1 Constructions

In this section, we motivate and give several constructions of searchable encryption.

1.1 Additional Notations

Let $W(\cdot)$ denotes a function that takes in a file and returns the set of keywords associated to it. We write $P_a^{W(db)}$ to mean a permutation on $W(db)$. The permutation is a product of $\frac{|W(db)|}{a}$ disjoint cycles of length a. When the keyword set is well-understood, we omit it from the notation. Let W be a set of keywords, we abuse the notation $P_a(W)$ to mean $P_a(W) = \{P_a^n(w) \mid w \in W\}$.

1.2 Motivation

From previous works, we know that the leakage of most searchable encryptions is repetition of keywords in files, meaning that if the same keyword appears in two different documents, the adversary is aware of that. There are constructions that hides this information from the adversary in the passive case but this is nonetheless leaked through queries. There are many known attacks that abuse this information to perform inference attack. Our goal is to construct a scheme that is secure against any inference attack.

To motivate, we start by describing the attacks. The basic attack is the well-known frequency analysis. Upon obtaining the information on the database, the adversary computes the frequency of each encrypted keyword. The information is then compared to his prior knowledge on the distribution. The encrypted keywords are matched to their most likely unencrypted counterparts. Naveed et al. [10] have shown that this attack is very efficient and accurate on attribute-based encrypted tables, and their attack can be extended to other forms of encrypted databases easily.

The more advanced attacks aim to improve accuracy of keyword recovery by making use of co-occurrences of keywords. In this case, instead of looking at frequency of keywords one by one, the adversary looks at pairs of keywords and find frequencies on them. The frequencies computed is usually put into a matrix, known as the co-occurrence matrix and some optimization algorithm is performed to find a permutation between keywords and their encryptions which fits the objective function the best. It has been shown that attacks of this form can recover a substantial amount of plaintext-ciphertext pairs even if frequency analysis fails so.

Attack of this genre does not stop here. We imagine there are adversaries who make use of higher order information such as frequency of every three keywords occur together and more. So ultimately, a necessary condition for a scheme to be secure is that it is secure against inference attack that abuses co-occurrence matrices at all levels.

All of our constructions rely on a permutation P_a. The idea is that we transform the original database into a form such that after permuting the keywords with P_a, the view of the encrypted database should stay the same. Thus, the adversary cannot distinguish if the original keywords are those without the permutation or the ones with it. Of course, since P_a is a permutation, we can apply P_a again and the view of the encrypted database is still unchanged. In fact, there are (at least) a different orientations of the transformed database such that the encrypted views are the same.

1.3 Co-occurrence Matrices and Permutability

In this subsection, we give definition of co-occurrence matrices and permutable co-occurrence matrices, and show how these can help us achieve security.

Definition 1 (Co-occurrence at level l with Documents)**.** *Co-occurrence at level l on database db with documents is defined to be $DC_l^{db} : W(db)^l \to \mathcal{P}(F)$.*

$$DC_l^{db}(w_1, \ldots, w_l) := \{f \mid (w_i \neq w_j \forall i, j) \wedge (f \in F) \wedge (\{w_1, \ldots, w_l\} = W(f))\}. \tag{1}$$

Sometimes, only the counts in the co-occurrence matrix is of interest to us. So we define the corresponding co-occurrence matrix to be:

Definition 2 (Co-occurrence at level l with Counts)**.** *Co-occurrence at level l on database db with counts is defined to be $CC_l^{db} : W(db)^l \to \mathbb{N}$.*

$$CC_l^{db}(w_1, \ldots, w_l) := \left|DC_l^{db}(w_1, \ldots, w_l)\right|. \tag{2}$$

When the database *db* is well-understood, we omit it from the notation. With this definition, we can talk about co-occurrence of arbitrary number of keywords. For example, counts of keywords corresponds to CC_1, and co-occurrence is CC_2. Similar definition can be made with the encrypted database, we give them briefly here.

Definition 3 (Co-occurrence at level d with Encrypted Documents)**.** *Co-occurrence at level l on encrypted database edb with encrypted documents is defined to be $EDC_l^{edb} : EW(edb)^l \to \mathcal{P}(EF)$.*

$$EDC_l^{edb}(ew_1, \ldots, ew_l) := \{ef \mid (ew_i \neq ew_j \forall i, j) \wedge (ef \in edb) \wedge (\{ew_1, \ldots, ew_l\} = \mathsf{Dec}(ef))\}. \tag{3}$$

Definition 4 (Co-occurrence at level l with Encrypted Counts)**.** *Co-occurrence at level d on database db with encrypted counts is defined to be $ECC_l^{edb} : EW(edb)^l \to \mathbb{N}$.*

$$ECC_l^{edb}(ew_1, \ldots, ew_l) := \left|EDC_l^{edb}(ew_1, \ldots, ew_l)\right|. \tag{4}$$

Finally, we can define the collection of co-occurrences at all levels with the following definition.

Definition 5 (Co-occurrences at all levels)**.** *Co-occurrences at all levels with documents is $DC^{db} = \left\{DC_1^{db}, \ldots, DC_{W(db)}^{db}\right\}$. Co-occurrences at all levels with counts is $CC^{db} = \left\{CC_1^{db}, \ldots, CC_{W(db)}^{db}\right\}$.*

Co-occurrences at all levels with encrypted documents is $EDC^{edb} = \left\{EDC_1^{edb}, \ldots, EDC_{W(edb)}^{edb}\right\}$. Co-occurrences at all levels with encrypted counts is $ECC^{edb} = \left\{ECC_1^{edb}, \ldots, ECC_{EW(edb)}^{edb}\right\}$.

All of our constructions rely on transforming co-occurrences into a way such that the adversary can no longer recover the information he wants. The property we achieve with our transformation is what we called permutability.

Definition 6 (Permutability)**.** *Given co-occurrence C of any form above, and a permutation P on a coordinate of the input, we say C is P-permutable if*

$$C_l(w_1, \ldots, w_l) = C_l(P(w_1), \ldots, P(w_l)) \tag{5}$$

for all $w_1, \ldots, w_l$.

We say that co-occurrence at all levels are permutable by P if all co-occurrences are permutable by P.

In dimensions 1 and 2, this is easier to understand in terms of vectors and matrices. The following is an example of $(w_1\ w_2\ w_3)$-permutable co-occurrence count in matrix form:

$$\begin{bmatrix} 0 & 5 & 5 \\ 5 & 0 & 5 \\ 5 & 5 & 0 \end{bmatrix} \tag{6}$$

Note that co-occurrences are always symmetric due to the definition. Now imagine these are the only documents in our database. If we use a secure deterministic encryption scheme to encrypt these keywords, the adversary is going to see a permuted version of this co-occurrence. But he is not able to tell which ciphertext corresponds to which keyword because all pairs of keywords have the same count. Our encryption schemes essentially takes co-occurrences (at all levels) in any shape, and transform them into these permutable matrices. With a clever choice of permutation P, we can achieve desired security with respect to some gain functions. In the next subsection, we will show how to transform arbitrary co-occurrence into a permutable one.

1.4 Our Constructions

In this section, we describe our constructions. All of the constructions are secure against keyword guessing attack (it is going to be a security notion slightly different from the one defined above, see later) but they achieve different level of security against other attacks.

1.5 Permutation with Adding Fake Documents

One straightforward way to achieve the permutation property is to pad fake documents. Let $(\overline{\mathsf{KGen}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}})$ be a secure encryption scheme on searchable encryption (i.e. with leakage being repetition of keywords). Let $\mathsf{KGen}'(a, W(db))$ be a key generation scheme that takes in a positive integer a which divides $|W(db)|$ and set of keywords in db, and returns a permutation on $W(db)$ that is a product of disjoint cycles of length a. Let $\texttt{FGen}(1^n, W)$ be a function that generates a random fake file with the given set of keywords W. We can define scheme 1 to be $\texttt{scheme1} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ in the following way.

```
$\mathsf{KGen}(n, a, W(db))$
$\mathrm{k} \leftarrow \overline{\mathsf{KGen}}(1^n)$
$P_a \leftarrow \mathsf{KGen}'(a, W(db))$
return $(\mathrm{k}, P_a)$
```

```
$\mathsf{Enc}(1^n, (\mathrm{k}, P_a), db)$
$db' = ()$
for $f \in db$ :
  $db' = db' + (W(f), (True, f))$
  for $i = 1, \dots, a-1$ :
    $db' = db' + ((P_a)^i(W(f)), (False, \texttt{FGen}(1^n, (P_a)^i(W(f))))$
return $\overline{\mathsf{Enc}}(1^n, \mathrm{k}, db')$
```

```
$\mathsf{Dec}(1^n, (\mathrm{k}, P_a), edb)$
$db = ()$
for $ef \in edb$ :
  $(W, (b, f)) \leftarrow \overline{\mathsf{Dec}}(1^n, (\mathrm{k}, P_a), ef)$
  if $b = True$ :
    $db = db + ((W, f))$
return $db$
```

Insertion and deletion are straightforward and other functions, namely trapdoor generation and search, are taken from the aforementioned encryption scheme directly so we will omit them here.

1.6 Scheme in IKK paper

In the paper written by Islam et al. [11], the authors introduced an encryption scheme that is secure against inference attack. Their security notion is different from the one we described in section **??**. We recall their definition and results here.

Definition 7 ((α, t)-secure index)**.** *We say an $m \times n$ binary index matrix ID is (α, t)-secure for a given $\alpha \in [1, m]$ and $t \in \mathcal{N}$ if there exists a set of partitions $\{S_i\}$ s.t. the following holds.*

1. *The partition set is complete. That is $\cup_i S_i = [1, m]$.*
2. *The partitions in the set are non-overlapping. That is, $\forall i, j, S_i \cap S_j = \emptyset$.*
3. *Each partition has at least α rows. That is, $\forall j, |S_j| \geq \alpha$.*
4. *Finally, the hamming distance between any two rows of a given partition j is at most t. That is, $\forall i_1, i_2 \in S_j, d_H(ID_{i_1}, ID_{i_2}) \leq t$.*

Intuitively, (α, t)-secure index is a measure on how indistinguishable keywords are. If we want to achieve better security, we want to have large α and small t. Indeed, the authors' goal is to find a lossless (there is no deletion of documents or keywords) transformation from the given index to an $(\alpha, 0)$-secure index, for some α specified by the user as the security parameter. The authors have given a theorem on $(\alpha, 0)$-secure index as follows.

Theorem 8. *Let ID be an $m \times n$ $(\alpha, 0)$-secure index for some $0 < \alpha \leq m$. Given the response of a query $q_i = Trapdoor_w$ for some keyword w and the complete ID matrix, an attacker can successfully identify the keyword w with probability at most $\frac{1}{a}$ by just using background information related to ID matrix.*

It is important to note that the adversary described in the above theorem is different from the one we discussed in definition **??**. In particular, we have here that the adversary cannot guess any keyword from its encryption with probability greater than $\frac{1}{a}$, which is independent from the keyword frequencies. In fact, all of our schemes have this property.

To transform an index to an $(\alpha, 0)$-secure one, the authors used a clustering algorithm to pad the original index with fake keywords.

1.7 Permutation with Adding Fake Keywords

In this section, we describe a scheme that is very similar to that in the IKK paper. The main difference is that our scheme is dynamic, i.e. it supports document insertions and deletions. The idea is similar: our goal is to pad fake keywords so that groups of a keywords are indistinguishable from each other. Again, we can use a permutation P_a to achieve this.

Let ($\overline{\mathsf{KGen}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}}$) be a secure encryption scheme on searchable encryption (i.e. with leakage being repetition of keywords). Let $\mathsf{KGen}'(a, W(db))$ be a key generation scheme that takes in a positive integer a which divides $|W(db)|$ and set of keywords in db, and returns a permutation on $W(db)$ that is a product of disjoint cycles of length a. We can define scheme 1 to be $\texttt{scheme2} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ in the following way.

```
$\mathsf{KGen}(n, a, W(db))$
    $\mathrm{k} \leftarrow \overline{\mathsf{KGen}}(1^n)$
    $P_a \leftarrow \mathsf{KGen}'(a, W(db))$
    return $(\mathrm{k}, P_a)$
```

```
$\mathsf{Enc}(1^n, (\mathrm{k}, P_a), db)$
    $db' = ()$
    for $f \in db$ :
    $W' = \cup_{i=0}^{a-1} (P_a)^i (W(f))$
        $db' = db' + (W', f)$
    return $\overline{\mathsf{Enc}}(1^n, \mathrm{k}, db')$
```

```
$\mathsf{Dec}(1^n, (\mathrm{k}, P_a), edb)$
    $db = ()$
    for $ef \in edb$ :
    $db = db + \mathsf{Dec}(ef)$
    return $db$
```

Insertion and deletion are straightforward and other functions, namely trapdoor generation and search, are taken from the aforementioned encryption scheme directly so we will omit them here.

1.8 Permutation with a Documents

Our last scheme takes a step further and apply the permutation to a set of files instead of one file each time. All functions except the encryption function for this scheme are the same as that of `scheme2` so we will not repeat it here. For simplicity, we assume that the number of documents in db is a multiple of a. The encryption scheme for `scheme3` is the following.

```
$\mathsf{Enc}(1^n, (\mathrm{k}, P_a), db)$
$db' = ()$
for $\{f_1, \cdots, f_a\} \leftarrow_{\$} db$ :
    $W' = \cup_{i=1}^{a} (P_a)^{a-i+1}(W(f_i))$
    $db' = db' + ((W', f(f_1)))$
    for $j = 2, \cdots, a$ :
        $W' = (P_a)(W')$
        $db' = db' + ((W', f(f_j)))$
    $db = db \backslash \{f_1, \cdots, f_a\}$
return $\overline{\mathsf{Enc}}(1^n, \mathrm{k}, db')$
```

Insertion (of a documents each time) is straightforward and other functions, namely trapdoor generation and search, are taken from the aforementioned encryption scheme directly so we will omit them here. The drawback of this scheme is that deletion cannot be done straightaway in a way such that security is still preserved. One way to achieve deletion is to delete the target file and retrieve the other $a-1$ files with keywords permuted from the target file, and insert them back with future insertions or documents that are already in local cache. It is also possible to store this information on the server side with an IND-CPA secure scheme without the the index structure.

2 Security Analysis

In this section, we analyse security of the schemes we described above. We begin by defining and proving the leakage of our schemes. The, we show that all of our schemes are secure against the inference attack (different from definition **??**) we will define in this section. We show that security against inference attack is not sufficient. Namely, a scheme that is secure against inference attack can still leak a lot of information about the unencrypted database. We demonstrate this with a gain function and analysis of this gain function on the schemes we proposed.

2.1 Cryptographic Proofs

IND-DCPA is equivalent to Indistinguishability under Permutation. In this part, we prove equivalence between IND-DCPA and indistinguishability under permutation. The first notion is used to exam security of deterministic encryption schemes. We recall it here.

Definition 9 (IND-DCPA). *Let* $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *be a deterministic encryption scheme. Let* $\mathcal{A}$ *be an adversary that has access to a left-right encryption oracle and* $b \in \{0, 1\}$ *a random bit. Consider the following experiment:*

```
$\underline{Exp_{\mathcal{A}}^{IND\text{-}DCPA\text{-}b}(n)}$
$k \leftarrow_{\$} \mathsf{KGen}(1^n)$
$b' \leftarrow \mathcal{A}^{\mathsf{Enc}(k, \mathcal{LR}(\cdot,\cdot,b))}$
**return** $b'$
```

It is required that the messages queried to the left oracle are all unique, so does the right oracle. The IND-DCPA advantage is defined to be

$$Adv_{\mathcal{A}}^{IND-DCPA}(n) = \left|\Pr\left[Exp_{\mathcal{A}}^{IND\text{-}DCPA\text{-}1}(n) = 1\right] - \Pr\left[Exp_{\mathcal{A}}^{IND\text{-}DCPA\text{-}0}(n) = 1\right]\right|. \tag{7}$$

A scheme is said to be secure in IND-DCPA if for any PPT adversary $\mathcal{A}$, the advantage above is negligible in n.

Intuitively, if a scheme is IND-DCPA secure, no adversary should be able to identify the right permutation between plaintexts and ciphertexts with probability better than random guessing. We formalise this idea as a security notion we call indistinguishability for deterministic encryption under permutation (IND-DP).

Definition 10 (IND-DP)**.** *Let $(\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ be a deterministic encryption scheme. Let $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ be an adversary where $\mathcal{A}_0$ is an adversary that given security parameter, generates a list of non-repeating messages and a permutation on messages; $\mathcal{A}_1$ is an adversary that given security parameter, a list of encrypted messages and a permutation on the plaintexts, determine if the encryption is an encryption of original messages or those in permuted order. Let $b \in \{0, 1\}$ be a random bit. The experiment for IND-DP can be described as follows.*

```
$\underline{Exp_{\mathcal{A}}^{IND\text{-}DP\text{-}b}(n)}$
$k \leftarrow_{\$} \mathsf{KGen}(1^n)$
$((m_0, \cdots, m_l), \sigma) \leftarrow \mathcal{A}_0(1^n)$
$M_0 = (m_0, \cdots, m_l); M_1 = (\sigma(m_0), \cdots, \sigma(m_l))$
$C \leftarrow \mathsf{Enc}(1^n, k, M_b)$
$b' \leftarrow \mathcal{A}_1(1^n, (m_0, \cdots, m_l), \sigma, C)$
**return** $b'$
```

It is required that m_i are all unique. The IND-DP advantage is defined as:

$$Adv_{\mathcal{A}}^{IND-DP}(n) = \left|\Pr\left[Exp_{\mathcal{A}}^{IND\text{-}DP\text{-}1}(n) = 1\right] - \Pr\left[Exp_{\mathcal{A}}^{IND\text{-}DP\text{-}0}(n) = 1\right]\right|. \tag{8}$$

A scheme is said to be secure in IND-DP if for any PPT adversary $\mathcal{A}$, the advantage above is negligible in n.

Now we will show that a scheme is IND-DCPA secure if and only if it is IND-DP secure. We formalise it in the following theorem.

Theorem 11 (IND-DCPA and IND-DP are Equivalent)**.** *A scheme is IND-DCPA secure if and only if it is IND-DP secure.*

Proof. (IND-DCPA ⇒ IND-DP) We show the contrapositive is true. Suppose a scheme is not secure under IND-DP, so there is some list of non-repeating messages M_0 and permutation σ such that the adversary for IND-DP can distinguish encryption of M_0 from M_1 (as those defined in the experiment above). We construct an IND-DCPA adversary $\mathcal{B}$ as follows:

- Let the message to the left oracle be M_0, and that to the right oracle be M_1.

- Query pairs of messages one by one until $C = \mathsf{Enc}(\mathrm{k}, M_b)$ is obtained.
- Run $\mathcal{A}_1(1^n, M_0, \sigma, C)$ for IND-DP, return what the adversary returns.

If $M = (m_0, \cdot, m_l)$, and σ a permutation on $\{m_0, m_1, \cdot, m_l\}$, we abuse the notation and write $\sigma(M)$ to mean $(\sigma(m_0), \cdot, \sigma(m_l))$. Working out the advantages, we see that:

$$\begin{aligned}
&\mathrm{Adv}_{\mathcal{B}}^{IND-DCPA}(n)\\
=&\left|\Pr\left[\mathrm{Exp}_{\mathcal{B}}^{\text{IND-DCPA-1}}(n) = 1\right] - \Pr\left[\mathrm{Exp}_{\mathcal{B}}^{\text{IND-DCPA-0}}(n) = 1\right]\right|\\
=&\left|\Pr[\mathcal{A}_1(1^n, M_0, \sigma, \mathsf{Enc}(\sigma(M_0))) = 1] - \Pr[\mathcal{A}_1(1^n, M_0, \sigma, \mathsf{Enc}(\sigma(M_0))) = 0]\right|\\
=&\left|\Pr\left[\mathrm{Exp}_{\mathcal{A}}^{\text{IND-DP-1}}(n) = 1\right] - \Pr\left[\mathrm{Exp}_{\mathcal{A}}^{\text{IND-DP-0}}(n) = 1\right]\right|\\
=&\mathrm{Adv}_{\mathcal{A}}^{IND-DP}(n).
\end{aligned}$$

So the forward implication is verified.

(IND-DP $\Rightarrow$ IND-DCPA) The reverse direction is straightforward. Suppose $\mathcal{A}$ is an adversary against IND-DCPA. We construct IND-DP adversary $\mathcal{B}$ as follows:

- Generate a list of non-repeating plaintexts M_0 and a permutation σ.
- Upon obtaining the list of ciphertexts C, run $\mathcal{A}^{\mathsf{Enc}(\mathrm{k}, \mathcal{LR}(M_0, \sigma(M_0), b))}$. Return what the adversary returns.

By analysing the advantage of the corresponding IND-DP adversary, we see that:

$$\begin{aligned}
&\mathrm{Adv}_{\mathcal{B}}^{IND-DP}(n)\\
=&\left|\Pr\left[\mathrm{Exp}_{\mathcal{B}}^{\text{IND-DP-1}}(n) = 1\right] - \Pr\left[\mathrm{Exp}_{\mathcal{B}}^{\text{IND-DP-0}}(n) = 1\right]\right|\\
=&\left|\Pr\left[\mathcal{A}^{\mathsf{Enc}(\mathrm{k}, \mathcal{LR}(M_0, \sigma(M_0), 1))} = 1\right] - \Pr\left[\mathcal{A}^{\mathsf{Enc}(\mathrm{k}, \mathcal{LR}(M_0, \sigma(M_0), 1))} = 0\right]\right|\\
=&\left|\Pr\left[\mathrm{Exp}_{\mathcal{A}}^{\text{IND-DCPA-1}}(n) = 1\right] - \Pr\left[\mathrm{Exp}_{\mathcal{A}}^{\text{IND-DCPA-0}}(n) = 1\right]\right|\\
=&\mathrm{Adv}_{\mathcal{A}}^{IND-DCPA}(n).
\end{aligned}$$

So the reverse implication is verified. □

In fact IND-DP says a bit more than what is in the experiment. From the experiment, we see that $\mathsf{Enc}(M)$ and $\mathsf{Enc}(\sigma(M))$ are indistinguishable. But since $\mathsf{Enc}(\cdot)$ is a deterministic encryption, there is some permutation τ on the ciphertexts such that $\mathsf{Enc}(\sigma(M)) = \tau(\mathsf{Enc}(M))$. Therefore, the order of ciphertext looks completely random in the view of the adversary. Since deterministic encryption itself can be seen as a bijection between plaintexts and ciphertexts, we conclude that IND-DCPA secure deterministic encryption schemes acts like a random bijection.

Leakage as Co-occurrences. We have so far not specified any encryption scheme to build the index so the leakage is not fixed for the overall scheme. Now we consider those schemes whose leakage is repetition of keywords in files. Let $db(W)$ be a function to return all documents containing keywords W, and $edb(EW)$ be a function to return all encrypted documents containing encrypted keywords EW. We abuse the notation $\mathsf{Enc}(W)$ to mean encryption of the keywords in the search index. Then formally, we have that for all keyword sets W, $|db(W)| = |edb(\mathsf{Enc}(W))|$.

Since we are only interested in the security of the index (assuming the documents are securely encrypted and there is no leakage), the unencrypted database is reduced to co-occurrence at all levels with documents, and the encrypted index is reduced to co-occurrence at all levels with encrypted documents. Leakage described above is just the count version of the two.

2.2 All of our Constructions are Secure against Keyword Guessing Attack

In this subsection, we show that all of our schemes are secure against keyword guessing attack. We will use extended posterior g-vulnerability to analyse the schemes. For simplicity, we assume that the database is a point distribution, i.e. the adversary knows exactly how the database looks like. The gain function we consider here is the following:

- $\mathcal{W} = W(db) \times EW(edb)$,
- $\mathcal{X} = DC \times \mathcal{P}$,
- $\mathcal{Y} = \mathcal{L}_{\mathcal{M}}(\mathcal{X}) = \{\mathsf{Enc}(DC, \sigma) \mid \sigma \in \mathcal{P}\}$,
- $g((m, c), (DC, \sigma), y) = \mathbb{1}(\sigma(m) = c)$.

Here $\mathcal{P}$ denotes the set of all possible bijections from $W(db)$ to $EW(edb)$, and $\mathsf{Enc}(DC, \sigma)$ is an idealised encryption function that encrypts with bijection σ and returns co-occurrences at all levels with encrypted documents. The gain function is 1 if the adversary can guess any pair of keyword and its encryption correctly in one try, and 0 otherwise. Here, the permutation P_a we used for the encryption is assumed to be known to the adversary. He can recover this piece of information by file injection attack or inspecting the algebraic structure of the leakage (since we are working with an information-theoretic adversary, this is permitted).

Furthermore, we assume in this section that the unencrypted database is singular, i.e. co-occurrences at all levels with counts is not permutable by any permutation except the identity permutation. This is an assumption that is opposite of that in [4]. We argue that their assumption is unrealistic because even for simple databases this assumption does not hold, e.g. $db = ((\{1\}, f_0), (\{1\}, f_1), (\{2\}, f_2))$ and one queries keyword 1 and 2 once each. In fact, it is evident from the known attacks that non-singularity is not a reasonable assumption. Furthermore, non-singularity is not a very strong classification on encrypted databases. For instance, one can achieve non-singularity by making only two keywords indistinguishable and reveal all other information.

In the next part of the subsection, we first show that any scheme that preserves singularity after encryption is not secure in our notion. We then prove that permutability can be used to achieve security. Finally, we prove that all of our constructions satisfies some form of permutability, thus they are secure under our notion.

Insecure Schemes are Insecure. We present the result as a theorem.

Theorem 12. *Let the distribution of database be a fixed distribution π, i.e. the adversary knows that the unencrypted database is db. Let $\mathcal{X}$ be the secrets and $\mathcal{Y}$ be the leakage. given $y \in \mathcal{Y}$, and $\mathcal{L}_{\mathcal{M}}$ the leakage function for the encryption scheme. We assume that there is a unique $x \in \mathcal{X}$ such that $\mathcal{L}_{\mathcal{M}}(x) = y$. Then, for the gain function g defined above,*

$$EV_g(\pi, \mathcal{C}_{\mathcal{L}_{\mathcal{M}}}) = 1 \tag{9}$$

Proof.

$$
\begin{aligned}
&EV_g(\pi, \mathcal{C}_{\mathcal{L}_{\mathcal{M}}}) \\
=&\sum_{y\in\mathcal{Y}} \Pr[\mathcal{Y}=y] \max_{w\in\mathcal{W}} \sum_{x\in\mathcal{X}} \Pr[\mathcal{X}=x \mid \mathcal{Y}=y]\, g(w,x,y) \\
=&\sum_{y\in\mathcal{Y}} \Pr[\mathcal{Y}=y] \max_{w\in\mathcal{W}} \Pr\left[\mathcal{X}=\mathcal{L}_{\mathcal{M}}^{-1}(y) \mid \mathcal{Y}=y\right] g(w,\mathcal{L}_{\mathcal{M}}^{-1}(y),y) \\
=&\sum_{y\in\mathcal{Y}} \Pr[\mathcal{Y}=y] \max_{w\in\mathcal{W}} 1 \cdot g(w,\mathcal{L}_{\mathcal{M}}^{-1}(y),y) \\
=&\sum_{y\in\mathcal{Y}} \Pr[\mathcal{Y}=y] \cdot 1 \\
=&1.
\end{aligned}
$$

The proof uses the property that $\Pr[\mathcal{X}=x \mid \mathcal{Y}=y]$ is 1 when $x=\mathcal{L}_{\mathcal{M}}^{-1}(y)$ and 0 otherwise. Furthermore, since the adversary knows the pre-image completely, his maximal guess is 1 for every $g(\cdot)$. □

Permutability of Encryption implies Security. The encryption schemes we proposed do not have invertible leakage function. This is because the padding part of our schemes transforms the database into one that is P_a permutable, where P_a is part of the secret key in our schemes. We always assume that document identifiers are permuted randomly by the encryption scheme so there is no leakage by looking at the document identifiers only. Therefore, it suffices to look at the co-occurrences with counts. We formalise this as a theorem.

Theorem 13. *Let CC^{db} be a P_a-permutable co-occurrences at all levels with counts. Let Q be a bijection from $W(db)$ to $EW(edb)$ that represents the idealised IND-DCPA secure encryption function on keywords. Then $Q\cdot(P_a)^i$ for $i=1,\dots,a-1$ generates the same leakage as Q. Furthermore, no PPT adversary can guess the correct idealised encryption function on keywords with probability greater than $\frac{1}{a}$.*

Proof. Since CC^{db} is P_a-permutable, $(P_a)^i(CC^{db})=CC^{db}$ for all $i=0,\dots,a-1$. By definition, this means for all $i=0,\dots,a-1$, for all $l=1,\dots,|EW(edb)|$, for all $w_1,\dots,w_l\in EW(edb)^l$,

$$CC_l^{db}((P_a)^i(w_1),\dots,(P_a)^i(w_l)) = CC_l^{db}(w_1,\dots,w_l). \tag{10}$$

Recall from above that the encryption scheme for the index leaks repetition of keywords. So

$$CC_l^{db}(w_1,\dots,w_l) = ECC_l^{edb}(Q(w_1),\dots,Q(w_l)) \tag{11}$$

and

$$CC_l^{db}((P_a)^i(w_1),\dots,(P_a)^i(w_l)) = ECC_l^{edb}(Q\cdot(P_a)^i(w_1),\dots,Q\cdot(P_a)^i(w_l)). \tag{12}$$

Using equation 10, we obtain the desired result as

$$ECC_l^{edb}(Q(w_1),\dots,Q(w_l)) = ECC_l^{edb}(Q\cdot(P_a)^i(w_1),\dots,Q\cdot(P_a)^i(w_l)). \tag{13}$$

The second half of the theorem follows immediately from theorem 11. □

Because P_a is a permutation, for any $i\neq 0 \mod a$, $Q\cdot(P_a)^i(w)\neq Q(w)$ for all $w\in W(db)$. That is, if the adversary's guess is based on the wrong permutation, he cannot guess any pair of keyword and its encryption correctly. With this observation, we can compute extended posterior vulnerability with gain function for inference attack with the following proposition.

Proposition 14. *Let the distribution of database be a fixed distribution π, i.e. the adversary knows that the unencrypted database is db. Let CC^{db} be a P_a-permutable co-occurrence at all levels with counts. Let Q be a bijection from $W(db)$ to $EW(edb)$ that represents the idealised IND-DCPA secure encryption on keywords. Let g be the gain function defined at the start of the subsection. Then*

$$EV_g(\pi, \mathcal{C}_{\mathcal{L}_\mathcal{M}}) = \frac{1}{a}. \tag{14}$$

Proof. From theorem 13, we know that if Q is a bijection between keywords and their encryptions as an idealised encryption which can generate some given leakage, then so does $Q \cdot (P_a)^i$ for $i = 1, \ldots, a-1$. Due to indistinguishability of the encryption scheme, all these can be the actual encryption function with probability $\frac{1}{a}$. Putting everything together, we get

$$\begin{aligned}
&EV_g(\pi, \mathcal{C}_{\mathcal{L}_\mathcal{M}}) \\
=&\sum_{y\in\mathcal{Y}} \Pr[\mathcal{Y}=y] \max_{w\in\mathcal{W}} \sum_{x\in\mathcal{X}} \Pr[\mathcal{X}=x \mid \mathcal{Y}=y]\, g(w,x,y) \\
=&\sum_{y\in\mathcal{Y}} \Pr[\mathcal{Y}=y] \max_{w\in\mathcal{W}} \sum_{i=0}^{a-1} \Pr\left[\mathcal{X}=(db, Q_y\cdot(P_a)^i) \mid \mathcal{Y}=y\right] g(w,(db,Q_y\cdot(P_a)^i),y) \\
=&\sum_{y\in\mathcal{Y}} \Pr[\mathcal{Y}=y] \max_{w\in\mathcal{W}} \sum_{i=0}^{a-1} \frac{1}{a}\cdot g(w,(db,Q_y\cdot(P_a)^i),y) \\
=&\sum_{y\in\mathcal{Y}} \frac{\Pr[\mathcal{Y}=y]}{a} \max_{w\in\mathcal{W}} \sum_{i=0}^{a-1} g(w,(db,Q_y\cdot(P_a)^i),y) \\
=&\sum_{y\in\mathcal{Y}} \frac{\Pr[\mathcal{Y}=y]}{a}\cdot 1 \\
=&\frac{1}{a}.
\end{aligned}$$

The second line is obtained by re-writing the vulnerability. The third and fourth lines are obtained as a result of theorem 13. The fifth line is just re-arranging the equation. The sixth line is due to the fact that if the guess is correct for some i, then it cannot be correct for any other i. So the maximum over the expression is 1. Hence, the extended posterior vulnerability is $\frac{1}{a}$. □

Our Schemes are Secure against Inference Attack. We are now ready to show that our schemes are secure against inference attack. It suffices to show that the (unencrypted) padded database is P_a-permutable for all the schemes.

Theorem 15. *All of our schemes produces P_a-permutable padded databases.*

Proof. Since document identifiers are not relevant in this proof, we can simply look at the co-occurrences at all levels with counts.

(`scheme1`) Let CC be the co-occurrences at all levels with counts for the original database. We compute an expression for the padded co-occurrences CC' at all levels with counts and show that it is P_a-permutable. Pick an arbitrary level l, for some $w_1, \ldots, w_l \in W(db)^l$, we see that

$$CC'_l(w_1, \ldots, w_l) = \sum_{i=0}^{a-1} CC((P_a)^i(w_1), \ldots, (P_a)^i(w_l)).$$

Therefore,

$$
\begin{aligned}
&CC'_l((P_a)(w_1),\dots,(P_a)(w_l))\\
=&\sum_{i=0}^{a-1} CC((P_a)^{i+1}(w_1),\dots,(P_a)^{i+1}(w_l))\\
=&\sum_{i=0}^{a-1} CC((P_a)^{i}(w_1),\dots,(P_a)^{i}(w_l))\\
=&CC'_l(w_1,\dots,w_l).
\end{aligned}
$$

Since our choice of l and $w_1,\dots,w_l$ are arbitrary, we conclude that CC' is P_a-permutable at all levels.

(`scheme2`) Let CC be the co-occurrences at all levels with counts for the padded database. We check that for any level l and for any $w_1,\dots,w_l \in W(db)^l$, $CC_l(w_1,\dots,w_l) > 0$ implies $CC_l(w_1,\dots,w_l) = CC_l((P_a)(w_1),\dots,(P_a)(w_l))$. We show that for each document insertion, this holds, and by induction on the number of documents, this holds for the whole database. Suppose we insert some document with keyword $W = \{w'_1,\dots,w'_m\} \subset W(db)$. Then the set of keywords we have with padding is $W' = \cup_{i=0}^{a-1}(P_a)^i(W)$. Clearly, $(P_a)(W') = (W')$, so the statement is verified for the document. By induction on the number of documents, this holds for the whole database. Thus, CC is P_a-permutable.

(`scheme3`) We show that for groups of a documents inserted together, the permutation property holds. And then by induction on the number of documents, we have that the co-occurrences at all levels with counts for the padded database is P_a-permutable. Let $W_1,\dots,W_a$ be the sets of keywords for the documents. Padded keyword sets for the document can be written as $W'_j =$

$$
\begin{aligned}
&W'_j\\
=&\cup_{i=1}^{a}\,(P_a)^{a-i+j}(W_i)\\
=&\cup_{i=1}^{a}\,(P_a)(P_a)^{a-i+j-1}(W_i)\\
=&(P_a)\left(\cup_{i=1}^{a}(P_a)^{a-i+j-1}(W_i)\right)\\
=&(P_a)(W'_{j-1}).
\end{aligned}
$$

Thus, the co-occurrences with counts generated by the padded keyword sets are indeed P_a-permutable. By induction on the number of documents, the co-occurrences at all levels with counts for the whole database is P_a-permutable. □

Note: do we have to write down the induction explicitly?

2.3 Security against Inference Attack is not Enough

In this subsection, we show that it is not sufficient to judge security of an encryption scheme by its security against inference attack. We show an attack on one of our schemes that can recover significant amount of information. We use extended posterior g-vulnerability to compare security of our schemes against that attack. The comparison cannot be done in the traditional setting of indistinguishability-based notions, thus demonstrating the advantages of our approach.

Exact Keyword Set Guessing Attack. The goal of adversary for exact keyword set guess attack is to recover the exact set of keywords for some document. Without loss of generality, we can assume that the goal of the adversary is to recover the keyword set for the first encrypted file.

From above, we know that the set of encryption functions indistinguishable by the adversary is $Q \cdot (P_a)^i$ for $i = 1,\dots,a-1$ where Q is the true idealised encryption function. So in the leakage channel, we only have to look at these as secrets corresponding to encryption because all others result in $p(x \mid y) = 0$. To make it more interesting, we assume that the adversary does not have access to the exact unencrypted database (otherwise there are trivial attacks for some databases). Note that in this case, the encryption function mentioned above can still be recovered. This is because the adversary can work with aggregated information such as aggregated co-occurrences at level 1 and 2 with counts. Thus, we can model the secret as the set of keywords in the first encrypted document and the encryption function, and the leakage as the encrypted keywords in the first encrypted document. We define the leakage channel and gain function as follows.

- $\mathcal{W} = \mathcal{P}(W(db))$,
- $\mathcal{X} = \mathcal{P}(W(db)) \times \{Q \cdot (P_a)^i \mid i = 0, \ldots, a-1\}$,
- $\mathcal{Y} = \mathcal{P}(EW(edb))$,
- $g(w, (W, Q \cdot (P_a)^i), EW) = \mathbb{1}(w = W)$.

Security of Scheme1 against Exact Keyword Set Guessing Attack. Under the gain function above, we will show that extended posterior g-vulnerability of `scheme1` is $\frac{1}{a}$. We will then argue that by combining exact keyword set guess attack with semantics of keywords, one can break the security of the scheme.

Theorem 16. *Let the leakage channel and gain function be those described above. Furthermore, we assume that for all $i = 1, \ldots, a-1$, the real keyword set W for the first encrypted document has the property that $(P_a)^i(W) \neq W$. Then*

$$EV_g^{\textit{scheme1}}(\pi, \mathcal{C}_{\mathcal{L}_\mathcal{M}}) = \frac{1}{a}. \tag{15}$$

Proof. We observe that for any leakage $y = EW$, if we know the encryption function is $Q \cdot (P_a)^i$ for some i, then the set of keywords for the unencrypted document is exactly $W = (Q \cdot (P_a)^i)^{-1}(EW)$. Because $(P_a)^i(W) \neq W$ for all $i = 1, \ldots, a-1$, $(Q \cdot (P_a)^i)^{-1}(EW)$ is the exact set of keywords only when $(Q \cdot (P_a)^i)^{-1}$ is the correct decryption function. This happens with probability $\frac{1}{a}$. In terms of extended posterior g-vulnerability, we have

$$\begin{aligned}
&EV_g^{\texttt{scheme1}}(\pi, \mathcal{C}_{l_{\texttt{scheme1}}}) \\
=&\sum_{y \in \mathcal{Y}} \Pr[\mathcal{Y} = y] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr[\mathcal{X} = x \mid \mathcal{Y} = y]\, g(w, x, y) \\
=&\sum_{y \in \mathcal{Y}} \Pr[\mathcal{Y} = y] \max_{w \in \mathcal{W}} \sum_{i=0}^{a-1} \Pr\left[\mathcal{X} = ((Q \cdot (P_a)^i)^{-1}(y), Q \cdot (P_a)^i) \mid \mathcal{Y} = y\right] g(w, ((Q \cdot (P_a)^i)^{-1}(y), Q \cdot (P_a)^i)), y) \\
=&\sum_{y \in \mathcal{Y}} \Pr[\mathcal{Y} = y] \max_{w \in \mathcal{W}} \sum_{i=0}^{a-1} \frac{1}{a} \cdot g(w, ((Q \cdot (P_a)^i)^{-1}(y), Q \cdot (P_a)^i)), y) \\
=&\sum_{y \in \mathcal{Y}} \frac{\Pr[\mathcal{Y} = y]}{a} \max_{w \in \mathcal{W}} \sum_{i=0}^{a-1} g(w, ((Q \cdot (P_a)^i)^{-1}(y), Q \cdot (P_a)^i)), y) \\
=&\sum_{y \in \mathcal{Y}} \frac{\Pr[\mathcal{Y} = y]}{a} \cdot 1 \\
=&\frac{1}{a}.
\end{aligned}$$

□

There is in fact a more detrimental attack based on this. In the adversary above, the sets of keywords $(Q \cdot (P_a)^i)^{-1}(y)$ are equally likely because we do not care if the first encrypted document is a real or fake one. Suppose now our goal is to recover the exact keyword set for some real document (not necessarily the first one). By the algebraic structure of the encryption scheme, all of $(Q \cdot (P_a)^i)^{-1}(y)$ can be the keyword set for a real document. But in practice, they are not equally likely to happen. This is because keywords have semantic meaning so some set of keywords occur together more frequently than the others. By using this extra information, one can guess with success rate higher than $\frac{1}{a}$.

`Scheme2` and `scheme3` are more Secure. Now we show that `Scheme2` and `scheme3` achieves better security with respect to the gain function. We will do this by comparing the extended posterior g-vulnerability of the three schemes.

Theorem 17. *Let the leakage channel and gain function be those described above. Furthermore, we assume that for all $i = 1, \dots, a-1$, the real keyword set W for the first encrypted document has the property that $(P_a)^i(W) \neq W$. Then*

$$EV_g^{\mathit{scheme1}}(\pi, \mathcal{C}_{\mathcal{L}_{\mathit{scheme1}}}) \geq EV_g^{\mathit{scheme2}}(\pi, \mathcal{C}_{\mathcal{L}_{\mathit{scheme2}}}) \geq EV_g^{\mathit{scheme3}}(\pi, \mathcal{C}_{\mathcal{L}_{\mathit{scheme3}}}). \tag{16}$$

Proof. Let y be the set of encrypted keywords for the first encrypted file and Q be the real encryption function on the database. We have already derived $EV_g^{\texttt{scheme1}}(\pi, \mathcal{C}_{\mathcal{L}_{\texttt{scheme1}}})$ for `scheme1` so it suffices to work out the expression for `scheme2` and `scheme3`. We begin by writing down inverse of leakage functions for `scheme2` and `scheme3`.

$$\begin{aligned}
\mathcal{L}^{-1}_{\texttt{scheme2}}(y) &= \{z \subset (Q \cdot (P_a)^i)^{-1}(y) \mid (i = 0, \dots, a-1) \wedge (\cup_{i=0}^{a-1} (P_a)^i(z) = y)\} \\
\mathcal{L}^{-1}_{\texttt{scheme3}}(y) &= \{z \subset (Q \cdot (P_a)^i)^{-1}(y) \mid (i = 0, \dots, a-1)\}.
\end{aligned}$$

These can be written as

$$\begin{aligned}
\mathcal{L}^{-1}_{\texttt{scheme2}}(y, i) &= \{z \subset (Q \cdot (P_a)^i)^{-1}(y) \mid \cup_{i=0}^{a-1} (P_a)^i(z) = y\} \\
\mathcal{L}^{-1}_{\texttt{scheme3}}(y, i) &= \{z \subset (Q \cdot (P_a)^i)^{-1}(y)\},
\end{aligned}$$

with $\cup_{i=0}^{a-1} \mathcal{L}^{-1}_{\texttt{scheme2}}(y, i) = \mathcal{L}^{-1}_{\texttt{scheme2}}(y)$ and $\cup_{i=0}^{a-1} \mathcal{L}^{-1}_{\texttt{scheme3}}(y, i) = \mathcal{L}^{-1}_{\texttt{scheme3}}(y)$.

(Comparing `scheme1` and `scheme2`) The extended posterior g-vulnerability of `scheme2` can be written as

$$\begin{aligned}
& EV_g^{\texttt{scheme1}}(\pi, \mathcal{C}_{l_{\texttt{scheme2}}}) \\
={}& \sum_{y \in \mathcal{Y}} \Pr[\mathcal{Y} = y] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr_{\texttt{scheme2}}[\mathcal{X} = x \mid \mathcal{Y} = y]\, g(w, x, y) \\
={}& \sum_{y \in \mathcal{Y}} \Pr[\mathcal{Y} = y] \max_{w \in \mathcal{W}} \sum_{i=0}^{a-1} \sum_{z \in \mathcal{L}^{-1}_{\texttt{scheme2}}(y,i)} \Pr_{\texttt{scheme2}}\left[\mathcal{X} = (z, Q \cdot (P_a)^i) \mid \mathcal{Y} = y\right] g(w, (z, Q \cdot (P_a)^i)), y)
\end{aligned}$$

Since g is a binary gain function, it is only possible for the adversary to achieve positive gain if his guess is in the set $\mathcal{L}^{-1}_{\texttt{scheme2}}(y)$. So the guess function takes the form $w : \mathcal{Y} \to \mathcal{L}^{-1}_{\texttt{scheme2}}(y)$. Since the guess needs to be exactly the same as the real keyword set, we have

$$\begin{aligned}
& EV_g^{\texttt{scheme2}}(\pi, \mathcal{C}_{l_{\texttt{scheme2}}}) \\
={}& \sum_{y \in \mathcal{Y}} \Pr[\mathcal{Y} = y] \sum_{i=0}^{a-1} \Pr_{\texttt{scheme2}}\left[\mathcal{X} = (w(y), Q \cdot (P_a)^i) \mid \mathcal{Y} = y\right] g(w(y), (w(y), Q \cdot (P_a)^i)), y) \\
={}& \sum_{y \in \mathcal{Y}} \Pr[\mathcal{Y} = y] \sum_{i=0}^{a-1} \Pr_{\texttt{scheme2}}\left[\mathcal{X} = (w(y), Q \cdot (P_a)^i) \mid \mathcal{Y} = y\right].
\end{aligned}$$

But $\sum_{i=0}^{a-1} \Pr_{\texttt{scheme2}}\left[\mathcal{X} = (w(y), Q \cdot (P_a)^i) \mid \mathcal{Y} = y\right] \leq \frac{1}{a}$ as

$$\Pr_{\texttt{scheme2}}\left[\mathcal{X} = (w(y), Q \cdot (P_a)^i) \mid \mathcal{Y} = y\right] = \Pr_{\texttt{scheme2}}\left[\mathcal{X} = (P_a^j(w(y)), Q \cdot (P_a)^{i+j}) \mid \mathcal{Y} = y\right]$$

for all $j = 1, \ldots, a-1$. So

$$EV_g^{\texttt{scheme1}}(\pi, \mathcal{C}_{l_{\texttt{scheme2}}}) \leq \frac{1}{a} = EV_g^{\texttt{scheme1}}(\pi, \mathcal{C}_{l_{\texttt{scheme1}}}).$$

(Comparing scheme2 and scheme3) To see the inequality from scheme2 to scheme3, we observe that $\mathcal{L}^{-1}_{\texttt{scheme3}}(y) \subseteq \mathcal{L}^{-1}_{\texttt{scheme2}}(y)$. The two schemes also share the same set of possible encryption functions. So for any $z \in \mathcal{L}^{-1}_{\texttt{scheme3}}(y)$ and $i = 0, \ldots, a-1$,

$$\Pr_{\texttt{scheme3}}\left[\mathcal{X} = (z, Q \cdot (P_a)^i) \mid \mathcal{Y} = y\right] \leq \Pr_{\texttt{scheme2}}\left[\mathcal{X} = (z, Q \cdot (P_a)^i) \mid \mathcal{Y} = y\right].$$

Let $w : \mathcal{Y} \to \mathcal{L}^{-1}_{\texttt{scheme3}}(y)$ be a deterministic strategy which maximises $EV_g^{\texttt{scheme3}}(\pi, \mathcal{C}_{l_{\texttt{scheme3}}})$, we get

$$\begin{aligned}
&EV_g^{\texttt{scheme3}}(\pi, \mathcal{C}_{l_{\texttt{scheme3}}}) \\
=&\sum_{y \in \mathcal{Y}} \Pr[\mathcal{Y} = y] \sum_{i=0}^{a-1} \Pr_{\texttt{scheme3}}\left[\mathcal{X} = (w(y), Q \cdot (P_a)^i) \mid \mathcal{Y} = y\right] \\
\leq&\sum_{y \in \mathcal{Y}} \Pr[\mathcal{Y} = y] \sum_{i=0}^{a-1} \Pr_{\texttt{scheme2}}\left[\mathcal{X} = (w(y), Q \cdot (P_a)^i) \mid \mathcal{Y} = y\right] \\
=&EV_g^{\texttt{scheme2}}(\pi, \mathcal{C}_{l_{\texttt{scheme2}}}).
\end{aligned}$$

□

There is an easy way to see this inequality. We simplify the attack to only guessing the number of real keywords in the first document. Let n be the number of keywords in the first encrypted document. For scheme1, since there is no padding of keywords to documents, so the number of real keywords is n. For scheme2, due to collisions, we only know that the number of keywords can fall in the range $\lceil \frac{n}{a} \rceil$ to n. For scheme3, the number of keywords can be any integer from 1 to n.

References

[1] M. S. Alvim, K. Chatzikokolakis, C. Palamidessi, and G. Smith. Measuring information leakage using generalized gain functions. In *2012 IEEE 25th Computer Security Foundations Symposium*, pages 265–279, June 2012.

[2] Eu-Jin Goh. Secure indexes. Cryptology ePrint Archive, Report 2003/216, 2003. http://eprint.iacr.org/2003/216.

[3] Yan-Cheng Chang and Michael Mitzenmacher. Privacy preserving keyword searches on remote encrypted data. In John Ioannidis, Angelos Keromytis, and Moti Yung, editors, *ACNS 05: 3rd International Conference on Applied Cryptography and Network Security*, volume 3531 of *Lecture Notes in Computer Science*, pages 442–455, New York, NY, USA, June 7–10, 2005. Springer, Heidelberg, Germany.

[4] Reza Curtmola, Juan A. Garay, Seny Kamara, and Rafail Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 06: 13th Conference on Computer and Communications Security*, pages 79–88, Alexandria, Virginia, USA, October 30 – November 3, 2006. ACM Press.

[5] Melissa Chase and Seny Kamara. Structured encryption and controlled disclosure. In Masayuki Abe, editor, *Advances in Cryptology – ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 577–594, Singapore, December 5–9, 2010. Springer, Heidelberg, Germany.

[6] Seny Kamara, Charalampos Papamanthou, and Tom Roeder. Dynamic searchable symmetric encryption. Cryptology ePrint Archive, Report 2012/530, 2012. `http://eprint.iacr.org/2012/530`.

[7] Seny Kamara and Charalampos Papamanthou. Parallel and dynamic searchable symmetric encryption. In Ahmad-Reza Sadeghi, editor, *FC 2013: 17th International Conference on Financial Cryptography and Data Security*, volume 7859 of *Lecture Notes in Computer Science*, pages 258–274, Okinawa, Japan, April 1–5, 2013. Springer, Heidelberg, Germany.

[8] David Cash, Joseph Jaeger, Stanislaw Jarecki, Charanjit S. Jutla, Hugo Krawczyk, Marcel-Catalin Rosu, and Michael Steiner. Dynamic searchable encryption in very-large databases: Data structures and implementation. In *ISOC Network and Distributed System Security Symposium – NDSS 2014*, San Diego, CA, USA, February 23–26, 2014. The Internet Society.

[9] Kaoru Kurosawa and Yasuhiro Ohtaki. UC-secure searchable symmetric encryption. In Angelos D. Keromytis, editor, *FC 2012: 16th International Conference on Financial Cryptography and Data Security*, volume 7397 of *Lecture Notes in Computer Science*, pages 285–298, Kralendijk, Bonaire, February 27 – March 2, 2012. Springer, Heidelberg, Germany.

[10] Muhammad Naveed, Seny Kamara, and Charles V. Wright. Inference attacks on property-preserving encrypted databases. In Indrajit Ray, Ninghui Li, and Christopher Kruegel:, editors, *ACM CCS 15: 22nd Conference on Computer and Communications Security*, pages 644–655, Denver, CO, USA, October 12–16, 2015. ACM Press.

[11] Mohammad Saiful Islam, Mehmet Kuzu, and Murat Kantarcioglu. Access pattern disclosure on searchable encryption: Ramification, attack and mitigation. In *ISOC Network and Distributed System Security Symposium – NDSS 2012*, San Diego, CA, USA, February 5–8, 2012. The Internet Society.

[12] David Cash, Paul Grubbs, Jason Perry, and Thomas Ristenpart. Leakage-abuse attacks against searchable encryption. In Indrajit Ray, Ninghui Li, and Christopher Kruegel:, editors, *ACM CCS 15: 22nd Conference on Computer and Communications Security*, pages 668–679, Denver, CO, USA, October 12–16, 2015. ACM Press.

[13] David Pouliot and Charles V. Wright. The shadow nemesis: Inference attacks on efficiently deployable, efficiently searchable encryption. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 16: 23rd Conference on Computer and Communications Security*, pages 1341–1352, Vienna, Austria, October 24–28, 2016. ACM Press.

[14] Annabelle McIver, Carroll Morgan, Geoffrey Smith, Barbara Espinoza, and Larissa Meinicke. Abstract channels and their robust information-leakage ordering. In Martín Abadi and Steve Kremer, editors, *Principles of Security and Trust*, pages 83–102, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[15] M. S. Alvim, K. Chatzikokolakis, A. McIver, C. Morgan, C. Palamidessi, and G. Smith. Axioms for information leakage. In *2016 IEEE 29th Computer Security Foundations Symposium (CSF)*, pages 77–92, June 2016.