Towards an Information Theoretic Analysis of Searchable Encryption (Extended Version)

Saeed Sedghi, Jeroen Doumen, Pieter Hartel, Willem Jonker

University of Twente, Enschede, The Netherlands

Abstract. Searchable encryption is a technique that allows a client to store data in encrypted form on a curious server, such that data can be retrieved while leaking a minimal amount of information to the server. Many searchable encryption schemes have been proposed and proved secure in their own computational model. In this paper we propose a generic model for the analysis of searchable encryptions. We then identify the security parameters of searchable encryption schemes and prove information theoretical bounds on the security of the parameters. We argue that perfectly secure searchable encryption schemes cannot be efficient. We classify the seminal schemes in two categories: the schemes that leak information upfront during the storage phase, and schemes that leak some information at every search. This helps designers to choose the right scheme for an application.

1 Introduction

Storage outsourcing is a popular approach towards reducing the total cost of ownership of enterprise data storage. Current solutions either store data in plain, such that the confidentiality of the data is easily compromised, or the data is stored encrypted, which severely limits the kind of service that can be provided. In particular, the ability to search encrypted data is much needed but difficult to provide. Searchable encryption has many applications, particularly where client privacy is a main concern such as in E-mail servers [3], keeping medical information of a client [16], storing private videos and photos, and backup applications [15].

There are two trivial, extreme approaches towards searching in encrypted data. The first trivial approach is for the server to send the client the entire encrypted data base, such that the client may decrypt, then query. Although this solution has a high security, the communication overhead between the server and the client is prohibitively high. The second trivial approach is for the server to decrypt the entire data base, then to execute the query. Although this solution is efficient, letting the server decrypt the data base offers poor security. The problem is thus to find a good compromise between query and data communication efficiency on the one hand, and security on the other hand.

Efficiency means that the query performance should not be influenced negatively by encryption, and that data communication to and from the server should be appropriate. Security means that the stored data, the query that the client

sends to the server to retrieve the data selectively, and executing the query on the stored data should not reveal any information to the server about the data except the data items matched with the query.

Three seminal searchable encryption schemes have been proposed [15, 3, 9], each with specific advantages and disadvantages. A good overview is provided in the PhD thesis of Brinkman [4]. The main problem with each of these proposals is that they assume different limitations: in each case the security analysis assumes a specific model consisting of an adversary with specific limitations. Therefore, the security analysis in each case does not show exactly which parameters reveal information to the server, in a manner that allows us to compare the leakage of schemes.

Contribution We propose a model for searchable encryption that facilitates an information theoretic security analysis against an adversary with *un*limited power. Since an information theoretic analysis implies no restriction on the computational model of the adversary, information leakage from the parameters is computable in this model. We stress that an information theoretic approach requires idealized encryption and hash functions, and as such this work is an exploration into the theoretical properties of searchable encryption. We apply our model and analysis method to the three seminal approaches towards searchable encryption to show that the model and the analysis method are both general and powerful. The scope of our analysis and results in this paper is limited to a single client using a single server to store a single data item. There can be any number of keywords associated with the data item, and the client may perform any number of keyword searches. The extension to multiple data items, multiple clients and multiple servers is future work.

The paper is organized as follows. Section 2 formalizes the problem. Section 3 presents our approach towards analyzing the security of searchable encryption schemes. Section 4 analyzes the security of the three seminal searchable encryption schemes. A summary of the related work is described in Sect. 5. We present some concrete examples of our analysis in Sect. 6. The last section concludes and suggests future work.

2 Statement of the Problem

Notation. We use the following notation, which is borrowed from Moulin et al [11]. Random variables are denoted by capital letters (e.g. X) and their individual values by lower case letters (e.g. x). The domain over which a random variable is defined is denoted by a script letter (e.g. $\mathcal{X}$) and the number of elements in the range of X is denoted by $|\mathcal{X}|$. The probability mass function (pmf) of a random variable $X \in \mathcal{X}$ is denoted by $P_X(x)$. When no confusion is possible, we drop the subscript to simplify the notation. We write $X \sim P_X$ to indicate that a random variable X is distributed according to P_X. Given random variables X and Y, we denote the entropy of X by $H(X)$, the entropy of X conditioned on Y by $H(X|Y)$ and the mutual information between X and Y by $I(X;Y)$.

2.1 Description of the Problem

There are various formulations of the searchable encryption problem. We propose the following generic formulation in this paper. Without loss of generality we assume that the client splits his data into a non-searchable part d and a set of searchable keywords m. Referring to Fig. 1, let us assume that a client is about to store a tuple $u =< d, m >$ consisting of a single data item d and an associated metadata item m on a server. The metadata m is actually a set of l keywords $m = \{w_1, ..., w_l\}$ where each keyword is taken from a finite set $\mathcal{W}$. The objectives of the client are:

1. The confidentiality of d and m is preserved.
2. d is retrieved in the case m contains a queried keyword.

All solutions to the searchable encryption problem proceed in four phases:

Setup: The client and the server may need to share some data and functions and each may need to prepare some private data.

Storage: The data and the metadata items are transformed to an appropriate format for storage on the server by the steps below:

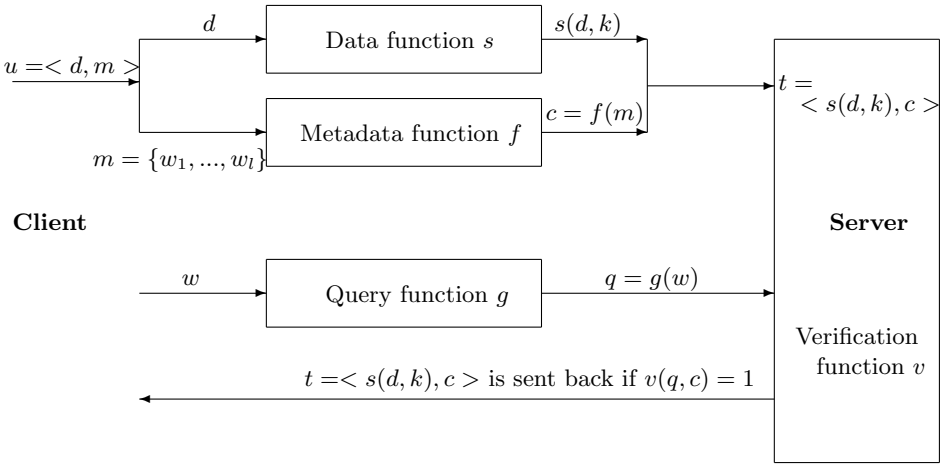

Fig. 1. Formulation of the searchable encryption problem. Here, d is a data item, and $m = \{w_1, ..., w_l\}$ is the associated metadata.

– The client transforms m to a searchable representation $c = f(m)$, where $f(.)$ is a metadata function.

– The client transforms d to an appropriate encrypted form for storage on the server $s(d,k)$, where $s(.,.)$ is a data function and $k \in \mathcal{K}$ is a secret key.
– The client sends the tuple $t =< s(d,k), c >$ to the server for storage.

Query: To query the server if a keyword $w \in \mathcal{W}$ occurs in m, the client sends the query $q = g(w)$ to the server, where $g(.)$ is a query function.

Search: Given q and c, using a verification function $v(.,.)$ the server checks if $v(q,c) = 1$, $t =< s(d,k), c >$ is sent back to the client in case of a match.

2.2 Problem Instances

The data d, the metadata m and the functions $s(.,.)$, $f(.)$, $g(.)$ and $v(.,.)$ are the parameters of our proposed searchable encryption formulation. To show that this is a realistic formulation we will instantiate these parameters in such a way that the formulation specializes to the three seminal searchable encryption schemes. These are: Song, Wagner, and Perrig (SWP) [15], Public key encryption with keyword search (PEKS) [3], and Secure indexes (SI) [9]. Below a description of the listed schemes is presented.

The SWP scheme. The first practical approach to the problem of searchable encryption has been proposed by Song, Wagner and Perrig [15]. This scheme does not search for keywords in the metadata; instead searching approaches the data directly. The SWP scheme requires the client to split the data item d into fixed size blocks $d = (b_1, ..., b_l)$ and calculates a searchable representation for each block b_i. However, to apply our formulation to the SWP scheme we consider each block to be a keyword, i.e. $w_i = b_i$.

Setup: The client and the server agree to use a hash function $h_1 : \{0,1\}^v \times \{0,1\}^n \longrightarrow \{0,1\}^{n-v}$, where n is the number of bits in each keyword and $1 \leq v < n$. The client also uses a hash function $h_2 : \mathcal{W} \longrightarrow \{0,1\}^n$.

Storage $f(m)$: To generate a searchable representation $c = f(m)$, the client:

1. Generates a sequence of random values $r_i \in \{0,1\}^v$, $1 \leq i \leq l$.
2. Generates a sequence of trapdoors $x_i = r_i || h_1(r_i, h_2(w_i))$, $1 \leq i \leq l$.
3. Produces $e(w_i, k)$ using a symmetric key encryption function $e(.,.)$, and a secret key $k \in \mathcal{K}$, for each keyword $w_i \in m$.
4. Generates $c_i = x_i \oplus e(w_i, k)$, for each keyword $w_i \in m$.
5. Gathers the sequence $c = (c_1, ..., c_l)$ since the data item is actually a sequence of keywords $d = (w_1, ..., w_l)$.

The client sends the tuple $t =< s(d,k), c >$ to the server for storage, where $s(d,k) = 0$.

Query $g(w)$: To search for a keyword $w \in \mathcal{W}$, the client sends the query $q =< e(w,k), h_2(w) >$ to the server.

Search $v(q,c)$: After receiving q, the server calculates the trapdoor $y_i' || y_i'' = c_i \oplus e(w,k)$ for each element c_i of c and checks if $y_i'' = h_1(y_i', h_2(w))$; the server sends back $t =< s(d,k), c >$ to the client in case of a match.

The PEKS scheme. The main disadvantage of the SWP scheme is that the encrypted keyword(s) must be sent to the server for the verification function. Boneh et al [3] propose the idea of a public key searchable encryption based on the Diffie-Hellman problem. In contrast with the SWP scheme, searching is performed on a set of keywords $m = \{w_1, ..., w_l\}$ associated with the data d.

Setup The client chooses two groups of prime order p, $\mathbb{G}_1$ and $\mathbb{G}_2$, using group generators g_1 and g_2 respectively and a non-degenerate bilinear function $b : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$. The server and the client agree to use two hash functions $h_1 : \{0,1\}^* \longrightarrow \mathbb{G}_1$ and $h_2 : \mathbb{G}_2 \longrightarrow \{0,1\}^{[log(p)]}$. The client picks a random numbers $\alpha \in \mathbb{Z}_p$ such that $g_1^\alpha \in \mathbb{G}_1$.

Storage $f(m)$: To generate a searchable representation $c = f(m)$, the client:

1. Generates a random value $r_i \in \mathbb{Z}_p$ for each keyword $w_i \in m$.
2. Generates a searchable representation $c_i =< g_1^{r_i}, h_2(b(h_1(w_i), g_1^{(r_i\alpha)})) >$ for each $w_i \in m$.
3. Gathers a set $c = \{c_1, ..., c_l\}$, since the metadata is a set of keywords.

The client sends the tuple $t =< s(d,k), c >$ to the server for storage, where $s(d,k)$ is any appropriate encryption of d.

Query $g(w)$: To query for a keyword $w \in \mathcal{W}$, the client sends $q = h_1(w)^\alpha$ to the server.

Search $v(q,c)$: The server checks for each element c_i of c, if $h_2(b(q, g^{r_i})) = h_2(b(h_1(w_i), g^{(r_i\alpha)}))$; $t =< s(d,k), c >$ is sent back to the client in case of a match.

The SI scheme. Both the SWP and the PEKS schemes have the disadvantage that a search takes time linear in the number of keywords. To obtain a secure, efficient and practical method, Goh [9] proposes an approach to map each keyword to a hash value. The client has a data item d and an associated set of keywords $m = \{w_1, ..., w_l\}$ to store on the server.

Setup The client chooses $z \geq 1$ independent hash functions $h_1, ..., h_z$, where each $h_i : \mathcal{W} \longrightarrow \{0,1\}^j, j \in \mathbb{N}$.

Storage $f(m)$: To generate a searchable representation $c = f(m)$, the client:

1. Calculates a trapdoor $x_i = \{h_1(w_i), ..., h_z(w_i)\}$ for each keyword $w_i \in m$.
2. Represents c by an array of 2^j bits, where each element c_n, $1 \leq n \leq 2^j$, takes the value 0 or 1 as follows:

$$c_n = \begin{cases} 1 \text{ if there is at least one } h_s(w_i) = n, i = 1, ..., l, s = 1, .., z \\ 0 \text{ Otherwise} \end{cases} \quad (1)$$

The client sends the tuple $t =< s(d,k), c >$ to the server, where $s(d,k)$ is any appropriate encryption of d.

Query $g(w)$: To query for a keyword $w \in \mathcal{W}$ the client sends $q = \{q_1, ..., q_z\}$, where $q_s = h_s(w), s = 1, ..., z$, to the server.

Search $v(q,c)$: The server checks for all $1 \leq s \leq z$ if $c_{q_s} = 1$; $s(d,k)$ is sent back to the client in case of a match.

Table 1 summarizes the functions to be used for $f(.)$, $g(.)$ and $v(.,.)$ by each of the cited methods.

	Parameters of the formulation			
Scheme	$t =< d, m >$ $m = \{w_1, ..., w_l\}$	$c = f(m)$	$q = g(w)$	$v(q, c)$
SWP [15]	$d = (b_1, .., b_l)$ $w_i = b_i$	$c_i = e(w_i, k) \oplus x_i$, where $x_i = r_i \| h(r_i)$	$q = e(w, k)$	$x'_i \| x''_i = c_i \oplus e(w, k)$ Check if $x''_i = h(x'_i)$
SI [9]	d m	$c = \{h_j(w_i)\}$ for $j = 1, ..., z$ and $i = 1, .., l$	$q = \{h_j(w)\}$ for $j = 1, ..., z$	Check if $q \subset c$
PEKS [3]	d m	$c_i =< x_i, y_i >$ where $x_i = g^{r_i}$ and $y_i = h_2(b(h_1(w_i), g^{r_i \alpha}))$	$q = (h_1(w))^{\alpha}$	Check if $h_2(b(q, x_i)) = y_i$

Table 1. Summary of the employed functions in searchable encryption approaches. In this table r is a random number, $h(.)$ is a hash function, $e(.,.)$ is an encryption function, $g(.)$ is a group generator, and $b(.,.)$ is a bilinear map.

3 Security Evaluation

In this section we evaluate the security of searchable encryption schemes. SWP [15] informally divide the security evaluation of searchable encryption schemes into the following properties:

- Provable security: A searchable encryption scheme is provably secure if the confidentiality of the data and the metadata is preserved before performing any search.
- Hidden query: A searchable encryption scheme is hidden query if a received query does not leak any information on the queried keyword.
- Query isolation: A searchable encryption scheme supports query isolation if the server learns nothing about the metadata after a search is performed.

There are two approaches for evaluating the security of cryptographic schemes: the information theoretic approach and the computational complexity approach. While the information theoretic approach evaluates the security of cryptographic schemes against an adversary with *unlimited* computational power, the computational complexity approach decides whether it is feasible for an adversary with reasonably *limited* computational power to extract information about the plaintext of a ciphertext.

All the previously proposed searchable encryption schemes are proved secure in the computational complexity setting. However, we are interested in the *theoretical* bounds on the security of searchable encryption schemes in general, and the seminal schemes mentioned earlier in particular. Therefore, we use an information theoretic approach to analyze the security of idealized searchable encryption schemes.

In Section 3 we present and motivate four assumptions that formalize the difference between the seminal schemes as published and their idealized interpretations analyzed here.

We now formalize the security evaluation of searchable encryption schemes using the tools of information theory as follows:

- **Provable security**: We formalize the provable security of d as $I(D; s(D,K), f(M))$ (i.e. the information that the stored tuple $< f(M), s(D,K) >$ leaks on D) and the provable security of m as $I(M; s(D,K), f(M))$ (i.e. the information that the stored tuple $< f(M), s(D,K) >$ leaks on M).
- **Hidden query**: We formalize hidden query as $I(W; g(W))$ (i.e. the information that a query leaks on the queried keyword).
- **Query isolation**: We formalize query isolation as: $I(M; v(f(M), g(W)))$ (i.e. the information that a search result leaks on the metadata).

3.1 Security Parameters of the Formulation

In this section we introduce a convenient notation for the information leakage from each searchable encryption parameter. In our formulation we have four different functions that can leak information:

- The stored data $s(d,k)$ leaks on d: $\varepsilon_s = I(D; s(D,K))$.
- The searchable representation $f(m)$ leaks on m: $\varepsilon_f = I(M; f(M))$.
- The query $g(w)$ leaks on w: $\varepsilon_g = I(W; g(W))$.
- The search result $v(f(m), g(w))$ leaks on m: $\varepsilon_v = I(M; v(g(W), f(M)))$.

Here, ε_g and ε_v correspond to hidden query and query isolation respectively. An idealized searchable encryption scheme has perfect security if:

$$\varepsilon_s = \varepsilon_f = \varepsilon_g = \varepsilon_v = 0 \tag{2}$$

Theorem 1, relates provable security to ε_s and ε_f by giving upper bounds on the uncertainty of the server about the data and the metadata items, under the condition that the server has access to $s(d,k)$ and $f(m)$.

Theorem 1. *The provable security of any searchable encryption scheme admits the following upper bound:*

$$I(D; s(D,K), f(M)) \leq \varepsilon_s + \varepsilon_f - H(M|D) + H(M|D, f(M))$$
$$I(M; s(D,K), f(M)) \leq \varepsilon_f + \varepsilon_s - H(D|M) + H(D|M, s(D,K))$$

Proof: see the appendix.
The intuition of the first inequality is as follows: ε_s is the information that the stored data $s(D,K)$ leaks on D directly, and $\varepsilon_f - H(M|D) + H(M|D, f(M))$ is the information that the searchable representation $f(M)$ leaks on D indirectly via M. A similar intuition applies to the second inequality.

4 Analysis of Known Schemes

The security of searchable encryption schemes is analyzed in related work using the computational complexity approach. The reason is that the cryptographic primitives which are used for the data, the metadata, the query and the verification function (e.g. block ciphers and hash functions) are not information theoretically secure. Here, we are interested in an adversary with unlimited power, who however cannot look inside the cryptographic primitives. In other words, we analyze whether searchable encryption schemes leak information to the server under the assumption that perfectly secure cryptographic primitives are used:

Assumption 1. Any encryption function is a one-time pad encryption since only the one time pad encryption has been proved to be information theoretically secure.

Assumption 2. The client uses a secret table by way of a hash function. In most cases considered here, the server does not need to know the hash function h. A secret table works as follows: given $y = h(x)$, and a string y it is *impossible* to find the corresponding bit string x, whereas in the case that $h(.)$ is a hash function, it is *hard* to find the string x. Therefore, a secret table is information theoretically secure and $I(W; h(W)) = 0$.

We make the following assumptions on the distribution of the keywords in the metadata:

Assumption 3. The distribution of the total number of keywords in m is $L \backsim P_L(l)$. i.e. clients tend to choose the number of keywords according to a distribution which depends on the application domain.

Assumption 4. Given the total number of keywords, l, the distribution of choosing a keyword $P(w \in \mathcal{M}|L = l)$ is uniform. i.e. the client picks each keyword from the set $\mathcal{W}$ with the same probability. Although this assumption might not be correct for all practical situations, the uniform distribution of keywords gives the highest security in comparison with the other distributions.

These assumptions are purely theoretical. In particular, we are not suggesting to use a one time pad encryption as in this case outsourcing data to the data base would not help the client (i.e. the secret key must be as large as the data itself). The hash table is just as impractical. However, these assumptions allow us to analyze how close the seminal schemes are to ideal security. In other words, by these assumptions the client uses the most secure cryptographic primitives (for the data, metadata, query and verification functions) and keyword distribution, and we explore the theoretical upper bounds on the security of these schemes. Moreover, our results show how much information the parameters of the searchable encryption schemes leak to the server.

When it comes to a practical implementation, the leakage of information from the functions can not be smaller than what is derived here. The reason is that although the cryptographic primitives are secure against an adversary

with limited power, they are not information theoretically secure. Therefore, our results show:

1. The minimum leakage of information from the functions of searchable encryption in all situations.
2. Which functions of a searchable encryption scheme leak more information, and which functions leak less information. This type of analysis is not possible in computational security settings.

4.1 Idealized SWP

The parameters of the SWP scheme are as follows: the data and the metadata are the same $d = m$, there is no stored data on the server $s(d, k) = 0$, the metadata is a sequence of keywords $m = (w_1, ..., w_l)$, for each keyword $w_i \in m$ a random value r_i is generated. According to assumption 1, the encryption function $e(.,.)$ is a one time-pad encryption. Hence, each keyword $w_i \in m$ is encrypted using a unique key $k_i \in \{0,1\}^n$ as $e(w_i, k_i)$.

– **The information leakage from** $s(D, K)$: Since there is no stored data in the SWP scheme ($s(D, K) = 0$),

$$\varepsilon_s = I(D; s(D, K)) = 0 \tag{3}$$

– **The information leakage from** $f(M)$: Let define the function

$$T(e(m, k)) = e(m, k) \oplus (r_1||h(r_1), ..., r_l||h(r_l)) \tag{4}$$

then $f(m) = T(e(m, k))$, where $k = (k_1, ..., k_l)$. Since $M \longrightarrow e(M, K) \longrightarrow T(e(M, K))$ forms a Markov chain:

$$I(M; T(e(M, K))) \leq I(M|e(M, K)) \tag{5}$$

Using one time pad encryption for $e(m, k)$, $I(M|e(M, K)) = 0$ [14]. Hence,

$$\varepsilon_f = I(M; f(M)) = 0 \tag{6}$$

Therefore, the searchable representation achieves perfect secrecy.

– **The information leakage from** $g(W)$: To query for a keyword w the client sends the query $g(w) =< e(w, k), h_1(w) >$ to the server. Hence,

$$\varepsilon_g = I(W; e(W, K), h_1(W)) \tag{7}$$

Using standard information theoretic formulas:

$$\varepsilon_g = I(W; h_1(W)) + I(W; e(W, K)|h_1(W)) \tag{8}$$

According to assumption 2, $I(W; h(W)) = 0$, and according to assumption 1, $I(W; e(W, K)) = 0$. Hence, $\varepsilon_g = 0$. Therefore, a query does not leak any information about the keyword queried.

– **The information leakage from** $v(f(M), g(W))$: The SWP scheme transforms each keyword w_i to a unique searchable representation c_i, since each keyword is transformed to a searchable representation by a unique random value r_i. Therefore, if the keywords w_i and w_j are the same keywords in the metadata, c_i and c_j are different. However, if the client sends the query $g(w_i)$ (or $g(w_j)$) to the server, by the search result the server learns that c_i and c_j are searchable representations of the same keywords. Therefore, the uncertainty of the server about c_i and c_j reduces to the uncertainty about c_i (or c_j) alone. The following theorem quantifies the information that a search leaks on M.

Theorem 2. *Let* $P(M|L) = \frac{1}{|\mathcal{W}|^l}$, $\mathcal{E}(S)$ *denotes the expected number of repeated keywords and* $\mathcal{E}(L)$ *denotes the expected number of keywords in* M. *Then,*

$$\varepsilon_v = I(M; v(f(M), g(W))) = H(W)(\mathcal{E}(S) - 1) \tag{9}$$

where $\mathcal{E}(S) \approx (\frac{1}{|\mathcal{W}|})$.

Proof: see the appendix.
Intuitively, Theorem 2 says that the uncertainty of the repeated keywords is the reductions in the uncertainty of M due to the knowledge of a search result. If each keyword occurs once in M, then $\mathcal{E}(S) = 1$ and $I(M; v(f(M), g(W))) = 0$, and if all the keywords are the same, then $\mathcal{E}(S)) = \mathcal{E}(L)$ and $I(M; v(f(M), g(W))) = \mathcal{E}(L) - 1$.

Summarizing, even using perfectly secure cryptographic primitives for the SWP scheme, a search reveals some information to the server. However, storing data does not leak any information.

4.2 Idealized SI

Without loss of generality we assume that the client uses only one hash function $h(.)$ to map each keyword w to a searchable representation $h(w)$. (i.e. $f(m) = \{h(w_1), ..., h(w_l)\}$). According to assumption 2, the client uses a secret table by way of the hash function. We also assume that $P(M|L) = \frac{1}{\binom{|\mathcal{W}|}{l}}$, since according to assumption 4 the distribution of keywords in the metadata is uniform and the metadata is a set of keywords.

– **The information leakage from** $s(D, K)$**:** Since we assume that encryption is one time-pad encryption (assumption 1), the stored data achieves perfect secrecy [14]. Hence,

$$\varepsilon_s = I(D; s(D, K)) = 0 \tag{10}$$

– **The information leakage from** $f(M)$**:** In contrast with the SWP scheme, the SI scheme admits false positives. A false positive occurs when two or more different keywords are mapped to the same searchable representation by $h(.)$. First we evaluate the information leakage in the case without false

positives. We then extend the evaluation in the case of a probability of a false positive.

- Without false positives:

$$\varepsilon_f = I(M; f(M)) = I(M; h(W_1), ..., h(W_L)) \tag{11}$$

Theorem 3 shows the information leakage from $f(M)$ in the case without false positive.

Theorem 3. *The information that the searchable representation leaks on the metadata is:*

$$\varepsilon_f = I(M; f(M)) = H(L) \tag{12}$$

where $H(L)$ is the uncertainty in the number of unique keywords in the metadata.

Proof: See the appendix.

Intuitively, Theorem 4 says that the uncertainty in the number of unique keywords contained in the metadata is the reduction in the uncertainty of the metadata due to the knowledge of the searchable representation.

- A False positive:

In this case the server cannot learn the precise number of unique keywords since more than one unique keywords could be mapped to the same searchable representation. The following theorem shows how the probability a false positive P reduces the information that $f(M)$ leaks on M.

Theorem 4. *Let j be the number of bits in the output of the hash values. Then,*

$$\varepsilon_f = I(M, f(M)) = H(L) - \beta \tag{13}$$

Here, $\beta = \sum_{i=1}^{|\mathcal{W}|} \sum_{x=1}^{i} (P(i) log(\frac{1}{\binom{\mathcal{W}}{i}}) + \frac{(2^j)! x^{i-l}}{(2^j - x)! 2^{ji}} P(i) log(\frac{(2^j)! x^{i-x}}{(2^j - x)! 2^{ji} \binom{\mathcal{W}}{i}}))$.

Proof: see the appendix.

Intuitively, Theorem 4 says that a false positive reduces the revealed information about the uncertainty of the number of keywords to the server.

– **The information leakage from $g(W)$:** To query for a keyword w the client sends the query $g(w) = h(w)$ to the server. According to assumption 2:

$$\varepsilon_g = I(W; h(W)) = 0 \tag{14}$$

Therefore, a query in the SI scheme does not reveal any information about the queried keyword.

– **The information leakage from $v(f(M), g(W))$:** In contrast with the SWP scheme, the metadata is a set of keywords. Hence, each keyword in the metadata is unique and by a search result the server learns nothing about the metadata.

$$\varepsilon_v = I(M; v(f(M), g(W))) = 0 \tag{15}$$

Summarizing, using even perfectly secure cryptographic primitives for the SI scheme, the searchable representation reveals information to the server. The probability of false positive increases the security of the SI.

4.3 Idealized PEKS

The PEKS scheme was originally proposed for transforming a *set* of keywords to a searchable representation. However, the scheme is capable of transforming a *sequence* of keywords to a searchable representation as well, since a different random value r_i is considered to transform each keyword w_i to a searchable representation (see section 2.4 PEKS). In other words, the same keywords can be mapped to a different searchable representation. Therefore we consider the metadata as a sequence of keywords. Since an information theoretic approach cannot handle public key cryptography, we analyze only the hash function of the PEKS scheme. In other words, our analysis relies on the security of the hash function only.

- **The information leakage from $s(D, K)$:** According to assumption 1, $s(D, K)$ is one time-pad encryption. Hence,

$$\varepsilon_s = I(D, s(D, K)) = 0 \tag{16}$$

  Therefore, the stored data does not reveal any information about the data.
- **The information leakage from $f(M)$:** Since the client uses a secure table and a unique random value to transform each keyword to a searchable representation, the server cannot learn anything about the metadata. Hence,

$$\varepsilon_f = I(M; f(M)) = 0 \tag{17}$$

  Therefore, the server cannot learn anything about the metadata due to the knowledge of the searchable representation.
- **The information leakage from $g(W)$:** To query for a keyword w the server sends the query $g(w) = h(w)^\alpha$ to the server. According to assumption 2,

$$\varepsilon_g = I(W; h(W)^\alpha) = 0 \tag{18}$$

  Therefore, the server cannot learn anything about the plaintext of the query after receiving a query.
- **The information leakage from $v(f(M), g(W))$:** Similar to the case of the SWP scheme, before performing any search by the client, the server cannot learn the unique keywords in the metadata. However, after a search for the keyword w, the server learns the repeated keywords in case of a match. By the same analysis as of Theorem 2, the information leakage from a search is calculated as follows:

$$\varepsilon_v = I(M; v(f(M), g(W))) = H(W)(\mathcal{E}(S) - 1) \tag{19}$$

Summarizing, using even perfectly secure cryptographic primitives for the PEKS scheme, a search reveals some information to the server.

Table 2 summarizes the information leakage from the parameters of the SWP, SI and PEKS schemes. In this table the analysis of the two trivial schemes from the introduction is included for comparison. Trivial solution 1 lets the server

return the entire encrypted data base to the client at each query. In this case the server never learns anything about the data, hence $\varepsilon_s = \varepsilon_f = \varepsilon_g = \varepsilon_v = 0$. Trivial solution 2 lets the server decrypt all data at the first query, hence, the uncertainty in the keywords $H(W)$ and the data $H(D)$ is leaked to the server at the first query.

	Information leakage from			
Scheme	Stored data ε_s	Searchable representation ε_f	Query ε_g	Search ε_v
Schemes that do not leak at all				
Trivial solution 1	0	0	0	0
Schemes that leak at each search				
Trivial solution 2	0	0	$H(W)$	$H(D)$
SWP	0	0	0	$(\mathcal{E}(S)-1)H(W)$
PEKS	0	0	0	$(\mathcal{E}(S)-1)H(W)$
schemes that only leak up front				
SI	0	$H(L)-\beta$	0	0

Table 2. Summary of the information leakage from the parameters of the SWP, SI, and PEKS schemes, as well as the two trivial schemes. In this table $\beta = \sum_{i=1}^{|\mathcal{W}|}\sum_{x=1}^{i}(P(i)log(\frac{1}{\binom{\mathcal{W}}{i}}) + \frac{(2^j)!x^{i-l}}{(2^j-x)!2^{ji}}P(i)log(\frac{(2^j)!x^{i-x}}{(2^j-x)!2^{ji}\binom{\mathcal{W}}{i}}))$ (see 4.2)

We conclude by categorizing the seminal searchable encryption schemes into three groups: (1) schemes that do not leak at all, (2) schemes that leak everything up front, and (3) schemes that leak nothing up front but which leak at each search. In practice this means that if an application is expected to perform many searches, a scheme from the second group is probably best. If we have a scenario where only a few searches are expected, a scheme from the third group is best.

5 Related Work

Our analysis of the information leakage of searchable encryption is limited to the three seminal schemes, SWP, PEKS and SI. However, we believe that the same analysis can be applied to more recent schemes based on the seminal schemes. Here, we discuss the most prominent searchable encryption schemes that follow the seminal schemes.

SWP based schemes. For XML documents Brinkman et al [5] modify the SWP scheme to facilitate a faster search by exploiting the tree structure of XML documents.

SI based schemes. Chang and Mitzenmacher [6] propose a scheme based on mapping keywords of the metadata to hash values. The proposed scheme has a lower probability of false positives than the SI scheme and the scheme also hides

the number of keywords in the metadata from the server. Curtmola et al [7] propose a symmetric key encryption approach that offers better efficiency than the SI scheme, and the scheme can be applied for multiple users scenarios by generating a random value for each user.

PEKS based schemes. The schemes proposed by Abdalla et al [10] lowers the probability of false positives in the PEKS scheme. Bellare et al [2] propose a deterministic searchable encryption scheme that offers faster search than the PEKS scheme by mapping each searchable representation to an index. Baek et al [1] propose a modified PEKS in such a way that the client could send a query through an insecure channel by mapping each keyword to several hash values. Park et al [13] propose a modified PEKS that gives a proxy the ability to decrypt searchable representations containing desired keywords. Fuhr at al [8] propose a scheme that allows the client to recover the plaintext of the keywords after transforming the metadata to a searchable encryption. The scheme applies the xor operation to the searchable representation and the hash value of a random value. The random value is kept by the client.

None of the extensions to the seminal schemes use primitives that cannot be idealized in the same way as we have idealized encryption and hashing. Therefore we believe that our methods are applicable to the extensions of the seminal schemes. To prove this is future work.

6 Example

In this section we present a few numerical examples to illustrate what the analysis actually means. Our cryptographic primitives in this example are chosen according to assumptions 1 and 2. According to assumption 4, the distribution of the keywords in the metadata is uniform and we consider a Poisson distribution for the distribution of the number of keywords ($P(l)$) in the metadata as follows [12]:

$$P(l) = \frac{\lambda^l e^{-\lambda}}{l!} \tag{20}$$

Here λ is the expected number of keywords in the metadata. Let the cardinality of keyword in metadata be equal to the size of Oxford dictionary, 126000 words. Moreover, let the expected number of keywords in the metadata be $\lambda = 100$.

SWP and PEKS: The expected number of the repeated keywords $\mathcal{E}(S) = 16.9\frac{1}{126000}$ and the enctropy of keywords is $H(W) = 16.9$ Hence the information leakage from the parameters is then as follows, $\varepsilon_d = \varepsilon_f + \varepsilon_g = 0$, and $\varepsilon_v = 16.9\frac{1}{126000}$.

SI: The SI scheme is applied to a set of keywords. Let assume there is no false positives. The entropy of the number of the number of keywords is $H(L) = 5.14$. Hence, the information leakage from the parameter is: $\varepsilon_d = \varepsilon_g = \varepsilon_v = 0$ and $\varepsilon_f = 5.14$. Now, let the number of in the hash of keywords be $j = 200$, Then, $H(L) = 4.7$.

The example shows that in the case that the client performs only a few searches on the stored data (e.g. backup scenarios), the SWP scheme or the

PEKS scheme would be ideal. However, In the case that the client intends to send many queries to the server, the SI scheme is a better choice since this scheme has higher efficiency.

7 Conclusion and Future Work

We present an information theoretic analysis of searchable encryption, where a single client stores a single data item on a single server. Any number of keywords may be associated with the data item and the client can perform any number of queries. We propose a generic formulation for the searchable encryption problem and apply the formulation to all three seminal schemes (SWP, SI, and PEKS) from the literature. We then formalize the security of searchable encryption using the tools of information theory and analyze the seminal schemes. The results of our analysis shows that even using perfectly secure cryptographic primitives, the parameters of all seminal schemes leak some information to the server. Our analysis shows that each scheme has its specific strengths and weaknesses in terms of provable security, hidden query and query isolation, and thus provides a useful tool to compare searchable encryption schemes. In future work we intend to extend the analysis to multiple data, multiple client and multiple server settings.

Acknowledgements

We thank Henk van Tilborg, Qiang Tang and Peter van Liesdonk for their help with the paper.

This research is supported by the SEDAN project, funded by the Sentinels program of the Technology Foundation STW, applied science division of NWO and the technology programme of the Ministry of Economic Affairs under project number EIT.7630.

References

1. J. Baek, R. Safiavi-Naini, and W. Susilo. Public key encryption with keyword search revisited. Available on Cryptology ePrint Archive, Report 2005/119, 2005.
2. Mihir Bellare, Alexandra Boldyreva, and Adam O'Neill. Deterministic and efficiently searchable encryption. In Alfred Menezes, editor, *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*, pages 535–552. Springer, 2007.
3. D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. In C. Cachin and J. Camenisch, editors, *23rd Int. Conf. on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, volume LNCS 3027, pages 506–522, Interlaken, Switzerland, May 2004. Springer.
4. R. Brinkman. *Searching in encrypted data*. PhD thesis, University of Twente, Enschede, The Netherlands, 2007. ISBN 9789036524889.
5. R. Brinkman, L. Feng, J. M. Doumen, P. H. Hartel, and W. Jonker. Efficient tree search in encrypted data. *Information Systems Security Journal*, 13(3):14–21, Jul 2004.

6. Yan C. Chang and Michael Mitzenmacher. *Privacy Preserving Keyword Searches on Remote Encrypted Data*, volume 3531. January 2005.
7. Reza Curtmola, Juan Garay, Seny Kamara, and Rafail Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. In *CCS '06: Proceedings of the 13th ACM conference on Computer and communications security*, pages 79–88, New York, NY, USA, 2006. ACM.
8. Thomas Fuhr and Pascal Paillier. Decryptable searchable encryption. In Willy Susilo, Joseph K. Liu, and Yi Mu, editors, *ProvSec*, volume 4784 of *Lecture Notes in Computer Science*, pages 228–236. Springer, 2007.
9. Eu-Jin Goh. Secure indexes. Cryptology ePrint Archive, Report 2003/216, 2003.
10. John Malone-Lee, Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. In *Advances in Cryptology - Proceedings of CRYPTO '05*, pages 205–222. Springer-Verlag, August 2005.
11. P. Moulin and J. A. O'Sullivan. Information-Theoretic analysis of information hiding. *IEEE Transactions on information theory*, 49(3):563–593, Mar 2003.
12. A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. 4th edition, 2002.
13. D. Park, J. Cha, and P. Lee. Searchable keyword-based encryption. Cryptology ePrint Archive, Report 2005/367, 2005. Available at http://eprint.iacr.org., 2005.
14. C. Shannon. Communication theory of secrecy systems. *Bell Sys. Tech*, 28(4):656–715, 1949.
15. D. X. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In *21st Symp. on Security and Privacy (S&P)*, pages 44–55, Berkeley, California, May 2000. IEEE Computer Society.
16. Juan Ramón Troncoso-Pastoriza, Stefan Katzenbeisser, and Mehmet Celik. Privacy preserving error resilient dna searching through oblivious automata. In *CCS '07: Proceedings of the 14th ACM conference on Computer and communications security*, pages 519–528, New York, NY, USA, 2007. ACM.

A Proof of Theorem 1

Here we prove Theorem 1. Using standard information theoretic rules,

$$I(D; s(S,K), f(M)) = I(D; s(D,K)) + I(D; f(M)|s(D,K)) \tag{21}$$

By information theoretic rules:

$$I(D; f(M)|s(D,K)) \leq I(D; f(M)) \tag{22}$$

$$I(D; f(M)) = H(D) - H(D|f(M)) \tag{23}$$

$$I(D; f(M)) = H(D) - H(D|f(M)) + H(D|M) - H(D|M) \tag{24}$$

The random variables $D \longrightarrow M \longrightarrow f(M)$ form a Markov chain since

$$P(d|m, f(m)) = P(d|m). \tag{25}$$

According to the Markov chain rule, $H(D|M) = H(D|M, f(M))$. Hence,

$$I(D; f(M)) = H(D) - H(D|f(M)) + H(D|M, f(M)) - H(D|M). \tag{26}$$

The mutual information $I(D; f(M)$ is evaluated below:

$$I(D; f(M)) = I(D; M) - I(D; M|f(M)) \tag{27}$$

Working at the right hand side:

$$I(D; M) = H(M) - H(M|D) \tag{28}$$

$$I(D; M|f(M)) = H(M|f(M)) - H(M|f(M), D) \tag{29}$$

By substituting 28 and 29 into 27 we obtain:

$$I(D; f(M)) = I(M; f(M)) - I(M; f(M)|D) \tag{30}$$

B Proof of Theorem 2

Here we prove Theorem 2. To evaluate the mutual information

$$I(M; v(f(M), g(W))) = H(M) - H(M|v(f(M), g(W))), \tag{31}$$

we first evaluate the entropy $H(M)$, we then evaluate the conditional entropy $H(M|v(f(M), g(W)))$.

Before performing any search, the server knows the number of keywords in the metadata. Hence,

$$H(M) = \sum_l P(l)(H(W_1) + ... + H(W_l)) \tag{32}$$

Here, W_i denotes the i-th keywords in the metadata. According to assumption 4 each keyword in the metadata is chosen uniformly, therefore for each i and j, $H(W_i) = H(W_j)$. Let, $H(W_i) = H(W)$, since, $H(M) = \sum_l lP(l)H(W)$ and the expected number of keywords is, $\mathcal{E}(L) = \sum_l P(l)l$, we obtain, $H(M) = \mathcal{E}(L)H(W)$.

We now evaluate the conditional entropy $H(M|v(f(M), g(W)))$. Let s be the expected number of keywords in metadata items that matches the query. After a search has performed, the server learns the keywords that match the query. Therefore, the conditional entropy is computed as follows:

$$H(M|v(f(M), g(W))) = \sum_l P(l)H(W_1)+...+H(W_l)-\sum_s P(s)H(W_1)+...+H(W_s) \tag{33}$$

Hence, the mutual information $I(M; v(f(M), g(W))$ is evaluated below:

$$I(M; v(f(M), g(W))) = \mathcal{E}(L)H(W) - (\mathcal{E}(L)H(W) + \mathcal{E}(S)H(W)) = \mathcal{E}(S)H(W) \tag{34}$$

Evaluating the expected number of repeated keywords s, we have, $\mathcal{E}(s) = \sum_s P(s)$. The probability $P(s)$ is calculated as $P(s) = (\frac{1}{|\mathcal{W}|})^s(\frac{|\mathcal{W}|-1}{|\mathcal{W}|})^{l-s}$. Let assume $|\mathcal{W}| >> 1$, then $P(s) = (\frac{1}{|\mathcal{W}}|)^s$. Therefore, we obtain, $\mathcal{E}(S) = \sum_{s=1}^{|\mathcal{W}|} s(\frac{1}{|\mathcal{W}|})^s$. In case $|\mathcal{W}| >> 1$, $\mathcal{E}(S) \approx \sum_{s=1}^{|\mathcal{W}|} \frac{1}{|\mathcal{W}|}$.

C Proof of Theorem 3

Here we prove Theorem 3. Write the mutual information $I(M; f(M)) = H(M) - H(M|f(M))$. We first evaluate the entropy $H(M)$, we then evaluate the conditional entropy $H(M|f(M))$.

According to standard information theoretic rules, the entropy $H(M) = -\sum_m P(m)log(P(m))$. We extend the probability function $P(m)$ to $P(m) = \sum_l P(m|l)P(l)$. Hence, the entropy $H(M)$ is evaluated below:

$$H(M) = -\sum_m \sum_l P(m|l)P(l)log(\sum_l P(m|l)P(l)). \tag{35}$$

The conditional probability $P(m|l)$ is evaluated as follows:

$$P(m|l) = \begin{cases} \frac{1}{\binom{\mathcal{W}}{l}} \text{ if } |m| = l \\ 0 \quad \text{Otherwise} \end{cases}$$

The conditional probability $P(m|l)$ below:

$$P(m|l) = \frac{\delta(|m|, l)}{\binom{\mathcal{W}}{l}} \tag{36}$$

where, $\delta(|m|, l) = \begin{cases} 1 \text{ if } |m| = l \\ 0 \text{ Otherwise} \end{cases}$ Hence,

$$\sum_l P(m|l) = \frac{1}{\binom{\mathcal{W}}{l}} \tag{37}$$

Hence,

$$H(M) = -\sum_l \sum_{m=1}^{\binom{\mathcal{W}}{l}} \frac{1}{\binom{\mathcal{W}}{l}} P(l)log(\frac{P(l)}{\binom{\mathcal{W}}{l}}) = -\sum_l P(l)log(\frac{P(l)}{\binom{\mathcal{W}}{l}}) \tag{38}$$

We now evaluate the conditional entropy $H(M|f(M))$. Knowing the searchable representation $f(m)$ the server learns the number of unique keywords l in the metadata. Hence, $H(M|f(M)) = H(M|L)$. The conditional Entropy $H(M|L)$ is evaluated below:

$$H(M|L) = -\sum_m \sum_l P(m|l)P(l)log(P(m|l)) \tag{39}$$

By substituting 37 into 39,

$$H(M|L) = -\sum_l \sum_{m=1}^{\binom{\mathcal{W}}{l}} P(l)log(\frac{1}{\binom{\mathcal{W}}{l}}) \tag{40}$$

Hence, the evaluation of the mutual mutual information $I(M; f(M)) = H(M) - H(M|L)$ is as follows,

$$H(M) - H(M|L) = -\sum_l P(l)log(P(l)) \tag{41}$$

Since $H(L) = -\sum_l P(l)log(P(l))$, obtain $I(M; f(M)) = H(L)$

D Proof of Theorem 4

Here we prove Theorem 4. Let j be the number of bits of hash values, l be the number of hash values in the searchable representation and $|m|$ be the number of keywords in the metadata. Using standard information theoretic rule, the conditional entropy $H(M|L)$ is evaluated as follows;

$$H(M|L) = P(m,l)logP(m|l) \tag{42}$$

We split the joint probability $P(m,l)$ to the number of keywords, $P(m,l) = \sum_{|m|} P(m,l||m|)P(|m|) = \sum_{|m|} P(m||m|)P(|m||l)P(|m|)$, where, $P(m||m|) = \frac{1}{\binom{|\mathcal{W}|}{|m|}}$. We evaluate the conditional probability $P(|m||l)$. Since $|m| \geq l$,

$$P(|m||l) = \begin{cases} 0 & \text{if } l > |m| \\ (\frac{2^j-1}{2^j})...(\frac{2^j-l+1}{2^j})(\frac{l}{2^j})^{|m|-l} & \text{if } l \leq |m| \end{cases} \tag{43}$$

Hence,

$$P(m|l) = \begin{cases} 0 & \text{if } l > |m| \\ \frac{(\frac{2^j-1}{2^j})...(\frac{2^j-l+1}{2^j})(\frac{l}{2^j})^{|m|-l}}{\binom{\mathcal{W}}{\|m|}} & \text{if } l \leq |m| \end{cases} \tag{44}$$

To evaluate $H(M|L)$, write, $H(M|L) = P(m,l)log(P(m|l))$. The conditional probability $P(m|l)$ is evaluated below,

$$P(m,l) = \sum_{|m|=l}^{|\mathcal{W}|} P(m|l,|m|)P(|m|) \tag{45}$$

Since, $P(m|l,|m|) = P(m||m|)$,

$$P(m|l,|m|) = \frac{1}{\binom{|\mathcal{W}|}{|m|}} \tag{46}$$

Hence, the conditional entropy is calculated as follows,

$$H(M|L) = \sum_{|m|=1}^{|\mathcal{W}|} \sum_{l=1}^{i} \frac{(2^j)!l^{|m|-l}}{(2^j-l)!2^{j|m|}} P(|m|)log(\frac{(2^j)!l^{|m|-l}}{(2^j)!2^{j|m|}\binom{\mathcal{W}}{|m|}}) \tag{47}$$

Using (38),

$$H(M) = -\sum_{l}^{|\mathcal{W}|}(\frac{P(l)}{\binom{\mathcal{W}}{l}})P(l)log(\frac{P(l)}{\binom{\mathcal{W}}{l}}) \tag{48}$$

Hence, $I(M; f(M))$ is evaluated as follows,

$$I(M; f(M)) = H(L) - \sum_{|m|} P(|m|)log(\frac{1}{\binom{\mathcal{W}}{i}}) + \sum_{|m|=1}^{|\mathcal{W}|} \sum_{l=1}^{i} \frac{(2^j)!l^{|m|-l}}{(2^j-l)!2^{j|m|}} P(|m|)log(\frac{(2^j)!l^{|m|-l}}{(2^j)!2^{j|m|}\binom{\mathcal{W}}{|m|}}) \tag{49}$$