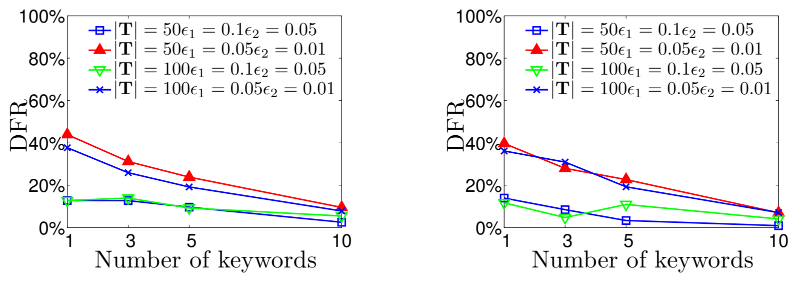
Figures 4 and 5 also show that the experimental DFRs match our theoretical values very well. Given a query group from the KOS dataset or the ERN dataset, we first obtain n, the number of queries per query group, $|\mathbf{T}|$, the number of topics, and m, the number of keywords in each query. We then obtain corresponding $\mathbb{E}[\hat{p}]$ from Figure 3a or 3b. We now calculate the expected DFR of the query group from Equation 4 given n, $|\mathbf{T}|$ and $\mathbb{E}[\hat{p}]$.

The run time of $\mathfrak{Dom}$ is very small. It takes less than 0.1s to find out dummy queries in a query group.

D. Effectiveness of $\mathfrak{Sec}$ Against TIO-1

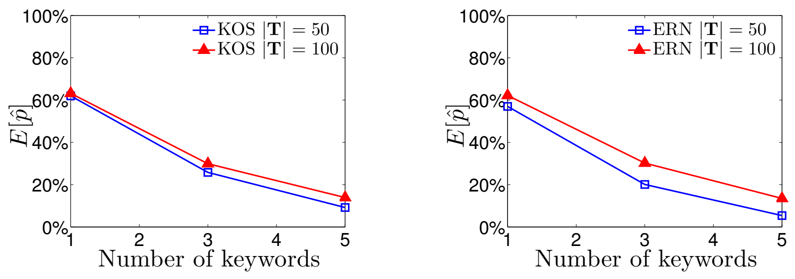
To test the performance of $\mathfrak{Sec}$ against *TIO-1*, we use the following parameters:

1) $m=\{1, 3, 5, 10\}$, $|\mathbf{T}|=\{50, 100\}$, $\{(\epsilon_1=0.1, \epsilon_2=0.05), (\epsilon_1=0.05, \epsilon_2=0.01)\}$, $l=\{\lfloor 0.125n \rfloor, \lfloor 0.25n \rfloor, \lfloor 0.5n \rfloor, \lfloor 0.75n \rfloor\}$.
2) For each combination of m, $|\mathbf{T}|$, (ϵ_1, ϵ_2), l, use *TIO-1* to generate 1000 query groups for 1000 real queries.
3) Use $\mathfrak{Sec}$ to filter out dummy queries.



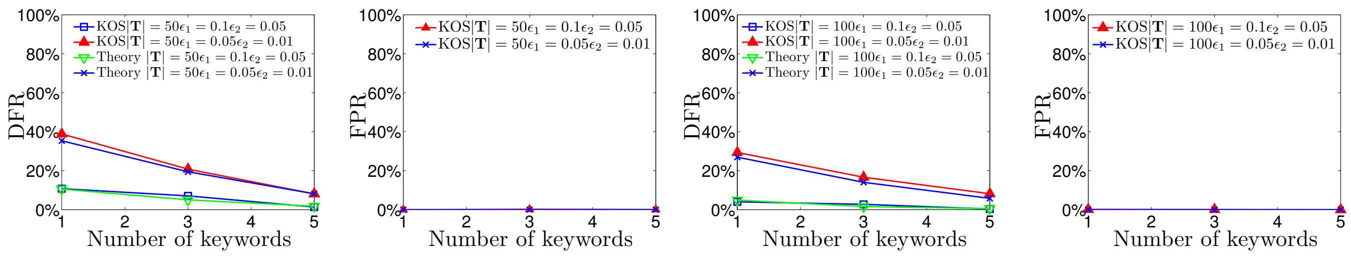
(a) KOS (b) ERN

Fig. 2: Performance of $\mathfrak{Clo}$.



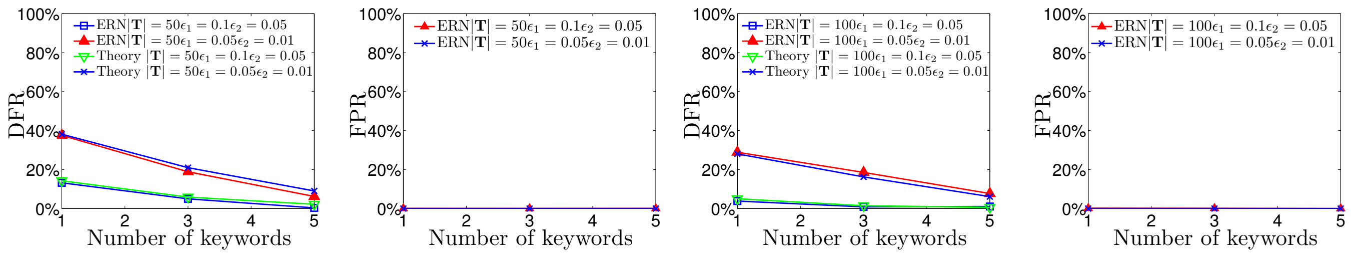
(a) KOS (b) ERN

Fig. 3: $\mathbb{E}[\hat{p}]$.



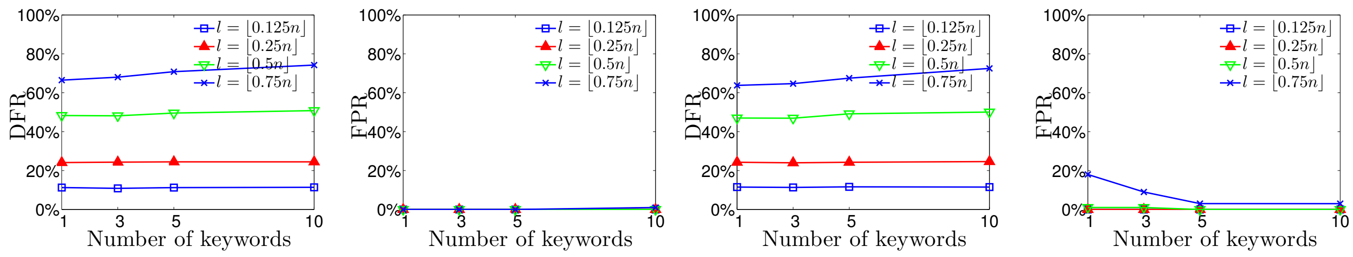
(a) DFR $|\mathbf{T}| = 50$ (b) FPR $|\mathbf{T}| = 50$ (c) DFR $|\mathbf{T}| = 100$ (d) FPR $|\mathbf{T}| = 100$

Fig. 4: Performance of $\mathfrak{Dom}$ on KOS dataset.



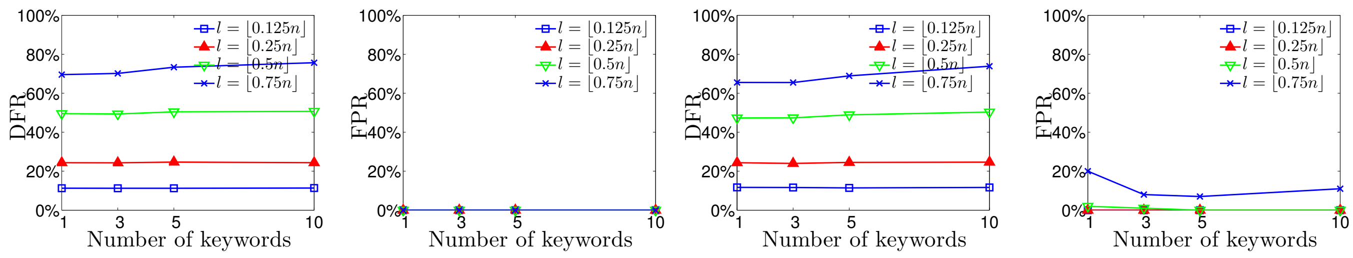
(a) DFR $|\mathbf{T}| = 50$ (b) FPR $|\mathbf{T}| = 50$ (c) DFR $|\mathbf{T}| = 100$ (d) FPR $|\mathbf{T}| = 100$

Fig. 5: Performance of $\mathfrak{Dom}$ on ERN dataset.



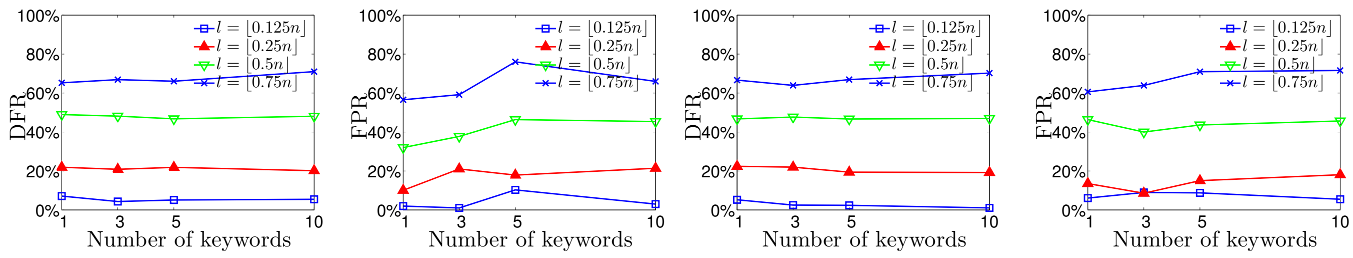
(a) DFR $|\mathbf{T}| = 50$ (b) FPR $|\mathbf{T}| = 50$ (c) DFR $|\mathbf{T}| = 100$ (d) FPR $|\mathbf{T}| = 100$

Fig. 6: Performance of $\mathcal{S}ec$ on KOS dataset when $\epsilon_1 = 0.05$, $\epsilon_2 = 0.01$.



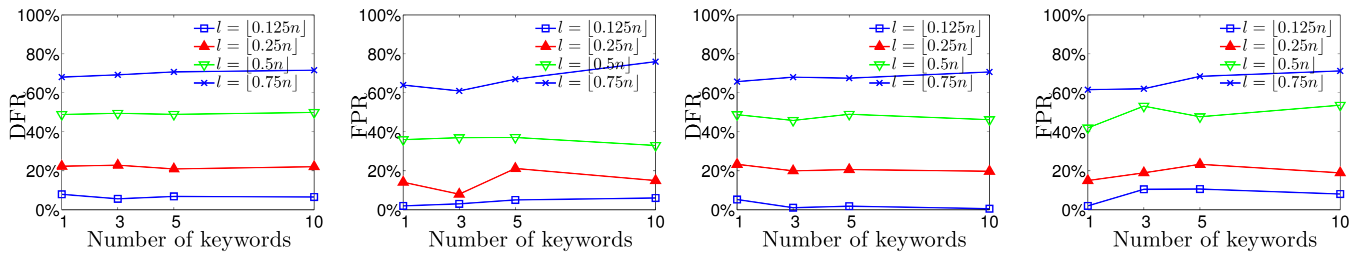
(a) DFR $|\mathbf{T}| = 50$ (b) FPR $|\mathbf{T}| = 50$ (c) DFR $|\mathbf{T}| = 100$ (d) FPR $|\mathbf{T}| = 100$

Fig. 7: Performance of $\mathcal{S}ec$ on ERN dataset when $\epsilon_1 = 0.05$, $\epsilon_2 = 0.01$.



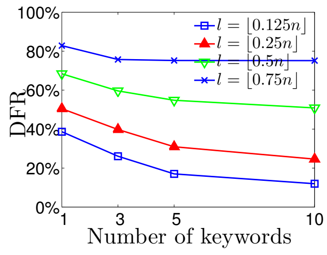
(a) DFR $|\mathbf{T}| = 50$ (b) FPR $|\mathbf{T}| = 50$ (c) DFR $|\mathbf{T}| = 100$ (d) FPR $|\mathbf{T}| = 100$

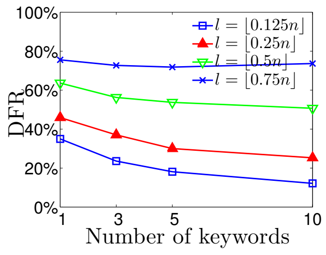
Fig. 8: Performance of $\mathcal{S}ec$ on KOS dataset when $\epsilon_1 = 0.1$, $\epsilon_2 = 0.05$.

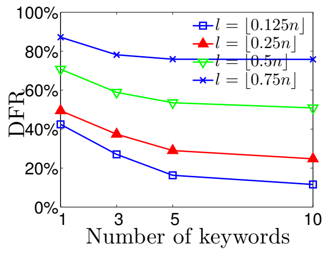


(a) DFR $|\mathbf{T}| = 50$ (b) FPR $|\mathbf{T}| = 50$ (c) DFR $|\mathbf{T}| = 100$ (d) FPR $|\mathbf{T}| = 100$

Fig. 9: Performance of $\mathcal{S}ec$ on ERN dataset when $\epsilon_1 = 0.1$, $\epsilon_2 = 0.05$.


(a) KOS $|\mathbf{T}| = 50$


(b) KOS $|\mathbf{T}| = 100$


(c) ERN $|\mathbf{T}| = 50$

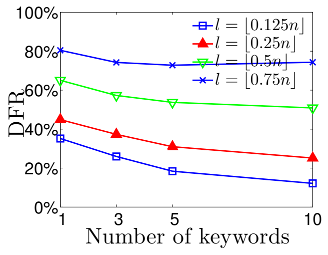
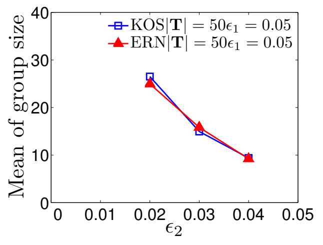
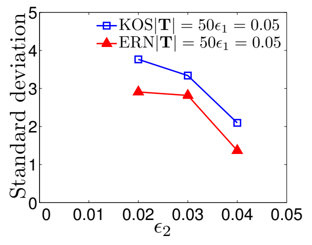
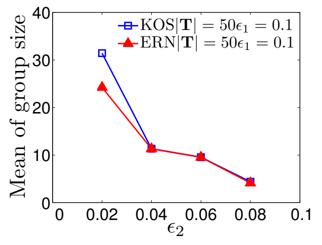

(d) ERN $|\mathbf{T}| = 100$

Fig. 10: Performance of the combined attack on KOS and ERN dataset when $\epsilon_1 = 0.05$, $\epsilon_2 = 0.01$.


(a) Mean of group size


(b) Standard derivation


(a) Mean of group size

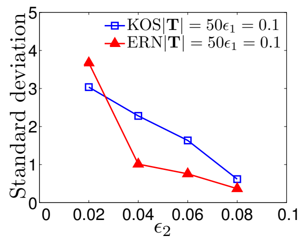
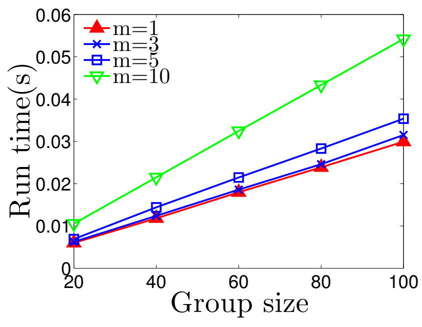

(b) Standard derivation

Fig. 11: Group size when $\epsilon_1 = 0.05$ and $\epsilon_2 = 0.04$ to 0.02.

Fig. 12: Group size when $\epsilon_1 = 0.1$ and $\epsilon_2 = 0.08$ to 0.02.


(a) KOS

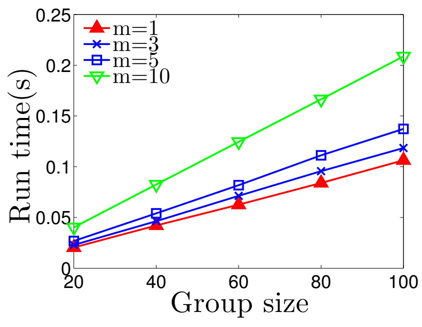

(b) ERN

Fig. 13: Time used in query group generation.

If randomly flagging l of n queries in a group as dummy queries, the upper bound for the DFR is $\frac{l}{n-1}$. and the expected FPR is $\frac{l}{n}$.

Figures 6 and 7 show the DFRs and FPRs for $\mathcal{Sec}$ on KOS and ERN, respectively, when $\epsilon_1 = 0.05, \epsilon_2 = 0.01$ and $|\mathbf{T}| = 50, 100$. Clearly, both DFAs and FPRs grow as l increases. DFRs are close to their upper bound of $\frac{l}{n-1}$, and FPRs are close to 0 when $l < \lfloor 0.75n \rfloor$, which is excellent.

Figures 8 and 9 show the DFRs and FPRs of $\mathcal{Sec}$ on KOS and ERN, when $\epsilon_1 = 0.1, \epsilon_2 = 0.05$ and $|\mathbf{T}| = 50, 100$. Again, DFRs and FPRs grow with l. DFRs are close to their upper bound, and FPRs are close to the predicted values based on random selection, which means $\mathcal{Sec}$ is not very effective in this setting.

Our experiments reveal an important truth, namely that smaller ϵ_1 and ϵ_2 do not make *TIO-1* more secure. *TIO-1* with $\epsilon_1 = 0.05$ and $\epsilon_2 = 0.01$ has worse resilience against $\mathcal{Sec}$. We also find $\mathcal{Sec}$ is not sensitive to the number of keywords in each query, so $\mathcal{Sec}$ remains effective for larger queries.

The run time of $\mathcal{Sec}$ is very small. It takes less than 0.1s to find out dummy queries in a query group.

E. Combining $\mathcal{Dom}$ with $\mathcal{Sec}$

$\mathcal{Dom}$ and $\mathcal{Sec}$ are complementary in the following aspects. $\mathcal{Dom}$ is sensitive to the number of keywords in queries, while $\mathcal{Sec}$ is not. $\mathcal{Sec}$ is more suitable for queries with more keywords. Second, $\mathcal{Sec}$ is more aggressive than $\mathcal{Dom}$. $\mathcal{Sec}$ can filter out more dummy queries than $\mathcal{Dom}$, but may introduce more false positives.

We therefore explore an attack which combines $\mathcal{Dom}$ and $\mathcal{Sec}$. We will use $\mathcal{Dom}$ and $\mathcal{Sec}$ separately to filter out dummy queries, then return the union of the dummy queries sets identified by each attack. Fig. 10 shows the DFRs of the attack on KOS and ERN datasets. We can see that the DFRs of the combined attack is superior to the DFRs of $\mathfrak{Sec}$, especially when m is small. Up to 80% dummy queries can be filtered out. This means dummy queries filtered out by $\mathfrak{Dom}$ is not a subset of those filtered out by $\mathfrak{Sec}$. We can filter out more dummy queries combining $\mathfrak{Dom}$ and $\mathfrak{Sec}$. We do not show FPRs here because $\mathfrak{Dom}$ rarely introduces false positive, so FRPs of the combined attack are the same as FPRs of $\mathfrak{Sec}$.

F. Correlation Between Query Group Size and ϵ_1, ϵ_2

Our experiments show a clear correlation between the two thresholds ϵ_1, ϵ_2 and the size of queries groups in *TIO-1*. This correlation indicates that attacks can be composed to precisely infer the exact values of ϵ_1, ϵ_2.

Figures 11a and 11b show the mean and standard deviation of the group size when $\epsilon_1 = 0.05$ and $\epsilon_2 = 0.04$ to 0.02. Figures 12a and 12b show the mean and standard deviation of the group size when $\epsilon_1 = 0.1$ and $\epsilon_2 = 0.08$ to 0.02. From these figures, we can see that when ϵ_1 is constant, reducing ϵ_2 increases the size of query groups, and when ϵ_2 is constant, reducing ϵ_1 increases the size of query groups.

These results shows that ϵ_1 and ϵ_2 are related to the size of query groups. This leaves the possibility of developing more sophisticated attacks using parameters such as query group size to precisely infer ϵ_1 and ϵ_2.

G. Run time of HDGA

We performed experiments to measure the average time required for query group generation in HDGA. Figures 13a and 13b show the average time used for query group generation for various n, m, for the KOS and ERN datasets respectively, with $|\mathbf{T}| = 100$, $n = \{20, 40, 60, 80, 100\}$, and $m = \{1, 3, 5, 10\}$. The run time is very small, but increases with n, as expected.

VIII. Conclusion

Topical intent obfuscation (TIO) is a new and promising approach to preserving user privacy in text-based search. TIO obscures a user's topical intent by embedding the user's query within a sufficiently large set of dummy queries. We reviewed current approaches to TIO, and identified numerous shortcomings. We presented several attacks that are highly effective against current methods, and showed that the assumptions they are based on, while seemingly sound, are flawed.

We presented several attacks on current schemes, supported by theoretical analysis and experiments. Our experiments show that our attacks are very effective in practice. Even an exhaustive search using our closure heuristic attack is surprisingly effective, even when given very limited resources. Other attacks, which we proposed based on other heuristics, are even more devastating against current schemes.

Finally, we proposed a new dummy query generation method called HDGA, which we proved to always generate closed groups, and immune to the attacks we describe. Our experiments show that HDGA is effective and efficient.

IX. Acknowledgement

This work was supported in part by contract number N00014-07-C-0311 with the Office of Naval Research.

References

[1] H. H. Pang, X. Xiao, and J. Shen, "Obfuscating the topical intention in enterprise text search," in *ICDE*, 2012, pp. 1168–1179.

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

[3] A. S. Josep Domingo-Ferrer and J. Castellà-Roca., "h(k)-private information retrieval from privacyuncooperative queryable databases," *Online Information Review*, vol. 33, no. 4, p. 720C744, 2009.

[4] D. Rebollo-Monedero and J. Forné, "Optimized query forgery for private information retrieval," *IEEE Transactions on Information Theory*, vol. 56, no. 9, pp. 4631–4642, 2010.

[5] J. Castellà-Roca, A. Viejo, and J. Herrera-Joancomartí, "Preserving user's privacy in web search engines," *Computer Communications*, vol. 32, no. 13-14, pp. 1541–1551, 2009.

[6] J. Katz, A. Sahai, and B. Waters, "Predicate encryption supporting disjunctions, polynomial equations, and inner products," in *EUROCRYPT*, 2008, pp. 146–162.

[7] Y. Lu, "Privacy-preserving logarithmic-time search on encrypted data in cloud," in *NDSS*. The Internet Society, 2012.

[8] P. Wang and C. V. Ravishankar, "Secure and efficient range queries on outsourced databases using rp-trees," in *ICDE*, 2013, pp. 314–325.

[9] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill, "Order-preserving symmetric encryption," in *EUROCRYPT*, 2009, pp. 224–241.

[10] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *TCC*, 2007, pp. 535–554.

[11] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *S&P*, 2000, pp. 44–55.

[12] A. Boldyreva, N. Chenette, and A. O'Neill, "Order-preserving encryption revisited: Improved security analysis and alternative solutions," in *CRYPTO*, 2011, pp. 578–595.

[13] M. L. Yiu, G. Ghinita, C. S. Jensen, and P. Kalnis, "Outsourcing search services on private spatial data," in *ICDE*, 2009, pp. 1140–1143.

[14] H. Pang, X. Ding, and X. Xiao, "Embellishing text search queries to protect user privacy," *PVLDB*, vol. 3, no. 1, pp. 598–607, 2010.

[15] X. Shen, B. Tan, and C. Zhai, "Privacy protection in personalized search," *SIGIR Forum*, vol. 41, no. 1, pp. 4–17, 2007.

[16] M. Murugesan and C. Clifton, "Providing privacy through plausibly deniable search," in *SDM*, 2009, pp. 768–779.

[17] D. C. Howe and H. Nissenbaum, "Trackmenot: Resisting surveillance in web search," in *Lessons from the Identity Trail: Anonymity, Privacy, and Identity in a Networked Society*, 2009, pp. 417–436.

[18] H. Pang, J. Shen, and R. Krishnan, "Privacy-preserving similarity-based text retrieval," *ACM Trans. Internet Technol.*, vol. 10, no. 1, pp. 4:1–4:39, Feb. 2010.

[19] Y. Xu, K. Wang, B. Zhang, and Z. Chen, "Privacy-enhancing personalized web search," in *WWW*, 2007, pp. 591–600.

[20] E. Balsa, C. Troncoso, and C. Díaz, "Ob-pws: Obfuscation-based private web search," in *IEEE Symposium on Security and Privacy*, 2012, pp. 491–505.

[21] D. G. Thaler and C. V. Ravishankar, "Using name-based mappings to increase hit rates," *IEEE/ACM Trans. Netw.*, vol. 6, no. 1, pp. 1–14, Feb. 1998.

[22] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.

[23] T. Kuflik, B. Shapira, Y. Elovici, and A. Maschiach, "Privacy preservation improvement by learning optimal profile generation rate," in *UM*, 2003, pp. 168–177.

[24] B. Shapira, Y. Elovici, A. Meshiach, and T. Kuflik, "Prawła privacy model for the web: Research articles," *J. Am. Soc. Inf. Sci. Technol.*, vol. 56, no. 2, pp. 159–172, Jan. 2005.

[25] R. Chow and P. Golle, "Faking contextual data for fun, profit, and privacy," in *Proceedings of the 8th ACM workshop on Privacy in the electronic society*, ser. WPES '09, 2009, pp. 105–108.

[26] 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*, 2010, pp. 19–37.

[27] O. Krafft and N. Schmitz, "A note on hoeffding's inequality," *Journal of the American Statistical Association*, vol. 64, pp. 907–912, 1969.

[28] "Uci machine learning repository." [Online]. Available: http://archive.ics.uci.edu/ml/datasets/Bag+of+Words

[29] "Keyword and search engines statistics." [Online]. Available: http://www.keyworddiscovery.com/keyword-stats.html