Analysis of Heuristic Based Access Pattern Obfuscation

Huseyin Ulusoy
The University of Texas at Dallas
huseyin.ulusoy@utdallas.edu

Murat Kantarcioglu
The University of Texas at Dallas
muratk@utdallas.edu

Bhavani Thuraisingham
The University of Texas at Dallas
bhavani.thuraisingham@utdallas.edu

Ebru Celikel Cankaya
The University of Texas at Dallas
ebru.cankaya@utdallas.edu

Erman Pattuk
The University of Texas at Dallas
erman.pattuk@utdallas.edu

***Abstract*—As cloud computing becomes popular, the security and privacy issues emerge as important hindrances to more widespread adoption of cloud computing. In particular, outsourcing sensitive data to untrusted cloud service providers creates important security and regulatory compliance challenges. Encryption of the outsourced data has been introduced as an alternative to protect privacy and security. In the context of searchable symmetric encryption, many solutions have been proposed to perform efficient search on the encrypted outsourced data. Some of them achieve protecting privacy of outsourced data, but may disclose the access patterns (i.e., they disclose which data items are retrieved based on the query execution). Recently, it has been shown that such access pattern disclosures could be exploited even further to infer sensitive information about underlying data, even if the data is stored in encrypted form. To address the access pattern disclosures, oblivious RAM and heuristic based techniques are proposed. However, the overhead of oblivious RAM based solutions is too high in many cases, and the security and scalability of heuristic based techniques have not been carefully analyzed yet. In this paper, we provide the first framework to analyze and compare the security and efficiency of such heuristics (e.g., caching, fake data access, and data duplication). In addition, we provide extensive empirical analysis that yields important insights into how to use such heuristics effectively in practice; and we discuss how such heuristics can be combined to improve security and efficiency.**

I. Introduction

The rapid development of cloud computing has enabled us to outsource our data perpetually and cost effectively. Now, it is easier to process enormous amount of data by leveraging the huge computation power provided by the cloud computing. The data owners can also focus on their business logic without worrying about the underlying infrastructure. Still, the cloud computing paradigm is not flawless. Since the data is stored in an untrusted environment, security and privacy have emerged as arguably the most significant concerns that hinder widespread adoption of cloud computing [1]. Any disclosure of the sensitive information (e.g., health records, personal emails) in outsourced data may potentially cause problems ranging from monetary loss to business interruption. Hence, it is essential to protect outsourced sensitive data from any adversarial purposes.

Encryption of the outsourced data has been proposed as a solution to mitigate privacy and security concerns in cloud computing. However, generic encryption based solutions prevent data owners from processing outsourced data efficiently. To address this problem, searchable encryption solutions [2]–[9] have been introduced.

Available searchable encryption schemes offer distinct privacy and scalability guarantees. Among such schemes, the relatively efficient ones disclose access pattern (i.e., they disclose which data items are retrieved as a result of a query) [4]–[6], [8], [10], and others are too costly to be cost efficient for cloud environment [3]. Although disclosing access pattern seems innocuous, recently it has been shown that access patterns can be used to infer sensitive information related to outsourced data [11]. In addition, some experts raised concerns about disclosure of "activity pattern" that could be exploited to create side-channels and inference attacks [1].

One of the solutions that could be used to hide access pattern is Oblivious RAM [12]. Oblivious RAM provides perfect privacy by completely hiding the access pattern, but it has a logarithmic overhead for each data access. Therefore, it is usually not applicable to cloud environments due to its huge additional cost for large datasets. Because of inapplicability of Oblivious RAM solutions to many scenarios, some heuristics have been proposed to hide access patterns such as caching and fake data access [13]–[15]. To our knowledge, these heuristics have not been analyzed with respect to the security and scalability that they provide in a cloud setting. In this paper, we provide a comprehensive framework to analyze and compare such heuristics with respect to security and efficiency. Then, we show how to parameterize such heuristics to achieve desired efficiency versus security trade-off. Additionally, we show that individual heuristic based access pattern obfuscation techniques do not provide effective security guarantees. To address this shortcoming, we propose the hybrid policies that combine multiple heuristic based obfuscation techniques (i.e., caching, fake data access, and redundant data storage) to provide better security and efficiency trade-offs.

The organization of the paper is as follows: Section II reviews related work. Section III introduces the searchable encryption model. Section IV analyzes the heuristics. Section V presents the empirical evaluations and Section VI concludes the paper.

COLLABORATECOM 2013, October 20-23, Austin, United States

DOI 10.4108/icst.collaboratecom.2013.254199

II. Related Work

Searchable encryption has been a popular research topic, where many protocols and security definitions have been proposed over the years. The proposed protocols can be put into two main categories: the protocols allowing practical search and protocols hiding access pattern. The first practical searchable encryption protocol was proposed by Song et al. [8], where the documents and the search index are encrypted with a symmetric key. This protocol is called *Searchable Symmetric Encryption (SSE)*. Later, Goh formalized the security requirements of SSE in [6] and introduced *Z-IDX* index, which is secure against adaptive chosen keyword attack (IND-CKA). Simultaneously, Chang et al. introduced another security definition for SSE based on simulations in [4], which is slightly stronger than the definition in [6]. The concept of SSE model was extended to involve multiple users by Park et al. in [16]. Curtmola et al. expanded the security definition of SSE to secure against adaptive adversaries in multiple users setup in [5]. Golle et al. enabled the secure conjunctive keyword search with a scheme based on decisional Diffie-Hellman assumption in [7]. Later, for conjunctive search, a more efficient scheme was proposed in [9]. Moreover, a public key scheme was proposed by Boneh and Waters in [3] with conjunctive, subset, and range queries over encrypted data. Although all these SSE schemes are practical in a cloud setting, they do not hide the access pattern [5]. As demonstrated in [11], the access patterns can be combined with some background information to infer sensitive information about the underlying data.

Oblivious RAM model was first presented by Goldreich and Ostrovsky in [12] in which the users can securely search over encrypted data without revealing their access patterns. Although the model has been improved by many studies [17]–[19], it incurs a logarithmic overhead in the number of outsourced data for each single data retrieval, which makes it very costly for big data. Moreover, Boneh et al. [20] proposed a searchable encryption model that hides access patterns based on public key encryption. Although these schemes provide provable security against access pattern attacks, they are impractical in the cloud setting. In addition to these schemes, di Vimercati et al. proposed a B^+-tree based shuffle index in [13] while incurring $O(log(n))$ additional accesses for each data retrieval, where n is the dataset size. Furthermore, some heuristics (i.e., caching, fake data access) have been proposed to hide access patterns without sacrificing efficiency in [13]–[15]. To our knowledge, this is the first paper that analyzes the security and efficiency of heuristic based access pattern obfuscation techniques.

III. Model

A typical searchable encryption scenario may be described as follows: Alice has a set of documents $\mathcal{D} = \{d_1, d_2, \ldots, d_n\}$ containing a set of keywords $\mathcal{Q} = \{q_1, q_2, \ldots, q_m\}$. She wants to outsource $\mathcal{D}$ into a public cloud, provided by Bob, while preserving the efficient search capabilities over outsourced data. To this end, she creates an encrypted index M, and sends it to Bob along with the encrypted documents. The encrypted index is an $m \times n$ binary matrix, where the keywords correspond to the rows and the document ids correspond to the columns. In other words, M_{ij} is set to 1 if q_i exists in d_j, and M_{ij} is 0 otherwise. To retrieve the set of documents that contain $q \in \mathcal{Q}$, Alice generates a trapdoor from q and sends it to Bob. When a search is made, Bob checks M, and returns the matching encrypted document ids to Alice. She decrypts them and retrieves the documents from Bob. Note that this model is general enough to simulate an arbitrary searchable encryption model as discussed in [11].

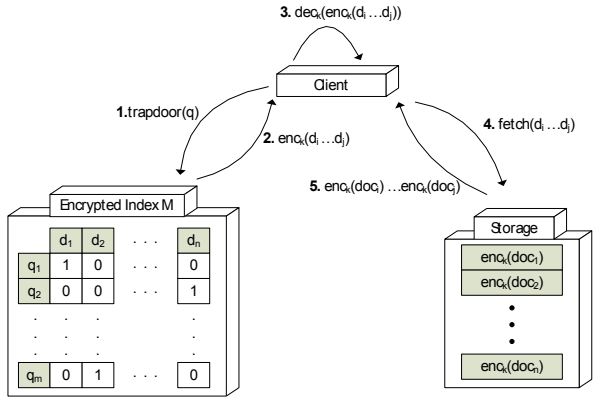


Fig. 1. Model

Fig. 1 shows the three entities of the overall structure of searchable encryption: A client, the binary matrix M, and the storage. We assume that the client is a trusted entity, while M, the storage and the network are untrusted. Thus, an adversary is assumed to observe all information stored in or transmitted through the untrusted components. To prevent attacks, it is important to determine the information that an adversary can have. The plaintext of the outsourced documents and trapdoors cannot be learned by the adversary if a secure pseudo-random permutation is adopted. Yet, the adversary can still gather statistical information about the access patterns of the queries and documents. Let $P(q)$ denote the probability of querying the particular keyword $q \in \mathcal{Q}$, and $P(d)$ denote the probability of retrieving the particular document $d \in \mathcal{D}$. Then, $P(d|q)$ denotes the probability that the retrieved document is $d \in \mathcal{D}$ when $q \in \mathcal{Q}$ is the searched keyword. It is clear that $P(q)$ and $P(d)$ can be monitored by adversary that has intruded M and the storage, respectively. Then, $P(d|q)$ can also be inferred by means of simultaneous observation of M and the storage. Note that once $P(d|q)$ is inferred, the decrypted matrix of M can also be obtained. A recent attack by Islam et al. [11] shows that the adversary knowledge $P(d|q)$ and some background knowledge are enough to infer most of the information in the underlying data. Note that by using current searchable encryption models, it is not possible to hide $P(q)$ unless a probabilistic encrypted index is employed. Yet, as discussed in [12], $P(d)$ and $P(d|q)$ can be hidden if $P(d)$ and $P(q)$ are independent.

Definition 3.1: (Oblivious Search) A search in a searchable encryption model is oblivious if and only if the probability distributions $P(d)$ and $P(q)$ are independent, where $\forall d \in \mathcal{D}$ and $\forall q \in \mathcal{Q}$

Definition 3.2: (Search Indistinguishability) The documents are indistinguishable if each document's retrieval probability is equal for a given arbitrary search query.

$$\forall d_i, d_k \in D \text{ and } \forall q \in Q,\ P(d_i \mid q) = P(d_k \mid q), \text{ where } d_i \neq d_k$$

Definition 3.2 leads us to two results:

Lemma 3.1: Given an arbitrary keyword search satisfying search indistinguishability, each document's retrieval probability is uniform and equal to $\frac{1}{n}$.

Proof:

$$\begin{aligned} 1 &= \sum_{d_i \in \mathcal{D}} P(d_i \mid q) = n\ P(d_i \mid q) && \text{by definition 3.2} \\ \frac{1}{n} &= P(d_i \mid q) \end{aligned}$$

■

Lemma 3.2: If all keyword searches satisfy search indistinguishability, each document's retrieval probability is uniform and equal to $\frac{1}{n}$.

Proof:

$$\begin{aligned} 1 &= \sum_{q \in Q} P(q) = \sum_{q \in Q} \sum_{d \in \mathcal{D}} P(q, d) \\ &= \sum_{q \in Q} \sum_{d \in \mathcal{D}} P(d \mid q) P(q) = \sum_{q \in Q} P(q) \sum_{d \in \mathcal{D}} P(d \mid q) \\ &= \sum_{q \in Q} P(q) n P(d \mid q) = n \sum_{q \in Q} P(d \mid q) P(q) && \text{by definition 3.2} \\ \frac{1}{n} &= P(d) \end{aligned}$$

■

Theorem 3.3: If all keyword searches satisfy the search indistinguishability (Def. 3.2), the search is oblivious by definition 3.1.

Proof: By lemma 3.1 and 3.2, $\forall q \in Q$ *and* $\forall d \in \mathcal{D}$, $P(q)$ and $P(d)$ are independent.

$$P(d \mid q) = P(d) = \frac{1}{n}$$

■

Given an arbitrary search keyword q_j, let x_j denote the number of documents where q_j exists, and $1 \leq x_j \leq n$. Then, the following holds:

$$\forall d_i \in \mathcal{D} \text{ and } \forall q_j \in Q,$$
$$P(d_i \mid q_j) = \begin{cases} \frac{1}{x_j} & q_j \in d_i \\ 0 & q_j \notin d_i \end{cases}, \textit{where } P(q_j) \neq 0 \textit{ and where } 1 \leq x_j \leq n$$

Let $P'(d_i|q_j)$ denote the probability that the retrieved document is $d_i \in \mathcal{D}$ if the query $q_j \in Q$ is searched in an oblivious manner (see def. 3.2). We claim that the total variation distance between $P(d_i \mid q_j)$ and $P'(d_i \mid q_j)$ can be used to measure the indistinguishability of the model.

$$\textit{By lemma } 3.1, \quad \forall d_i \in \mathcal{D} \textit{ and } \forall q_j \in Q, \quad P'(d_i \mid q_j) = \frac{1}{n}$$

Definition 3.3: (Total variation distance of probability measures [21]) Let μ *and* η be two probability measures over a finite set Ω. The total variation distance between μ *and* η is the largest possible $L_1 - distance$ between the two probability measures:

$$\begin{aligned} \|\mu - \eta\|_{TV} &= \sup_{x \in \Omega} |\mu(x) - \eta(x)| \\ &= \frac{1}{2} \sum_{x \in \Omega} |\mu(x) - \eta(x)| \end{aligned} \tag{1}$$

Definition 3.4: (ϵ − distance) An encrypted search satisfies $\epsilon - distance$ if the maximum L_1 distance between $P(d_i|q_j)$ *and* $P'(d_i|q_j)$ is less than a constant ϵ for all documents in $\mathcal{D}$ and all keywords in Q. Then, the total variation distance (Def. 3.3) can be used to measure the $\epsilon - distance$.

$$\sup_{d_i \in \mathcal{D}, q_j \in Q} |P(d_i|q_j) - P'(d_i|q_j)| \leq \epsilon \tag{2}$$

The $\epsilon - distance$ of an arbitrary searchable encryption model is as follow:

$$\|P(d_i \mid q_j) - P'(d_i|q_j)\|_{TV} = \tag{3}$$
$$\sup_{q_j \in Q} \{ \frac{1}{2} \sum_{d_i \in \mathcal{D}} |P(d_i \mid q_j) - P'(d_i|q_j)| \} =$$
$$\sup_{q_j \in Q} \{ \frac{1}{2} \sum_{q_j \in d_i, d_i \in \mathcal{D}} |P(d_i|q_j) - P'(d_i|q_j)| + \frac{1}{2} \sum_{q_j \notin d_i, d_i \in \mathcal{D}} |P(d_i|q_j) - P'(d_i|q_j)| \} =$$
$$\sup_{1 \leq x_j \leq n} \{ \frac{1}{2} x_j (\frac{1}{x_j} - \frac{1}{n}) + \frac{1}{2}(n - x_j)\frac{1}{n} \} =$$
$$\sup_{1 \leq x_j \leq n} \{ 1 - \frac{x_j}{n} \} = 1 - \frac{1}{n} = \approx 1 \tag{4}$$

Smaller values of $\epsilon \in [0, 1]$ implies less access pattern disclosure. Furthermore, when a model satisfies $\epsilon = 0$, it completely hides the access pattern.

IV. Heuristics

The $\epsilon - distance$ given in Eq. 3 can be reduced by adopting some heuristics such as caching, fake document access and document duplication. In this section, we analyze the performance of these techniques with respect to their $\epsilon - distance$ and costs.

A. Caching

In the context of searchable encryption, caching is the process of storing a subset of the entire data in a local storage, so that a faster response is provided whenever the requested data is in the cache. It helps to reduce the overall monetary cost [22], by reducing the network traffic between the client and the storage. Moreover, as shown in Prop. 4.1, it has an improving effect on the security guarantees of the model. We will analyze two basic cache policies, the random and locality caching, with respect to $\epsilon - distance$ metric and cache hit rates.

1) Random Caching: In random caching policy, the client chooses uniformly random k documents to be stored in the cache for each query.

Proposition 4.1: When searchable encryption model employs random caching policy, $\epsilon - distance$ of the model is decreased approximately by $\frac{k}{2n}$.

Proof: Let $HR(d)$ denote the probability that the document d is in the cache, which is also called the *cache hit rate*. Then, $HR(d)$ can be written as follows:

$$\forall d \in D,\ HR(d) = \sum_{i=0}^{k-1} \frac{1}{n-i} \geq \sum_{i=0}^{k-1} \frac{1}{n} = \frac{k}{n}\ ,\ \textit{where } k < n$$

When random caching is employed by the model, let $P_{cache}(d_i \mid q_j)$ denote the probability that a retrieved document is d_i given the keyword q_j, and $x_{j,c}$ denote the expected number of documents retrieved for the keyword search q_j. Then, the following holds:

$$\forall d_i \in D \textit{ and } \forall q_j \in Q,\ x_{j,c} = x_j(1 - HR(d_i))$$
$$P_{cache}(d_i \mid q_j) = \begin{cases} \frac{1}{x_j} & q_j \in d_i \\ 0 & q_j \notin d_i \end{cases}$$

Then, the $\epsilon - distance$ of model after random caching can be written as follows:

$$\begin{aligned}
&\|P_{cache}(d_i \mid q_j) - P'(d_i \mid q_j)\|_{TV} = \\
&\sup_{q_j \in Q}\{\ \frac{1}{2}\sum_{d_i \in D}|P_{cache}(d_i|q_j) - P'(d_i|q_j)|\ \} = \\
&\sup_{q_j \in Q}\{\ \frac{1}{2}\sum_{q_j \in d_i,\ d_i \in D}(P_{cache}(d_i|q_j) - P'(d_i|q_j)) \\
&\qquad + \frac{1}{2}\sum_{q_j \notin d_i,\ d_i \in D}(P'(d_i|q_j) - P_{cache}(d_i|q_j))\ \} = \\
&\sup_{q_j \in Q}\{\ \frac{1}{2}\{\ x_{j,c}(\frac{1}{x_j} - \frac{1}{n}) + (n - x_{j,c})(\frac{1}{n} - 0)\ \}\ \} = \\
&\sup_{1 \le x_j \le n}\{\ 1 - \frac{x_j}{n} - \frac{k}{2n} + \frac{x_j k}{n^2}\ \} = 1 - \frac{1}{n} - \frac{k}{2n} + \frac{k}{n^2} \approx 1 - \frac{k}{2n}
\end{aligned}$$

■

The cost of random caching policy is to store k data items in the client and to process at most k additional document accesses for each query. Because, if the documents chosen for caching have not already been fetched by the previous query, they will be retrieved from the storage.

2) Locality Caching: The locality caching policy is introduced to maximize the cache hit rate by exploiting the locality of data. To this end, the client keeps track of the document access frequencies and stores the most frequently accessed ones in the local computer. The locality principle of the data [23] ensures better cache hit rate than the random selection. Yet, its effect on $\epsilon - distance$ is limited. Because, all data items cannot find a chance to be cached, but only the frequently accessed ones do. When the cache eventually finds an equilibrium state, it holds the same data items for each query.

The cost of locality caching policy is to store k data items in the cache. It does not need to process additional document accesses like random caching, because the documents can be chosen from among the documents that are already fetched for previous queries.

B. Random Fake Access

The random fake access is a heuristic, where the documents are retrieved with a uniform random probability α. More formally, given an arbitrary query q, let α be the probability of fake access to $d_i \in \mathcal{D}$ when $P(d_i|q) = 0$.

Proposition 4.2: If searchable encryption employs random fake access policy with a probability α, $\epsilon - distance$ of the model is decreased approximately by $\sqrt{\alpha}$.

Proof: When the random fake access policy is employed by the model, let $P_{fake}(d_i \mid q_j)$ denote the probability that a retrieved document is d_i given a keyword q_j, and $x_{j,f}$ denote the expected number of retrieved documents by q_j. Then, the following holds:

$$\forall d_i \in \mathcal{D}\ and\ \forall q_j \in Q,\ \ x_{j,f} = x_j + (n - x_j)\alpha$$

$$P_{fake}(d_i \mid q_j) = \begin{cases} \frac{1}{x_{j,f}} & q_j \in d_i \\ \frac{\alpha}{x_{j,f}} & q_j \notin d_i \end{cases}$$

After the random fake access policy is employed, the $\epsilon - distance$ of model can be written as follows:

$$\begin{aligned}
&\|P_{fake}(d_i \mid q_j) - P'(d_i \mid q_j)\|_{TV} = &(5)\\
&\sup_{q_j \in Q}\{\ \frac{1}{2}\sum_{d_i \in D}|P_{fake}(d_i|q_j) - P'(d_i|q_j)|\ \} = \\
&\sup_{q_j \in Q}\{\ \frac{1}{2}\sum_{q_j \in d_i,\ d_i \in D}(P_{fake}(d_i|q_j) - P'(d_i|q_j)) &(6)\\
&+ \frac{1}{2}\sum_{q_j \notin d_i,\ d_i \in D}(P'(d_i|q_j) - P_{fake}(d_i|q_j))\ \} = \\
&\sup_{q_j \in Q}\{\ \frac{1}{2}\{\ x_{j,f}(\frac{1}{x_{j,f}} - \frac{1}{n}) + (n - x_{j,f})(\frac{1}{n} - \frac{\alpha}{x_{j,f}})\ \}\ \} = \\
&\sup_{q_j \in Q}\{\ 1 - \frac{\alpha}{2} - \frac{x_j}{n}(1 - \alpha) - \frac{n\alpha}{2(n\alpha + x_j(1-\alpha))}\ \} \Rightarrow &(7)\\
&\frac{\partial}{\partial x_j}\left(1 - \frac{\alpha}{2} - \frac{x_j}{n}(1-\alpha) - \frac{n\alpha}{2(n\alpha + x_j(1-\alpha))}\right) \Rightarrow \\
&x_j = \frac{n(\sqrt{\frac{\alpha}{2}} - \alpha)}{1 - \alpha},\ where \quad \alpha < 1 \Rightarrow \\
&\|P_{fake}(d_i \mid q_j) - P'(d_i \mid q_j)\|_{TV} \le 1 + \frac{\alpha}{2} - \sqrt{2\alpha} \approx 1 - \sqrt{\alpha}
\end{aligned}$$

Note that the Eq. 5 can be calculated by taking derivative with respect to x_j. ■

The estimated cost of random fake access policy is to process $\alpha(n - x_j)$ additional accesses when q_j is queried. Note that, as the number of documents increases, the cost of random fake access policy increases as well.

C. Document Duplication

Document duplication is another heuristic we analyze, where multiple copies of documents are stored redundantly in the storage. When a keyword search needs a document, one of the copies of the document is retrieved randomly. Since the selection process is uniformly at random, $P(d_i|q_j)$ decreases with respect to the number of copies.

Proposition 4.3: In searchable encryption model, if c copies are stored for each document $d_i \in \mathcal{D}$, $\epsilon - distance$ of the model decreases with respect to c.

Proof:

Let Φ denote a function for the duplication process, where $\Phi : \mathcal{D} \rightarrow \mathcal{D}^c$, and $\mathcal{D}'$ denote the set of documents, where $\mathcal{D}' = \bigcup_{d_i \in \mathcal{D}} \Phi(d_i)$ and $|\Phi(d_i)| = c$

The document $d \in \Phi(d_i)$ is selected uniformly at random when a searched keyword q_j exists in d. Let $P_{dup}(d_i \mid q_j)$ denote the probability that a retrieved document is one of c copies of document d_i, given the keyword q_j, when the model employs the document duplication policy. Then, the following holds:

$$\forall d_i \in \mathcal{D}'\ and\ \forall q_j \in Q,$$

$$P_{dup}(d_i \mid q_j) = \begin{cases} \frac{1}{x_j{}^c} & q_j \in d_i \\ 0 & q_j \notin d_i \end{cases}$$

When document duplication is employed, the $\epsilon - distance$ of model can be written as follows:

$$\|P_{dup}(d_i \mid q_j) - P'(d_i \mid q_j)\|_{TV} =$$
$$\sup_{q_j \in Q}\{\ \frac{1}{2}\sum_{d_i \in D}|P_{dup}(d_i|q_j) - P'(d_i|q_j)|\ \} =$$
$$\sup_{q_j \in Q}\{\ \frac{1}{2}\sum_{q_j \in d_i,\ d_i \in D}(P_{dup}(d_i|q_j) - P'(d_i|q_j)) + \frac{1}{2}\sum_{q_j \notin d_i,\ d_i \in D}(P'(d_i|q_j) - P_{dup}(d_i|q_j))\ \} =$$
$$\sup_{q_j \in Q}\{\ \frac{1}{2}\{\ x_j(\frac{1}{x_j c} - \frac{1}{cn}) + (n - x_j)(\frac{1}{cn} - 0)\ \}\ \} =$$
$$\frac{1}{c} sup_{1 \le x_j \le n}\{\ 1 - \frac{x_j}{n}\ \} = \frac{1}{c}(\ 1 - \frac{1}{n}\ ) \quad , \ where\ 1 \le c$$

■

The cost of document duplication policy is to store extra $(c-1)n$ documents, $c > 1$, in the storage.

Table I summarizes the $\epsilon - distance$ and cost of introduced policies.

Policy	$\epsilon - distance$	cost
Random Caching	$1 - \frac{k}{2n}$	k (data access/query)
Locality Caching	$1 - \frac{1}{n}$	0
Random Fake Access	$1 - \sqrt{\alpha}$	$\alpha(n - x_j)$ (data access/query)
Document Duplication	$\frac{1}{c}(1 - \frac{1}{n})$	$(c-1)n$ (redundant document storage)

TABLE I. HEURISTIC COMPARISON

D. Hybrid Policies

The heuristics introduced above can be combined to achieve lower $\epsilon - distance$ values while dividing the total cost into different policies. In theory, any combination of these three heuristics can be used, but we analyze two of these hybrid policies below, in terms of the gain with respect to $\epsilon - distance$ and incurred costs.

1) Hybrid Policy I: Random Caching + Random Fake Access: In hybrid policy I, the random caching (Sec. IV-A1) is utilized along with the random fake access (Sec. IV-B). Suppose that the client can store k out of n documents in the local storage and uses the random fake access policy with probability α.

Proposition 4.4: If a searchable encryption model employs *hybrid policy I* with k documents in the cache and fake access probability α, $\epsilon - distance$ of the model is decreased approximately by $\frac{k}{2n} + \sqrt{\alpha}$.

Proof: When the model employs the hybrid policy I, let $P_{hybrid_I}(d_i \mid q_j)$ denote the probability that a retrieved document is d_i given the keyword q_j, and $x_{j,h}$ denote the expected number of retrieved documents by q_j. Then, the following holds:

$$\forall d_i \in \mathcal{D}\ and\ \forall q_j \in Q,\ x_{j,h} = x_j - \frac{x_j k}{n} + (n - x_j)\alpha$$
$$P_{hybrid_I}(d_i \mid q_j) = \begin{cases} \frac{1-\frac{k}{n}}{x_{j,h}} & q_j \in d_i \\ \frac{\alpha}{x_{j,h}} & q_j \notin d_i \end{cases}$$

When *hybrid policy I* is employed, the $\epsilon - distance$ of an arbitrary query q_j can be written as follows:

$$\|P_{hybrid_I}(d_i \mid q_j) - P'(d_i \mid q_j)\|_{TV} = \quad (8)$$
$$\sup_{q_j \in Q}\{\ \frac{1}{2}\sum_{d_i \in D}|P_{hybrid_I}(d_i|q_j) - P'(d_i|q_j)|\ \} =$$
$$\sup_{q_j \in Q}\{\ \frac{1}{2}\sum_{q_j \in d_i,\ d_i \in D}(P_{hybrid_I}(d_i|q_j) - P'(d_i|q_j)) + \frac{1}{2}\sum_{q_j \notin d_i,\ d_i \in D}(P'(d_i|q_j) - P_{hybrid_I}(d_i|q_j))\ \} = \quad (9)$$
$$\sup_{q_j \in Q}\{\ \frac{1}{2}\{\ x_{j,h}(\frac{1 - \frac{k}{n}}{x_{j,h}} - \frac{1}{n}) + (n - x_{j,h})(\frac{1}{n} - \frac{\alpha}{x_{j,h}})\ \}\ \} =$$
$$\sup_{q_j \in Q}\{\ 1 - \frac{k}{2n} - \frac{\alpha}{2} - \frac{x_j}{n}(1 - \frac{k}{n} - \alpha) - \frac{n\alpha}{2(n\alpha + x_j(1 - \frac{k}{n} - \alpha))}\ \} \Rightarrow \quad (10)$$
$$0 = \frac{\partial}{\partial x_j}\left(1 - \frac{k}{2n} - \frac{\alpha}{2} - \frac{x_j}{n}(1 - \frac{k}{n} - \alpha) - \frac{n\alpha}{2(n\alpha + x_j(1 - \frac{k}{n} - \alpha))}\right) \Rightarrow$$
$$x_j = \frac{n(\sqrt{\frac{\alpha}{2}} - \alpha)}{1 - \frac{k}{n} - \alpha},\ where \quad \alpha < 1 \Rightarrow$$
$$\|P_{fake}(d_i \mid q_j) - P'(d_i \mid q_j)\|_{TV} \le 1 - \frac{k}{2n} + \frac{\alpha}{2} - \sqrt{2\alpha} \approx 1 - \frac{k}{2n} - \sqrt{\alpha}$$

Note that Eq. 8 can be calculated by taking derivative with respect to x_j. ■

If $\alpha n > k$, the client can choose k documents for caching from the documents, retrieved by the random fake access policy in the previous query. Therefore, the additional document access cost of random caching is eliminated. Total cost of the policy is to store k documents in the local machine and process αn fake document accesses.

2) Hybrid Policy II: Random Caching + Random Fake Access + Document Duplication: In hybrid policy II, document duplication is added to hybrid policy I. Suppose that the client can store k out of n documents in her local machine, it employs the random fake access policy with the probability α and document duplication policy with c copies.

Proposition 4.5: If a searchable encryption model employs *hybrid policy II* with k documents in the cache, fake access probability α and c copies in the storage, $\epsilon - distance$ of the model is decreased with respect to k, α and c.

Proof: Let $P_{hybrid_{II}}(d_i \mid q_j)$ denote the probability that a retrieved document is d_i, given the keyword q_j, when the model employs the hybrid policy II. Then, the following holds:

$$\forall d_i \in D'\ and\ \forall q_j \in Q,\ P_{hybrid_{II}}(d_i \mid q_j) = \begin{cases} \frac{1-\frac{k}{n}}{x_{j,h}c} & q_j \in d_i \\ \frac{\alpha}{x_j^h c} & q_j \notin d_i \end{cases}$$

When *hybrid policy II* is employed, the $\epsilon - distance$ of an arbitrary query q_j can be written as follows:

$$\|P_{hybrid_{II}}(d_i \mid q_j) - P'(d_i \mid q_j)\|_{TV} = \tag{11}$$

$$\sup_{q_j \in Q}\{\ \frac{1}{2}\sum_{d_i \in D} |P_{hybrid_{II}}(d_i|q_j) - P'(d_i|q_j)|\ \} =$$

$$\sup_{q_j \in Q}\{\ \frac{1}{2}\sum_{q_j \in d_i,\ d_i \in D} (P_{hybrid_{II}}(d_i|q_j) - P'(d_i|q_j)) + \tag{12}$$

$$\frac{1}{2}\sum_{q_j \notin d_i,\ d_i \in D} (P'(d_i|q_j) - P_{hybrid_{II}}(d_i|q_j))\ \} =$$

$$\sup_{q_j \in Q}\{\ \frac{1}{2}\{\ x_j^h(\frac{1-\frac{k}{n}}{x_{j,h}c} - \frac{1}{nc}) + (n - x_{j,h})(\frac{1}{nc} - \frac{\alpha}{x_{j,h}c})\ \}\ \} =$$

$$\frac{1}{c}\sup_{q_j \in Q}\{\ 1 - \frac{k}{2n} - \frac{\alpha}{2} - \frac{x_j}{n}(1 - \frac{k}{n} - \alpha) - \frac{n\alpha}{2(n\alpha + x_j(1 - \frac{k}{n} - \alpha))}\ \} \Rightarrow \tag{13}$$

$$0 = \frac{\partial}{\partial x_j}\left(1 - \frac{k}{2n} - \frac{\alpha}{2} - \frac{x_j}{n}(1 - \frac{k}{n} - \alpha) - \frac{n\alpha}{2(n\alpha + x_j(1 - \frac{k}{n} - \alpha))}\right) \Rightarrow \tag{14}$$

$$x_j = \frac{n(\sqrt{\frac{\alpha}{2}} - \alpha)}{1 - \frac{k}{n} - \alpha},\ where \quad \alpha < 1$$

$$\|P_{fake}(d_i \mid q_j) - P'(d_i \mid q_j)\|_{TV} \leq \frac{1}{c}\{1 - \frac{k}{2n} + \frac{\alpha}{2} - \sqrt{2\alpha}\} \approx \frac{1}{c}\{1 - \frac{k}{2n} - \sqrt{\alpha}\}$$

Note that Eq. 11 can be calculated by taking derivative with respect to x_j. ■

Since the document duplication policy increases the size of corpus by c times, the number of fake data accesses also increases by c times. Therefore, total cost of the policy is to store k documents in the local storage, process αcn fake document accesses, and store $(c-1)n$ redundant documents in the storage.

V. Experimental Analysis

In this section, we present empirical results and analyses of the proposed heuristics on the searchable encryption model. To our knowledge, this is the first extensive empirical study of access pattern obfuscation techniques such as caching,fake data access and data duplication. The analysis presents comparison of proposed heuristics, with respect to scalability performance, under different security constraints. Our results suggest that hybrid approaches can lower $\epsilon - distance$ to desired values for acceptable costs although the individually usage of heuristics is not very effective.

A. Experiment setup

The experiment is conducted on a realistic scenario. In this scenario, a real world dataset (ENRON) [24] is uploaded to our free account in Dropbox file hosting service [25]. Since Dropbox does not provide computation power, the encrypted index is deployed into a private server that we manage. The server is an 8 cores Intel(R) Xeon(R) D Linux machine clocked at 2.50GHz with 32 GB RAM. Our personal desktop, which is an Intel(R) Pentium(R) D Windows 7 machine clocked at 2.80GHz with 4.00 GB RAM, is used as the client. The client and Dropbox are connected (via Internet) through the REST API of Dropbox. The client and the server are connected via the UT Dallas local area network (LAN).

The ENRON dataset [24], which is a well studied email log, is employed as our corpus. The dataset is composed of many folders from approximately 150 users. We randomly choose 10000 documents[1] from the inbox folders as our document set D, in which there are totally 75147 distinct keywords. Since the documents have some email related meta data at the first few lines, we filtered these lines from the documents. Then, the words in the documents are stemmed for improving the search process. At the end of this process, the documents become a composition of stemmed words. Moreover, the documents are split into tree structured folders because of Dropbox's limit on the number of files that can be stored in a folder.

Although any query distribution can be incorporated in the experiment, we assume that more common words have more chance to be searched. Thus, 1000 queries are generated by randomly selecting keywords from the documents. Moreover, 120 stop words that do not make sense for a query keyword (i.e., "the","to","a") are omitted in the search queries.

In the experiment, various security scenarios are explored by assigning different $\epsilon - distances$ (i.e., 0.001, 0.01, 0.1, 0.2, 0.3) constraints. We argue that a searchable encryption model that satisfies $0.001 - distance$ definition is more secure than a model satisfying $0.3 - distance$, because it is closer to the definition 3.2 which is oblivious by theorem 3.3. Then, the cost of hybrid policies are presented as number of fake document accesses for each document retrieval, and the fake document access cost of ORAM is compared with it.

B. Results

If only the random caching policy is employed in the model, the size of cache to achieve $\epsilon - distance$ constraint can be calculated by means of proposition 4.1:

$$k \geq 2n(1-\epsilon), \qquad lim_{\epsilon \to 0} k = 2n - 2$$

"$\lim_{\epsilon \to 0}\ k = 2n-2$" renders the cache meaningless, because if the client can hold all documents in the local machine, there is no need for searchable encryption model in the cloud setting.

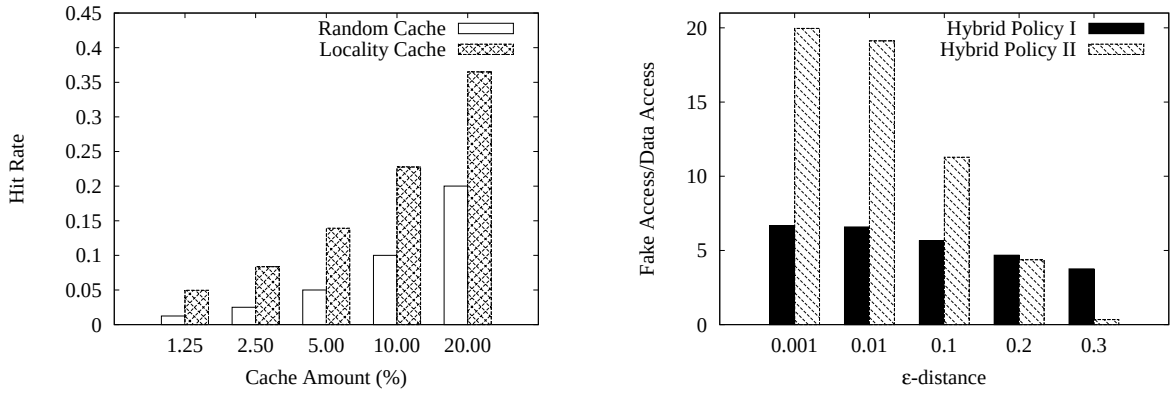
The locality caching cannot be used to reduce $\epsilon - distance$ as explained in section IV-A2. However, its hit rate is superior to random caching (see Fig.2(a)) especially when cache size is relatively small. When 1.25% of the documents are cached, the locality caching ($HR(d_i) = 0.0496$) is four times better than the random caching($HR(d_i) = 0.0125$). Therefore, a combination of small amount of locality caching and random caching will provide a good balance between the security and scalability gains. For instance, if 5% locality caching is employed along with 15% random caching, its hit rate is 0.09 higher than 20% random caching, but the reduction in $\epsilon - distance$ is only 0.05 smaller than 20% random caching.

If only the fake document access policy is used in the model, the amount of access probability α to succeed $\epsilon - distance$ constraint can be calculated as follows:

$$\alpha \geq (1-\epsilon)^2, \qquad lim_{\epsilon \to 0} \alpha = 1$$

"$lim_{\epsilon \to 0}\alpha = 1$" implies the retrieval of all documents in the storage. This is the trivial oblivious strategy, where all data items are scanned for a single query. Yet, it cannot be tolerated in the cloud setting due to the overhead it introduces with a big corpus.

[1] medium sized data for a searchable encryption model



(a) Hit Rate Comparison

(b) Hybrid Policy Comparison

Fig. 2. Results

If only the document duplication policy is used in the model, the number of copies c to succeed $\epsilon - distance$ constraint can be calculated as follows:

$$c \geq (1 - \frac{1}{n})\frac{1}{\epsilon}, \qquad lim_{\epsilon \to 0} c = \infty$$

"$\lim_{\epsilon \to 0}\ c = \infty$" makes it impossible to individually adopt document duplication to succeed small ϵ values. Note that keeping too much copies for each data item in the corpus can be very expensive in the cloud [1] .

Although the heuristics cannot be employed individually to ensure small $\epsilon - distance$, their contribution can be significant when combined in a hybrid approach. In Figure 2(b), the fake data access costs of the two hybrid policies are listed. *Hybrid policy I* is composed of random caching and fake document access policies, where $\alpha = \frac{k}{n}$. In *Hybrid policy II*, document duplication is added to *Hybrid policy I* setting, where $\alpha = \frac{k}{n}$ and $c = 3$[2]. Then, five security premises $\epsilon = \{0.001, 0.01, 0.1, 0.2, 0.3\}$ are defined, and k and α are determined by Equations 4.4 and 4.5.

As shown in Figure 2(a), *Hybrid policy II* performs more fake additional data accesses than *Hybrid policy I* for small ϵ values because of the larger size of document number. Yet, *Hybrid policy II* outperforms *Hybrid policy I* for relatively larger ϵ values.

C. Comparison of ORAM and Hybrid Policies

Oblivious RAM has been proposed by Goldreich and Ostrovsky [12] to completely hide access patterns. After its first introduction, many Oblivious RAM constructions have been proposed. Among these constructions, to our knowledge, Williams et. al. [19] have proposed one of the most efficient ORAM constructions available in the literature. In this protocol, the computational overhead is $O(log\ n \times\ log\ logn)$ per request, where n is the number of data items. This overhead contains the constant factor of $1.44c$ for an allowed error probability of 2^{-c} with other constant factors in the big O [17]. If we assume, c = 64, then the constant 1.44c is 92 (Please see [17] for more details). Based on this analysis, the value of the constant c can be accurately approximated to be around 100. Now, suppose the number of data items are 2^{10}. Then, $log\ n = 10$. Hence, even the most efficient ORAM presented in [19] requires more than $100 \times 10 = 1000$ data items to be retrieved for a single data access, which is far from practical. Moreover, the overhead gets bigger as the number of data items (n) increases. On the other hand, the analyzed hybrid policy I retrieves approximately 6 documents for each document access of searchable encryption when $\epsilon = 0.001$. Still, Oblivious RAM completely hides the access pattern, but the *hybrid policies I and II* leak $\epsilon - distant$ access pattern.

[2]The current redundancy in contemporary cloud systems like Hadoop

VI. Conclusion

In this paper, we analyze the heuristics for the access pattern obfuscation over searchable encryption model. To this end, we first aim to measure the amount of access pattern leakage of an arbitrary model. Then, we determine the obfuscation amount for each heuristic and two different combination of those. By using these calculated obfuscation amount, we compare the cost of heuristics while restricting the access pattern leakage by a predetermined constant. An empirical analysis is conducted on a real world data set. The empirical evaluations indicates the practical nature of the hybrid heuristics.

References

[1] Y. Chen, V. Paxson, and R. H. Katz, "Whats new about cloud computing security?" EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2010-5, Jan 2010. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-5.html

[2] M. Bellare, A. Boldyreva, and A. O'Neill, "Deterministic and efficiently searchable encryption," ser. CRYPTO'07, 2007, pp. 535–552. [Online]. Available: http://dl.acm.org/citation.cfm?id=1777777.1777820

[3] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," ser. TCC'07, 2007. [Online]. Available: http://dl.acm.org/citation.cfm?id=1760749.1760788

[4] Y.-C. Chang and M. Mitzenmacher, "Privacy preserving keyword searches on remote encrypted data," ser. ACNS'05, 2005, pp. 442–455. [Online]. Available: http://dx.doi.org/10.1007/11496137_30

[5] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient constructions," ser. CCS '06, 2006, pp. 79–88. [Online]. Available: http://doi.acm.org/10.1145/1180405.1180417

[6] E.-J. Goh *et al.*, "Secure indexes," *the Cryptology ePrint Archive*, 2003.

[7] P. Golle, J. Staddon, and B. Waters, "Secure conjunctive keyword search over encrypted data," in *ACNS 04*, 2004, pp. 31–45.

[8] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," ser. SP '00, 2000, pp. 44–. [Online]. Available: http://dl.acm.org/citation.cfm?id=882494.884426

[9] L. Ballard, S. Kamara, and F. Monrose, "Achieving efficient conjunctive keyword searches over encrypted data," ser. ICICS'05, 2005, pp. 414–426.

[10] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," ser. ICDCS '10, 2010, pp. 253–262. [Online]. Available: http://dx.doi.org/10.1109/ICDCS.2010.34

[11] M. K. Mohammad Islam, Mehmet Kuzu, "Access pattern disclosure on searchable encryption: Ramification, attack and mitigation," in *NDSS Symposium*, 2012.

[12] O. Goldreich and R. Ostrovsky, "Software protection and simulation on oblivious rams," *J. ACM*, vol. 43, no. 3, pp. 431–473, May 1996. [Online]. Available: http://doi.acm.org/10.1145/233551.233553

[13] S. De Capitani di Vimercati, S. Foresti, S. Paraboschi, G. Pelosi, and P. Samarati, "Efficient and private access to outsourced data," ser. ICDCS '11, 2011, pp. 710–719. [Online]. Available: http://dx.doi.org/10.1109/ICDCS.2011.37

[14] P. Lin and K. S. Candan, "Enabling access-privacy for random walk based data analysis applications," *Data Knowl. Eng.*, vol. 63, no. 3, pp. 667–683, 2007.

[15] M. Kantarcioglu and C. Clifton, "Security issues in querying encrypted data," in *DBSec*, 2005.

[16] H.-A. Park, J. W. Byun, and D. H. Lee, "Secure index search for groups," ser. TrustBus'05, 2005. [Online]. Available: http://dx.doi.org/10.1007/11537878_14

[17] B. Pinkas and T. Reinman, "Oblivious ram revisited," ser. CRYPTO'10, 2010, pp. 502–519. [Online]. Available: http://dl.acm.org/citation.cfm?id=1881412.1881447

[18] E. Stefanov, E. Shi, and D. Song, "Towards practical oblivious ram," *CoRR*, vol. abs/1106.3652, 2011.

[19] P. Williams, R. Sion, and B. Carbunar, "Building castles out of mud: practical access pattern privacy and correctness on untrusted storage," ser. CCS '08, 2008, pp. 139–148. [Online]. Available: http://doi.acm.org/10.1145/1455770.1455790

[20] D. Boneh, E. Kushilevitz, R. Ostrovsky, and W. E. S. III, "Public key encryption that allows pir queries," Cryptology ePrint Archive, Report 2007/073, 2007.

[21] L. Devroye and G. Lugosi, *Combinatorial methods in density estimation*. Springer, 2001.

[22] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, and M. Zaharia, "Above the clouds: A berkeley view of cloud computing," Tech. Rep., 2009.

[23] P. J. Denning, "The locality principle," *Commun. ACM*, vol. 48, no. 7, pp. 19–24, Jul. 2005. [Online]. Available: http://doi.acm.org/10.1145/1070838.1070856

[24] B. Klimt and Y. Yang, "Introducing the enron corpus," in *CEAS 2004*, 2004.

[25] "Dropbox cloud storage," www.dropbox.com, 2012.