OB-WSPES: A Uniform Evaluation System for Obfuscation-based Web Search Privacy

Chengkun Wei, Qinchen Gu, Shouling Ji, Wenzhi Chen, Zonghui Wang, Raheem Beyah

Abstract—Web search queries reveal extensive sensitive information about users' interests and preferences to the search engines and eavesdroppers. Obfuscation-based private web search solutions automatically generate dummy queries and send the obfuscated queries to the search engine to hide users' search intentions. Despite many obfuscation methods and tools have been developed, there is no practical system for evaluating their utility performance and the vulnerability against modern privacy attacks. In this paper, we propose and develop OB-WSPES, a uniform evaluation system for obfuscation-based web search privacy, which allows researchers to conduct fair analysis and evaluation of existing or newly developed web search privacy protection/attack techniques. Leveraging OB-WSPES, we model the obfuscation activities and systematically implement and evaluate five obfuscation schemes and ten modern web search attacks on the public AOL dataset. Our results demonstrate that, counter-intuitively, adding more fake queries to a user's real data does not necessarily yield better privacy. The query utility of obfuscated queries declines with the increasing amount of dummy queries, while the application utility does not. We discuss the experimental results and point out the four important factors that affect the web search privacy and utility. Further, we propose possible directions for future research.

Index Terms—Obfuscation, Web Search, Privacy, Utility, Evaluation System

1 Introduction

Web search has become one of the most effective ways to get information. However, search activities can also give extensive insights into users' interests and intentions. For example, query logs can be used by the search engines or eavesdroppers to generate users' portraits and infer users' sensitive information.

To this end, many approaches have been developed to protect users' privacy in web search. Generally, we can classify them into three categories: *cryptography-based solutions*, *proxy-based solutions* and *obfuscation-based solutions*.

In *cryptography-based* solutions [42], [43], [44], [45], also known as system-centric solutions, Private Information Retrieval (PIR) techniques are used to retrieve information from the search engines without revealing queries and search activities. These solutions provide strong privacy guarantees. However, one drawback is that their computational cost may be unaffordable for a large search engine. Another drawback is that they require the search engines to cooperate with PIR protocols. Thus it may not be realistic to implement these solutions. In *proxy-based* solutions [53], [54], [55], [56], also known as network-centric solutions, users connect to the search engine through an anonymous communication system to hide their identities. This technique can prevent an adversary from constructing users' profile to some extent, but various tracking techniques (e.g., cookies, device/web fingerprinting and browser plugins) can be utilized to link a user with his/hers queries. In *obfuscation-based* solutions [1], [2], [5], [6], [8], [9], [11], [60], [61], also known as user-centric solutions, a user can conceal real queries by generating and issuing dummy queries. Nevertheless, adversaries may employ prior knowledge about obfuscation schemes and logs of users' search activities to filter out fake queries from a set of observed data. Note that these solutions are complementary, and it is possible to combine these solutions for designing a hybrid web search privacy protection mechanism. Specifically, a series of interesting studies integrate proxy-based and obfuscation-based methods and combine unlinkability and indistinguishably of the private web search mechanisms [57], [58], [59].

In this paper, we focus on obfuscation-based solutions. Surprisingly, although there have already been many obfuscation techniques [1], [3], [23], [32] and powerful web search privacy attacks [22], [27], [31], *there are no practical systems to evaluate obfuscation mechanisms' performance and their resistance against modern privacy attacks.* To address this challenge, we propose and implement OB-WSPES, an obfuscation-based web search privacy evaluation system that enables users to obfuscate their data with different obfuscation techniques, measure utilities of obfuscated queries and comprehensively evaluate the obfuscation resilience against web search privacy attacks.

We construct a generic model which characterizes the obfuscation-based web privacy search activities as six-tuples. Based on this, we could consider the obfuscation mechanism as a black box and model any obfuscation scheme's activities. Subsequently, we use various feature extraction methods to extract features from those search activities. We also assume that adversaries might have prior knowledge about the obfuscation mechanisms and have access to the log history of users' queries. Then, we leverage different attack methods against diverse obfuscation mechanisms under various settings. In the evaluation module, we evaluate the utility of obfuscated data from two aspects, query utility and application utility. In addition, we measure the resistance/vulnerability of obfuscation methods against attacks based on the attack reports.

However, through the analysis and evaluation of modern obfuscation schemes against privacy attacks, we find that in the event of a powerful attack, no obfuscation mechanisms could achieve the desired effect and adding more fake queries to users' real data does not necessarily yield better privacy. Furthermore, four major

C. Wei, S. Ji, W. Chen and Z. Wang are with the College of Computer Science and Technology, Zhejiang University, China. E-mail: {weichengkun, sji, chenwz, zhwang}@zju.edu.cn.
Q. Gu and R. Beyah are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, USA. E-mail: qgu7@gatech.edu, raheem.beyah@ece.gatech.edu.

factors affect the utility and privacy of the obfuscated queries (the obfuscation algorithm, the content and the size of thesaurus, the manner of interleaving obfuscated queries, and the ratio of dummy queries to real queries).

Contributions. The main contributions of this paper are as follows.

(a) We develop and implement an obfuscation-based evaluation system for web search privacy (OB-WSPES). To the best of our knowledge, OB-WSPES is the first such practical system publicly available to both academia and industry. More importantly, OB-WSPES provides the first uniform platform that enables researchers to conduct accurate comparative studies of obfuscation and web search privacy attack techniques, and to comprehensively understand the effectiveness of existing or newly developed web search obfuscation/attack techniques.

(b) In OB-WSPES, we model the obfuscation activities and systematically analyze, implement, and evaluate five query obfuscation schemes. We conduct experiments with eight utility metrics on the AOL dataset. The results demonstrate that the query utility declines with the increase of inserted queries, while the application utility (the result of web search) does not.

(c) In OB-WSPES, we summarize and analyze the fundamental properties of existing web search privacy attacks. Then, we systematically implement and evaluate 10 modern web search privacy attacks on the public AOL dataset. Our results show that modern web search privacy attacks are powerful and robust against noise, and classification-based machine learning attacks are more effective than clustering-based and linkage-function-learning attacks.

(d) Leveraging OB-WSPES, we find a more efficient feature vectorization scheme for attacks, which results in high attack performance (with the average accuracy of 0.83) and on average 9 times faster than existing state-of-the-art methods.

(e) We analytically and experimentally evaluate the performance of query obfuscation schemes on defending against modern web search privacy attacks. We find that existing obfuscation techniques are vulnerable to modern web search privacy attacks. *Surprisingly, adding more fake queries to users' real data does not necessarily yield better privacy.* The degree of vulnerability depends on the thesaurus used by obfuscation schemes, the utility and semantic information preserved, the strategy of sending obfuscated queries, etc.

We open source OB-WSPES [1] to enhance the diversity and availability of the platform, and to facilitate the research in this area

2 Background & Motivation

In this section, we study existing obfuscation-based private web search schemes and attack techniques. Furthermore, we introduce our two major motivations for building OB-WSPES.

2.1 OB-PWS Status Quo

Generally, existing Obfuscation-based Private Web Search (OB-PWS) techniques can be classified into three categories. We discuss each category as follows.

TABLE 1: Abbreviation and acronyms.

Terms	OB-PWS	Obfuscation Based Private Web Search
	WSP	Web Serch Privay
	ODP	Open Directory Projcet [64]
Obfuscation mechanisms	TMN	TrackMeNot [1]
	GooPIR	GooPIR [2]
	DisPA	Dissociating Privacy Agent [5]
	PRAW	PRivAcy model for the Web [8]
	PDS	Plausibly Deniable Search [9]
	NQI	Naive Query Injection
	NISPP	Noise Injection for Search Privacy Protection [11]
	OQF-PIR	Optimized Query Forgery for Private Information Retrieval [29]
Attack methods	QWSP	Gervais et al.'s attack [27]
	SimAttack	Petit et al.'s attack [31]
	K-means	k-means clustering
	SVM	Support Vector Machine
	NB	Naive Bayes
	LR	Logistic Regression
	RF	Random Forest
	GBC	Gradient Boosting Classifier
	NC	Nearest Centroid
	DTC	Decision Tree Classifier
	MLP	Multilayer Perception

Query Injection. To obfuscate user queries, a simple method is injecting dummy queries into users' query set. A straightforward scheme is Naive Query Injection (NQI), which randomly samples other users' queries as dummy queries for the target user. This method is scalable and easily deployable. Yet, it has been proven vulnerable to Web Search Privacy (WSP) attacks [27]. Ye et al. proposed Noise Injection for Search Privacy Protection (NISPP) [11] and gave the first theoretical analysis on query injection, which models the search privacy threat as an information inference problem and injects noises into users' queries to minimize privacy breaches. TrackMeNot (TMN) [1] is a popular and publicly available browser plugin, and protects web users against data profiling by simulating HTTP search requests to the search engines. Its fake queries are extracted from a static seed list of query terms or a dynamic query list which is generated by observing the search results of user queries.

Profile-based Obfuscation. Another popular idea to protect query privacy is profile-based obfuscation. Generally, there are two ways. One is to make the target user profile more general. The other is to split the user profile into several parts. In PRivAcy model for the Web (PRAW) [8], Elovici et al. used a profile meter to monitor user profile, and generated dummy queries to make the user profile more general. Like PRAW, Monedero et al. proposed Optimized Query Forgery for Private Information Retrieval (OQF-PIR) [29] that provides a theoretical approach to generate fake queries by measuring the Kullback-Leibler divergence between the user profile and the population distribution, which was designed to achieve perfect user profile obfuscation by assimilating the user profile to the average population profile. Recently, there have been works that seek to split user profile. In Dissociating Privacy Agent (DisPA) [3], [5], Juarez et al. classified user queries into several sets of categories based on taxonomy of Open Directory Projcet (ODP) [64], and gave each set of queries a virtual identity. It follows that user profile was divided into several parts, and it was difficult to perform re-identification by using dissociated logs [4].

***K*-anonymity.** K-anonymity has been widely used in anonymizing data. A release of queries is said to have the k-anonymity property if the query cannot be distinguished from at least k-1 other queries contained in the release [6]. One of the most popular schemes is GooPIR [2], which provides k-

[1] OB-WSPES: https://github.com/ChengkunWei/OB-WSPES

anonymity and disguises the query keywords by adding a certain number of dummy keywords to each real query, then connects real query and fake queries with *or*, and sends them together to Google. Another work is [37], where Xu et al. considered the dynamics of web search users and proposed the notion of online anonymity which provides *k*-online-anonymity. Furthermore, in [9], [10], Murugesan et al. proposed Plausibly Deniable Search (PDS) by using the singular-value-decomposition approach to generate cover queries that have characteristics similar to the real queries, which aims to prevent the real queries "standing out" from the cover queries. Ahmad et al. [60], [61] proposed intent aware query obfuscation mechanisms which submit *k* additional cover queries and corresponding clicks with each real query, which act as decoys to mask users' genuine search intent from a search engine.

Motivation 1: Based on current obfuscation techniques, we have the following remarks.

- ***No unified utility metrics to evaluate the obfuscation mechanisms.*** On one hand, most, if not all, existing OB-PWS works have not been thoroughly evaluated with respect to the utility and the resistance to WSP attacks. They only conducted limited evaluations on their utility preservation (e.g., category entropy and user profile) which are insufficient to understand their value of web search privacy and applications (e.g., personalization of web search). On the other hand and more importantly, to the best of our knowledge, no work has proposed unified evaluation criteria for obfuscation schemes to evaluate OB-PWS utility and the resistance against modern WSP attacks. Although there are many obfuscation mechanisms, we still do not have good understanding of the main factors that affect the privacy and utility, and how to choose the appropriate method to protect users' web search privacy. To address this problem, we present a unified utility metrics system in Section 4.

2.2 WSP Attacks

The goal of the adversary is to identify the real queries from a set of observed data. Conceptually, there are two types of WSP attacks. One is *linkage attack* whose objective is to determine whether the observed queries belong to the target user. The other is *distinguishing attack* which aims to separate users' real queries from the mix of real and fake data. Both attack mechanisms are independent yet related. They are independent because they are based on fundamentally different principles. Yet, they resemble similar binary classification problems where one tries to divide the instances of two classes when mixed together. Because of this, from a technical perspective, we classify WSP attacks into two categories: *Classification/Clustering Attack* and *Linkage Function Learning Attack*. We discuss them as follows.

Classification/Clustering Attack. When an adversary attempts to distinguish users' real queries and infer users' interests, it is straightforward to take advantage of classification or clustering algorithms to filter out valuable information. For example, Al-Rfou et al. [32] made a simple clustering method (Partitioning Around Medoids) against TMN, while achieving high precision in finding users' real queries. Another work by Shapira et al. [22] examined PRAW's ability to withstand clustering attack. It shows that an eavesdropper cannot derive an accurate estimation of the user profile, and claims that PRAW can resist against clustering attack to some extent. In [30], Petit et al. measured TMN using several classification algorithms, e.g., Logistic Regression, Decision Tree, Random Forest and ZeroR. Its results demonstrate that a search engine equipped with only a short-term history of a user's queries can break the privacy guarantees of TMN by only utilizing off-the-shelf machine learning techniques. Furthermore, other powerful classification methods [49], [50], [51] could also be used to identify users' real queries, e.g., Support Vector Machine, Naive Bayesian and Gradient Boosting Classifier.

TABLE 2: Analysis of existing WSP attack techniques. DF= training data free, Scal.=scalable, Prac=practical, Rob.= robust to noise, √=true, ◐=partially true, ◆ =conditionally true, and ✗=false.

	DF	Scal.	Prac.	Rob.
SVM	✗	√	◆	√
NB	✗	√	◆	√
RD	✗	√	◆	√
NC	✗	√	◆	√
DTC	✗	√	◆	√
GBC	✗	√	◆	√
LR	✗	√	◆	√
MLP	✗	√	◆	√
K-means	√	√	◆	√
SimAttack	✗	√	◐	√
QWSP	✗	✗	◐	√

Linkage Function Learning Attack. Generally, the key idea of the linkage function learning attack is to learn a linkage function that predicts the relationship between two query activities (e.g., whether they are issued by the same user or not), irrespective of the type of obfuscation algorithms. Recently, more schemes on linkage function learning attack have been proposed. In [31], Petit et al. proposed SimAttack that learns a linkage function to capture the distance between a query and the target user's profile. In another work [27], Gervais et al. proposed a generic quantitative framework, applying machine learning techniques (e.g., Gradient Boosted Regression Trees (GBRT) [28]) to learn a linkage function that predicts the relation between two queries. Then, a clustering method was used to partition the set of observed queries. Note that the above methods can complement each other.

2.2.1 *WSP Attack Analysis*

In this subsection, we discuss the performance of WSP attacks. We analyze them under different conditions, including training data/training data free, robustness to noise, scalability and practicality to each obfuscation scheme. For convenience, in the rest of this paper, we denote classification-based attack as CLA, clustering-based attack as CLU, Petit et al.'s attack [31] as SimAttack, and Gervais et al.'s attack [27] as QWSP. CLA attacks are very effective, which have been proven by previous work [30]. For CLA, we discuss representative methods including Support Vector Machines (SVM), Naive Bayes (NB), Logistic Regression (LR), Random Forest (RF), Gradient Boosting Classifier (GBC), Nearest Centroid (NC), Decision Tree Classifier (DTC) and Multilayer Perception (MLP). For CLU, although many powerful algorithms have been proposed, the methods used in WSP attacks are *K*-means or its variant, e.g., PAM. Thus, we discuss *K*-means in CLU. As for linkage learning attacks, we discuss SimAttack and QWSP. We show our analysis results in Table 2 and discuss as follows.

As for training data/training data free, all CLA attacks are supervised learning, which needs training data (e.g., a periodic

TABLE 3: Analysis of Obfuscation vs WSP attacks, √, ◆, ✗ = the obfuscation scheme is vulnerable, conditionally vulnerable, and invulnerable to WSP attacks, respectively.

	NQI	TMN	NISPP	PRAW	OQF−PIR	DisPA	GooPIR	PDS
SVM	√	√	◆	◆	◆	✗	◆	◆
NB	√	√	◆	◆	◆	✗	◆	◆
RD	√	√	◆	◆	◆	✗	◆	◆
NC	√	√	◆	◆	◆	✗	◆	◆
DTC	√	√	◆	◆	◆	✗	◆	◆
GBC	√	√	◆	◆	◆	✗	◆	◆
LR	√	√	◆	◆	◆	✗	◆	◆
MLP	√	√	◆	◆	◆	✗	◆	◆
K-means	√	√	◆	◆	◆	◆	◆	◆
SimAttack	√	√	◆	◆	◆	◆	◆	◆
QWSP	√	√	◆	◆	◆	◆	◆	◆

tagged data of users' real queries and dummy queries) to train the classifier. However, CLU attacks may perform well without the seed information, because their key idea is to assign n objects into a cluster and minimize the cluster's width while maximizing the cluster separation. Similar to CLA, most linkage function learning schemes need training data to enhance attack effectiveness, e.g., SimAttack needs user query logs to build the user profile and QWSP needs labeled training data to build the linkage function.

We consider both CLA and CLU are conditionally practical, and CLA methods are more practical and scalable than CLU. This is because most linkage or distinguishing attacks resemble binary classification problems. Thus, CLA is more widely applicable than CLU. Many articles have analyzed CLU/CLAs' advantages and disadvantages. Thus, we omit the details of the comparison.

For SmiAttack [31], compared with naive classification methods, it improves the performance of the attack results on TMN by 23%, while decreasing the execution time by two orders of magnitude. However, it may only be partially practical. This is because SimiAttack makes use of user's profile to learn the linkage function. If the obfuscation mechanism aims to change user's profile or injects too many dummy queries, it will result in a more general or completely different user profile, which may leads to a precipitous decline in SimAttack's effectiveness.

Specifically, we consider QWSP [27] conditionally practical and not scalable. Although the key idea of QWSP is great which inspired us to build an evaluation system, in practice, the effectiveness of its algorithm depends too much on structural feature extraction. It is difficult to extract enough important features from the queries that are protected by obfuscation schemes. Furthermore, the computational complexity limits the scalability, and the computation time grows polynomial with the number of queries. Thus, this scheme may not be suitable for large-scale query processing.

2.2.2 OB-PWS vs WSP Attack Analysis

In this subsection, we analyze the vulnerability of obfuscation mechanisms against WSP attacks. Specifically, understanding the vulnerability/resistance of state-of-the-art query obfuscation schemes against modern WSP attacks is still an open problem. After carefully analyzing existing obfuscation and WSP attack techniques, we summarize the vulnerability of existing obfuscation schemes in Table 3 and discuss as follows.

In query injection based obfuscation schemes, it has been shown in both academia and practice that the NQI and TMN schemes cannot protect the user search query privacy against WSP attacks [27], [30], [31], [32]. Diverse approaches have been successfully applied to attack TMN or NQI, including classification-based [30], clustering-based [32] and linkage function learning methods [27], [31]. Therefore, NQI and TMN are vulnerable to all the existing WSP attacks. For NISPP, it could control the ratio between dummy queries and real queries. The effect of NISPP may change with different ratios of dummy queries to real queries. Thus, NISPP is conditionally vulnerable.

For profile based obfuscation schemes, as we have mentioned before, two different ways can be used to protect users' profile. On one hand, PRAW and OQF-PIR generate dummy queries to make users' profile more general or dissimilar with the original. It follows that obfuscation methods like PRAW and OQF-PIR are conditionally vulnerable to WSP attacks. The extent of private information leakage depends on the column of injected dummy queries and the dissimilarity between fake and original profiles. On the other hand, DisPA divides user queries into parts according to the ODP taxonomy and gives each part of queries a different identity, so as to mislead the search engines or adversaries to make an incomplete profile of the user. In the face of classification based attack algorithms DisPA is invulnerable because of lack of tagged fake data to train the classifier. The linkage function learning attacks and the CLU methods may be helpful in this scenario. It follows that DisPA is conditionally vulnerable to both CLU and linkage function learning attacks. The degree of vulnerability depends on the extent of dissimilarity between fake and original profiles, and how much utility is preserved.

K-anonymity based obfuscation schemes are conditionally vulnerable to modern WSP attacks. The reasons for this are as follows. As k-anonymity was initially designed for traditional relational data, which makes a user semantically indistinguishable from k-1 other users. When researchers extended k-anonymity to obfuscate users' query, they designed schemes to make k queries structurally indistinguishable with respect to structural semantics, e.g., term numbers and query entropy. However, even if users' queries cannot be distinguished with respect to some structural semantics, they can be discovered by additional semantic features, e.g., topic distance, ODP distribution, ego-surfing and combinations of several semantic features. Therefore, as long as utilities are preserved in the obfuscated queries, the k-anonymity based schemes are susceptible to modern WSP attacks. The degree of vulnerability is dependent upon the level of utilities it preserved.

In summary, based on our analysis, obfuscation schemes are still vulnerable to modern WSP attacks. The fundamental reasons include: first, existing obfuscation schemes only destroy partial semantics features (e.g., query topics, query terms frequency) to ensure that obfuscated queries are indistinguishable with respect to some properties. However, additional semantic features (e.g., queries' topic distance, temporal pattern) can still enable effective WSP attack to user queries; and second, as one of the main objectives, all the obfuscation schemes try to preserve as much utility as possible. However, the more utilities it preserves, the easier real queries to be spilted from obfuscated data. There is always a tradeoff between utility and security.

Motivation 2: Based on current attack techniques, we have the following remarks.

- ***No practical system comprehensively evaluate the resilience of obfuscation algorithms in defending against web search privacy attacks.*** On one hand, some abstract models and evaluation frameworks only focus on theoretical analysis. For example, Balsa et al. [52] proposed an abstract model and an associated analysis framework to systematically evaluate the privacy protection offered by OB-PWS algorithms. In this model, we can only study

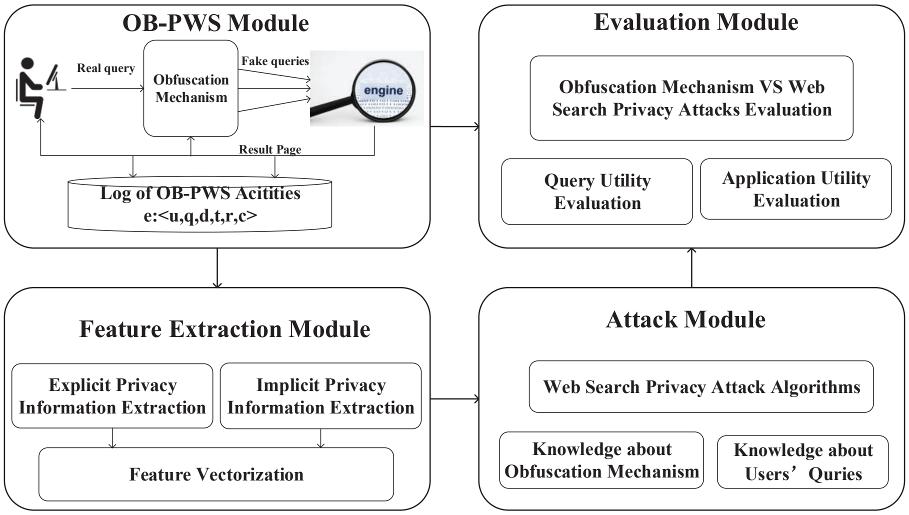

OB-PWS Module
Real query
Obfuscation Mechanism
Fake queries
engine
Result Page
Log of OB-PWS Acitities
e:<u,q,d,t,r,c>
Evaluation Module
Obfuscation Mechanism VS Web Search Privacy Attacks Evaluation
Query Utility Evaluation
Application Utility Evaluation
Feature Extraction Module
Explicit Privacy Information Extraction
Implicit Privacy Information Extraction
Feature Vectorization
Attack Module
Web Search Privacy Attack Algorithms
Knowledge about Obfuscation Mechanism
Knowledge about Users' Quries

Fig. 1: System overview: Obfuscation-based Web Search Privacy Evaluation System.

the operation principles of OB-PWS algorithms in theory. On the other hand, most quantification frameworks only focus on one aspect of obfuscation schemes or attack techniques. For example, Gervais et al. [27] quantified privacy of the users with respect to linkage attacks, and Peddinti et al. [30] focused on TMN and Tor. There is no practical system which allows researchers or users to conduct comprehensively study of existing or newly developed obfuscation/WSP attack techniques. To solve this problem, we present a practical evaluation system OB-WSPES in Section 3.

3 OB-WSPES Framework

In this Section, we introduce OB-WSPES, a practical system to understand the resistance/vulnerability of state-of-the-art obfuscation schemes against modern WPS attacks. Firstly, we present the framework of OB-WSPES, which composes of 4 modules and aggregates 5 obfuscation schemes and 10 attack mechanisms. Then, we introduce a set of unified evaluate metrics for obfuscation mechanisms.

As shown in Figure 1, the system is composed of the following main components: *OB-PWS Module*, *Feature Extraction Module*, *Attack Module*, and *Evaluation Module*. We provide high-level details of these components.

3.1 OB-PWS Activity Modeling

When it comes to model OB-PWS activity, two natural questions arise: 1) What is OB-PWS activity? and 2) How to model OB-PWS activities?

To answer the first question, we describe OB-PWS activities into five steps: 1) users' real queries are sent or intercepted by the obfuscation mechanism; 2) the obfuscation mechanism generates dummy queries based on the algorithm and parameter settings; 3) the real or dummy queries are transmitted to the search engine according to the sending policy (e.g., send in sequential or regular intervals); 4) in response, the search engine retrieves a result page consisting of a ranked list of web pages; and 5) the user clicks and browses relevant documents or further refined queries to fulfill the information need.

Generally, there are two key points of obfuscation mechanisms. One is the dummy queries content, which relies on sources of fake queries, generation algorithms, etc. Another key parameter is the way of interleaving fake and real queries. For example, the obfuscation mechanism may send real queries together with fake queries or send fake queries at regular intervals.

To evaluate obfuscation mechanisms, we model OB-PWS activity as six-tuples e: $\langle u, q, d, t, r, c\rangle$, where u is the user identity (e.g., user name, cookie identifier and IP), q is the real query sent to the search engine, d is the list of dummy queries generated by the obfuscation mechanism, t is the list of time-stamps at which users' real or dummy queries are issued, r is the result page of the search engine, c is the web page clicked by the user. For a target user u, the search activities of user u could be denoted as S_u: $\{e_1, e_2, ...e_n\}$. Based on these six-tuples, we could treat the obfuscation mechanism as a black box and model any obfuscation algorithm or tool's activities.

3.2 Feature Extraction

In the process of feature extraction, we need to solve two problems: 1) What privacy information should be extracted? 2) How to vectoring such information?

Generally, the privacy information extracted from users' queries could be classified into two categories: *explicit privacy information* and *implicit privacy information*.

Explicit Privacy Information refers to Personally Identifiable Information (PII) embedded in the query itself, such as the user's vehicle registration plate number and the living address. In addition, people might search his/her own name for a variety of reasons (e.g., entertainment, finding celebrities with the same name) which is also known as " Ego-surfing" [33].

Implicit Privacy Information refers to sensitive information that cannot be learned directly from the queries. In this case, machine learning algorithms and data mining techniques (e.g., Natural Language Processing) are used to aggregate user information (possibly from various sources), extract user privacy information and build a profile of the user. For example, it is possible to accurately infer demographics traits (e.g., age, gender, political and religious views) from users' search logs [34].

Besides information extraction, feature vectorization is also a very important part of feature extraction. A feature vector in web search is an n-dimensional vector that represents user's query. Different vectorization methods may lead to different results in accuracy and efficiency. Currently, the feature vectorization methods applied by most attack schemes are statistical techniques. For example, word count frequency models like n-grams model and simple bag-of-words models like TF-IDF [31] [30]. However, with the evolution of text-based feature extraction techniques, word embedding approaches (e.g., Word2Vec [35], Glove [36]) may perform much better than word count frequency models in feature vectorization. We will evaluate the effect of different vectorization methods in subsection 5.4.

3.3 Obfuscation Mechanism Attacking

In the attack stage, the objective of the adversary is to partition the set of events in S and determines which one associated with the target user. By exploiting the prior knowledge of the obfuscation mechanism and user's query logs, attack methods could learn a prediction function $P(q_i, u_i)$, where q_i is the queries observed by the attacker, u_i is the target user and $P(q_i, u_i)$ indicates the probability that the query q_i belongs to the target user u_i. In addition, we assume the following about the prior knowledge of the attacker.

Obfuscation Mechanism. An attacker may have knowledge about the obfuscation mechanism in advance, or he/she can infer the behavior of the mechanism by observing its output. Apart from knowing the exact obfuscation mechanism, the adversary is likely to be aware of the parameter settings in the mechanism (e.g., number of fake queries, the interval of sending queries).

Log History of Users' Queries. The adversary possibly has accessed the log history of users' queries. He/she could infer the time continuity and content relevance of user's queries (e.g., query sessions, topic distribution of queries) based on the collection of search activity, and additionally build exact models for users. Thus, further empowering him to predict users' real queries effectively.

For attack methods we have analyzed in Section 2, their unanimous goal is to separate the fake queries from real queries. Thus, we could relieve the differences in specific algorithms and treat the attack function as a black box. We feed the prior knowledge and obfuscated queries to the attack mechanism. The output of the attack mechanism are the prediction functions $P(q_i, u_i)$ that quantify the relationship between queries and users. The attack results could be further used to evaluate the performance of obfuscation mechanisms or WSP attacks.

3.4 Evaluation Module

Obfuscation mechanisms could be evaluated from two perspectives: *the level of utility retained* and *the ability to resist attacks*. To this end, two main aspects of the evaluation could be done.

The Level of Utility Retained. The utility is one of the essential attributes for obfuscation schemes. Lower usability means the user could not get the desired retrieval results. In this case, users will abandon the obfuscation tool, and the obfuscation algorithm will be meaningless. In OB-WSPES, we propose utility metrics mentioned in Section 4 to quantify the level of utility that obfuscated queries retained, and assess the availability and effectiveness of the obfuscation mechanisms.

The Ability to Resist Attack. Although there are many obfuscation schemes and attack methods, there is no practical system to comprehensively evaluate the performance of obfuscation mechanisms in defending against state-of-the-art attack methods. To solve this problem, we use different parameter settings for different obfuscation mechanisms to deal with various attacks. Through the analysis of attack results, we can find out the optimal parameter setting in the presence of a specific attack, and the most effective attack methods on a specific obfuscation scheme. We can also make suggestions for the strengths or weaknesses of the obfuscation/attack mechanisms.

TABLE 4: Metrics for query evaluation.

Metric	Description
NQC	Number of characters in the query
QE	Entropy of the frequency of terms in the query
ODP1D	Distribution of level 1 categories in the ODP taxonomy
ODP1E	Entropy of ODP level 1 category
ODP2D	Distribution of level 2 categories in the ODP taxonomy
ODP2E	Entropy of ODP level 2 category
PWS_J	Jaccard distance of the query result lists
PWS_E	Edit distance of the query result lists
UP	User profile

4 Unified Evaluation Metric

In this section, we propose a set of unified query utility metrics to evaluate the performance of different obfuscation schemes. We comprehensively analyze the utility of existing OB-PWS algorithms and conduct the detailed resistance analysis in Section 4.3.

We focus on semantic features that are used in Natural Language Processing (NLP). White et al. [26] analyzed the extractable features of search activity in detail. Further, Gervais et al. [27] extracted 12 semantic features for query events and analyzed the relative importance of features in attacks (Appendix, Table 9 and Table 10). Based on this, we leverage the most important features whose relative importance is higher than 10 in Table 10 and group them into *query utility metrics* and *application utility metrics*.

4.1 Query Utility Metric

Query utility captures how many basic attributes are retained in the obfuscated queries. The first two basic features are the frequency of terms in the query and the number of characters in the query which have relative high importance in Table 10. For the number of characters in the query, we define the Number of Query Characters (NQC) metric which refers to the number of characters n_c in queries. For the the frequency of terms in the query, we define the Query Entropy (QE) metric which refers to the entropy of the frequency of users' queries and defined as $H(F) = -\sum_{i=1}^{n} f_i \log_2(f_i)$, where $F = \{f_1, \cdots, f_n\}$, and f_i is the i-th query term frequency.

The topic of query statements is another important attribute. We compute the topics associated with the query according to the Open Directory Project (ODP) [64], an openly available hierarchical ontology. We use ODP for categorizing queries into different semantic categories. The ODP dataset contains about 3.48 million web sites. The categories are organized within a tree, having the root as the common top category. Every category in the ontology has a path to the root. There are about 1.03 million categories as the leaves in the ODP dataset. Given a query, we calculate the ODP category of the query. First, we retrieve the top 10 relevant URLs in the ODP dataset. Second, for each URL (e.g., http://curlie.org/Arts/Crafts) or domain of the URL (e.g., http://curlie.org), we find the categories associated with it in the ODP dataset. Finally, we categorize the query as the most common category of the top 10 URLs.

After obtaining the ODP categories of queries, we calculate the topic based metrics which include the level 1 (e.g., Top/Arts) and level 2 (e.g., Top/Arts/Music) ODP categories. Specifically, we calculate: 1) ODP Level 1 Category Distribution (ODP1D), $D_1 = \{d_{11}, \cdots, d_{1n}\}$, which refers to the distribution of level 1 categories in the ODP taxonomy; 2) ODP Level 1 Category Entropy (ODP1E), which is defined as the entropy of level 1 categories in the ODP taxonomy and defined as $H(D_1) = -\sum_{i=1}^{n} p(d_{1i}) \log_2 p(d_{1i})$, where $p(d_{1i})$ is the probability of the i-th category in D_1; 3) ODP Level 2 Category Distribution (ODP2D), $D_2 = \{d_{21}, \cdots, d_{2n}\}$, which refers to the distribution of level 2 categories in the ODP taxonomy; 4) ODP Level 2 Category Entropy (ODP2E), which is defined as the entropy of level 2 categories in the ODP taxonomy and defined as $H(D_2) = -\sum_{i=1}^{n} p(d_{2i}) \log_2 p(d_{2i})$, where $p(d_{2i})$ is the probability of the i-th category in D_2.

4.2 Application Utility Metric

In reality, queries are used for information retrieval, high-level applications, etc. Therefore, besides examining queries' fundamental utilities, it is crucial to ensure that the obfuscated queries are valuable for practical applications. Toward this end, we evaluate two important application utility metrics as follows.

Personalization of Web Search (PWS). Personalization of web search, where different users searching for the same terms may observe different results, has been implemented by major search engines (e.g., Bing, Google) [12]. Personalization provides obvious benefits to users, including disambiguation and retrieval of locally relevant results to optimize users' decisions. Generally, there are two approaches to personalize search results, one involves modifying the user's query and the other re-ranks the search results [14].

To measure PWS, we set the control group as the original queries and the test group as the obfuscated queries. We submit both groups of queries to the search engine and collect the search results. Note that the detailed methods for collecting search activity is presented in Section 5.2. To measure the difference of each real query's search results between the test group and the control group. We apply the similar metrics used by [13] to measure personalization characteristics.

First, to measure the overlap of search results, we measure the *jaccard index* of PWS (PWS_J), which views the result lists as sets and is defined as the size of the intersection of the union.

$$PWS\_J = J(R_{test}, R_{control}) = \frac{|R_{test} \cap R_{control}|}{|R_{test} \cup R_{control}|}$$

where $R_{test} = \{r_1, \cdots, r_{10}\}$ (resp., $R_{control}$) is the search result list of the test group (resp., control group). A Jaccard Index of 0 represents no overlap between the search lists, while 1 indicates they contain the same result set.

Second, to measure the ranking of web search results, we compute the *edit distance* of PWS (PWS_E), which computes the number of the list elements that must be inserted, deleted, substituted, or swapped to make the control group result list identical to the test group result list and defined as $PWS\_E = D_{DL}(R_{test}, R_{control})$, where D_{DL} is the Damerau-Levenshtein distance [63]. For example, if $R_{control} = \{r_1, r_2, r_3, r_4, \cdots, r_{10}\}$ and $R_{test} = \{r_4, r_2, r_1, r_3, \cdots, r_{10}\}$, which needs two steps to change $R_{control}$ to R_{test}, $\{r_1, r_2, r_3, r_4, \cdots, r_{10}\} \rightarrow \{r_4, r_2, r_3, r_1, \cdots, r_{10}\} \rightarrow \{r_4, r_2, r_1, r_3, \cdots, r_{10}\}$, and thus the Damerau-Levenshtein distance between $R_{control}$ and R_{test} is 2. The smaller the value of edit distance, the closer the ranking of the two result lists.

TABLE 5: Analysis of existing query obfuscation techniques. √=preserving the utility, ◐=partially preserving the utility, ◆=conditionally preserving the utility depending on parameters and considered data, ✗=not preserving the utility, and **n/a** = evaluation not available in existing works.

	NQI	TMN	NISPP	PRAW	OQF-PIR	DisPA	GooPIR	PDS
NQC	√	◐	◐	◆	◆	◆	◐	✗
QE	√	◐	◐	◆	◆	◆	◐	✗
ODP1D	√	◐	◐	◆	◆	✗	◆	◆
ODP2D	√	◐	◐	◆	◆	✗	◆	◆
ODP1E	√	◐	◐	◆	◆	✗	◆	◆
ODP2E	√	◐	◐	◆	◆	✗	◆	◆
PWS	√	◐	◆	◆	◆	◐	◆	◆
UP	√	◐	◆	◆	◆	◐	◆	◆
R2WSPA	✗	✗	n/a	✗	n/a	n/a	✗	n/a

User Profile (UP). A user profile is the representation of a user model, which refers to the explicit digital representation of a person's identity [16]. User profile technologies can be applied in many meaningful applications, e.g., web mining, recommendation systems and social networking. We could construct user profile in two ways: explicitly, e.g., asking for registration information about user interests or ratings, or implicitly, e.g., using Natural Language Processing (NLP) techniques to extract user interesting topics. Generally, the search engine generates user profiles based on click logging, query history, browsing history, click-through history, etc. [17].

In this paper, we focus on query history and build the user profile on queries sent by each user [18]. First, we vectorize the query statements with Natural Language Processing (NLP) techniques. Then, we build the profile of each user based on its query history. We calculate the *cosine simmilarity* between the profile of obfuscated queries and the original queries.

4.3 Privacy vs Utility Analysis

We discuss the utility performance of existing query obfuscation techniques and summarize the query utility metrics, application utility metrics, and Resistance to WSP Attacks (R2WSPA) (e.g., [22], [27], [31]) of existing query obfuscation schemes in Table 5 and analyze as follows.

In query injection schemes, diverse strategies have been applied to inject dummy queries. There are two main differences between query injection schemes, fake data sources and obfuscation algorithms. For example, Naive Query Injection (NQI) randomly samples other users' queries as dummy queries for the target user. It follows that the correlation between dummy queries and real queries is very low, and it is obvious that NQI preserves most utility properties. However, due to the fact that the semantic similarity (e.g., similarity of query topics) of user's actual query statements is very high, NQI is also the most vulnerable scheme to WSP attacks. Compared with NQI, TMN contains two fake query sources, one is a static feed list of query terms and the other is a dynamic query-list which generates fake queries by observing a user's previous search activities. It follows that TMN partially preserves query and application utility metrics. Another work NISPP [11] gets dummy queries from a set of other user queries based on a certain probability. In addition, it can manually set the privacy level and change the sequence of issued queries. It follows that NISPP can partially preserve query utility and conditionally preserve application utility.

For profile-based schemes, the fundamental idea is to perturb the adversary's observed profiles so as to protect users' privacy. On one hand, DisPA [3] splits a user profile into several sub-profiles, which change the structural properties of the user's profile. It follows that it cannot preserve the ODP1D, ODP2D, ODP1E and ODP2E utility, while conditionally or partially preserving the NQC, PWS and UP utility. On the other hand, PRAW [8] and OQF-PIR [29] make use of dummy queries to make the user profile more general, which conditionally preserve all the utility. Particularly, the more dummy queries are added, the fewer utility it is preserved. Furthermore, PRAW defends against clustering attacks was shown in [22].

K-anonymity schemes [2] [23] can partially or conditionally preserve most query and application utility metrics. This is because the fundamental idea of the *k*-anonymity based scheme is to make *k* queries similar in some statistic features. However, there is always a trade-off between obfuscation and utility. If the *k* is large, more features will be destroyed, and more utilities will be lost. If the *k* is selected to be small, more utilities will be preserved at the cost of lower obfuscation level. Furthermore, most *k*-anonymity schemes choose fake queries from the thesaurus, hence the query utility NQC and QE may be partially lost. For example, GooPIR checks the popularity of keywords in the real query and selects *k*-1 words from a prepared thesaurus. The selected words have a similar level of popularity with the real query so as to prevent the real query from appearing more frequently. As for PDS [9], query privacy is achieved by replacing a real query with a set of canonical queries. It uses seed documents to construct discrete canonical sets and only the canonical queries could be sent to the search engine. Consequently, this scheme destroys the NQC and QE utility metrics. In addition, there is a definite connection between canonical queries and real queries. Thus, PDS conditionally preserves ODP1D, ODP1E, ODP2D, ODP2E, PWS, and UP. Furthermore, GooPIR could not defend against WSP attacks as shown in [31].

In summary, most of the obfuscation schemes can partially or conditionally preserve most utility metrics. Multiple factors can affect the availability of obfuscation mechanisms, e.g., source of fake queries, obfuscation algorithms, and the ratio of dummy queries to real queries. No existing work evaluates the resistance of state-of-the-art obfuscation schemes against modern WSP attacks. Although most of the schemes have theoretical privacy guarantees, unfortunately, the privacy analysis cannot guarantee that they can defend against modern PWS attacks due to the improper model of the adversary's auxiliary information, problematic assumptions, etc. To address this open problem, we analyze and verify the effectiveness of existing query obfuscation schemes against modern WSP attacks in Sections 2 and 5 respectively.

We make further remarks on OB-WSPES and its modules and functions as follows.

(a) OB-WSPES provides a platform which enables the user to evaluate the obfuscation methods, and conveniently choose modern obfuscation algorithms to obfuscate queries. In addition, they can also employ different attack methods and evaluation modules to examine whether the obfuscated data could satisfy the security/privacy and utility requirements.

(b) OB-WSPES is a uniform platform that allows web search obfuscation researchers to compare their obfuscation schemes with existing solutions as well as to examine their schemes' resistance against modern PWS attacks. OB-WSPES also permits WSP attack researchers to evaluate the performance of new WSP attacks by attacking the obfuscated queries for state-of-the-art obfuscation schemes. Therefore, OB-WSPES is helpful to both data owners and researchers in conveniently applying existing schemes, comprehensively understanding existing algorithms and effectively developing novel techniques.

(c) Besides providing a uniform platform, OB-WSPES is an easily portable and extendable system. First, the algorithm in OB-WSPES are implemented in Python and thus they are system independent. Second, as shown in Fig.1, multiple modules can work together to perform query obfuscation and WSP attack evaluation. Additionally, all the modules of OB-WSPES can work individually. Third, all the schemes/measurements within each module are independent, which means that they can be implemented, evaluated and employed independently. Furthermore, newly developed obfuscation/WSP attack schemes and utility metrics can be easily integrated into OB-WSPES.

5 Evaluation of OBWSPES

5.1 Primary Dataset

We employ the AOL dataset [40], which is a real web search dataset in 2006. This dataset contains approximately 21 million queries from nearly 650,000 users during a three month period. To collect real and rich search activities, we focus on the most active users. Specifically, we choose 174 users, and they have 301,389 queries. Note that [27] has shown the effectiveness of sampled dataset is adequate to estimate the overall distribution of attack performance.

5.2 Search Activity Collection

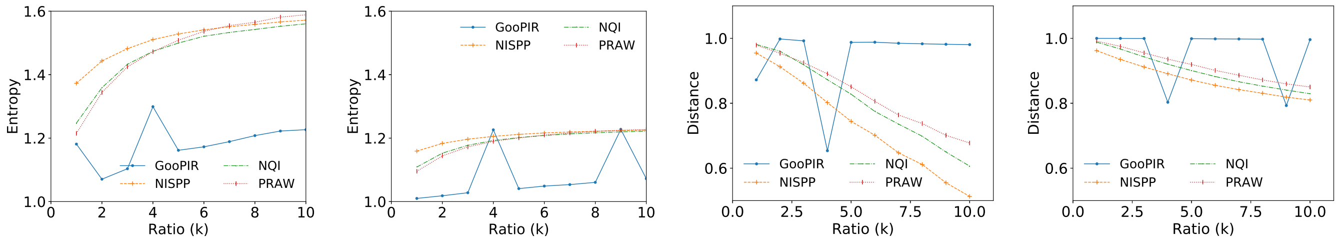
In order to gather realistic user search activities, we simulate and log each user's real search activities taking advantage of web crawler (Selenium) to control browser. First, we associate each user with a fake account. The crawler will initialize each user's profile in the browser (e.g. new cookie and empty history) when it starts searching for the user's queries. Second, each user's real queries will be sent in the original order of the AOL dataset. On one hand, users' search behaviors are relevant in terms of content and temporal characteristics (e.g., search sessions and search refinements) [62]. On the other hand, the search history will affect the personalized results returned by the search engine [13]. Finally, we collect search activities as described in Section 3.1. The real query is sent to the obfuscation mechanism, and then real query and dummy queries are issued to the search engine according to the sending strategy. We record the real query, dummy queries, and the time-stamps each query is issued. Then, we save the pages returned by the search engine, in which we focus on the top 10 results [13].

Particularly, as for TMN which is a popular plugin, we install the TMN (Version 0.10.4) into the browser and use the default setting (e.g., query frequency 10 per minute, enable burst and default RSS feed). TMN stores the search activities (e.g., real queries, dummy queries and URL) in the search log when we issue user's queries. To collect search activities, we issue user's queries with a TMN equipped browser and collect the real-time search logs in TMN.

As for GooPIR, we download and extract the GooPIR (Version 1.0). We use the original files (one month wiki news) contained

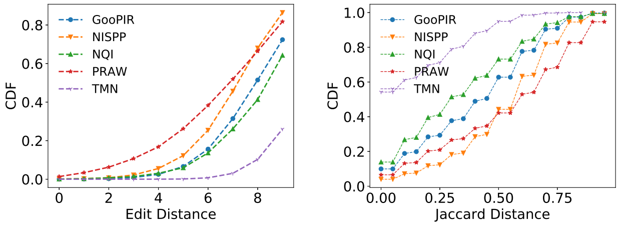
TABLE 6: Obfuscation vs Utility. Average utility metrics of obfuscated data vs original queries. *k* is the ratio of dummy queries to real queries.

	k	NQC	QE	ODP2E	ODP2D	ODP3E	ODP3D	PWS_J	PWS_E	UP
TMN	Def.	0.9971	1.005	1.0310	0.9993	1.0932	0.9778	**0.1507**	**9.5961**	0.7959
NQI	1	0.9938	1.0096	1.1087	**0.9883**	1.2475	**0.9811**	0.3372	8.4952	0.8958
NQI	4	0.9907	1.0095	1.1923	0.9202	1.4735	0.8724	0.7133	5.7739	0.8972
NQI	8	0.9882	1.0101	1.2167	**0.8527**	1.5425	**0.6983**	**0.4714**	**7.4948**	0.8975
GooPIR	1	0.9742	0.9744	1.0099	**0.9999**	1.1811	**0.8720**	**0.4266**	**8.1509**	0.6182
GooPIR	4	0.9673	0.9733	1.2260	0.8031	1.2989	0.6537	**0.5519**	**7.4377**	0.6086
GooPIR	8	0.9569	0.9657	1.0604	**0.9973**	1.2078	**0.9831**	0.3088	8.7427	0.6526
NISPP	1	0.9775	0.9997	1.1590	**0.9620**	1.3728	**0.9547**	0.5400	7.5488	0.9007
NISPP	4	0.9516	0.9992	1.2053	0.8906	1.5105	0.8018	0.6705	6.2594	0.9011
NISPP	8	0.9502	1.0000	1.2215	**0.8306**	1.5584	**0.6112**	0.6237	6.6431	0.9015
PRAW	1	0.9935	1.0663	1.0954	**0.9914**	1.2149	**0.9790**	0.5559	6.9296	0.8785
PRAW	4	0.9905	1.0652	1.1896	0.9358	1.4724	0.8906	**0.3053**	**8.7191**	0.8851
PRAW	8	0.9905	1.0586	1.2215	**0.8717**	1.5650	**0.7374**	0.3353	8.5639	0.8877



(a) ODP3E (b) ODP2E (c) ODP3D (d) ODP2D

Fig. 2: The average ODP utility metrics with different *k*. *k*=dummy queries/real queries.



(a) Edit Distance *k*=1 (b) Jaccard distance *k*=1

Fig. 3: CDF of the PWS utility metrics with *k*=1.

in the program as the external knowledge and generate dummy queries with the same method in [2]. The real and dummy queries are connected by "or" and sent to the search engine.

For NQI and NISPP, we build the external knowledge based on the queries of the sampled users in the AOL dataset. Specifically, the NQI mechanism randomly selects dummy queries from other users' queries. NISPP samples dummy queries from the external knowledge based on the privacy breaches of search queries [11].

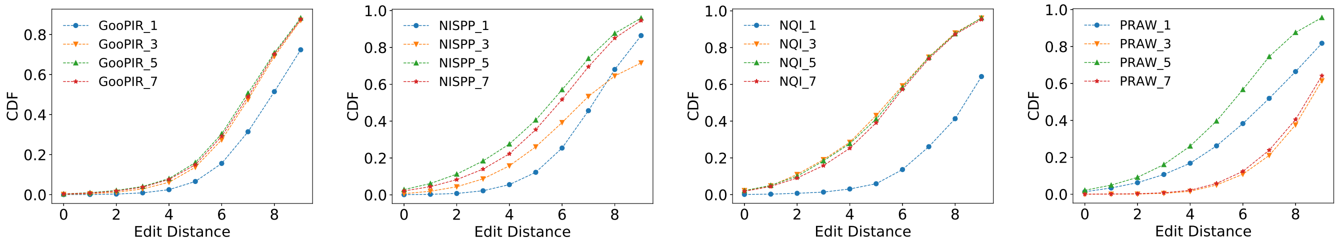
For PRAW, the fake query is constructed from a mix of terms from two sources. One source is the terms provided by the users' actual search history. The other source is a database of terms which are extracted from the user requested pages that provides random terms relating to the general topic interest of the user. Note that except TMN, in our framework, the external knowledge provided for GooPIR, NQI, NISPP and PRAW could be configured.

5.3 Utility Evaluation

In this subsection, we evaluate the utility performance of obfuscation algorithms. We obfuscate the original user queries, then measure how much utility are preserved in obfuscated queries. Specifically, when measuring the utility metrics of NQC, ODP1D, OPD2D and UP, we calculate the *cosine similarity* between the distributions of the obfuscated queries and original queries; when measuring QE, ODPE1, and ODPE2, we take the ratio between the obfuscated queries and original queries; when measuring PWS, we compute the *Edit Distance* (i.e., the Damerau–Levenshtein distance [41]) and *Jaccard similarity* between the obfuscated queries and original queries.

The criteria for obfuscation settings is that we follow the same or similar settings as the original works of these obfuscation schemes. For TMN, we apply the default parameter settings, and the query frequency is 10 per minute. In NQI, GooPIR, NISPP, and PRAW, we adjust the ratio of the dummy queries to the real queries from one to ten, which is enough to measure the effect of the volume change of injected noise [8]. We denote the ratio of dummy queries to real queries as *k* later in the paper. We demonstrate the results in Table 6 and analyze the results as follows.

Query Utility. As described in Table 6, among all the query utilities, NQC and QE are the most insensitive metrics, where the obfuscated queries are almost the same as the original data with respect to both metrics. Meanwhile, ODP3E and ODP3D are very sensitive to noise changes. They are the easiest to be destroyed by obfuscation mechanisms. Furthermore, we have the following discussions:



(a) GooPIR k=1,3,5,7

(b) NISPP k=1,3,5,7

(c) NQI k=1,3,5,7

(d) PRAW k=1,3,5,7

Fig. 4: CDF of Edit Distance for GooPIR, NISPP, NQI, and PRAW with different k. k=dummy queries/real queries

(a) *Generally, the query utility declines with the increasing amount of dummy queries.* For example, in Figure 2, ODP3D in NQI changes from 0.9811 (k=1) to 0.6061 (k=10) and the ODP3D for NQI, PRAW, NISPP significant declines with the increase of k. The entropy (ODP3E and ODP2E) becomes larger with k, which also means the decreases of query utility. However, the utility performance for GooPIR is unstable. This is because the source of fake queries for GooPIR is small and out-of-date. In many cases, the sample method could not get enough fake queries. Thus, the results of GooPIR may change greatly.

(b) *With the same k, different obfuscation mechanisms preserve different degrees of utilities.* Compared with GooPIR, NQI only preserved 69.83% (k=8) ODP3D utility. This is because the thesaurus used by NQI is much larger than GooPIR whose thesaurus was extracted from specific articles. Although there is no big thesaurus as the source of dummy queries for NISPP and PRAW, they apply different strategies of obfuscation, and show their ability to obfuscate user queries. For example, compare with the original queries, each of them preserves 73.74% (PRAW, k=8) and 61.12% (NISPP, k=8) ODP3D utility.

(c) *The more detailed the data is described, the more sensitive it is to noise.* For example, the character distribution of the obfuscated queries remains the same as the original queries, while the detailed categories of the query change significantly. The fact that ODP3 (ODP3E and ODP3D) metrics are more sensitive than ODP2 (ODP2E and ODP2D) metrics shows that the more detailed user queries classify, the more sensitive it is to query changes. This gives us an inspiration that if we map the data to the meaningful high dimensional space, we will get a better understanding of obfuscation mechanisms.

Application Utility. In Table 6, we can see that PWS application utility metrics are more sensitive than query utility metrics. For example, most NQC and ODP2D metric results are bigger than 0.8, while almost all PWS_J metric results are less than 0.6, which means that about 40% search results are different from the original ones. We have the further discussions as follows:

(a) *The manner of interleaving real queries and dummy queries affects the search results.* In Figure 3, compared with other mechanisms, TMN destroys most PWS metrics (PWS_J=0.1507, PWS_E=9.5961). This is because TMN sends fake queries at a fixed time interval and in parallel with user's real queries. In this way, the search engine is unable to determine the user's true intentions which results in low availability. PWS metric results of GooPIR are better than that of NQI. This is because NQI sends obfuscated queries in a definite order. While GooPIR connects real queries and fake queries with *OR* and sends them together to the search engine which is more convenient to retrieve effective information.

(b) *Adding more dummy queries to users' real data does not yield more web search result changes.* Intuitively, the PWS result will get worse with the increase of k. However, we observe that different obfuscation mechanisms and k lead to different PWS results. In Figure 4, the sort of PWS performance on NISPP is NISPP_1 > NISPP_3 > NISPP_7 > NISPP_5, and on PRAW the sort of PWS performance is PRAW_3 > PRAW_7 > PRAW_1 > PRAW_5. While for GooPIR and NQI, they always have the best PWS performance when k=1. When the k is greater than 2, the PWS performance on GooPIR and NQI almost has no changes. Furthermore, in Table 6, we can see that the search results of larger k may perform better than smaller k (e.g, GooPIR k=4 (PWS_J=0.5519 and PWS_E=7.4377) is better than GooPIR k=1 (PWS_J=0.4266 and PWS_E=8.150) and NQI k=8 (PWS_J =0.4714 and PWS_E=7.4948) is better than PRAW k=4 (PWS_J =0.3053 and PWS_E=8.7191). This is because many factors may affect the application utilities, e.g., queries contents, the manner of sending queries, the ratio of dummy queries to real queries, and the obfuscation algorithm. Therefore, when it comes to obfuscate user queries, we should select the obfuscation tools/methods and parameter settings depending on the corresponding scenario.

Summary. In this subsection, we evaluate the utility of obfuscated queries. The evaluation results in Table 6 are consistent with our analysis in Table 5. Most obfuscation algorithms can partially or conditionally preserve query and application utilities. Therefore, the obfuscated data can be employed for query log analysis and data mining tasks. In addition, we find that query utility declines with the increasing amount of dummy queries. However, the application utility does not have the same nature as query utility.

No obfuscation algorithm is optimal in preserving every utility and more dummy queries do not mean more application utility destroyed. Four major factors affect the utility of obfuscated queries are: 1) the content and the size of the thesaurus; 2) the obfuscation algorithm; 3) the manner of interleaving dummy and real queries; 4) the ratio of dummy queries to real queries. Therefore, when choosing obfuscation algorithms or tools, it is better to take into account the corresponding application scenario.

5.4 Feature Vectorization Scheme

To the best of our knowledge, almost all previous trials use the one-hot encoding scheme, while no previous literature analyzes the effect of different vectorization schemes in the PWS area. To fill this gap, in this subsection, we measure the impact of different vectorization schemes (word embedding and one-hot encoding) on the effectiveness of the attack algorithms.

Specifically, we experiment 1) Word2Vec (W2V) which has 300 dimensions and is trained by AOL, 2) One-hot encoder High

TABLE 7: Average accuracy and average CPU time to analyze one user's data.

	Time (s)			Accuracy		
Module	W2V	OHH	OHL	W2V	OHH	OHL
RF	0.1015	3.6754	0.6518	0.8774	0.9031	0.8972
SVM	**0.3827**	**31.0410**	**5.6986**	**0.8204**	**0.8207**	**0.8178**
NB	0.2659	21.7176	3.9781	0.7968	0.7808	0.7698
GBC	1.0762	41.3206	7.1235	0.8284	0.8159	0.8015
LR	0.9047	36.0651	6.3222	0.8411	0.8331	0.8201
NC	0.7589	30.4581	5.3380	0.8334	0.8406	0.8191
DTC	0.6816	26.6342	4.6858	0.8310	0.8472	0.8288
MLP	1.5245	91.5149	16.3423	0.8412	0.8549	0.8375
K-means	**1.4686**	**83.0000**	**15.1997**	**0.8061**	**0.8085**	**0.7952**

dimension (OHH) which has 28,939 dimensions and is trained by AOL, 3) One-Hot encoder Low dimension (OHL) which has 5,218 dimensions and is trained by the sampled user's query data. We use TrackMeNot (TMN), which has been used in previous literature [27] [30], to obfuscate real queries. The AOL dataset contains three months (March, April and May) of query logs. We take OB-PWS activities in March and April as training data and the test data are users' last month OB-PWS activities. We train the model based on the training data and detect the CPU time (second) and the accuracy of the model on the testing dataset.

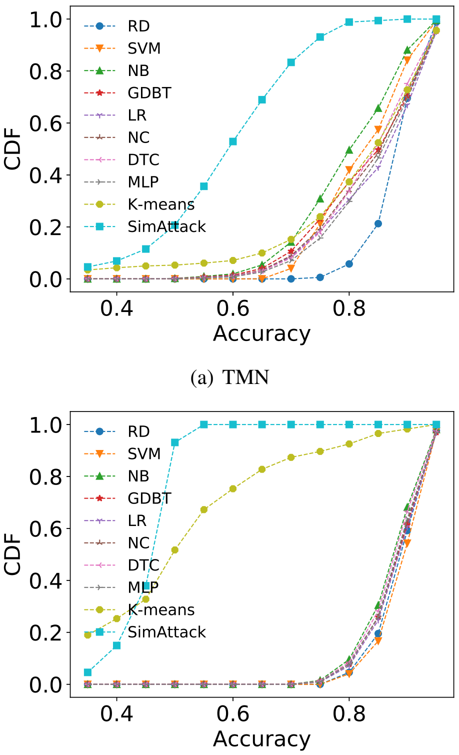
Table 7 shows the average accuary and average CPU time to analyze one user's data, we can conclude that W2V gets high accuracy attack results, which is basically the same with OHH and higher than OHL. More importantly, the average time used in W2V to analyze one user's data drastically reduced. For example, in the GBC method, the average CPU time to analyze one user's data for OHH and OHL is 40 (41.31) times and 7 (7.12) times longer than that of W2V (1.07) respectively.

Summary. In this subsection, we find that the word embedding scheme is more efficient than previous schemes. There are two reasons: First, users' queries are more relevant in semantic. Second, the word embedding techniques map data to high dimensional semantic space (lower than one-hot encoding) and preserve more semantic information than one-hot encoder solutions, which results in the attack methods classifying real queries out of obfuscated data more accurately and efficiently.

5.5 Obfuscation vs Attack

In this subsection, we measure the effectiveness of the state-of-the-art obfuscation techniques against modern WSP attacks. We employ the same AOL dataset as before. The methodology is that we first employ different obfuscation techniques to obfuscate AOL user data, and change the ratio of dummy queries to real queries from 1 to 10. Then, we employ different WSP attack algorithms to attack the obfuscated data. Specially, we employ RD, SVM, NB, GBC, LR, NC, DTC, MLP, *K*-means and SimAttack against all obfuscation mechanisms (TMN, GooPIR, NISPP, NQI and PRAW). We show the accuracy results of WSP attacks to each obfuscation mechanism in Table 8. From the results, we have the following findings.

(a) *Classification based attack algorithms are more effective than clustering based and linkage function learning based methods.* Figure 5 shows the cumulative distribution of accuracies of different attacks against TMN and GooPIR. In TMN, the average accuracy of SimAttack is 0.5820 which is much smaller than classification based or clustering based attack algorithms. In GooPIR, the difference is more noticeable, as the average accuracy of classification based attacks is 1.6 times and 1.98 times of that of *K*-means and SimAttack respectively.



(a) TMN

(b) GooPIR

Fig. 5: CDF of accuracy of different attacks against TMN and GooPIR with *k*=1. *k*=dummy queries/real queries.

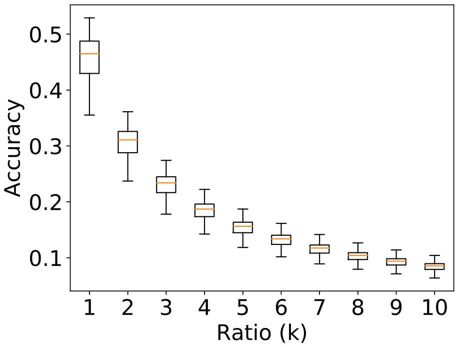


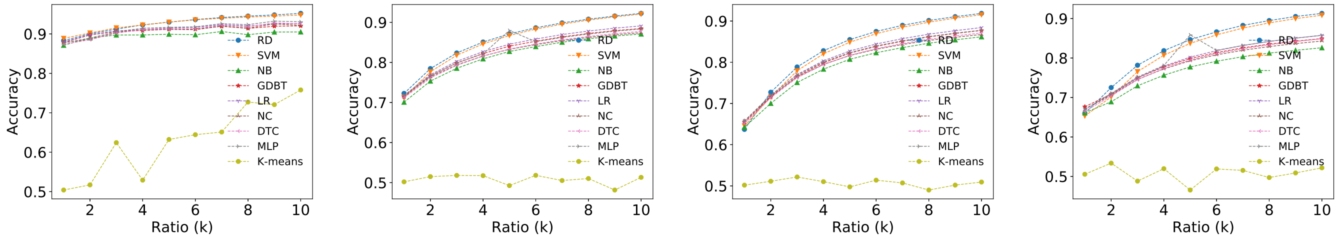
Fig. 6: The accuracy of SimAttack with the change of *k*. *k*=dummy queries/real queries.

(b) An interesting finding is that: ***adding more fake queries to users' real data does not necessarily yield better privacy.*** This finding was contrary to our intuition. Figure 6 shows that the attack performance in SimAttack decreases rapidly with the increase of *k*. However, Figure 7 shows that no matter what obfuscation mechanism is, attack performance of classification based attack schemes improves with the increase of *k*. In addition, the attack performance in *K*-means is basically unchanged with the increase of *k*. This finding further confirms that the classification based attack algorithms are more effective than clustering based and linkage function learning based methods. Besides, it shows that in front of powerful modern WSP attacks, more dummy queries may make an adversary more aware of the obfuscation mechanism. Thus, the adversary could easily access valuable information.

(c) *Different obfuscation strategies and sources of fake queries result in different effects of privacy protection.* Figure 8 presents

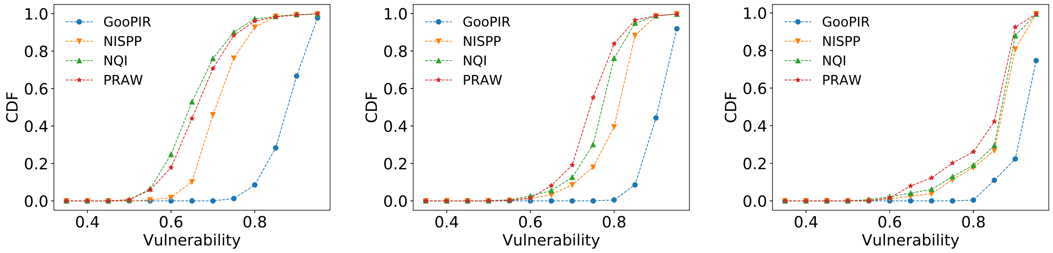
TABLE 8: Obfucation vs Attack. The average accuracy of attacks against obfuscation mechanisms. k=dummy queries/real queries.

OB-PWS	*k*	RD	SVM	NB	GBDT	LR	NC	DTC	MLP	K-means	SimAttack
TMN	1	**0.8774**	0.8204	0.7968	0.8284	0.8411	0.8334	0.8308	0.8412	0.8035	**0.5820**
GoopPIR	1	0.8830	0.8886	0.8711	0.8782	0.8798	0.8777	0.8737	0.8771	**0.5041**	**0.4472**
GoopPIR	3	0.9121	0.9152	0.8971	0.9063	0.9076	0.9031	0.9020	0.9056	0.6243	0.2250
GoopPIR	5	0.9298	0.9297	0.8992	0.9117	0.9148	0.9114	0.9125	0.9166	0.6323	0.1506
GoopPIR	7	0.9415	0.9396	0.9060	0.9192	0.9233	0.9196	0.9217	0.9260	0.6512	0.1129
GoopPIR	9	**0.9481**	**0.9451**	0.9046	0.9194	0.9251	0.9253	0.9279	0.9323	**0.7206**	**0.0905**
NISPP	1	**0.7226**	0.7124	0.7007	0.7141	0.7174	0.7160	0.7121	0.7179	0.5021	0.3049
NISPP	3	0.8235	0.8178	0.7853	0.7985	0.8026	0.7939	0.7926	0.7984	0.5181	0.1890
NISPP	5	0.8711	0.8672	0.8273	0.8404	0.8454	0.8326	0.8333	0.8796	0.4927	0.1380
NISPP	7	0.8985	0.8955	0.8503	0.8637	0.8694	0.8544	0.8563	0.8622	0.5053	0.1110
NISPP	9	**0.9157**	0.9136	0.8658	0.8793	0.8854	0.8690	0.8718	0.8778	**0.4816**	**0.0933**
NQI	1	0.6371	0.6457	0.6423	0.6544	0.6569	0.6574	0.6515	**0.6575**	**0.5018**	**0.4497**
NQI	3	0.7885	0.7794	0.7508	0.7661	0.7703	0.7643	0.7625	0.7690	0.5216	0.2308
NQI	5	0.8550	0.8486	0.8072	0.8222	0.8274	0.8164	0.8168	0.8233	**0.4976**	**0.1583**
NQI	7	0.8896	0.8849	0.8355	0.8509	0.8573	0.8431	0.8453	0.8521	0.5072	0.1218
NQI	9	**0.9106**	0.9071	0.8543	0.8698	0.8766	0.8606	0.8639	0.8707	**0.5019**	**0.1001**
PRAW	1	0.6597	0.6532	0.6612	0.6766	0.6713	**0.6699**	0.6649	0.6695	0.5055	0.4478
PRAW	3	0.7817	0.7655	0.7297	0.7497	0.7500	0.7458	0.7449	0.7518	0.4881	0.2338
PRAW	5	0.8467	0.8372	0.7774	0.7987	0.8022	0.7930	0.7952	0.8031	0.4656	0.1610
PRAW	7	0.8827	0.8759	0.8034	0.8258	0.8317	0.8197	0.8235	0.8319	0.5153	0.1235
PRAW	9	**0.9052**	0.8998	0.8195	0.8427	0.8504	0.8366	0.8418	0.8504	**0.5088**	**0.1015**



(a) GooPIR (b) NISPP (c) NQI (d) PRAW

Fig. 7: The average accuracy of different attacks aganist GooPIR, NISPP, NQI and PRAW with the change of *k*. *k*=dummy queries/real queries.



(a) $k = 1$ (b) $k = 3$ (c) $k = 7$

Fig. 8: CDF of vulnerability for different obfuscation mechanisms with *k*=1, 3, 7. *k*=dummy queries/real queries.

the vulnerability of different obfuscation mechanisms. The vulnerability is defined as the average accuracy of different attacks. We can see that: with lower *k*, the sort of privacy protection effect is NQI > PRAW > NISPP> GooPIR. While, with larger *k* the sort of privacy protection effect is PRAW > NQI > NISPP> GooPIR. The main reason for GooPIR being the most vulnerable to attack is that the source of fake queries of GooPIR is a small static data source (one month Wiki news in Octorber 2007). It is easy for an adversary to get and analyze these data, and split user's real data from obfuscated queries. As we implement PRAW, NQI and NISPP in the same source of fake queries, their principal difference is the obfuscation strategy. Therefore, we can conclude that strategies and fake query sources play important roles in obfuscation mechanisms, which is consistent with our findings in Subsection 5.2.

Summary. Based on the above results and findings, we have the following discussions.

Among all the obfuscation techniques, we find that TMN is the most vulnerable. There are many reasons for this, e.g., the amount of dummy queries is not proportional to real queries, thesaurus or RSS feeds could be learned by the adversary, the obfuscated queries do not resemble original user queries, etc. In most cases, PRAW performs better than the other mechanisms. This is because the strategy used by PRAW is measuring the change of user profile (internal and external). In addition, it not only uses static fake query source, but also generates dummy queries based on user search results. According to the results in Table 8 and Table 6, it turns out that the obfuscation strategy, *the ratio of dummy queries to real queries*, *the size of fake queries source*, *the obfuscation algorithm* and *the strategy of sending obfuscated queries* will affect the performance of obfuscation mechanisms. Hence, it is possible to design an efficient, scalable and portable obfuscation tool in particular scenes.

All the obfuscation algorithms are vulnerable to some or all of the modern WSP attacks, which confirms our analytical results in Section 2. Generally, the state-of-the-art query obfuscation algorithms are vulnerable because: First, in existing obfuscation schemes, the explicit privacy information is not fully covered or protected by dummy information. Because of this, naive feature extraction methods could be used to de-anonymize user queries. Second, users' queries are closely related in semantic space, and it is easy for an adversary to separate users' real queries out of obfuscated data when mapping user data to high dimensional semantic space. Furthermore, the design philosophy of existing obfuscation schemes is to preserve as much utility as possible. However, utilities can also be used to conduct powerful WSP attacks. There is always a tradeoff between privacy and utility. Therefore, it is still an open problem to design effective obfuscation algorithms which can defend against modern WSP attacks.

Among all the attack algorithms, classification based machine learning methods such as GBC, RD and MLP perform much better than other attacks in most scenarios. This is because these algorithms can effectively combine implicit and explicit text characteristics and assign appropriate weights to features. Other attacks might depend too much on a single attribute. For example, SimAttack depends on the user profile. Our evaluation results in Section 5 consistent with our utility analysis in Section 2 that obfuscated queries conditionally or partially preserve most utility metrics (e.g., ODP2E, ODP3D and PWS). Furthermore, it turns out that the more attributes or utilities the attack method can capture, the more effective the attack is. It is possible to combine the advantages of different attacks to develop more efficient and flexible methods.

6 Discussion

6.1 Evolution of Obfuscation-based Web Search Privacy

To enforce unlinkability and indistinguishably of privacy web search, a series of studies [57], [58], [59] integrated proxy-based and obfuscation-based methods in recent years. We evaluate the obfuscation methods of PEAS [57], X-Search [59] and CYCLOSA [58], and discuss the results in Appendix B. The experimental results again confirm our findings in Section 5. Next, we discuss the evolution of obfuscated-based privacy web search mechanisms.

The primitive idea of protecting users' queries was simple at the beginning. A straightforward scheme is naive query injection, which randomly samples fake queries from fake query sources (e.g., other user's queries, seed files, and HTML pages) and injects the selected fake queries to real queries [1]. With the development of data anonymization, k-anonymization has been widely used in obfuscating queries. A release of queries is said to have the k-anonymity property if any query in the release cannot be distinguished from at least k-1 other queries contained in the release [2], [6], [9], [10], [37], [60], [61]. Furthermore, another popular idea to protect query privacy is profile-based obfuscation [3], [5], [8], [29]. It uses statistic techniques to monitor user profile, and add fake queries to make it difficult to re-identify user's real profile. Note that the development of obfuscation technologies are correlated. For example, PRAW [8] measures users' profile, injects k-1 fake queries, and samples fake queries from both static and dynamic query sources.

With the aggravation of the game between query protection and attack, researchers combine the advantages of obfuscation-based solutions with proxy-based or cryptography-based solutions. For example, PEAS [57] combines a new proxy protocol with a new obfuscation method, and X-Search [59] and CYCLOSA [58] send users' queries (fake and real) to the search engine through a proxy, and improve security by relying on Intel SGX. To advance the development of obfuscation-based technologies, our OB-WSPES provides a uniform evaluation system for obfuscation-based web search privacy, which allows researchers to conduct fair analysis and evaluation of existing or newly developed web search privacy protection/attack techniques. On one hand, the combination of different query protection techniques will reduce the success possibility of re-identification attacks and distinguish attacks. One the other hand, we believe that the progress of NLP technologies will further promote the development of query obfuscation technologies. We discuss the future works in Section 6.3.

6.2 Limitations

As a practical obfuscation based web search privacy evaluation framework (OB-WSPES), we believe our work can be improved in several perspectives.

Integrate More Obfuscation and Attack Schemes. In OB-WSPES, we focus on five obfuscation techniques and evaluate them against ten privacy attack methods. In practice, there might exist additional obfuscation mechanisms and attack methods. Though our research and findings are useful and effective, in order to make the system more practical and useful for industry and academic researches, it would be much better to integrate more obfuscation and attack schemes.

Consider More Features. The features extracted by each attack method may be different. For example, SimAttack extracts profile features and QWSP extracts textual characteristics, behavioral characteristics, and temporal characteristics. In order to compare the various attack algorithms fairly, in OB-WSPES, the main features extracted are textual features. Although our methods of feature extraction have reflected the quality of the attack algorithms, it would be much better to extract more useful information from users' explicit and implicit data, e.g., the number of clicks, temporal and regional characteristics. We believe that the success rate of attacks would be improved if we use more features.

More Evaluation Metrics. In OB-WSPES, we evaluate obfuscation tools and methods with respect to utilities and the ability to resist attacks. Specifically, we have evaluated 6 query utility metrics, 2 application utility metrics and defended against 10 WSP attacks. These metrics can reflect the basic properties of these tools/methods. The evaluation would be more valuable if we consider more evaluation metrics. For example, metrics for search result ranking, the time to retrieve useful information, and the impact on network throughput. Furthermore, if a comprehensive evaluation index is given, it will be much easier for users to choose proper obfuscation methods or tools.

6.3 Future Work

In order to better protect web search privacy, more future works could be done in the following directions.

Query Obfuscation. According to our analytically results in Section 2 and evaluation results in Section 5, most obfuscation techniques are vulnerable to modern WSP attacks. However, it is very important to protect web search privacy. We consider improving web search privacy in the following ways:

1) *Semantic space based obfuscation mechanism.* As we have analysis that most existing schemes obfuscate queries based on the statistical characteristics, and there is no scheme that obfuscates user queries in semantic space. However, the attackers can easily successful attack at semantic level. With the develop of NLP techniques, it is possible to build a semantic based obfuscation mechanism which maps user real queries to semantic space, introduces calculated noise to queries and converts fake queries from semantic space to natural-language spatial.

2) *Combining obfuscation and unlinkability*. Web search privacy faces at least two aspects attacks: identity attack and de-anonymization attack. Limitations of web search privacy using only obfuscation-based mechanisms, proxy-based mechanisms or cryptography-based mechanisms have been pointed out. To achive the unlinkability and the indistinguishability of search activities, it is interesting to combine the advantages of the three machanisms (e.g., [57], [58], [59]). Through careful design, combining obfuscation and unlinkability may get an universal powerful web search privacy protection tool.

3) *Specific scenario aware protection*. A possible research direction could be developing multiple-roles and specific scenario based obfuscation techniques. This is because a user's web search is multifaceted, and different facets require different privacy intensity. We could divide user's queries into parts based on user's different facets, give each group of queries a new identity, and obfuscate each group of queries. In this way, the user's profile could be decoupled and obfuscated. This may achieve better obfuscation and meanwhile support some application utility, e.g., personalization of web search and recommender system.

Web Search Privacy Attack. Based on our WSP attacks evaluation result, future WSP attack researches may be improved in the following two directions.

1) *Neural network based attacks*. Neural network based techniques can extract more semantic features and their connections. The new NLP and tools to analyze and manipulate text become mature and can provide room in the design of attacks to break the obfuscation. For example, they might use deep neural networks with various pre-trained user or word embedding to classify collected queries. In addition, the neural network model can be continuously updated according to the query data stream.

2) *Combining fingerprinting and de-anonymization*. In addition to queries, we can get rich information in the real world such as location, URLs, time delay, virtual address and device-specific information. Through the analysis of the rich information, we can use fingerprinting technologies to identify the user's real identity and link each user with its queries, which is accurately identified attacks. Browser fingerprinting has reached a state of maturity where it is now used by many companies alongside cookies to identify and track devices for a wide range of purposes from targeted advertising to fraud prevention [65], [66], [68]. For example, the behavior of communication traffic may reveal some patterns (such as, packet size, packet direction, and inter-packet time, etc.) that can expose users' identities. If we successfully link the senders to queries, although the users apply obfuscation technology to protect their queries, it is more easy for us to perform de-anonymization attacks to distinguish users' real intention and interests.

Evaluation Platform. In this paper, we focus on implementing and evaluating query obfuscation and web search privacy attack techniques. It is also attractive to integrate the web search protection and attack techniques, e.g., proxy-based and cryptography-based schemes. In the future, we will develop a uniform and open-source evaluation system supporting multi-type web search protection and attack schemes.

7 Conclusion

In this paper, we propose, implement and evaluate OB-WSPES, a uniform evaluation system for obfuscation based web search privacy. Within OB-WSPES, we systematically analyze, implement, and evaluate five query obfuscation algorithms, two feature vectorization methods, eight query utility metrics, and ten modern WSP attacks. To the best of our knowledge, OB-WSPES is the first such system that provides a practical platform enabling the user to obfuscate queries and evaluate the security of their data. In addition, it allows researchers to conduct fair studies of existing or newly developed obfuscation/WSP attack techniques.

Leveraging OB-WSPES, we conduct extensive experiments. The result demonstrates that (*i*) adding more fake queries to users' real data does not yield better privacy. In front of powerful attack methods, more dummy queries result in higher success attack rate; (*ii*) the query utility preserved by obfuscated queries declines with the increasing amount of dummy queries, while the application utility does not; and (*iii*) all the state-of-the-art obfuscation schemes are vulnerable to several or all of the modern web search privacy attacks. The degree of vulnerability depends on several factors, e.g., the content and the size of the thesaurus, the obfuscation algorithm, the strategy of sending obfuscated queries and the ratio of dummy queries to real queries. Finally, based on our findings and analysis, we discuss the future research directions of query obfuscation and web search privacy attack.

References

[1] D. C. Howe and H. Nissenbaum, "Trackmenot: Resisting surveillance in web search," *Lessons from the Identity trail: Anonymity, privacy, and identity in a networked society*, vol. 23, pp. 417–436, 2009.

[2] J. Domingo-Ferrer, A. Solanas, and J. Castellà-Roca, "h(k)-private information retrieval from privacy-uncooperative queryable databases," *Online Information Review*, vol. 33, no. 4, pp. 720–744.

[3] M. Juarez and V. Torra, "Dispa: An intelligent agent for private web search," in *Advanced Research in Data Privacy*. Springer, 2015, pp. 389–405.

[4] M. Juárez and V. Torra, "A self-adaptive classification for the dissociating privacy agent," in *Privacy, Security and Trust (PST), 2013 Eleventh Annual International Conference on*. IEEE, 2013, pp. 44–50.

[5] M. Juarez and V. Torra. (2017) Dispa. [Online]. Available: https://code.google.com/archive/p/dispa-framework/source

[6] Sweeney, Latanya. "k-anonymity: A model for protecting privacy." International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10.05 (2002): 557-570.

[7] A. Arampatzis, G. Drosatos, and P. S. Efraimidis, "Versatile query scrambling for private web search," *Information Retrieval Journal*, vol. 18, no. 4, pp. 331–358, 2015.

[8] B. Shapira, Y. Elovici, A. Meshiach, and T. Kuflik, "Praw—a privacy model for the web," *Journal of the Association for Information Science and Technology*, vol. 56, no. 2, pp. 159–172, 2005.

[9] M. Murugesan and C. Clifton, "Plausibly deniable search," in *Proceedings of the Workshop on Secure Knowledge Management (SKM 2008)*, 2008, pp. 3–4.

[10] Murugesan, Mummoorthy, and Chris Clifton. "Providing privacy through plausibly deniable search." Proceedings of the 2009 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, 2009.

[11] S. Ye, F. Wu, R. Pandey, and H. Chen, “ Noise injection for search privacy protection,” in *Computational Science and Engineering, 2009. CSE’09. International Conference on*, vol. 3. IEEE, 2009, pp. 1–8.

[12] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. Torr, “Bing: Binarized normed gradients for objectness estimation at 300fps,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 3286–3293.

[13] A. Hannak, P. Sapiezynski, A. Molavi Kakhki, B. Krishnamurthy, D. Lazer, A. Mislove, and C. Wilson, “Measuring personalization of web search,” in *Proceedings of the 22nd international conference on World Wide Web*. ACM, 2013, pp. 527–538.

[14] J. Teevan, S. T. Dumais, and E. Horvitz, “Personalizing search via automated analysis of interests and activities,” in *ACM SIGIR Forum*, vol. 51, no. 3. ACM, 2018, pp. 10–17.

[15] “Personalized search,” https://en.wikipedia.org/wiki/Personalized_search, 2018, "[Online; 2018-05-03]".

[16] Wiki. (2018) User profile. [Online]. Available: https://en.wikipedia.org/wiki/User_profile

[17] K. Mivule, “Web search query privacy, an end-user perspective,” *Journal of Information Security*, vol. 8, no. 01, p. 56, 2016.

[18] M. Speretta and S. Gauch, “Personalized search based on user search histories,” in *Web Intelligence, 2005. Proceedings. The 2005 IEEE/WIC/ACM International Conference on*. IEEE, 2005, pp. 622–628.

[19] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, *Recommender systems handbook*. Springer, 2015.

[20] M. Göksedef and Ş. Gündüz-Öğüdücü, “Combination of web page recommender systems,” *Expert Systems with Applications*, vol. 37, no. 4, pp. 2911–2922, 2010.

[21] S. Ji, W. Li, P. Mittal, X. Hu, and R. A. Beyah, “Secgraph: A uniform and open-source evaluation system for graph data anonymization and de-anonymization.” in *USENIX Security Symposium*, 2015, pp. 303–318.

[22] Y. Elovici, B. Shapira, and A. Meshiach, “Cluster-analysis attack against a private web solution (praw),” *Online Information Review*, vol. 30, no. 6, pp. 624–643, 2006.

[23] C. Carpineto and G. Romano, “Semantic search log k-anonymization with generalized k-cores of query concept graph,” in *European Conference on Information Retrieval*. Springer, 2013, pp. 110–121.

[24] A. Arampatzis, P. Efraimidis, and G. Drosatos, “Enhancing deniability against query-logs,” *Advances in Information Retrieval*, pp. 117–128, 2011.

[25] A. Arampatzis, P. S. Efraimidis, and G. Drosatos, “A query scrambler for search privacy on the internet,” *Information retrieval*, vol. 16, no. 6, pp. 657–679, 2013.

[26] R. W. White, A. Hassan, A. Singla, and E. Horvitz, “From devices to people: Attribution of search activity in multi-user settings,” in *Proceedings of the 23rd international conference on World wide web*. ACM, 2014, pp. 431–442.

[27] A. Gervais, R. Shokri, A. Singla, S. Capkun, and V. Lenders, “Quantifying web-search privacy,” in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2014, pp. 966–977.

[28] J. H. Friedman, “Stochastic gradient boosting,” *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, 2002.

[29] 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.

[30] S. T. Peddinti and N. Saxena, “Web search query privacy: Evaluating query obfuscation and anonymizing networks,” *Journal of Computer Security*, vol. 22, no. 1, pp. 155–199, 2014.

[31] A. Petit, T. Cerqueus, A. Boutet, S. B. Mokhtar, D. Coquil, L. Brunie, and H. Kosch, “Simattack: private web search under fire,” *Journal of Internet Services and Applications*, vol. 7, no. 1, p. 2, 2016.

[32] R. Al-Rfou, W. Jannen, and N. Patwardhan, “Trackmenot-so-good-after-all,” *arXiv preprint arXiv:1211.0320*, 2012.

[33] Wiki. (2018) Ego surfing. [Online]. Available: https://en.wikipedia.org/wiki/Egosurfing

[34] B. Bi, M. Shokouhi, M. Kosinski, and T. Graepel, “Inferring the demographics of search users: Social data meets search queries,” in *Proceedings of the 22nd international conference on World Wide Web*. ACM, 2013, pp. 131–140.

[35] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[36] J. Pennington, R. Socher, and C. Manning, “Glove: Global vectors for word representation,” in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[37] Y. Xu, K. Wang, G. Yang, and A. W. Fu, “Online anonymity for personalized web services,” in *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM, 2009, pp. 1497–1500.

[38] Q. Zhu, T. Zhao, and S. Wang, “Privacy preservation algorithm for service-oriented information search,” *Jisuanji Xuebao(Chinese Journal of Computers)*, vol. 33, no. 8, pp. 1315–1323, 2010.

[39] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, “Collaborative filtering recommender systems,” in *The adaptive web*. Springer, 2007, pp. 291–324.

[40] G. Pass, A. Chowdhury, and C. Torgeson, “A picture of search.” in *InfoScale*, vol. 152, 2006, p. 1.

[41] M. Li, Y. Zhang, M. Zhu, and M. Zhou, “Exploring distributional similarity based models for query spelling correction,” in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2006, pp. 1025–1032.

[42] D. Demmler, A. Herzberg, and T. Schneider, “Raid-pir: Practical multi-server pir,” in *Proceedings of the 6th edition of the ACM Workshop on Cloud Computing Security*. ACM, 2014, pp. 45–56.

[43] C. Aguilar-Melchor, J. Barrier, L. Fousse, and M.-O. Killijian, “Xpir: Private information retrieval for everyone,” *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 2, pp. 155–174, 2016.

[44] T. Gupta, N. Crooks, W. Mulhern, S. T. Setty, L. Alvisi, and M. Walfish, “Scalable and private media consumption with popcorn.” in *NSDI*, 2016, pp. 91–107.

[45] J. Cappos, “Avoiding theoretical optimality to efficiently and privately retrieve security updates,” in *International Conference on Financial Cryptography and Data Security*. Springer, 2013, pp. 386–394.

[46] S. L. Blond, P. Manils, C. Abdelberi, M. A. D. Kaafar, C. Castelluccia, A. Legout, and W. Dabbous, “One bad apple spoils the bunch: exploiting p2p applications to trace and profile tor users,” *arXiv preprint arXiv:1103.1518*, 2011.

[47] S. Dutta Chowdhuri, “Measurements of a latency-biased expander topology in the tor anonymity system,” 2013.

[48] L. Xin and W. Neng, “Design improvement for tor against low-cost traffic attack and low-resource routing attack,” in *Communications and Mobile Computing, 2009. CMC’09. WRI International Conference on*, vol. 3. IEEE, 2009, pp. 549–554.

[49] D. Meyer and F. T. Wien, “Support vector machines,” *R News*, vol. 1, no. 3, pp. 23–26, 2001.

[50] B. P. Roe, H.-J. Yang, J. Zhu, Y. Liu, I. Stancu, and G. McGregor, “Boosted decision trees as an alternative to artificial neural networks for particle identification,” *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 543, no. 2-3, pp. 577–584, 2005.

[51] Q. Wang, G. M. Garrity, J. M. Tiedje, and J. R. Cole, “Naive bayesian classifier for rapid assignment of rrna sequences into the new bacterial taxonomy,” *Applied and environmental microbiology*, vol. 73, no. 16, pp. 5261–5267, 2007.

[52] E. Balsa, C. Troncoso, and C. Diaz, “Ob-pws: Obfuscation-based private web search,” in *Security and Privacy (SP), 2012 IEEE Symposium on*. IEEE, 2012, pp. 491–505.

[53] D. Goldschlag, M. Reed, and P. Syverson, “Onion routing for anonymous and private internet connections,” NAVAL RESEARCH LAB WASHINGTON DC CENTER FOR HIGH ASSURANCE COMPUTING SYSTEMS . . . , Tech. Rep., 1999.

[54] R. Dingledine, N. Mathewson, and P. Syverson, “Tor: The second-generation onion router,” Naval Research Lab Washington DC, Tech. Rep., 2004.

[55] D. I. Wolinsky, H. Corrigan-Gibbs, B. Ford, and A. Johnson, “Dissent in numbers: Making strong anonymity scale,” in *Presented as part of the 10th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 12)*, 2012, pp. 179–182.

[56] S. B. Mokhtar, G. Berthou, A. Diarra, V. Quéma, and A. Shoker, “Rac: A freerider-resilient, scalable, anonymous communication protocol,” in *2013 IEEE 33rd International Conference on Distributed Computing Systems*, 2013, pp. 520–529.

[57] A. Petit, T. Cerqueus, S. B. Mokhtar, L. Brunie, and H. Kosch, “Peas: Private, efficient and accurate web search,” in *2015 IEEE Trustcom/BigDataSE/ISPA*, vol. 1. IEEE, 2015, pp. 571–580.

[58] R. Pires, D. Goltzsche, S. B. Mokhtar, S. Bouchenak, A. Boutet, P. Felber, R. Kapitza, M. Pasin, and V. Schiavoni, “Cyclosa: Decentralizing private web search through sgx-based browser extensions,” in *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2018, pp. 467–477.

[59] S. B. Mokhtar, A. Boutet, P. Felber, M. Pasin, R. Pires, and V. Schiavoni, “X-search: revisiting private web search using intel sgx,” in *Proceedings*

of the 18th ACM/IFIP/USENIX Middleware Conference. ACM, 2017, pp. 198–208.

[60] W. U. Ahmad, M. M. Rahman, and H. Wang, "Topic model based privacy protection in personalized web search," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 2016, pp. 1025–1028.

[61] W. U. Ahmad, K.-W. Chang, and H. Wang, "Intent-aware query obfuscation for privacy protection in personalized web search," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 2018, pp. 285–294.

[62] Y. Liu, X. Wu, M. Ching, S. Feng, J. A. Beynon, and D. Goldberg, "Determining search results using session based refinements," May 29 2018, uS Patent 9,984,151.

[63] F. J. Damerau, "A technique for computer detection and correction of spelling errors," *Communications of the ACM*, vol. 7, no. 3, pp. 171–176, 1964.
Proceedings of the 22nd international conference on World Wide Web. ACM, 2013, pp. 527–538.

[64] "Open Dictionary Project (ODP) ontology," https://curlie.org, 2019.

[65] P. Laperdrix, B. Baudry, and V. Mishra, "Fprandom: Randomizing core browser objects to break advanced device fingerprinting techniques," in *International Symposium on Engineering Secure Software and Systems*. Springer, 2017, pp. 97–114.

[66] S. Englehardt and A. Narayanan, "Online tracking: A 1-million-site measurement and analysis," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 1388–1401.

[67] Q. Wang, Y. Zhang, X. Lu, Z. Wang, Z. Qin, and K. Ren, "Real-time and spatio-temporal crowd-sourced social network data publishing with differential privacy," *IEEE Transactions on Dependable and Secure Computing*, no. 4, pp. 591–606, 2018.

[68] G. Acar, C. Eubank, S. Englehardt, M. Juarez, A. Narayanan, and C. Diaz, "The web never forgets: Persistent tracking mechanisms in the wild," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2014, pp. 674–689.

Appendix

.1 Features extracted by privious works

TABLE 9: Features extracted from the query events by Gervas et al. [27].

Feature	Description
TFQuery	Frequency of terms in the query
TFLandingPage	Frequency of terms in landin pages
NumQueryTerms	Number of terms in the query
NumQueryChar	Number of characters in the query
TFQueryAdult	Frequency of adult terms in the query
TFlandingPageAdult	Frequency of adult terms in the landing pages
NumSpellingErrors	Number of misspelled terms
TopicODP	Set of ODP categories of the top 8 result pages
CityQuery	Cities mentioned in the query
CountriesQuery	Countries mentioned in the query
TFURL	Keywords in URLs of the top 8 result pages
QueryTermPopularity	Frequency of the query terms in AOL dataset

.2 Evaluation of the obfuscation mechanisms in PEAS, X-Search and CYCLOSA

In this subsection, we evaluate the obfuscation mechanisms PEAS, X-Search and CYCLOSA. Note that PEAS [57], X-Search [59] and CYCLOSA [58] are proxy-based techniques combined with obfuscation methods. **Therefore, in this paper, we focus on evaluating the obfuscation mechanisms in PEAS, X-Search and CYCLOSA**, and the evaluation methodology is the same as in Section 5.

.2.1 Utility Evaluation

We present the average utility results in Table 11. First, we can observe that the query utility declines with the increasing amount of dummy queries. For example, in Figure 10, ODP3D in PEAS changes from 0.9555 (k=1) to 0.8946 (k=8), and ODP2 in CYCLOSA changes from 0.9788 (k=1) to 0.9469 (k=8). Second, adding more dummy queries to users' real data does not yield more changes on the web search result. The PWS_E increases with k in PEAS. However, in CYCLOSA, the performances is almost identical for k=1 (PWS_E= 8.2727) and k = 4 (PWS_E = 8.2609). The obfuscation mechanism in X-Search performs better than PEAS and CYCLOSA in preserving utility. This is because X-Search randomly aggregates the original query with k fake queries with logical OR operators. These fake queries come from users' past queries maintained by X-Search, which are similar to real queries in both the structure and the semantic space. For example, the ODP2D results in X-Search are similar for different k, and the average profile similarity in X-Search (0.9660, k=1) is higher than PEAS (0.8166, k=1) and CYCLOSA (0.8892, k=1). *The utility results of PEAS, X-Search and CYCLOSA further confirm our findings in Section 5.3.*

TABLE 10: Relative importance of the features in linkage function Importance^{USR} and Importance^{TMN} defined by Gervas et al. [27], where Importance^{USR} is the feature importance of the linkage function learned for an attack against an obfuscation mechanism using queries from another user, and Importance^{TMN} is the importance of the linkage function learned for the attack against an obfuscation using autogenerated queries (TMN).

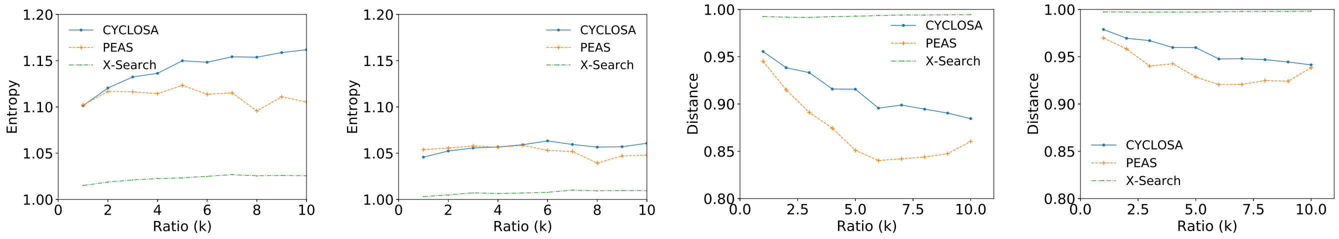
Feature relation	Importance^{USR}	Importance^{TMN}
Difference in query term weights	100	24
Difference of timestamps	56	100
Levenshtein distance of queries	32	20
Average ODP tree distance	31	7
Queries adult terms bool difference	26	15
JaccardC of query adult terms	20	16
Both queries have adult terms	19	17
Difference of clicked landing pages	18	5
Difference of query terms len	14	8
JaccardC of top 8 landing page URLs	12	22
Difference of query characters len	10	13
Difference of spelling errors	7	4
Both queries have spelling error	6	1
Same ODP level 2 category	4	no
Queries spelling error bool difference	4	no
JaccardC of landing page adult terms	4	4
JaccardC of the query terms	no	11

.2.2 Obfuscation vs Attack

We present the average accuracy results of attacks against PEAS, X-Search, and CYCLOSA in Table 12. *The results are consistent with our findings in Section 5.5*. First, in Table 12, the classification based attacks are more effective than the clustering based and the linkage based methods. For example, RD (0.9124) > K-means (0.8580) > SimAttack (0.1008) with k=9 in PEAS. Second, the performance of SimAttack against PEAS, X-Search and CYCLOSA is similar to the results in Section 5.5. Third, in Figure 10, we can observe that the attack accuracy results of PEAS, X-Search and CYCLOSA increase with the ratio k (k=dummy queries/real queries). These results again demonstrate that adding more fake queries to users' real data does not yield better privacy. Finally, compared with X-Search and CYCLOSA, PEAS is more vulnerable. This is because the fake queries generated by the co-occurrence matrix of PEAS have a gap with natural language and easier to be distinguished in the semantic space. Thus, in Figure 11, PEAS is the most vulnerable method.

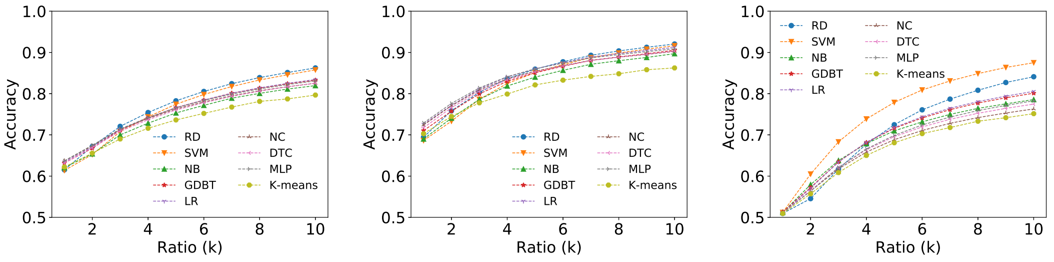
TABLE 11: Obfuscation vs Utility. Average utility metrics of obfuscated data vs original queries in PEAS, X-Search and CYCLOSA. *k* is the ratio of dummy queries to real queries.

	k	NQC	QE	ODP2E	ODP2D	ODP3E	ODP3D	PWS_J	PWS_E	UP
CYCLOSA	1	0.9994	1.0070	1.0457	**0.9788**	1.1013	**0.9555**	0.4593	8.2727	**0.8892**
CYCLOSA	4	0.9967	1.0156	1.0568	0.9598	1.1363	0.9158	0.4222	7.7143	0.8896
CYCLOSA	8	0.9914	1.0229	1.0566	**0.9469**	1.1538	**0.8946**	0.4593	8.2609	0.8884
PEAS	1	0.9995	1.0054	1.0539	**0.9698**	1.1020	**0.9452**	0.5083	**7.6000**	**0.8166**
PEAS	4	0.9936	1.0163	1.0565	0.9426	1.1144	0.8745	0.4593	8.6800	0.8176
PEAS	8	0.9811	1.0239	1.0393	**0.9248**	1.0958	**0.8440**	0.4083	9.4211	0.8179
X-Search	1	0.9992	1.0062	1.0031	**0.9974**	1.0150	0.9925	**0.5222**	7.9600	**0.9660**
X-Search	4	0.9901	1.0173	1.0064	**0.9972**	1.0226	0.9924	**0.5222**	7.6471	0.9673
X-Search	8	0.9714	1.0238	1.0094	**0.9979**	1.0256	0.9941	**0.5593**	7.4737	0.9677



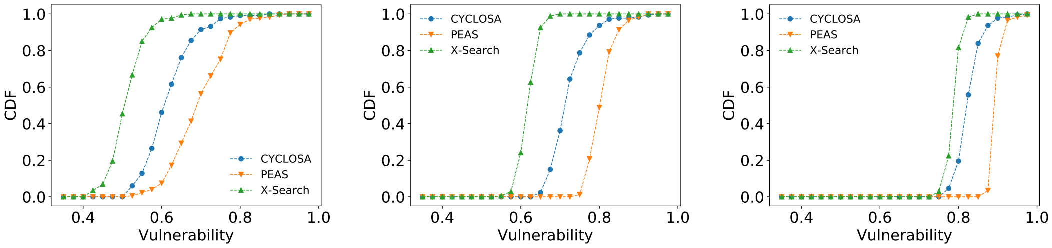
(a) ODP3E (b) ODP2E (c) ODP3D (d) ODP2D

Fig. 9: The average ODP utility metrics of PEAS, X-Search and CYCLOSA with different *k*. *k*=dummy queries/real queries.



(a) CYCLOSA (b) PEAS (c) X-Search

Fig. 10: The average accuracy of different attacks aganist PEAS, X-Search and CYCLOSA with the change of *k*. *k*=dummy queries/real queries.



(a) k = 1 (b) k = 3 (c) k = 7

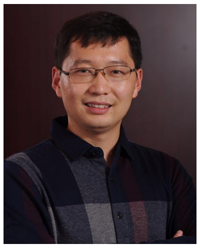
Fig. 11: CDF of vulnerability for obfuscation mechanisms in PEAS, X-Search and CYCLOSA with *k*=1, 3, 7. *k*=dummy queries/real queries.

TABLE 12: Obfucation vs Attack. The average accuracy of attacks against PEAS, X-Search and CYCLOSA.. k=dummy queries/real queries.

OB-PWS	*k*	RD	SVM	NB	GBDT	LR	NC	DTC	MLP	K-means	SimAttack
CYCLOSA	1	**0.6152**	0.6137	0.6205	0.6319	**0.6348**	0.6371	0.6315	0.6370	0.6221	**0.4536**
CYCLOSA	3	0.7206	0.7077	0.6999	0.7120	0.7128	0.7132	0.7090	0.7143	0.6901	**0.2971**
CYCLOSA	5	0.7826	0.7744	0.7532	0.7660	0.7660	0.7628	0.7603	0.7656	0.7362	**0.2280**
CYCLOSA	7	0.8242	0.8173	0.7890	0.8011	0.8012	0.7948	0.7934	0.7987	0.7674	**0.1844**
CYCLOSA	9	**0.8512**	0.8458	0.8112	0.8239	0.8250	0.8169	0.8165	0.8219	0.7872	**0.1577**
PEAS	1	**0.6943**	0.6862	0.6901	0.7105	0.7260	0.7238	0.7173	0.7290	**0.7029**	**0.4507**
PEAS	3	0.8032	0.7862	0.7862	0.7995	0.8114	0.8088	0.8052	0.8139	0.7777	0.2314
PEAS	5	0.8594	0.8510	0.8401	0.8506	0.8602	0.8544	0.8528	0.8601	0.8212	0.1590
PEAS	7	0.8930	0.8874	0.8712	0.8805	0.8888	0.8808	0.8804	0.8867	0.8418	0.1220
PEAS	9	**0.9124**	0.9080	0.8878	0.8968	0.9047	0.8951	0.8958	0.9018	**0.8580**	**0.1008**
X-Search	1	**0.5094**	**0.5119**	0.5119	0.5128	0.5110	0.5119	0.5115	0.5108	0.5093	**0.5052**
X-Search	3	0.6172	**0.6830**	0.6379	0.6347	0.6340	0.6176	0.6209	0.6210	0.6086	0.2918
X-Search	5	0.7246	**0.7788**	0.7098	0.7160	0.7178	0.6883	0.6956	0.6988	0.6809	0.2266
X-Search	7	0.7868	**0.8309**	0.7498	0.7611	0.7646	0.7280	0.7377	0.7432	0.7178	0.1937
X-Search	9	**0.8271**	**0.8635**	0.7753	0.7899	0.7940	0.7530	0.7643	0.7715	0.7417	**0.1729**

Chengkun Wei is currently a Ph.D. student of Computer Science and Technology at Zhejiang University. He received his M.S. degree of Computer Science and Technology from the Second Institute of China Aerospace Science. His current research interests primarily focus on privacy protection and blockchain.

Qinchen Gu (Member, IEEE) received his M.S. degree in electrical and computer engineering from Georgia Institute of Technology (Georgia Tech), Atlanta, GA, USA in 2015. He is currently a Ph.D. student in the School of Electrical and Computer Engineering at Georgia Tech, and a Graduate Research Assistant of the Communications Assurance and Performance (CAP) group. His research primarily focuses on the security for cyber-physical systems.



Shouling Ji is a ZJU 100-Young Professor in the College of Computer Science and Technology at Zhejiang University and a Research Faculty in the School of Electrical and Computer Engineering at Georgia Institute of Technology. He received a Ph.D. in Electrical and Computer Engineering from Georgia Institute of Technology, a Ph.D. in Computer Science from Georgia State University. His current research interests include AI Security, Data-driven Security, Privacy and Data Analytics. He is a member of IEEE and ACM and was the Membership Chair of the IEEE Student Branch at Georgia State (2012-2013).

Wenzhi Chen is a Professor in the College of Computer Science and Technology, Zhejiang University and the director of Information Technology Center of Zhejiang University, he used to be the Vice Dean of college of Computer Science and Technology. He received his Ph.D. in the college of Computer Science and Engineering at Zhejiang University. His current research interests include Embedded System and its Application, Computer Architecture, Computer System Software and Information Security. He is a member of IEEE, ACM and ACM Education Council.

Zonghui Wang is a senior engineer in the college of Computer Science and Engineering at Zhejiang University in Hangzhou, China. He received his Ph.D. in the college of Computer Science and Engineering at Zhejiang University in 2007. His current research interests focus on system security, privacy protection, cloud computing, and computer architecture.

Raheem Beyah is the Motorola Foundation Professor and Associate Chair in the School of Electrical and Computer Engineering at Georgia Tech, where he leads the Communications Assurance and Performance Group (CAP) and is a member of the Communications Systems Center (CSC). Prior to returning to Georgia Tech, Dr. Beyah was an Assistant Professor in the Department of Computer Science at Georgia State University, a research faculty member with the Georgia Tech CSC, and a consultant in Andersen Consulting's (now Accenture) Network Solutions Group. He received his Bachelor of Science in Electrical Engineering from North Carolina A&T State University in 1998. He received his Masters and Ph.D. in Electrical and Computer Engineering from Georgia Tech in 1999 and 2003, respectively. His research interests include network security, wireless networks, network traffic characterization and performance, and critical infrastructure security. He received the National Science Foundation CAREER award in 2009 and was selected for DARPA's Computer Science Study Panel in 2010. He is a member of AAAS and ASEE, is a lifetime member of NSBE, and is a senior member of ACM and IEEE.