

# 1 Introduction

This document is a summary of work so far on defining security notion for searchable encryptions and finding good/bad constructions with respect to the notions. In particular, we have considered using idea of g-leakage to measure security of a given scheme with respect to a target (gain) function and shown it is equivalent to an experiment-based adversary. We have studied three schemes we proposed, and shown that even though all of the schemes are secure against keyword guessing attack, they are not equally secure against other types of attacks. This demonstrates that security against keyword guessing attack is not sufficient for an encryptions scheme to be secure. For instance, one of the scheme we proposed is secure against inference attack but it leaks all keywords in all documents information-theoretically.

# 2 Notation

The following notations are used for sets we used to describe a searchable database.

Notation	Explanation
$DB$	Set of searchable databases
$EDB$	Set of encrypted searchable databases
$\mathcal{W}$	Set of keywords in a database
$\mathcal{EW}$	Set of encrypted keywords in a encrypted searchable databases
$\mathcal{EI}$	Set of encrypted indices for keyword searching
$F$	Set of files
$EF$	Set of encrypted files
$Q$	Set of queries on the database
$EQ$	Set of encrypted queries on encrypted database
$CST$	Set of states of the client
$R$	Set of responses from a query
$ER$	Set of encrypted responses from an encrypted query

The following notations are used to describe realisations of a database.

Notation	Explanation
$db$	Database that has not been encrypted
$edb$	Encrypted database
$W$	Keyword set associated to some file
$w$	A keyword from keyword set $\mathcal{W}$
$EW$	Encrypted set associated to some encrypted file
$ew$	An encrypted keyword from encrypted keyword set $\mathcal{EW}$
$EI$	An encrypted index associated to an encrypted database
$f$	A file in the database
$ef$	An encrypted file
$q$	A query on the database
$eq$	An encrypted query on the encrypted database
$cst$	A state of the client
$r$	A response from a query
$er$	An encrypted response from an encrypted query

We use  $\Pi_i$  to mean projection of a product of sets to its  $i$ -th component. For example,  $\Pi_1(A \times B) = A$ . We may also use it to return the  $i$ -th element in a tuple. For example,  $\Pi_1((A, B)) = A$ . We abuse  $W(f)$

and  $F(W)$  to mean set of keywords associated to file  $f$  and the set of files associated to keyword set  $W$ . Similarly, we abuse  $EW(ef)$  and  $EF(W)$  to mean set of encrypted keywords associated to encrypted file  $ef$  and the set of encrypted files associated to keyword set  $W$ .

### 3 g-leakage

G-leakage is first studied by Alvim et al. in [1]. To understand how it works, we first define (classic) vulnerability, leakage and capacity.

**Definition 1** (Information Channel). *A channel is a triple  $(\mathcal{X}, \mathcal{Y}, \mathcal{C})$ , where  $\mathcal{X}$  and  $\mathcal{Y}$  are finite sets and  $\mathcal{C}$  is a channel matrix, an  $|\mathcal{X}| \times |\mathcal{Y}|$  matrix whose entries are between 0 and 1 and whose rows each sum to 1.*

**Definition 2** (Vulnerabilities). *Given prior distribution  $\pi$  on  $\mathcal{X}$  and channel  $\mathcal{C}$ , the prior vulnerability is given by:*

$$V(\pi) = \max_{x \in \mathcal{X}} \pi[x], \quad (1)$$

*and the posterior vulnerability is given by:*

$$V(\pi, \mathcal{C}) = \sum_{y \in \mathcal{Y}} \max_{x \in \mathcal{X}} \pi[x] \mathcal{C}[x, y]. \quad (2)$$

**Definition 3** (Leakage and Capacity). *We define min-entropy leakage  $\mathcal{L}(\pi, \mathcal{C})$  and min-capacity  $\mathcal{ML}(\mathcal{C})$  to be:*

$$\mathcal{L}(\pi, \mathcal{C}) = -\log V(\pi) + \log V(\pi, \mathcal{C}) = \log \frac{V(\pi, \mathcal{C})}{V(\pi)} \quad (3)$$

$$\mathcal{ML}(\mathcal{C}) = \sup_{\pi} \mathcal{L}(\pi, \mathcal{C}). \quad (4)$$

G-leakage is different from the classic definition by allowing the adversary to guess part of  $x$  from observation  $y$ , and his gain is measured by a gain function  $g$ . We define g-leakage with the following definitions.

**Definition 4** (Gain function). *Given a set  $\mathcal{X}$  of possible secrets and a finite, non-empty set  $\mathcal{W}$  of allowable guesses, a gain function is a function  $g : \mathcal{W} \times \mathcal{X} \rightarrow [0, 1]$ .*

There is no restriction on how the gain function behaves from this definition. In particular, it is possible to set the gain to be 0 even if the adversary makes a perfect guess from what he has observed. Therefore, in our work, we need to define a class of well-behaved gain functions in order to talk about semantic meaning of the leakage. (**Note: not done yet.**)

**Definition 5** (Prior g-vulnerability). *Given gain function  $g$  and prior  $\pi$ , the prior g-vulnerability is*

$$V_g(\pi) = \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi[x] g(w, x). \quad (5)$$

**Definition 6** (Posterior g-vulnerability). *Given gain function  $g$ , prior  $\pi$ , and channel  $\mathcal{C}$ , the posterior g-vulnerability is*

$$V_g(\pi, \mathcal{C}) = \sum_{y \in \mathcal{Y}} \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi[x] \mathcal{C}[x, y] g(w, x). \quad (6)$$

**Definition 7** (g-leakage and g-capacity). *G-leakage  $\mathcal{L}_g(\pi, \mathcal{C})$  and g-capacity  $\mathcal{ML}_g(\mathcal{C})$  are defined as:*

$$\begin{aligned} \mathcal{L}_g(\pi, \mathcal{C}) &= -\log V_g(\pi) + \log V_g(\pi, \mathcal{C}) = \log \frac{V_g(\pi, \mathcal{C})}{V_g(\pi)} \\ \mathcal{ML}_g(\mathcal{C}) &= \sup_{\pi} \mathcal{L}_g(\pi, \mathcal{C}). \end{aligned}$$

For application to searchable databases,  $\mathcal{X}$  in the definition is the database and other secrets the adversary tries to guess, and  $\mathcal{Y}$  is the encrypted database he observes. The adversary may not only be interested in recover some information about the unencrypted database but also some relation between the unencrypted database and its encryption. For instance, the adversary may want to know what is the encryption of each keyword, i.e. keyword guessing attack. Hence, we extend the gain function as:

**Definition 8** (Extended gain function). *Given a set  $\mathcal{X}$  of possible secrets, a set  $\mathcal{Y}$  derived from  $\mathcal{X}$  and a finite, non-empty set  $\mathcal{W}$  of allowable guesses, a gain function is a function  $g : \mathcal{W} \times \mathcal{X} \times \mathcal{Y} \rightarrow [0, 1]$ .*

In our application, the prior vulnerability has no operational meaning so we decide to look at posterior vulnerability only. The posterior vulnerability with the new gain function is defined as follows.

**Definition 9** (Extended posterior g-vulnerability). *Given gain function  $g$ , prior  $\pi$ , and channel  $C$ , the extended posterior g-vulnerability is*

$$EV_g(\pi, C) = \sum_{y \in \mathcal{Y}} \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi[x] C[x, y] g(w, x, y). \quad (7)$$

## 4 Syntax of Searchable Encrypted Database

### 4.1 Database

A database  $db$  consists of a set of keywords  $\mathcal{W}$  and a set of files  $F$ , so  $db = (\mathcal{W}, F)$ . Each file  $f \in F$  contains some keywords specified by the user. For simplicity of notation, we denote keywords associated to a file  $W = \mathcal{W}(f)$ . The database supports queries via function  $\text{Query} : DB \times Q \rightarrow DB \times R$ , where  $Q$  is a set of queries on the database and  $R$  is some response. Since we are talking about encryptions for searching, searching of keywords must be one of the supported query types. In terms of notation, we may write  $\mathcal{W}(db)$  to mean the set of keywords associated to the database  $db$ , i.e.  $\mathcal{W}(db) = \Pi_1(db)$ .

### 4.2 Encrypted Database

An encrypted database  $edb$  consists of a set of encrypted keywords  $\mathcal{EW}$ , a set of encrypted files  $EF$  and an encrypted index  $\mathcal{EI}$  for encrypted queries, so  $edb = (\mathcal{EW}, EF, \mathcal{EI})$ . Each encrypted file  $ef \in EF$  contains some encrypted keywords which are encryptions of corresponds keywords for the plain files. Similar to unencrypted databases, we denote encrypted keywords associated to an encrypted file by  $EW = \mathcal{EW}(ef)$ . The encrypted database supports encrypted queries via function  $\text{EQuery} : EDB \times EQ \rightarrow EDB \times ER$ . In terms of notation, we may write  $\mathcal{EW}(edb)$  to mean the set of keywords associated to the encrypted database  $edb$ , i.e.  $\mathcal{EW}(edb) = \Pi_1(edb)$ .

### 4.3 Functions between Database and Encrypted Database

A mechanism for searchable encrypted database supports the following operations:

- **Key generation:**  $\text{KGen} : 1^n \rightarrow \mathcal{K}$ . The key generation algorithm takes in the security parameter and outputs keys used for the mechanism.
- **Initialisation:**  $\text{Init} : 1^n \times \mathcal{K} \rightarrow CST$ . The initialisation function takes in the security parameter and the key, and outputs the initial state of the client.

- **Encryption:**  $\text{Enc} : 1^n \times \mathcal{K} \times DB \times CST \rightarrow EDB \times CST$ . The encryption scheme takes in security parameter, a key, a database and the state of the client, and produce an encrypted database and a state.
- **Decryption:**  $\text{Dec} : 1^n \times \mathcal{K} \times EDB \times CST \rightarrow DB$ . The decryption scheme takes in security parameter, a key, an encrypted database and a state, and produce a database.
- **Query:**  $\text{Query} : DB \times Q \rightarrow DB \times R$ . A query function takes in a database and a query and outputs a response.
- **Query encryption:**  $\text{EncQ} : \mathcal{K} \times Q \times CST \rightarrow EQ \times CST$ . The query encryption function takes in a key and a query, and produces an encrypted query and a state.
- **Encrypted queries:**  $\text{EQuery} : EDB \times EQ \rightarrow EDB \times ER$ . An encrypted query takes an encrypted database and an encrypted query, and returns an encrypted database and an encrypted response.
- **Response decryption:**  $\text{DecR} : \mathcal{K} \times ER \times CST \rightarrow R \times CST$ . The response decryption function takes a key, an encrypted response, and a state, and returns a response and a state.

When the security parameter and the key are obvious, we may omit them from the notation. We may abuse  $\text{Enc}(\cdot)$  and  $\text{Dec}(\cdot)$  as encryption and decryption functions on keywords, for example,  $ew = \text{Enc}(w)$ .

#### 4.4 Correctness Requirements

Given key  $k$ , database  $db_0$ , and queries  $q_0, \dots, q_l$ , and the following procedure:

1.  $cst \leftarrow \text{Init}(1^n, k)$ ,
2.  $(edb_0, cst_0) \leftarrow \text{Enc}(1^n, k, db_0, cst)$ ,
3. For  $i = 0, \dots, l$ ,  
 $(eq_i, cst'_i) \leftarrow \text{EncQ}(k, q_i, cst_i); (edb_{i+1}, er_i) \leftarrow \text{EQuery}(edb_i, eq_i); (r_i, cst_{i+1}) \leftarrow \text{DecR}(k, er_i, cst'_i)$ .

And an imaginary procedure:

1. For  $i = 0, \dots, l$ ,  
 $(db_{i+1}, r'_i) \leftarrow \text{Query}(db_i, q_i)$ .

We require that for all security parameter  $1^n$ , key  $k$ , database  $db_0$ , and sequence of queries  $q_0, \dots, q_l \in Q^*$  the following conditions hold:

1. **Correctness of encryption and execution of queries on databases:**  $db_i = \text{Dec}(edb_i)$ .
2. **Correctness of query responses:**  $r_i = r'_i$ .

## 5 Security Notions

In this section, we discuss why previous notions are not sufficient in defining security of searchable encryption schemes. We then motivate with a security notion for keyword guessing attack on searchable encrypted databases. We modify the experiment by considering a leakage function instead of the real encryption function. After that, we generalise the experiment to allow for arbitrary adversarial goals.

## 5.1 Previous Notions

Several security notions have been proposed in the literature. The first notion is based on an indistinguishability game which tries to capture the idea that the encrypted index does not leak any information about the underlying files. The notion is known as indistinguishability under chosen keyword attack (IND-CKA) is proposed by Goh in [2]. A slightly stronger notion known as IND2-CKA has been proposed later in the same paper. However this notion is not sufficient because it only captures security of encrypted database and does not consider interaction between client and server via queries. Curtmola et al. [3] improved on the notion and proposed non-adaptive indistinguishability security and adaptive indistinguishability security for searchable symmetric encryption (SSE). The most remarkable difference between Curtmola’s notion and Goh’s is that the trace<sup>1</sup> of database is allowed to be leaked. The authors have also defined simulator-based security notions and proved equivalence of those to the indistinguishability-based notions. The idea of allowing some leakage is then generalised to be beyond just the trace [4, 5, 6, 7].

However, security with leakage does not imply security in practice. One needs to understand what those leakages really reveals about the underlying database. In some cases, that is obvious. For example, if the leakage function is everything of the database then we know the encryption scheme is not secure, and if the leakage function reveals nothing then the scheme is secure. But with anything in between, we are not able to draw conclusion immediately. In fact, many types of leakages can be abused to recover information about the underlying database. If a scheme leaks repetition of keywords then it may be vulnerable to frequency-based statistical attack [8]. If a scheme leaks co-occurrence of keywords then it can be exploited by IKK attack [9] and count attack [10]. Notably, co-occurrence is part of search pattern and access pattern and those are leaked by most of the schemes. More complicated leakage pattern involving bloom filter can also be misused [11]. Therefore, in order to prove that a scheme is secure for practical use, we have to show that the leakage does not reveal information of the underlying database we try to hide.

The first step towards this goal is to consider a snapshot adversary who only has access to the encrypted database in the static manner and tries to recover some information. To make it more concrete, we consider an adversary who tries to recover some encrypted keywords. We call the security notion security against keyword guessing attack.

## 5.2 Security Notion for Keyword Guessing Attack

There is a multitude of ways to evaluate how well the adversary has recovered information through keyword guessing attack. We will present with one of the simplest here. Namely, the adversary wins if he recovers a correct pair of keyword and its encryption (with one guess).

For simplicity, we consider snapshot attack here so the adversary is given the leakage of the scheme and there is no interaction between the encrypted database and the adversary. We may abuse  $\text{Dec}(\cdot)$  to be the function that decrypts an encrypted document or a set of encrypted keywords associated to an encrypted document, and outputs the set of real keywords.

**Definition 10** (Security against Keyword Guessing Attack). *We define a keyword guessing attack adversary to be  $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ , where  $\mathcal{A}_0$  is a probabilistic polynomial time (PPT) adversary which takes in the security parameter and generates a state **state** and a database  $db$ , and  $\mathcal{A}_1$  is a PPT adversary which takes in the security parameter, state **state** and encrypted database  $edb$ , and returns a pair of keyword and encrypted keyword. The experiment can be described as follows:*

---

<sup>1</sup>Trace is defined to be the collection of document sizes, search pattern and access pattern.

---

$Exp_A^{Keyword-Guessing}(n)$

---

```

1:   $k \leftarrow \text{\$KGen}(1^n)$ 
2:   $(state, db) \leftarrow \mathcal{A}_0(1^n)$ 
3:   $edb \leftarrow \text{Enc}(1^n, k, db)$ 
4:   $(m, c) \leftarrow \mathcal{A}_1(1^n, state, edb)$ 
5:  if  $(m \in \mathcal{W}(db)) \wedge (c \in \mathcal{EW}(edb)) \wedge (\text{Dec}(c) = m)$ 
6:    return  $1 - f(m, db)$ 
7:  else return  $-f(m, db)$ 

```

where  $f(m, db)$  is a function that returns the frequency of keyword  $m$  in database  $db$ .

Advantage  $Adv_A^{Keyword-Guessing}(n)$  of an inference attack adversary  $\mathcal{A}$  is defined to be

$$Adv_A^{Keyword-Guessing}(n) = \mathbb{E} \left[ Exp_A^{Keyword-Guessing}(n) \right]. \quad (8)$$

We say that a scheme is secure against keyword guessing attack if  $Adv_A^{Keyword-Guessing}(n)$  is less than some negligible function in  $n$ .

Intuitively, a scheme is secure against keyword guessing attack if there is no way he can guess any plaintext-ciphertext pair with probability greater than the frequency of the plaintexts. There are other possible definitions, for instance, one can look at the probability the adversary guessing all pairs of plaintexts and ciphertexts correctly.

This security notion is hard to work with because the database the adversary can generate is literally anything and the encryption scheme can generate any ciphertext. But in our game, we really do not care if keywords are ‘1’ and ‘2’ or ‘one’ and ‘two’, nor do we care how the ciphertexts look like. To simplify the problem, we want to work with idealised databases and encryption schemes which allows us to ignore information that are not relevant to our security game. Due to necessity of efficient searchability, encryption schemes for searchable encryption cannot achieve IND-CPA security, and the ‘insecure part’ of encryption function (and query functions) are understood in terms of leakage functions. Informally speaking, leakage function associated to a scheme captures what is leaked about the unencrypted database by looking at its encryption. For example, number of documents is part of leakage for any scheme that does not pad fake documents because after enough queries, all documents will be returned at least once, unless part of the database is never going to be returned by any query, but that defeats the purpose of putting them there in the first place.

In the next experiment, the adversary is only given the leakage associated to the mechanism applied to the generated database. We argue that by choosing the leakage carefully, the advantage for the experiment will be the same as the previous one, up to an additive negligible term. We begin by defining leakage of searchable encryption.

**Definition 11** (Leakage of Encryption Schemes for Searching). *Let  $\text{Enc}$  be the encryption algorithm for some searchable encryption scheme. We say  $\mathcal{L}_M : db \times \mathcal{K} \rightarrow \{0, 1\}^*$  is a valid leakage function for the mechanism if for all adversaries  $\mathcal{A}$  there exists a simulator  $\mathcal{S}$  such that for any PPT distinguishers  $\mathcal{D}$ , the following two games are indistinguishable.*

$Real_A(n)$	$Sim_{A, \mathcal{S}}(n)$
1: $k \leftarrow \text{\$KGen}(1^n)$	1: $k \leftarrow \text{\$KGen}(1^n)$
2: $db \leftarrow \mathcal{A}(1^n)$	2: $db \leftarrow \mathcal{A}(1^n)$
3: $edb \leftarrow \text{Enc}(1^n, k, db)$	3: $l \leftarrow \mathcal{L}_M(1^n, k, db)$
4: <b>return</b> $edb$	4: $edb \leftarrow \mathcal{S}(1^n, l)$
	5: <b>return</b> $edb$

$\mathcal{L}_{\mathcal{M}}$  is said to be minimal if there exists a simulator  $\mathcal{S}$  such that for all adversaries  $\mathcal{A}$ , no PPT distinguishers  $\mathcal{D}$  can distinguish the two games above.

Now we are ready to define security against idealised keyword guessing attack.

**Definition 12** (Security against Idealised Keyword Guessing Attack). *We define an idealised keyword guessing attack adversary to be  $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ , where  $\mathcal{A}_0$  is a PPT adversary which takes in the security parameter and generates a state **state** and a database  $db$ , and  $\mathcal{A}_1$  is any adversary (not necessarily polynomial time bounded) which takes in the security parameter, state **state** and leakage  $l$  generated by the minimal leakage function  $\mathcal{L}_{\mathcal{M}}$ , and returns a pair of keyword and encrypted keyword. The experiment can be described as follows:*

---

$Exp_{\mathcal{A}}^{Keyword-Guessing-Ideal}(n)$

---

```

1:   $k \leftarrow \text{\$KGen}(1^n)$ 
2:   $(\text{state}, db) \leftarrow \mathcal{A}_0(1^n)$ 
3:   $l \leftarrow \mathcal{L}_{\mathcal{M}}(1^n, k, db)$ 
4:   $(m, c) \leftarrow \mathcal{A}_1(1^n, \text{state}, l)$ 
5:  if  $(m \in \mathcal{W}(db)) \wedge (c \in \mathcal{EW}(edb)) \wedge (\text{Dec}(c) = m)$ 
6:    return  $1 - f(m, db)$ 
7:  else return  $-f(m, db)$ 

```

where  $f(m, db)$  is a function that returns the frequency of keyword  $m$  in database  $db$ .

Advantage  $Adv_{\mathcal{A}}^{Keyword-Guessing-Ideal}(n)$  of an idealised inference attack adversary  $\mathcal{A}$  is defined to be

$$Adv_{\mathcal{A}}^{Keyword-Guessing-Ideal}(n) = \mathbb{E} \left[ Exp_{\mathcal{A}}^{Keyword-Guessing-Ideal}(n) \right]. \quad (9)$$

We say that a scheme is secure against idealised keyword guessing attack if  $Adv_{\mathcal{A}}^{Keyword-Guessing-Ideal}(n)$  is less than some negligible function in  $n$ .

Notice that in the definition,  $\mathcal{A}_1$  does not need to be PT bounded. This is because we are in the idealised world where the cryptographic part has already been dealt with in the leakage. We will see later that with a well-defined leakage function  $\mathcal{L}_{\mathcal{M}}$ , advantage of the idealised experiment can be used to upper bound that of the real experiment.

### 5.3 Moving from Fixed Target Function to Generic Gain Function

In the two games for keyword guessing attack we defined above, the content returned by the games are fixed so proving security with respect to the games only says the scheme is secure against this specific adversary who tries to recover this specific information. This is very inflexible as a security notion for application in searchable encryption. In addition, wrong guesses in these games are completely punished in this setting in the sense that they will never show up in the advantage term, even though wrong guesses can still reveal some information. Consider the case where the adversary always tries to guess decryption of some ciphertext  $c$  and he always fails because his guess is some  $m$  such that  $\text{Dec}(c) \neq m$ . But ultimately, the goal of the adversary can be to recover some keywords in some documents. If  $m$  always appears together with  $\text{Dec}(c)$  in all documents, then in fact, the adversary recovers a significant amount of information about the underlying database, even though the guess itself is wrong. So we need to reward those guesses that are not completely correct but still reveals information about the database appropriately.

To do so, we use idea of gain function from g-leakage paper [1]. To put it in simple words, instead returning something from the experiment and analyse it in the advantage term, we compute the gain at the end of the

experiment, and the advantage is defined to be the expectation of the result of the experiment for the best adversary. We now define the games with gain functions.

**Definition 13** (Real Game with Gain Function). *We define a real adversary with gain function  $g$  to be  $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ , where  $\mathcal{A}_0$  is a PPT adversary which takes in the security parameter and generates a database  $db$ , and  $\mathcal{A}_1$  is a PPT adversary which takes in the security parameter, and encrypted database  $edb$ , and returns some guess  $w$  in the set of allowed guesses  $\mathcal{W}$ . The experiment can be described as follows:*

---

$Real_{\mathcal{A}}^g(n)$

---

```

1:   $k \leftarrow \text{\$KGen}(1^n)$ 
2:   $db \leftarrow \mathcal{A}_0(1^n)$ 
3:   $edb \leftarrow \text{Enc}(1^n, k, db)$ 
4:   $w \leftarrow \mathcal{A}_1(1^n, edb)$ 
5:  return  $g(w, (db, k), edb)$ 

```

Advantage  $Adv_{\mathcal{A},g}^{Real}(n)$  is defined to be:

$$Adv_{\mathcal{A},g}^{Real}(n) = \mathbb{E}[Real_{\mathcal{A}}^g(n)]. \quad (10)$$

**Definition 14** (Ideal Game with Gain Function). *We define an ideal adversary with gain function  $g$  to be  $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ , where  $\mathcal{A}_0$  is a PPT adversary which takes in the security parameter and generates a database  $db$ , and  $\mathcal{A}_1$  is an adversary which takes in the security parameter, and a valid leakage function  $\mathcal{L}_{\mathcal{M}}$ , and returns some guess  $w$  in the set of allowed guesses  $\mathcal{W}$ . The experiment can be described as follows:*

---

$Ideal_{\mathcal{A}}^{\mathcal{L}_{\mathcal{M}},g}(n)$

---

```

1:   $k \leftarrow \text{\$KGen}(1^n)$ 
2:   $db \leftarrow \mathcal{A}_0(1^n)$ 
3:   $l \leftarrow \mathcal{L}_{\mathcal{M}}(1^n, k, db)$ 
4:   $w \leftarrow \mathcal{A}_1(1^n, l)$ 
5:  return  $g(w, (db, k), l)$ 

```

Advantage  $Adv_{\mathcal{A},\mathcal{L}_{\mathcal{M}},g}^{Ideal}(n)$  is defined to be:

$$Adv_{\mathcal{A},\mathcal{L}_{\mathcal{M}},g}^{Ideal}(n) = \mathbb{E}[Ideal_{\mathcal{A}}^{\mathcal{L}_{\mathcal{M}},g}(n)]. \quad (11)$$

Note that we are not putting any condition to quantify when a scheme is secure under these notions, because the advantage really depends on the choice of  $g$ . But with well-defined  $g$ , the advantage will have an operational meaning and we can understand the level of security offered by our scheme from there.

## 5.4 Games with Gain Function are equivalent to some Extended Posterior g-vulnerability

In this subsection, we prove that in fact, we can look at the advantage from the games or some corresponding extended posterior g-vulnerability exchangeably. To do so, we need a slight tweak to the games above.

Let  $\pi$  be the prior on the joint distribution of databases and keys. Then the games above is equivalent to the following.



Real $\mathcal{A}^{\pi,g}(n)$	Ideal $\mathcal{A}^{\pi,\mathcal{L}_{\mathcal{M}},g}(n)$
1 : $(db, k) \leftarrow \pi$	1 : $(db, k) \leftarrow \pi$
2 : $edb \leftarrow \text{Enc}(1^n, k, db)$	2 : $l \leftarrow \mathcal{L}_{\mathcal{M}}(1^n, k, db)$
3 : $w \leftarrow \mathcal{A}_1(1^n, edb)$	3 : $w \leftarrow \mathcal{A}_1(1^n, l)$
4 : <b>return</b> $g(w, (db, k), edb)$	4 : <b>return</b> $g(w, (db, k), l)$

**Theorem 15** (Ideal game is equivalent to an extended posterior g-vulnerability). *Let  $\pi$  be the prior distribution of databases and keys. Let  $\mathcal{X} = DB \times \mathcal{K}$ . Let  $\mathcal{L}_{\mathcal{M}}$  be a valid leakage function. Let  $\mathcal{Y} = \{\mathcal{L}_{\mathcal{M}}(1^n, k, db) \mid k \in \mathcal{K}, db \in DB\}$ . Let  $\mathcal{C}_{\mathcal{L}_{\mathcal{M}}}$  be the channel matrix from  $\mathcal{X}$  to  $\mathcal{Y}$ . Let  $g : \mathcal{W} \times \mathcal{X} \times \mathcal{Y} \rightarrow [0, 1]$ . Then*

$$\sup_{\pi \in \text{Dist}} \sup_{\mathcal{A}} \text{Adv}_{\mathcal{A}, \pi, \mathcal{L}_{\mathcal{M}}, g}^{\text{Ideal}}(n) = EV_g(\pi, \mathcal{C}_{\mathcal{L}_{\mathcal{M}}}). \quad (12)$$

where  $\text{Dist}$  is the set of valid prior for the database.

*Proof.* For simplicity, we assume  $\text{Dist}$  is a point distribution. So it suffices to look at the expression  $\sup_{\mathcal{A}} \text{Adv}_{\mathcal{A}, \pi, \mathcal{L}_{\mathcal{M}}, g}^{\text{Real}}(n)$  with  $\mathcal{L}_{\mathcal{M}}$  being a valid leakage. Suppose  $\mathcal{L}_{\mathcal{M}}$  is indeed a valid leakage, then

$$\sup_{\mathcal{A}} \text{Adv}_{\mathcal{A}, \pi, \mathcal{L}_{\mathcal{M}}, g}^{\text{Ideal}}(n) \quad (13)$$

$$= \sup_{\mathcal{A}} \mathbb{E} \left[ \text{Ideal}_{\mathcal{A}}^{\mathcal{L}_{\mathcal{M}}, g}(n) \right] \quad (14)$$

$$= \mathbb{E}_{\mathcal{L}_{\mathcal{M}}} \left[ \max_{w \in \mathcal{W}} \mathbb{E}_{\pi} [g(w, (db, k), l \mid l)] \right] \quad (15)$$

$$= \mathbb{E}_{\mathcal{Y}} \left[ \max_{w \in \mathcal{W}} \mathbb{E}_{\pi} [g(w, \mathcal{X}, y) \mid y] \right] \quad (16)$$

$$= \sum_{y \in \mathcal{Y}} \Pr[\mathcal{Y} = y] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr[\mathcal{X} = x \mid \mathcal{Y} = y] g(w, x, y) \quad (17)$$

$$= \sum_{y \in \mathcal{Y}} \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr[\mathcal{X} = x, \mathcal{Y} = y] g(w, x, y) \quad (18)$$

$$= \sum_{y \in \mathcal{Y}} \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi[x] \mathcal{C}_{\mathcal{L}_{\mathcal{M}}}[x, y] g(w, x, y) \quad (19)$$

$$= EV_g(\pi, \mathcal{C}_{\mathcal{L}_{\mathcal{M}}}). \quad (20)$$

In the proof, equation 14 is just writing out the adversary explicitly. Equation 15 uses the fact that for any given leakage, there is a deterministic guess that maximises the conditional expectation and the best adversary overall does this for any leakage  $l$  he sees. The next few lines of the proof are just re-writing the expectation into algebraic form and re-arranging the terms.  $\square$

We can use the theorem above to draw some immediate conclusion for the real game.

**Proposition 16** (Real game is equivalent to an extended posterior g-vulnerability). *Let  $\pi$  be the prior distribution of databases and keys. Let  $\mathcal{X} = DB \times \mathcal{K}$ . Let  $\mathcal{Y} = EDB$ . Let  $\mathcal{C}_{\mathcal{M}}$  be the channel matrix from  $\mathcal{X}$  to  $\mathcal{Y}$ . Let  $g : \mathcal{W} \times \mathcal{X} \times \mathcal{Y} \rightarrow [0, 1]$ . Then*

$$\sup_{\pi \in \text{Dist}} \sup_{\mathcal{A}} \text{Adv}_{\mathcal{A}, \pi, g}^{\text{Real}}(n) = EV_g(\pi, \mathcal{C}_{\mathcal{M}}). \quad (21)$$

where  $\text{Dist}$  is the set of valid prior for the database.

*Proof.* Let  $\mathcal{L}_{\mathcal{M}}$  in the ideal experiment be the encryption function  $\text{Enc}(\cdot)$  then the idea game is exactly the real game. The result follows immediately.  $\square$

## 5.5 Bound the Advantage in the Real World using Advantage in the Idealised World

It would not make sense to look at the advantage in the idealised world if it has no connection to the advantage in the real world. In this subsection, we prove that by picking appropriate gain function in the idealised world, we can use the advantage in the idealised world to bound that in the real world.

**Theorem 17.** *Let  $\pi$  be the prior distribution of databases and keys. Let  $\mathcal{X} = DB \times \mathcal{K}$ . Let  $\mathcal{Y}_{Real} = EDB$ . Let  $\mathcal{C}_{\mathcal{M}}$  be the channel matrix from  $\mathcal{X}$  to  $\mathcal{Y}_{Real}$ . Let  $g_{Real} : \mathcal{W} \times \mathcal{X} \times \mathcal{Y}_{Real} \rightarrow [0, 1]$ .*

*Let  $\mathcal{L}_{\mathcal{M}}$  be a valid leakage function and  $\text{Sim}$  an simulator associated to  $\mathcal{L}_{\mathcal{M}}$ . Let  $\mathcal{Y}_{Ideal} = \{\mathcal{L}_{\mathcal{M}}(1^n, k, db) \mid k \in \mathcal{K}, db \in DB\}$ . Let  $\mathcal{C}_{\mathcal{L}_{\mathcal{M}}}$  be the channel matrix from  $\mathcal{X}$  to  $\mathcal{Y}_{Ideal}$ . Let  $g_{Ideal} : \mathcal{W} \times \mathcal{X} \times \mathcal{Y}_{Ideal} \rightarrow [0, 1]$  such that  $g_{Ideal}(w, x, y) = g_{Real}(w, x, \text{Sim}(y))$ . Then*

$$EV_{g_{Real}}(\pi, \mathcal{C}_{\mathcal{L}_{\mathcal{M}}}) \leq EV_{g_{Ideal}}(\pi, \mathcal{C}_{\mathcal{M}}) \quad (22)$$

*with overwhelming probability.*

*Proof.* The proof takes two steps. We will first relate the extended g-vulnerability in the idealised world to some other extended g-vulnerability. We then argue that the game related to the extended g-vulnerability we found is indistinguishable from that of the real world. Therefore, concluding the desired inequality.

We begin by re-writing extended g-vulnerability in the idealised world in the following way:

$$EV_{g_{Ideal}}(\pi, \mathcal{C}_{\mathcal{M}}) \quad (23)$$

$$= \sum_{y \in \mathcal{Y}_{Ideal}} \Pr[y] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr[x \mid y] g_{Ideal}(w, x, y) \quad (24)$$

$$= \sum_{y \in \mathcal{Y}_{Ideal}} \Pr[y] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr[x \mid y] g_{Real}(w, x, \text{Sim}(y)) \quad (25)$$

$$= \sum_{\text{Sim}(y) \in \text{Sim}(\mathcal{Y}_{Ideal})} \sum_{y \in \mathcal{Y}_{Ideal}} \Pr[\text{Sim}(y)] \Pr[y \mid \text{Sim}(y)] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr[x \mid y] g_{Real}(w, x, \text{Sim}(y)) \quad (26)$$

$$\geq \sum_{\text{Sim}(y) \in \text{Sim}(\mathcal{Y}_{Ideal})} \Pr[\text{Sim}(y)] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}_{Ideal}} \Pr[x \mid y] \Pr[y \mid \text{Sim}(y)] g_{Real}(w, x, \text{Sim}(y)) \quad (27)$$

$$= \sum_{\text{Sim}(y) \in \text{Sim}(\mathcal{Y}_{Ideal})} \Pr[\text{Sim}(y)] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}_{Ideal}} \Pr[x, y \mid \text{Sim}(y)] g_{Real}(w, x, \text{Sim}(y)) \quad (28)$$

$$= \sum_{\text{Sim}(y) \in \text{Sim}(\mathcal{Y}_{Ideal})} \Pr[\text{Sim}(y)] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr[x \mid \text{Sim}(y)] g_{Real}(w, x, \text{Sim}(y)). \quad (29)$$

To go from equation 26 to equation 27, we used Jensen's inequality. To arrive at equation 28, we use the fact that  $\mathcal{X} \rightarrow \mathcal{Y}_{Ideal} \rightarrow \text{Sim}(\mathcal{Y}_{Ideal})$  is a Markov chain. Finally, we get equation 29 due to law of total probability.

We observe that equation 29 is equivalent to the following game:

$$\begin{array}{l} \text{Simulated}_{\mathcal{A}}^{\text{Sim} \circ \mathcal{L}_{\mathcal{M}}, g_{Real}}(n) \\ \hline 1 : \quad k \leftarrow \text{KGen}(1^n) \\ 2 : \quad db \leftarrow \mathcal{A}_0(1^n) \\ 3 : \quad edb \leftarrow \text{Sim}(\mathcal{L}_{\mathcal{M}}(1^n, k, db)) \\ 4 : \quad w \leftarrow \mathcal{A}_1(1^n, edb) \\ 5 : \quad \textbf{return } g_{Real}(w, (db, k), edb) \end{array}$$

But this game is indistinguishable from the real game as  $\text{Sim}$  is a simulator that makes  $\text{Sim} \circ \mathcal{L}_{\mathcal{M}}(\cdot)$  indistinguishable from  $\text{Enc}(\cdot)$  by definition. Therefore, we conclude that

$$EV_{g_{\text{Real}}}(\pi, \mathcal{C}_{\mathcal{L}_{\mathcal{M}}}) = \sum_{\text{Sim}(y) \in \text{Sim}(\mathcal{Y}_{\text{Ideal}})} \Pr[\text{Sim}(y)] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr[x \mid \text{Sim}(y)] g_{\text{Real}}(w, x, \text{Sim}(y)) \quad (30)$$

with overwhelming probability. Therefore, we conclude

$$EV_{g_{\text{Real}}}(\pi, \mathcal{C}_{\mathcal{L}_{\mathcal{M}}}) \leq EV_{g_{\text{Ideal}}}(\pi, \mathcal{C}_{\mathcal{M}}) \quad (31)$$

as desired.  $\square$

## 5.6 Bounding Extended g-vulnerability using Extended g-vulnerability on Sub-channels

In this subsection, we highlight how to use extended g-vulnerability on sub-channels to bound extended g-vulnerability of some channel. It is well-known from information theory that if  $\mathcal{X} \rightarrow \mathcal{Y} \rightarrow \mathcal{Z}$  is a Markov chain, then  $I(\mathcal{X}; \mathcal{Y}) \geq I(\mathcal{X}; \mathcal{Z})$ , where  $I(\cdot)$  is the mutual information. Similar result has been shown for g-leakage in [1, 12, 13] for gain functions of the form  $g : \mathcal{W} \times \mathcal{X} \rightarrow [0, 1]$ . However, it also holds for mutual information that  $I(\mathcal{Y}; \mathcal{Z}) \geq I(\mathcal{X}; \mathcal{Z})$  but there is no corresponding result for g-vulnerability in the literature. We show that it is possible to obtain a similar result with our definition of extended g-vulnerability. We then extend our result to a channel consisting of more than two sub-channels.

**Theorem 18.** *Let  $\mathcal{X} \rightarrow \mathcal{Y} \rightarrow \mathcal{Z}$  be a Markov chain. Let  $\pi$  be the prior distribution on  $\mathcal{X}$ . Let  $\mathcal{C}_{\mathcal{X}, \mathcal{Y}}$  be the channel matrix between  $\mathcal{X}$  and  $\mathcal{Y}$ . Let  $\mathcal{C}_{\mathcal{X}, \mathcal{Z}}$  be the channel matrix between  $\mathcal{X}$  and  $\mathcal{Z}$ . Let  $\mathcal{W}$  be the set of allowed guesses. Let  $g_{\mathcal{X}, \mathcal{Y}} : \mathcal{W} \times \mathcal{X} \times \mathcal{Y} \rightarrow [0, 1]$  be a gain function on sub-channel  $\mathcal{X} \rightarrow \mathcal{Y}$ , and  $g^* : \mathcal{W} \times \mathcal{X} \times \mathcal{Z} \rightarrow [0, 1]$  be a gain function on the full channel. If*

$$g^*(w, x, z) \leq \sum_{y \in \mathcal{Y}} \Pr[y \mid x, z] g_{\mathcal{X}, \mathcal{Y}}(w, x, y) \quad (32)$$

for all  $w \in \mathcal{W}, x \in \mathcal{X}$ , and  $z \in \mathcal{Z}$ , then

$$EV_{g^*}(\pi, \mathcal{C}_{\mathcal{X}, \mathcal{Z}}) \leq EV_{g_{\mathcal{X}, \mathcal{Y}}}(\pi, \mathcal{C}_{\mathcal{X}, \mathcal{Y}}).$$

*Proof.* The proof is no more than just manipulating the probabilities.

$$\begin{aligned} & EV_{g_{\mathcal{X}, \mathcal{Y}}}(\pi, \mathcal{C}_{\mathcal{X}, \mathcal{Y}}) \\ &= \sum_{y \in \mathcal{Y}} \Pr[y] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr[x \mid y] g_{\mathcal{X}, \mathcal{Y}}(w, x, y) \\ &= \sum_{z \in \mathcal{Z}} \sum_{y \in \mathcal{Y}} \Pr[z] \Pr[y \mid z] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr[x \mid y] g_{\mathcal{X}, \mathcal{Y}}(w, x, y) \\ &\geq \sum_{z \in \mathcal{Z}} \Pr[z] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} \Pr[x \mid y] \Pr[y \mid z] g_{\mathcal{X}, \mathcal{Y}}(w, x, y) \\ &= \sum_{z \in \mathcal{Z}} \Pr[z] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} \Pr[x, y \mid z] g_{\mathcal{X}, \mathcal{Y}}(w, x, y) \\ &= \sum_{z \in \mathcal{Z}} \Pr[z] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr[x \mid z] \sum_{y \in \mathcal{Y}} \Pr[y \mid x, z] g_{\mathcal{X}, \mathcal{Y}}(w, x, y) \\ &\geq \sum_{z \in \mathcal{Z}} \Pr[z] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr[x \mid z] g^*(w, x, z) \\ &= EV_{g^*}(\pi, \mathcal{C}_{\mathcal{X}, \mathcal{Z}}). \end{aligned}$$

$\square$

**Corollary 19.** Let  $\mathcal{X} \rightarrow \mathcal{Y} \rightarrow \mathcal{Z}$  be a Markov chain. Let  $\pi$  be the prior distribution on  $\mathcal{X}$ . Let  $\mathcal{C}_{\mathcal{X},\mathcal{Y}}$  be the channel matrix between  $\mathcal{X}$  and  $\mathcal{Y}$ . Let  $\mathcal{C}_{\mathcal{X},\mathcal{Z}}$  be the channel matrix between  $\mathcal{X}$  and  $\mathcal{Z}$ . Let  $\mathcal{W}$  be the set of allowed guesses. Let  $g_{\mathcal{X},\mathcal{Y}} : \mathcal{W} \times \mathcal{X} \times \mathcal{Y} \rightarrow [0, 1]$  be a gain function on sub-channel  $\mathcal{X} \rightarrow \mathcal{Y}$ , and  $g^* : \mathcal{W} \times \mathcal{X} \times \mathcal{Z} \rightarrow [0, 1]$  be a gain function on the full channel. If there exists some function  $G : \mathcal{W} \times \mathcal{X} \rightarrow [0, 1]$  such that  $g^*(w, x, z) = g_{\mathcal{X},\mathcal{Y}}(w, x, y) = G(w, x)$  for any  $w \in \mathcal{W}, x \in \mathcal{X}, y \in \mathcal{Y}$  and  $z \in \mathcal{Z}$ , then

$$EV_{g^*}(\pi, \mathcal{C}_{\mathcal{X},\mathcal{Z}}) \leq EV_{g_{\mathcal{X},\mathcal{Y}}}(\pi, \mathcal{C}_{\mathcal{X},\mathcal{Y}}).$$

*Proof.* It suffices to verify equation 32 in theorem 18. We have

$$\begin{aligned} & \sum_{y \in \mathcal{Y}} \Pr[y | x, z] g_{\mathcal{X},\mathcal{Y}}(w, x, y) \\ &= \sum_{y \in \mathcal{Y}} \Pr[y | x, z] G(w, x) \\ &= G(w, x) \sum_{y \in \mathcal{Y}} \Pr[y | x, z] \\ &= G(w, x) \\ &= g^*(w, x, z). \end{aligned}$$

So the desired result follows.  $\square$

**Theorem 20.** Let  $\mathcal{X} \rightarrow \mathcal{Y} \rightarrow \mathcal{Z}$  be a Markov chain. Let  $\pi$  be the prior distribution on  $\mathcal{X}$ . Let  $\mathcal{C}_{\mathcal{Y},\mathcal{Z}}$  be the channel matrix between  $\mathcal{Y}$  and  $\mathcal{Z}$ . Let  $\mathcal{C}_{\mathcal{X},\mathcal{Z}}$  be the channel matrix between  $\mathcal{X}$  and  $\mathcal{Z}$ . Let  $\mathcal{W}$  be the set of allowed guesses. Let  $g_{\mathcal{Y},\mathcal{Z}} : \mathcal{W} \times \mathcal{Y} \times \mathcal{Z} \rightarrow [0, 1]$  be a gain function on sub-channel  $\mathcal{Y} \rightarrow \mathcal{Z}$ , and  $g^* : \mathcal{W} \times \mathcal{X} \times \mathcal{Z} \rightarrow [0, 1]$  be a gain function on the full channel. If

$$g^*(w, x, z) \leq \sum_{y \in \mathcal{Y}} \Pr[y | x, z] g_{\mathcal{Y},\mathcal{Z}}(w, y, z) \quad (33)$$

for all  $w \in \mathcal{W}, x \in \mathcal{X}$ , and  $z \in \mathcal{Z}$ , then

$$EV_{g^*}(\pi, \mathcal{C}_{\mathcal{X},\mathcal{Z}}) \leq EV_{g_{\mathcal{Y},\mathcal{Z}}}(\pi, \mathcal{C}_{\mathcal{Y},\mathcal{Z}}).$$

*Proof.*

$$\begin{aligned} & EV_{g_{\mathcal{Y},\mathcal{Z}}}(\pi, \mathcal{C}_{\mathcal{Y},\mathcal{Z}}) \\ &= \sum_{z \in \mathcal{Z}} \Pr[z] \max_{w \in \mathcal{W}} \sum_{y \in \mathcal{Y}} \Pr[y | z] g_{\mathcal{Y},\mathcal{Z}}(w, y, z) \\ &= \sum_{z \in \mathcal{Z}} \Pr[z] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} \Pr[x, y | z] g_{\mathcal{Y},\mathcal{Z}}(w, y, z) \\ &= \sum_{z \in \mathcal{Z}} \Pr[z] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr[x | z] \sum_{y \in \mathcal{Y}} \Pr[y | x, z] g_{\mathcal{Y},\mathcal{Z}}(w, y, z) \\ &\geq \sum_{z \in \mathcal{Z}} \Pr[z] \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \Pr[x | z] g^*(w, x, z) \\ &= EV_{g^*}(\pi, \mathcal{C}_{\mathcal{X},\mathcal{Z}}) \end{aligned}$$

$\square$

## 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] 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.
- [4] 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.
- [5] David Cash, Stanislaw Jarecki, Charanjit S. Jutla, Hugo Krawczyk, Marcel-Catalin Rosu, and Michael Steiner. Highly-scalable searchable symmetric encryption with support for Boolean queries. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 353–373, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.
- [6] Emil Stefanov, Charalampos Papamanthou, and Elaine Shi. Practical dynamic searchable encryption with small leakage. In *ISOC Network and Distributed System Security Symposium – NDSS 2014*, San Diego, CA, USA, February 23–26, 2014. The Internet Society.
- [7] Sky Faber, Stanislaw Jarecki, Hugo Krawczyk, Quan Nguyen, Marcel-Catalin Rosu, and Michael Steiner. Rich queries on encrypted data: Beyond exact matches. In Günther Pernul, Peter Y. A. Ryan, and Edgar R. Weippl, editors, *ESORICS 2015: 20th European Symposium on Research in Computer Security, Part II*, volume 9327 of *Lecture Notes in Computer Science*, pages 123–145, Vienna, Austria, September 21–25, 2015. Springer, Heidelberg, Germany.
- [8] 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.
- [9] 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.
- [10] 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.
- [11] 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.
- [12] 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.
- [13] 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.