1 Introduction

Searchable encryption aims to store user data remotely and facilitate searching functionality on the data, while protecting user privacy. Searchable encryption schemes are categorised into symmetric searchable encryption (SSE) schemes and public key encryption schemes with keyword search (PEKS). For SSE schemes, only the user with private key can amend the encrypted data structure and issue encrypted queries, whereas PEKS allows any user with public-key to insert new data into the encrypted data structure but only the users with the private keys can search. Clearly PEKS is more powerful than SSE, but due to the public-key nature, schemes in this category often require more complicated techniques and are less efficient in practice. For our interest, we focus on SSE schemes.

Over the past few years, a multitude of SSE schemes have been proposed, with different trade-offs between efficiency and security, and rich functionalities beyond single keyword search. However, many security vulnerabilities have been found on the schemes. Some of the attacks proposed in the literature requires very little prior knowledge and computing power, and are able to decipher overwhelming amount of encrypted data. All these attacks exploits what is called leakage of the schemes, which are permitted by the security notion.

In this report, we begin by introducing the readers to security notions for SSE and outline a few constructions satisfying the notions. Then we list existing attacks on SSE schemes. Most of the attacks are leakage-based attacks, meaning that the attacks will work against a class of schemes leaking at least the required amount of information. After that, we will explain why leakage abuse attacks are not captured by current notions.

Our work attempts to address the problems of current notions by proposing our own. Instead of traditional indistinguishability-based notions, our idea is to look at expected gain of the adversary, parametrised by a leakage function in the same way as the previous notions, a gain function which describes the goal of the adversary, distribution of databases and auxiliary information associated to the databases. In this way, we can reason exactly what is leaked under different assumptions on adversarial knowledge.

To compute expectation of our notion, we relate our notion to its idealised counterpart. It turns out that it is not always possible to bound the expectation of our notion by the idealised one. We theorise necessary conditions for this to hold. We found that our notion is closely related to g-leakage [1] and results on g-leakage can be used to evaluate expectation in our notion. In particular, data processing inequality can be used to isolate part of the leakage, allowing one to bound expectation of our notion. However, data processing inequality in the literature is incomplete in the sense that it is one-sided. We give our results on data processing inequality to complete the theorem.

2 Literature Review

2.1 Previous Notions and Constructions

In this subsection, we will introduce the most noteworthy security notions and schemes.

2.1.1 Goh's Notion and Scheme

Goh's work [7] is one of the first to explore search on encrypted data. At that point in time, it was not clear what is a good security notion for the task, so Goh proposed his own notions, known as IND1-CKA and IND2-CKA. On a high level, Goh's notion captures the idea that given an encrypted index, the adversary should not be able to guess its content after some adaptive queries, except the number of keywords in the index. Later, he proposed a stronger notion for which the adversary should not recover the number of

keywords too. We give syntax of index scheme and definition of the first notion here.

Definition 1 (Index Scheme)**.** An index scheme consists of the following four algorithms.

- $\mathsf{KGen}(1^n)$: Given a security parameter 1^n, outputs the master private key $\mathrm{k}_{\mathrm{priv}}$.
- $\mathsf{Trapdoor}(\mathrm{k}_{\mathrm{priv}}, w)$: Given the master key $\mathrm{k}_{\mathrm{priv}}$ and word w, outputs the trapdoor T_w for w.
- $\mathsf{BuildIndex}(\mathrm{d}, \mathrm{k}_{\mathrm{priv}})$: Given a document d and the master key $\mathrm{k}_{\mathrm{priv}}$, outputs the index $\mathcal{I}_d$.
- $\mathsf{SearchIndex}(T_w, \mathcal{I}_d)$: Given the trapdoor T_{k} for word w and the index $\mathcal{I}_d$ for document d, outputs 1 if $w \in d$ and 0 otherwise.

Definition 2 (IND-CKA)**.** Let (KGen, Trapdoor, BuildIndex, SearchIndex) be an index scheme. We use the following game between a challenger $\mathcal{C}$ and an attacker $\mathcal{A}$ to define semantic security against an adaptive chosen keyword attack (IND-CKA).

- **Setup.** The challenger $\mathcal{C}$ creates a set $\mathcal{S}$ of q words and gives this to the adversary $\mathcal{A}$. $\mathcal{A}$ chooses a number of subsets from $\mathcal{S}$. This collection of subsets is called $\mathcal{S}^*$ and is returned to $\mathcal{C}$. Upon receiving $\mathcal{S}^*$, $\mathcal{C}$ runs KGento generate the master key $\mathrm{k}_{\mathrm{priv}}$, and for each subset in $\mathcal{S}^*$, $\mathcal{C}$ encodes its contents into an index using BuildIndex. Finally, $\mathcal{C}$ sends all indexes with their associated subsets to $\mathcal{A}$.
- **Queries.** $\mathcal{A}$ is allowed to query $\mathcal{C}$ on a word x and receive the trapdoor T_x for x. With T_x, $\mathcal{A}$ can invoke SearchIndexon an index $\mathcal{I}$ to determine if $x \in \mathcal{I}$.
- **Challenge.** After making some Trapdoor queries, $\mathcal{A}$ decides on a challenge by picking a nonempty subset $V_0 \in \mathcal{S}^*$, and generating another non-empty subset V_1 from $\mathcal{S}$ such that $|V_0 - V_1| \neq 0$, $|V_1 - V_0| \neq 0$, and the total length of words in V_0 is equal to that in V_1. Lastly, $\mathcal{A}$ must not have queried $\mathcal{C}$ for the trapdoor of any word in $V_0 \triangle V_1$.

  Next, $\mathcal{A}$ gives V_0 and V_1 to $\mathcal{C}$ who chooses $b \leftarrow_{\$} \{0,1\}$, invokes $\mathsf{BuildIndex}(V_b, \mathrm{k}_{\mathrm{priv}})$ to obtain the index $\mathcal{I}_{V_b}$ for V_b, and returns $\mathcal{I}_{V_b}$ to $\mathcal{A}$. The challenge for $\mathcal{A}$ is determine b. After the challenge is issued, $\mathcal{A}$ is not allowed to query $\mathcal{C}$ for the trapdoors of any word $x \in V_0 \triangle V_1$.
- **Response.** $\mathcal{A}$ eventually outputs a bit b', representing its guess for b. The advantage of $\mathcal{A}$ in winning this game is defined as $Adv_{\mathcal{A}} = |\Pr[b = b'] - 1/2|$, where the probability is over $\mathcal{A}$ and $\mathcal{C}$'s coin tosses.

We say that an adversary $\mathcal{A}$ (t, ϵ, q)-breaks an index if $Adv_{\mathcal{A}}$ is at least ϵ after $\mathcal{A}$ takes at most t time and makes q trapdoor queries to the challenger. We say that $\mathcal{I}$ is an (t, ϵ, q)-IND-CKA secure index if no adversary can (t, ϵ, q)-break it.

Goh's construction relies on Bloom filter. If one wants to build an unencrypted search index with Bloom filter, the easiest way is to just use the Bloom filter with the same hash keys for each document and insert keywords in the most natural way. This is not secure as it is possible to infer similarity of document contents by simply looking at the Bloom filters. More recently, Pouliot and Wright [16] have shown that it is possible to recover keywords from Bloom filters with high accuracy. One also has to ensure that queries do not reveal underlying keywords. To address the challenges above, Goh suggests to use one way function to mask the keywords, and use one way function again on the masked keywords and documents so that Bloom filters depend on the documents themselves. To search, the user sends the server masked keywords and the server is able to check if the underlying keyword is in each of the document.

2.1.2 Chang and Mitzenmacher's Notion and Scheme

Chang and Mitzenmacher [4] approached the problem in a slightly different way. They propose to use a simulation-based security notion. For simplicity, we ignore some less important constraints in their definition

and focus on syntax and security requirement. We define Privacy Preserving Keyword Searches on Remote Encrypted Data (PPSED) and its security as follows.

Definition 3 (Syntax and Security of PPSED)**.** PPSED is a multi-round protocol between a remote file server $\mathcal{S}$ and a user $\mathcal{U}$. $\mathcal{S}$ has a set of n encrypted files $\xi = \{\mathcal{E}_1(m_1), \mathcal{E}_2(m_2), \cdots, \mathcal{E}_n(m_n)\}$ where for each $i \in [n]$, $\mathcal{E}_i$ is an encryption function and m_i is a file. The user $\mathcal{U}$ has decryption algorithms $\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_n$ such that $\mathcal{D}_1(\mathcal{E}_1(m_1)) = m_1, \mathcal{D}_2(\mathcal{E}_2(m_2)) = m_2, \cdots, \mathcal{D}_n(\mathcal{E}_n(m_n)) = m_n$. Moreover, in each round $j \in \mathbb{N}$, $\mathcal{U}$ prepares a keyword $w_j \in \{0,1\}^*$. An implementation of PPSED with security parameter 1^nmust satisfy the following:

1. Correctness: In round j, for $i \in [n]$, if w_j is a keyword of m_i, $\mathcal{U}$ can obtain $\mathcal{E}_i(m_i)$.
2. Security requirement: For $k \in \mathbb{N}$, let C_k be all the communications $\mathcal{S}$ receives from $\mathcal{U}$ before round k, and $C_k^* = \{\xi, Q_0 \equiv \phi, Q_1, \cdots, Q_{k-1}\}$, where for each $j \in [k-1]$, Q_j is an n-bit string such that for $i \in [n]$, $Q_j[i] = 1$ if and only if w_j is a keyword of m_i.

   For $k \in \mathbb{N}$, for any PPT algorithm $\mathcal{A}$, any $\Delta_k = \{m_1, \cdots, m_n, w_0 \equiv \phi, w_1, \cdots, w_{k-1}\}$, any function h, there is a PPT algorithm $\mathcal{A}^*$ such that the following value is negligible in 1^n:

$$\left|\Pr[\mathcal{A}(C_k, 1^n) = h(\Delta_k)] - \Pr[\mathcal{A}^*(C_k^*, 1^n) = h(\Delta_k)]\right|.$$

Intuitively, Chang and Mitzenmacher's security notion captures the idea that whatever computable about Δ_k given C_k can also be computed from C_k^*.

Their construction is very straight forward. For each document, they are going to keep an index of length d, where d is some constant. Let P_j be the pseudo-random permutation used to encrypt index of file m_j, then if w_i is a keyword in m_j, $P_j(i)$-th position of the index is marked as one, otherwise it is marked as 0. The index is further masked by a pseudo-random function depending on the file and the keyword.

2.1.3 Curtmola et al.'s Notion and Scheme

Curtmola et al. [6] argues that the previous two notions are insufficient in protecting user privacy. The main problem is that both schemes do not require the trapdoor to be secure, which means everything about the database can be leaked from queries even though the scheme is proven to be secure under the two notions. In their own notions, they fix the problem by requiring the encrypted files, encrypted index and trapdoors to be secure at the same time.

They make separation between non-adaptive and adaptive security of searchable symmetric encryption (SSE). In the non-adaptive setting (IND-CKA1), the adversary has to choose all his queries ahead of their executions, while in the adaptive setting (IND-CKA2), he can choose queries depending on the previous queries and their responses. They give indistinguishability-based definition and simulation-based definition in both settings, and prove that in each setting, indistinguishability-based definition is equivalent to simulation-based definition. We will refer non-adaptive simulation-based definition by them as SS-CKA.

Syntax of SSE is the same as that of index scheme so we invite interested readers to refer to definition 1. Security notions proposed by Curtmola et al. are parametrised by trace of history, later known as leakage functions. History (H) is the collection of documents and past queries, and trace (Tr) is whatever the adversary learns throughout the execution of the protocol. To define security, we also need to define the view of the adversary. View (V_k) under secret key k is everything the adversary sees during the execution of the protocol. For simplicity, we will only give non-adaptive security notions here. Adaptive notions can be easily derived from there.

Definition 4 (Non-Adaptive Indistinguishability Security for SSE)**.** A SSE scheme is secure in the sense of non-adaptive indistinguishability if for all $q \in \mathbb{N}$, for all (non-uniform) probabilistic polynomial-time

adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, for all polynomials p and all sufficiently large k:

$$\Pr[b' = b; \mathrm{k} \leftarrow \mathsf{KGen}(1^n); (H_0, H_1, \mathbf{state}) \leftarrow \mathcal{A}_1; b \leftarrow_{\$} \{0,1\}; b' \leftarrow \mathcal{A}_2(V_K(H_b), \mathbf{state})] < \frac{1}{2} - \frac{1}{p(k)},$$

where (H_0, H_1) are histories over q queries such that $Tr(H_0) = Tr(H_1)$, where **state** is a polynomially bounded string that captures $\mathcal{A}_1$'s state, and the probability is taken over the internal coins of $\mathsf{KGen}, \mathcal{A}$, and the underlying $\mathsf{BuildIndex}$ algorithm.

Definition 5 (Non-Adaptive Semantic Security for SSE)**.** A SSE scheme is non-adaptively semantically secure if for all $q \in \mathbb{N}$ and for all (non-uniform) probabilistic polynomial-time adversaries $\mathcal{A}$, there exists a (non-uniform) probabilistic polynomial-time algorithm (the simulator) $\mathcal{S}$ sucht aht for all traces Tr_q fo length q, all polynomially samplable distributions H_q over $\{H_q \in 2^{2^{\Delta}} \times \Delta^q : Tr(H_q) = Tr_q\}$ (i.e., the set of histories with trace Tr_q), all functions $f : \{0,1\}^m \to \{0,1\}^{l(m)}$ (where $m = |H_q|$ and $l(m) = poly(m)$), all polynomials p and sufficiently large k:

$$|\Pr[\mathcal{A}(V_{\mathrm{k}}(H_q)) = f(H_q)] - \Pr[\mathcal{S}(Tr(H_q)) = f(H_q)]| < \frac{1}{p(k)},$$

where $H_q \leftarrow_{\$} \mathcal{H}_q, \mathrm{k} \leftarrow_{\$} \mathsf{KGen}(1^n)$, and the probabilities are taken over $\mathcal{H}_q$ and the internal coins of KGen, $\mathcal{A}$, $\mathcal{S}$ and teh underlying $\mathsf{BuildIndex}$ algorithm.

The security notions captures the idea that the adversary learns at most as much as the trace. In the paper, trace is defined as the collection of document identifiers, length of documents and access pattern. Curtmola et al. prove that non-adaptive indistinguishability security is equivalent to non-adaptive semantic security. Similarly, they show adaptive indistinguishability security is equivalent to adaptive semantic security. The most popular notion in the literature is adaptive semantic security.

Curtmola et al.'s scheme is different from the previous ones as it is based on inverted index. An inverted index is an index data structure storing a mapping from keywords to the list of documents containing it. Such structure makes searching faster as one no longer has to check queried keyword on all documents but a secure construction with inverted index is more complicated. To begin with, we imagine we have an inverted index to search for documents containing each of the keywords. This data structure by itself is apparently insecure as the keywords are left in plain, and the server can learn the number of occurrence of each keyword with no effort. The idea is to shuffle the entire inverted index into a single array in a way such that the server can retrieve the set of documents for each keywords, but only with the help of the client. To achieve this, for each keyword, the client inserts the corresponding list of document identifiers into random locations of the array, specified by a pseudo-random permutation and a counter. The document identifiers inserted are coupled with the address of the next document identifier so the server can access all the document identifiers for a given keyword like a virtual linked list. Of course, to prevent the server from seeing the contents, the array is encrypted using IND-CPA secure scheme. At the same time, the client has to build a lookup table to specify the first location in the array for each keyword. He simply puts this information into an array and mask the entries with some one way function applied to the keywords.

2.1.4 Kurosawa and Ohtaki's Notion

Kurosawa and Ohtaki [12] studied the problem of verifiable SSE. Verifiable SSE is essentially SSE with the additional functionality that the user can verify if results returned by the queries are correct. As we are only interested in basic SSE, we will not derail into how to construct a verifiable SSE. However, the notion proposed by Kurosawa and Ohtaki is closely related to that of basic SSE. Their notion of security (privacy of verifiable SSE in their words) is identical to that of [6], except that their definition is in black-box model, meaning that they require a single simulator to work for all adversaries. Curiously, even though most papers use the definition in [6], the proofs are done in the black-box model, so security of those schemes is stronger than what is claimed. It is important that this is the case, as our security notion relies on black-box model.

2.1.5 Other Notions

Simulation-based security notion with leakage [6] is by far the most popular notion in the literature. Most recent works use variants of it. Chase and Kamara [5] modified the notion by asking the adversary in the game to output a bit. Works later [10, 9, 3] further generalised the notion to allow for multiple leakage functions and the adversary engages with the system in the random oracle model.

2.2 Attacks on SSE

Attacks on SSE start by assuming some adversarial power, for example, some of the attacks only require a passive adversary while the others require the adversary to inject files to the encrypted database. In addition, some attacks rely on auxiliary information the adversary has on the data. The most common auxiliary information used are frequency of keywords and frequency of co-occurrence of keywords. There are two main attack goals. First being to recover keywords from their encryptions. Second being to recover queries from the encrypted queries (usually come as search tokens). In the following literature review, we organise the attacks by their techniques, namely file injection attacks, frequency-based attacks, and co-occurrence-based attacks.

2.2.1 File Injection Attacks

Zhang et al. [18] studied power of file injection attack on recovering queries. The attack assumes that the underlying SSE leaks access pattern. The basic attack, which they call binary-search attack inject files with keywords crafted in a way such that the adversary can tell which keyword is queried by looking at access pattern on the injected files. To do so, one only need to inject number of files equal to logarithm of the number of keywords, as it gives enough bits to encode which keyword has been queried. The authors modified binary-search attack to show that simple countermeasure such as limiting the number of keywords per document does not alleviate security vulnerability. With partial knowledge on the database, the authors demonstrated that file injection attack can be mounted very efficiently and accurately.

2.2.2 Frequency-based Attacks

Naveed et al. [15] demonstrated vulnerability of encrypted databases using deterministic encryption and order-preserving encryption. Such encryption schemes are known as legacy schemes as they are fully compatible with implementations of database programs such as SQL. As deterministic encryption and order-preserving encryption leak frequency, the adversary is able to observe frequency of encrypted keywords. Given some prior knowledge on the frequency of keywords, the adversary can match encrypted keywords to the unencrypted counterparts by matching the frequencies. Naveed et al. studied different ways of doing this and their attacks are able to recover majority of encrypted medical records in their experiment. Wright and Pouliot [17] studied the problem from statistical point of view. They suggested a framework to perform statistical attacks on the encryption schemes, and demonstrated its usefulness in some attacks.

2.2.3 Co-occurrence-based Attacks

Frequency-based attacks works well if the prior knowledge on the frequencies are accurate. When this is not the case, such attacks can struggle in terms of attack accuracy. Islam et al. [8] are one of the first to look at co-occurrence of keywords. In simple terms, the adversary is assumed to have prior knowledge on frequencies of occurrences of pairs of keywords, and his goal is to recover encrypted queries by using this information. We assume that the underlying SSE leaks access pattern, so that the adversary can construct

the co-occurrence matrix of pairs of encrypted queries, where each entry in the matrix represent the number of documents returned by the corresponding queries in common. The goal of the adversary then is to fit this co-occurrence matrix with the co-occurrence matrix he generates from his prior knowledge. By this, we mean that the adversary has to find the best assignment of keywords to encrypted queries so that the distance between the two co-occurrence matrices is the smallest. The optimisation problem itself is known to be NP-hard, so Islam et al. suggested to use simulated annealing to find an approximate solution. Their optimisation procedure is later improved by Pouliot and Wright in [16] using graph-matching algorithms. In the same paper, Pouliot and Wright have shown that Bloom filter based schemes can be as vulnerable to co-occurrence-based attacks as deterministic encryption. Cash et al. [2] proposed a different attack known as count attack which works better than the other two attacks but it assumes that the adversary knows the co-occurrence of keywords exactly.

2.2.4 Other Attacks

Although not attacks on keyword search schemes, it is worth to mention the following attacks here. [11, 13] are attacks on order-preserving and order-revealing encryption. The leakage in the attacks are assumed to be access pattern and by simply manipulating with sets of documents, the authors are able to recover keywords for all documents. Grubbs et al.[1] demonstrates that leaking volume of range queries is already detrimental enough to the schemes as one is able to recover frequency of each value in the range.

2.3 Discussion

We will now address the question why the adversary can still mount those attacks even though the schemes in concern are proven to be secure under SS-CKA. The reason is quite simple: the notion captures the idea that whatever happens in the real world can be simulated by the ideal process. Assume that there is an attack in the real world, then SS-CKA guarantees that there is an attack that performs equally well in the ideal world, and it does not quantify if an attack is viable in the first place. In the worst case, consider an adversary who maintains his state as the database he challenged, then he is able to do everything he wants in the real world, but he does it equally well in the ideal world, and SS-CKA is not violated.

From the discussion above, we identify two shortcomings of SS-CKA. First of all, SS-CKA only parametrise over the leakage function and does not treat adversaries with different prior knowledges on the database differently. Secondly, it is not enough to prove that the encryption scheme can be simulated by some ideal process. We need to quantify precisely how well the adversary does under his prior knowledge.

In the next section, we will begin by defining syntax of SSE, and then give our definition of security. In the sections after, we will investigate basic properties of our security notion, and show how our notion can be used.

3 Our Definitions

3.1 Syntax of SSE

In this section, we define syntax of searchable encrypted database and then refine it for keyword search.

[1]Their paper has not been published yet so there is no proper citation.

3.1.1 Database and Encrypted Database

We define database db in the most general setting to be $db \in \{0,1\}^*$. The database supports queries via function Query : $DB \times Q \to DB \times R$, where Q is a set of queries on the database and R is some response.

Similarly, the encrypted database edb is $edb \in \{0,1\}^*$. The encrypted database supports encrypted queries via function EQuery : $EDB \times EQ \to EDB \times ER$.

3.1.2 Functions between Database and Encrypted Database

A mechanism for searchable encrypted database supports the following operations:

- **Key generation:** $\mathsf{KGen} : 1^n \to \mathcal{K}$. The key generation algorithm takes in the security parameter and outputs keys used for the mechanism.
- **Initialisation:** Init : $1^n \times \mathcal{K} \to CST$. The initialisation function takes in the security parameter and the key, and outputs the initial state of the client.
- **Encryption:** $\mathsf{Enc} : 1^n \times \mathcal{K} \times DB \times CST \to EDB \times CST$. 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:** $\mathsf{Dec} : 1^n \times \mathcal{K} \times EDB \times CST \to DB$. The decryption scheme takes in security parameter, a key,an encrypted database and a state, and produce a database.
- **Query:** Query : $DB \times Q \to DB \times R$. A query function takes in a database and a query and outputs a response.
- **Query encryption:** EncQ : $\mathcal{K} \times Q \times CST \to EQ \times CST$. The query encryption function takes in a key and a query, and produces an encrypted query and a state.
- **Encrypted queries:** EQuery : $EDB \times EQ \to EDB \times ER$. An encrypted query takes an encrypted database and an encrypted query, and returns an encrypted database and an encrypted response.
- **Response decryption:** DecR : $\mathcal{K} \times ER \times CST \to R \times CST$. 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 $\mathsf{Enc}(\cdot)$ and $\mathsf{Dec}(\cdot)$ as encryption and decryption functions on keywords, for example, $ew = \mathsf{Enc}(w)$.

3.1.3 Refinement to Keyword Search

A database db for keyword search 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 $W = \mathcal{W}(f)$. Since we are talking about encryptions for keyword search, 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. $\mathcal{W}(db) = \cup_{f \in db} \mathcal{W}(f)$.

An encrypted database edb for keyword search 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 $EW = \mathcal{EW}(ef)$. Encrypted keyword search must be one of the supported encrypted query types. In terms

of notation, we may write $\mathcal{EW}(edb)$ to mean the set of keywords associated to the encrypted database edb, i.e. $\mathcal{EW}(edb) = \cup_{ef \in edb} \mathcal{EW}(ef)$.

3.2 Our Security Notion

In this subsection we will present our security notion. We begin by motivating with a security notion that captures a specific attack known as keyword guessing attack. We then generalise this notion to capture any attack we wish to describe, and refine the notion further to quantify over classes of adversarial knowledges.

3.2.1 Security Notion for (Passive) Keyword Guessing Attack

In keyword guessing attacks in the literature [15, 8, 16, 2], the goal of the adversary is usually to math encrypted keywords or queries to unencrypted keywords. The success of the adversary is measured in terms of percentage of correct matches. Our keyword guessing attack has a slightly different goal and we argue that our goal is more meaningful and practical.

Consider the following SSE scheme where the index is a forward index. Each keyword in the documents is encrypted with some deterministic encryption scheme under the same key. We all know that deterministic encryption leaks frequency but if all documents contains all keywords, then frequency information is no longer relevant. The best adversary with the knowledge of frequencies of underlying plaintexts is not going to guess keywords and their encryptions correctly all the times. On the other hand, a wrong guess does not change the adversary's view on the set of keywords in all of the documents. He guesses the set of keywords in all of the documents correctly with certainty.

We define the goal of adversary in keyword guessing attack to be maximising the number of keywords he guesses correctly for each of the documents. His success is measured by the percentage of keywords he guesses correctly for each of the documents. Abstractly, we can represent how well the adversary has done with a function g between his guess and the actual database. We call this function *gain function*. We can naturally represent the adversary as a security game. We measure security of a scheme given some adversary by the expectation of the game.

$\underline{\mathrm{Real}^{g}_{\Pi,\mathcal{A}}(n)}$

1 : $(db, aux) \leftarrow\!\!\$\ \Pi$
2 : $\mathrm{k} \leftarrow\!\!\$\ \mathsf{KGen}(1^n)$
3 : $edb \leftarrow \mathsf{Enc}(1^n, \mathrm{k}, db)$
4 : $w \leftarrow \mathcal{A}(1^n, edb, aux)$
5 : **return** $g(w, db)$

Figure 1: Our real security game.

There is no reason why this gain function should be the only target of the adversary. In general, we can consider any gain function we want.

3.2.2 Comparison between Our Notion and IND-CKA2

Our notion has a very similar structure as the real game of SS-CKA. Both notions starts by picking a database and some auxiliary information and state. It goes through the encryption process and the adversary, based

on his view and auxiliary information or state, has to return something. The main differences between the two notions are the follows. First of all, in our notion, the database is drawn from some distribution instead of chosen by the adversary himself. We believe that our model is more realistic as distributions of practical databases are not completely random. If our goal is to protect a specific type of database, we only need to reason that our scheme is secure for the application. Secondly, the state of the adversary is given to him by Π instead of letting him control it himself. Since the auxiliary information represent adversarial knowledge, previous notion has no control over what the adversary knows, and the adversary can always just keep the database itself as his auxiliary information. With our notion, one can parametrise auxiliary information and study how different classes of auxiliary information affects security. Moreover, in our game, the adversary outputs his guess and it is measured with the gain function g. This gives our game much richer vocabulary than a game returning a single bit. Lastly, we quantify security of a scheme by the expectation of output of our security game. One can imagine this as expected loss of a company if it chooses some scheme. This is easier for practitioners to understand than complicated security notions.

4 Preliminary Results

In this section, we will derive preliminary results on our definition of security. We will first show implications between previous notions and relate those notions to our notion. We will then show how to compute the expectation of our security game by considering the idealised counterpart.

4.1 Implications between Previous Notions

We will show later that in order to bound the expectation in our security game, the scheme need to be secure in the black-box model. By that we mean the scheme has to satisfy semantic security of [6] with the order of quantifiers reversed, i.e. there exists a simulator that works for all adversaries. Black-box model appears in [12], and all proofs of schemes satisfying SS-CKA. For simplicity of notation, we prove relations with passive adversaries. All results can be converted to dynamic adversaries easily. We begin by giving definitions of the security notions.

Definition 6 (Semantic Security Notion for Bit Adversaries)**.** *Consider the following game.*

$Real_{\mathcal{A}}(n)$		$\mathsf{Sim}_{\mathcal{A},\mathcal{S}}(n)$	
1 :	$k \leftarrow\!\!\$\ \mathsf{KGen}(1^n)$	1 :	$(db, \boldsymbol{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_0(1^n)$
2 :	$(db, \boldsymbol{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_0(1^n)$	2 :	$l \leftarrow \mathcal{L}_{\mathcal{M}}(1^n, db)$
3 :	$edb, \leftarrow \mathsf{Enc}(1^n, k, db)$	3 :	$edb \leftarrow \mathcal{S}(1^n, l)$
4 :	$b \leftarrow \mathcal{A}_1(1^n, edb, \boldsymbol{st}_{\mathcal{A}})$	4 :	$b \leftarrow \mathcal{A}_1(1^n, edb, \boldsymbol{st}_{\mathcal{A}})$
5 :	**return** b	5 :	**return** b

We say that the underlying searchable encryption scheme is $\mathcal{L}_{\mathcal{M}}$-semantic secure in the black-box model for bit adversary if there exists a simulator $\mathcal{S}$ such that for all PPT adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ where $\mathcal{A}_1$ outputs a bit,

$$\Pr[Real_{\mathcal{A}}(n) = 1] - \Pr[\mathsf{Sim}_{\mathcal{A},\mathcal{S}}(n) = 1] \leq \mathsf{negl}(n)\,. \tag{1}$$

We say that the underlying searchable encryption scheme is $\mathcal{L}_{\mathcal{M}}$-semantic secure in the non-black-box model for bit adversary if for all PPT adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ where $\mathcal{A}_1$ outputs a bit, there exists a simulator $\mathcal{S}$ such that,

$$\Pr[Real_{\mathcal{A}}(n) = 1] - \Pr[\mathsf{Sim}_{\mathcal{A},\mathcal{S}}(n) = 1] \leq \mathsf{negl}(n)\,. \tag{2}$$

Definition 7 (Semantic Security Notion for String Adversaries)**.** *Consider the following game.*

$Real_{\mathcal{A}}(n)$		$\mathsf{Sim}_{\mathcal{A},\mathcal{S}}(n)$	
1 :	$k \leftarrow\!\!\$\ \mathsf{KGen}(1^n)$	1 :	$(db, \boldsymbol{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_0(1^n)$
2 :	$(db, \boldsymbol{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_0(1^n)$	2 :	$l \leftarrow \mathcal{L}_{\mathcal{M}}(1^n, db)$
3 :	$edb, \leftarrow \mathsf{Enc}(1^n, k, db)$	3 :	$edb \leftarrow \mathcal{S}(1^n, l)$
4 :	$s \leftarrow \mathcal{A}_1(1^n, edb, \boldsymbol{st}_{\mathcal{A}})$	4 :	$s \leftarrow \mathcal{A}_1(1^n, edb, \boldsymbol{st}_{\mathcal{A}})$
5 :	**return** s	5 :	**return** s

We say that the underlying searchable encryption scheme is $\mathcal{L}_{\mathcal{M}}$-semantic secure in the black-box model for string adversary if there exists a simulator $\mathcal{S}$ such that for all PPT adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, for all PPT distinguishers $\mathcal{D}$,

$$\Pr[\mathcal{D}(Real_{\mathcal{A}}(n)) = 1] - \Pr[\mathcal{D}(\mathsf{Sim}_{\mathcal{A},\mathcal{S}}(n)) = 1] \leq \mathsf{negl}(n)\,. \tag{3}$$

We say that the underlying searchable encryption scheme is $\mathcal{L}_{\mathcal{M}}$-semantic secure in the non-black-box model for string adversary if for all PPT adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, there exists a simulator $\mathcal{S}$, such that for all PPT distinguishers $\mathcal{D}$,

$$\Pr[\mathcal{D}(Real_{\mathcal{A}}(n)) = 1] - \Pr[\mathcal{D}(\mathcal{S}_{\mathcal{A},\mathcal{S}}(n)) = 1] \leq \mathsf{negl}(n)\,. \tag{4}$$

For simplicity, we call the four notions **B-BB-SS**, **B-NBB-SS**, **S-BB-SS** and **S-NBB-SS** respectively. We will now present the implications.

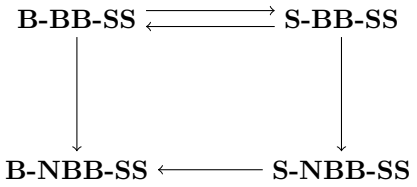

Figure 2: Implication between notions

To summarise the implications, we have:

1. In the black-box model, bit adversary and string adversary are equally strong.
2. Security in black-box model implies security in non-black-box model.
3. In the non-black-box model, bit adversary and string adversary are not equivalent.

We will now give a proof of the implications.

Proof. To begin with, we show that security in black-box model implies security in non-black-box model. Suppose there is a bit adversary in the black-box model, then this adversary works for all simulators, hence by using this adversary in the non-black box model, he wins with probability equal to the amount he wins in the black-box model. Hence for bit adversaries, security in black-box model implies security in non-black-box model. We use a very similar argument for string adversaries, the only difference is that we use the $(\mathcal{A}, \mathcal{D})$ that breaks the security in the black-box model to break security in the non-black-box model.

Since bit adversary is a special type of string adversary, we have implications from string adversary to bit adversary for free. It remains to show that in the black-box model, security against bit adversary implies security against string adversary. Suppose that the scheme is not secure against string adversary, then for all

simulators, there exists an adversary $\mathcal{A}$ such that there is a distinguisher $\mathcal{D}$ wining the security game with non-negligible probability. The bit adversary can be easily constructed by composing $\mathcal{A}$ and $\mathcal{D}$, i.e. we run the two adversaries one after another and returns the bit from $\mathcal{D}$ as the answer of the bit adversary. The bit adversary wins with probability equal to that of the corresponding string adversary. □

4.2 Computational Indistinguishability of Gain Functions

In the big picture, we will show that our definition of security game is indistinguishable from some ideal process up to the string returned by the adversary. Then we want to show that after applying the gain function g and taking expectation, we get expectations to be close to each other between the real world and ideal world. However, this is not the case for any g. Consider the following situation. Let X and Y be random variables defined as follows:

$$\Pr[X = x] = 2^{-n} \text{ for } x = [2^n],$$

$$\Pr[Y = y] = \begin{cases} 0, & \text{for } \mathrm{y} = 0 \\ 2^{-n+1}, & \text{for } \mathrm{y} = 1 \\ 2^{-n}, & \text{otherwise.} \end{cases}$$

It is easy to see that X is computationally indistinguishable from Y. Now, we set $g(x) = 2^n \cdot \mathbb{1}\{x = 0\}$. Then $\mathbb{E}[g(X)] = 1$, and $\mathbb{E}[g(Y)] = 0$. The difference is by no means negligible in n.

The reason why the phenomenon happens is because the difference between $\mathbb{E}[g(X)]$ and $\mathbb{E}[g(Y)]$ is statistical, and we only know that X and Y are computationally close. However, for well-behaved g, we can bound the difference between $\mathbb{E}[g(X)]$ and $\mathbb{E}[g(Y)]$. The result is presented as a theorem below.

Theorem 8. *Let X and Y be random variables and $g : X \cup Y \to \mathbb{R}$ a function. Assume that the following conditions hold:*

1. *X and Y are computationally indistinguishable,*
2. *$|g[X \cup Y]| \in \mathsf{poly}(n)$.*

Then

$$\mathbb{E}[g(X)] \leq \mathbb{E}[g(Y)] + |g[X \cup Y]| \cdot \sup_z |g(z)| \cdot \Delta_c(X, Y),$$

where $\Delta_c(\cdot, \cdot)$ denotes the computational distance between X and Y.

Proof. We start by expressing the difference between the expectations.

$$\begin{aligned} & |\mathbb{E}[g(X)] - \mathbb{E}[g(Y)]| \\ = & \sum_{z \in g[X]} z \cdot \Pr[g(X) = z] - \sum_{z \in g[Y]} z \cdot \Pr[g(Y) = z] \\ = & \sum_{z \in g[X \cup Y]} z \cdot (\Pr[g(X) = z] - \Pr[g(Y) = z]) \end{aligned}$$

So in particular, there is a z such that

$$|z \cdot (\Pr[g(X) = z] - \Pr[g(Y) = z])| \geq \frac{|\mathbb{E}[g(X)] - \mathbb{E}[g(Y)]|}{|g[X \cup Y]|}.$$

We can take a weaker bound on the right hand side by replacing z on the left by $\sup_z |g(z)|$, and we get

$$\begin{aligned}
\left|\sup_z |g(z)| \cdot (\Pr[g(X) = z] - \Pr[g(Y) = z])\right| &\geq \frac{|\mathbb{E}[g(X)] - \mathbb{E}[g(Y)]|}{|g[X \cup Y]|} \\
\sup_z |g(z)| \, |\Pr[g(X) = z] - \Pr[g(Y) = z]| &\geq \frac{|\mathbb{E}[g(X)] - \mathbb{E}[g(Y)]|}{|g[X \cup Y]|} \\
|\Pr[g(X) = z] - \Pr[g(Y) = z]| &\geq \frac{|\mathbb{E}[g(X)] - \mathbb{E}[g(Y)]|}{|g[X \cup Y]| \cdot \sup_z |g(z)|}
\end{aligned}$$

Since statistical difference (or variation distance) is defined as $\sum_{z \in g[X \cup Y]} |\Pr[g(X) = z] - \Pr[g(Y) = z]|$, we see that the quantity on the right is bounded by the statistical difference between $g(X)$ and $g(Y)$. Furthermore, as $g[X \cup Y] \in \mathsf{poly}(n)$, there exists a computational adversary achieving exactly this advantage, so the quantity on the right is bounded by the computational distance between $g(X)$ and $g(Y)$. Finally, we note that the advantage in distinguishing $g(X)$ and $g(Y)$ is the same as that of X and Y. By re-arranging the inequality, we get the desired result.

$$\begin{aligned}
\sum_{z \in g[X \cup Y]} |\Pr[g(X) = z] - \Pr[g(Y) = z]| &\geq \frac{|\mathbb{E}[g(X)] - \mathbb{E}[g(Y)]|}{|g[X \cup Y]| \cdot \sup_z |g(z)|} \\
\Delta_c(g(X), g(Y)) &\geq \frac{|\mathbb{E}[g(X)] - \mathbb{E}[g(Y)]|}{|g[X \cup Y]| \cdot \sup_z |g(z)|} \\
\Delta_c(X, Y) &\geq \frac{|\mathbb{E}[g(X)] - \mathbb{E}[g(Y)]|}{|g[X \cup Y]| \cdot \sup_z |g(z)|} \\
\mathbb{E}[g(X)] &\leq \mathbb{E}[g(Y)] + |g[X \cup Y]| \cdot \sup_z |g(z)| \cdot \Delta_c(X, Y)
\end{aligned}$$

□

Corollary 9. *Let X and Y be random variables and $g : X \cup Y \to \mathbb{R}$ a function. Assume that the following conditions hold:*

1. *X and Y are computationally indistinguishable,*
2. *$|g[X \cup Y]| \in \mathsf{poly}(n)$,*
3. *$\sup_z |g(z)| \in \mathsf{poly}(n)$.*

Then

$$\mathbb{E}[g(X)] \leq \mathbb{E}[g(Y)] + \mathsf{negl}(n). \tag{5}$$

4.3 Bounding Expectation in Our Notion

It is hard to compute expectation of gain function in our notion in the real world, so we take an approach similar to that of SS-CKA. In this section, we will first define idealised version of our notion, and then show if the underlying scheme is secure under **S-BB-SS** then the expectation of the gain function in our notion can be bounded by that of the ideal notion.

4.3.1 Idealised Notion

Let $\mathcal{L}_{\mathcal{M}} : DB \to \{0,1\}^*$ be a leakage function of mechanism $\mathcal{M}$ taking input a database generated by the adversary and outputs some leakage. Let $\mathcal{S} : \{0,1\}^* \to EDB$ be a PPT simulator taking input some leakage and outputs an encrypted database. We define idealised version of our notion as follows.

$\underline{\mathsf{Sim}^{g}_{\Pi,\mathcal{A},\mathcal{S}}(n)}$

1 : $(db, aux) \leftarrow\!\!\$\ \Pi$
2 : $l \leftarrow \mathcal{L}_{\mathcal{M}}(db)$
3 : $edb \leftarrow \mathcal{S}(1^n, l)$
4 : $w \leftarrow \mathcal{A}(1^n, edb, aux)$
5 : **return** $g(w, db)$

Figure 3: Our idealised security game.

4.3.2 S-BB-SS and our Notion

One key idea we have is that we want to be compatible with previous notions as much as possible so schemes proven to be secure in those settings can be easily adapt to our notion. We observed that works using SS-CKA as their security notion usually proves security of their schemes in the black box model. In another word, security of those schemes satisfies **S-BB-SS** with adaptive queries. For simplicity of notation, we will consider passive adversary in this work, but all the results can be easily generalised to adaptive security. We will prove our desired result by using a series of lemmas.

Lemma 10. Let mechanism $\mathcal{M}$ be an SSE with security parameter 1^n. Suppose that it satisfies **S-BB-SS** with leakage function $\mathcal{L}_{\mathcal{M}}$, then there exists a simulator $\mathcal{S}$ such that the outputs of the following experiments are computationally indistinguishable.

$\underline{\mathrm{Real}_{\mathcal{A}}(n)}$

1 : $\mathrm{k} \leftarrow\!\!\$\ \mathsf{KGen}(1^n)$
2 : $(db, \mathbf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_0(1^n)$
3 : $edb, \leftarrow \mathsf{Enc}(1^n, \mathrm{k}, db)$
4 : $s \leftarrow \mathcal{A}_1(1^n, edb, \mathbf{st}_{\mathcal{A}})$
5 : **return** (s, db)

$\underline{\mathsf{Sim}_{\mathcal{A},\mathcal{S}}(n)}$

1 : $(db, \mathbf{st}_{\mathcal{A}}) \leftarrow \mathcal{A}_0(1^n)$
2 : $l \leftarrow \mathcal{L}_{\mathcal{M}}(1^n, db)$
3 : $edb \leftarrow \mathcal{S}(1^n, l)$
4 : $s \leftarrow \mathcal{A}_1(1^n, edb, \mathbf{st}_{\mathcal{A}})$
5 : **return** (s, db)

Proof. Assume the contrary that the outputs of the experiments are computationally distinguishable, then there is a distinguisher $\mathcal{D}$ that distinguishes the two experiments. This distinguisher can be used to break **S-BB-SS** if the first adversary's state $\mathbf{st}_{\mathcal{A}} = db$, which is permitted by the security notion, and the lemma is proven. □

Lemma 11. Let mechanism $\mathcal{M}$ be an SSE with security parameter 1^n. Suppose that it satisfies **S-BB-SS** with leakage function $\mathcal{L}_{\mathcal{M}}$, then there exists a simulator $\mathcal{S}$ such that the outputs of the following experiments are computationally indistinguishable.

$\mathrm{Real}_{\Pi,\mathcal{A}}(n)$	
1 :	$(db, aux) \leftarrow\!\!\$\ \Pi$
2 :	$\mathrm{k} \leftarrow\!\!\$\ \mathsf{KGen}(1^n)$
3 :	$edb, \leftarrow \mathsf{Enc}(1^n, \mathrm{k}, db)$
4 :	$s \leftarrow \mathcal{A}_1(1^n, edb, aux)$
5 :	**return** (s, db)

$\mathsf{Sim}_{\Pi,\mathcal{A},\mathcal{S}}(n)$	
1 :	$(db, aux) \leftarrow\!\!\$\ \Pi$
2 :	$l \leftarrow \mathcal{L}_{\mathcal{M}}(1^n, db)$
3 :	$edb \leftarrow \mathcal{S}(1^n, l)$
4 :	$s \leftarrow \mathcal{A}_1(1^n, edb, aux)$
5 :	**return** (s, db)

Proof. The experiments above are slight modification from lemma 10 where the first adversary $\mathcal{A}_0$ is replaced by input distribution Π. As discussed before, parametrising over input distribution Π allows us to reason about security better. In fact, security in this sense is weaker than **S-BB-SS** as Π is essentially a restricted adversary $\mathcal{A}_0$ in lemma 10. □

Lemma 12. *Let mechanism $\mathcal{M}$ be an SSE with security parameter 1^n. Suppose that it satisfies **S-BB-SS** with leakage function $\mathcal{L}_{\mathcal{M}}$, then there exists a simulator $\mathcal{S}$ such that the expectations of our real and ideal notions are close, i.e. only differed by a negligible function in the security parameter.*

The gain function g in our notions must satisfy the following properties. Let X and Y be the sets of outputs of the real and ideal experiments in lemma 11 respectively. Then

1. $|g[X \cup Y]| \in \mathsf{poly}(n)$,
2. $\sup_z |g(z)| \in \mathsf{poly}(n)$.

Proof. Our experiments are nothing but the ones in 11 with gain function g applied to the outputs. From the lemma above, we know that those outputs are computationally indistinguishable. So by applying corollary 9, we conclude that the expectations in our real and ideal experiments are computationally close. □

In the next section, we will connect our notion to information theory. In particular, we will show how to use techniques in g-leakage to compute the expectation in our notion.

5 Results on g-vulnerability

5.1 Motivation

In information theory, researchers model information flow as a channel, and one can define quantitatively how much information is leaked through the channel. In the same vein, we can think of SSE as a channel and try to measure how much of the underlying database is leaked. The classic measure of information leakage is the min-entropy model [14]. Recently, Alvim et al. [1] proposed another measure of information leakage called g-leakage. In g-leakage, the adversary is awarded differently for each guess he makes via a gain function chosen by the user, and this is able to model many operational scenarios. In terms of SSE, the adversary does not need to guess the entire database to do harm with it, so g-leakage can be applied to SSE schemes to analyse their security. Maybe not so surprisingly, we found that our security notion can be modelled and computed using g-leakage. In this section, we will first introduce min-entropy and g-leakage to the readers. Then we will show the equivalence between g-leakage and our security notion. After that, we will prove a few results that are not seen in the original paper.

5.2 Introduction to g-leakage

Definition 13 (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 14 (Vulnerabilities)**.** *Given prior distribution π on $\mathcal{X}$ and channel C, the prior vulnerability is given by:*

$$V(\pi) = \max_{x \in \mathcal{X}} \pi[x], \tag{6}$$

and the posterior vulnerability is given by:

$$V(\pi, \mathcal{C}) = \sum_{y \in \mathcal{Y}} \max_{xin\mathcal{X}} \pi[x]\mathcal{C}[x, y]. \tag{7}$$

Definition 15 (Leakage and Capacity)**.** *We define min-entropy leakage $\mathcal{L}(\pi, \mathcal{C})$ and min-capacity $\mathcal{ML}(\mathcal{C})$ to be:*

$$\mathcal{L}(\pi, \mathcal{C}) = -\log V(\pi) + \log V(\pi, \mathcal{C}) = \log \frac{V(\pi, \mathcal{C})}{V(\pi)}, \tag{8}$$

$$\mathcal{ML}(\mathcal{C}) = \sup_{\pi} \mathcal{L}(\pi, \mathcal{C}). \tag{9}$$

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.

Definition 16 (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 $g : \mathcal{W} \times \mathcal{X} \to [0, 1]$.*

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.

Definition 17 (Prior g-vulnerability)**.** *Given gain function g and prior π, the prior g-vulnerability is*

$$V_g(\pi) = \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi[x] g(w, x). \tag{10}$$

Definition 18 (Posterior g-vulnerability)**.** *Given gain function g, prior π, and channel C, the posterior g-vulnerability is*

$$V_g(\pi, \mathcal{C}) = \sum_{y \in \mathcal{Y}} \max_{w \in \mathcal{W}} \sum_{x \in \mathcal{X}} \pi[x]\mathcal{C}[x, y] g(w, x). \tag{11}$$

Definition 19 (g-leakage and g-capacity)**.** *G-leakage $\mathcal{L}_g(\pi, \mathcal{C})$ and g-capacity $\mathcal{ML}_g(\mathcal{C})$ are defined as:*

$$\mathcal{L}_g(\pi, \mathcal{C}) = -\log V_g(\pi) + \log V_g(\pi, \mathcal{C}) = \log \frac{V_g(\pi, \mathcal{C})}{V_g(\pi)},$$

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

In application to SSE, we do not find prior g-vulnerability, g-leakage and g-capacity to be useful. The reason is because g-vulnerability measures the adversarial knowledge before observing the encrypted database, but it has no operational meaning in terms of attacks. G-leakage measures of how much more information one obtains after observing the outcome, but the measure is in terms of logarithm of ratios between posterior g-vulnerability and prior g-vulnerability. This does not directly reflect how much the adversary learns from the attack. Finally, g-capacity is supremum of g-leakage over all distributions π, but for databases, the distribution is much more structured. On the other hand, we will show equivalence between posterior g-vulnerability and our security notion, so results on posterior g-vulnerability can be used to reason about security of the underlying SSE scheme.

5.3 Bounding Expectations of Our Games with g-vulnerability

In this section, we will show how to bound expectations of our games with g-vulnerability.

Theorem 20 (Posterior g-Vulnerability and Real Game)**.** Let 1^n be the security parameter of some SSE scheme, and Π be the prior distribution on databases and auxiliary information. Let π be prior distribution on databases induced by Π. Let $\mathcal{X} = \{(db, aux) \mid (db, aux) \in \Pi\}$ and $\mathcal{Y} = \{(edb, aux) \mid (db, aux) \in \Pi, \mathrm{k} \leftarrow_{\$} \mathsf{KGen}(1^n), edb \leftarrow \mathsf{Enc}(1^n, \mathrm{k}, db)\}$. Let $\mathcal{C} : \mathcal{X} \times \mathcal{Y} \to [0, 1]$ be the natural channel matrix induced between $\mathcal{X}$ and $\mathcal{Y}$, i.e. $\mathcal{C}(x, y) = \Pr[\mathsf{Enc}(\pi_1(x)) = \pi_1(y)]$, where $\pi_1(\cdot)$ is a projection onto the first coordinate, and the probability is taken over all possible keys. Let $\mathcal{W}$ be the set of allowed guesses, and $g : \mathcal{W} \times DB \to [0, 1]$ be a gain function. Then

$$\sup_{\mathcal{A}} \mathbb{E}\left[\mathrm{Real}^{g}_{\Pi,\mathcal{A}}(1^n)\right] \leq V_g(\pi, \mathcal{C}).$$

Proof.

$$\begin{aligned}
&\sup_{\mathcal{A}} \mathbb{E}\left[\mathrm{Real}^{g}_{\Pi,\mathcal{A}}(1^n)\right] && (12)\\
=&\sup_{\mathcal{A}} \mathbb{E}_{\Pi,\mathcal{A}}[g(w, db)] && (13)\\
=&\sup_{\mathcal{A}} \mathbb{E}_{\mathcal{Y}}\left[\mathbb{E}_{\Pi,\mathcal{A}}[g(w, db) \mid (edb, aux)]\right] && (14)\\
=&\sup_{\mathcal{A}} \sum_{(edb,aux)\in\mathcal{Y}} \Pr[\mathcal{Y} = (edb, aux)] \sum_{(db,aux)\in\mathcal{X}} \sum_{w\in\mathcal{W}} \Pr[\mathcal{X} = (db, aux) \mid \mathcal{Y} = (edb, aux)] \\
&\Pr[\mathcal{A}(1^n, edb, aux) = w]\, g(w, db) && (15)\\
\leq& \sum_{(edb,aux)\in\mathcal{Y}} \Pr[\mathcal{Y} = (edb, aux)] \max_{w\in\mathcal{W}} \sum_{(db,aux)\in\mathcal{X}} \Pr[\mathcal{X} = (db, aux) \mid \mathcal{Y} = (edb, aux)]\, g(w, db) && (16)\\
=& \sum_{(edb,aux)\in\mathcal{Y}} \max_{w\in\mathcal{W}} \sum_{(db,aux)\in\mathcal{X}} \Pr[\mathcal{X} = (db, aux), \mathcal{Y} = (edb, aux)]\, g(w, db) && (17)\\
=& \sum_{(edb,aux)\in\mathcal{Y}} \max_{w\in\mathcal{W}} \sum_{(db,aux)\in\mathcal{X}} \Pr[\mathcal{X} = (db, aux)] \Pr[\mathcal{Y} = (edb, aux) \mid \mathcal{X} = (db, aux)]\, g(w, db) && (18)\\
=& \sum_{(edb,aux)\in\mathcal{Y}} \Pr[\mathcal{Y} = (edb, aux)] \max_{w\in\mathcal{W}} \sum_{(db,aux)\in\mathcal{X}} \pi[(db, aux)]\mathcal{C}[(db, aux), (edb, aux)]g(w, db) && (19)\\
=&V_g(\pi, \mathcal{C}). && (20)
\end{aligned}$$

Going from equation 12 to equation 13, we rewrite the expectation of real experiment as expectation of the gain function over randomness of Π and the adversary $\mathcal{A}$. To derive equation 14, we use tower rule and consider conditional expectation over all possible observed tuples of (edb, aux). Equation 15 expands equation 14 into summations. From optimisation, we know there is a deterministic strategy for the adversary $\mathcal{A}$ to maximise the function, hence we can convert supremum over adversaries into an adversary who deterministically pick a guess for each observed (edb, aux) as of equation 16, the reason why we have inequality between the equations is due to the fact that the deterministic strategy may not be polynomial in time and size. Equation 17 brings $\Pr[\mathcal{Y} = (edb, aux)]$ into the inner summation, and equation 18 rewrites $\Pr[\mathcal{X} = (db, aux), \mathcal{Y} = (edb, aux)]$ into conditional probability over $\mathcal{X}$. Finally, we realise the probabilities are in the desired forms in posterior g-vulnerability and the inequality is proven. □

We may not use this theorem often as the real game is hard to reason about. However, using a very similar argument as the proof above, we can bound supremum of ideal game with some posterior g-vulnerability.

Theorem 21 (Posterior g-Vulnerability and Ideal Game)**.** Let 1^n be the security parameter of some $\mathcal{L}_{\mathcal{M}}$-SS-CKA secure SSE scheme with simulator $\mathcal{S}$, and Π be the prior distribution on databases and auxiliary information. Let π be prior distribution on databases induced by Π. Let $\mathcal{X} = \{(db, aux) \mid (db, aux) \in \Pi\}$ and $\mathcal{Y} = \{(edb, aux) \mid (db, aux) \in \Pi, edb \leftarrow \mathcal{S}(1^n, \mathcal{L}_{\mathcal{M}}(\pi_1(x)))\}$. Let $\mathcal{C} : \mathcal{X} \times \mathcal{Y} \to [0, 1]$ be the idealised channel matrix induced between $\mathcal{X}$ and $\mathcal{Y}$, i.e. $\mathcal{C}(x, y) = \Pr[\mathcal{S}(1^n, \mathcal{L}_{\mathcal{M}}(\pi_1(x))) = \pi_1(y)]$, where $\pi_1(\cdot)$ is a projection onto the first coordinate, and the probability is taken over internal randomness of $\mathcal{S}(\cdot)$. Let $\mathcal{W}$ be the set of allowed guesses, and $g : \mathcal{W} \times DB \to [0, 1]$ be a gain function. Then

$$\sup_{\mathcal{A}} \mathbb{E}\left[\mathsf{Sim}^{g}_{\Pi,\mathcal{A},\mathcal{S}}(1^n)\right] \leq V_g(\pi, \mathcal{C}).$$

Using the theorem above and corollary 9, we can bound the expected gain of the best adversary in the real world by some idealised posterior g-vulnerability.

Corollary 22. Let 1^n be the security parameter of some $\mathcal{L}_{\mathcal{M}}$-SS-CKA secure SSE scheme with simulator $\mathcal{S}$, and Π be the prior distribution on databases and auxiliary information. Let π be prior distribution on databases induced by Π. Let $\mathcal{X} = \{(db, aux) \mid (db, aux) \in \Pi\}$ and $\mathcal{Y} = \{(edb, aux) \mid (db, aux) \in \Pi, edb \leftarrow \mathcal{S}(1^n, \mathcal{L}_{\mathcal{M}}(\pi_1(x)))\}$. Let $\mathcal{C} : \mathcal{X} \times \mathcal{Y} \to [0, 1]$ be the idealised channel matrix induced between $\mathcal{X}$ and $\mathcal{Y}$, i.e. $\mathcal{C}(x, y) = \Pr[\mathcal{S}(1^n, \mathcal{L}_{\mathcal{M}}(\pi_1(x))) = \pi_1(y)]$, where $\pi_1(\cdot)$ is a projection onto the first coordinate, and the probability is taken over internal randomness of $\mathcal{S}(\cdot)$. Let $\mathcal{W}$ be the set of allowed guesses, and $g : \mathcal{W} \times DB \to [0, 1]$ be a gain function with the following properties:

1. $|g[\mathcal{W} \times DB]| \in \mathsf{poly}(n)$,
2. $\sup_{(w,db)\in\mathcal{W}\times DB} |g(w, db)| \in \mathsf{poly}(n)$.

Then

$$\sup_{\mathcal{A}} \mathbb{E}\left[\mathrm{Real}^{g}_{\Pi,\mathcal{A}}(1^n)\right] \leq V_g(\pi, \mathcal{C}) + \mathsf{negl}(n)\,.$$

5.4 Data Processing Inequality on Posterior g-Vulnerability

In this subsection, we will introduce the readers to data processing inequality (DPI) and present our results on posterior g-vulnerability. DPI is a very useful tool to compute and bound posterior g-vulnerability as it allows us to reason about sub-channels.

In classic information theory, the amount of information shared by two random variables is measured by mutual information $I(\cdot, \cdot)$. Then data processing inequality states that post-processing cannot increase information. More precisely, if $X \to Y \to Z$ is a Markov chain, then $I(X; Y) \geq I(X; Z)$. Alvim et al. [1] proved a similar result on posterior g-vulnerability. Their result states that if $X \to Y \to Z$ is a Markov chain, then for any prior distribution π, and any gain function g, $V_g(\pi, X \to Y) \geq V_g(\pi, X \to Z)$.

However, there is a separation between mutual information and posterior g-vulnerability. Given that $X \to Y \to Z$ is a Markov chain, we also know that the reverse chain is Markovian. Which means we can use DPI on mutual information to prove that $I(Y; Z) \geq I(X; Z)$. On the other hand, there is no natural way to compare posterior g-vulnerability between $Y \to Z$ and $X \to Z$ as the gain functions take different inputs if we wish to compute posterior g-vulnerabilities on the two channels. We give conditions on the channels and gain functions where we actually have a DPI between posterior g-vulnerabilities of the two channels.

Theorem 23. Let $X \to Y \to Z$ be a Markov chain. Let π_X be prior distribution on X and $\pi_Y[y] = \sum_{x\in X} \pi_X[x]\mathcal{C}_{X\to Y}[x, y]$ be prior distribution on Y. Let $\mathcal{W}$ be the set of allowed guesses, $g_X : \mathcal{W} \times X \to [0, 1]$ and $g_Y : \mathcal{W} \times Y \to [0, 1]$ be gain functions on X and Y respectively. If $g_Y(w, y) \geq \sum_{x\in X} \Pr[x \mid y]\, g_X(w, X)$ for all $w \in \mathcal{W}, y \in Y$, then

$$V_{g_X}(\pi_X, X \to Z) \leq V_{g_Y}(\pi_Y, Y \to Z).$$

Proof.

$$
\begin{aligned}
& V_{g_X}(\pi_X, X \to Z) \\
=& \sum_{z \in Z} \Pr[z] \max_{w \in \mathcal{W}} \sum_{x \in X} \Pr[x \mid z]\, g_X(w, x) \\
=& \sum_{z \in Z} \Pr[z] \max_{w \in \mathcal{W}} \sum_{y \in Y} \sum_{x \in X} \Pr[x, y \mid z]\, g_X(w, x) \\
=& \sum_{z \in Z} \Pr[z] \max_{w \in \mathcal{W}} \sum_{y \in Y} \sum_{x \in X} \Pr[x \mid y] \Pr[y \mid z]\, g_X(w, x) \\
=& \sum_{z \in Z} \Pr[z] \max_{w \in \mathcal{W}} \sum_{y \in Y} \Pr[y \mid z] \sum_{x \in X} \Pr[x \mid y]\, g_X(w, x) \\
\leq& \sum_{z \in Z} \Pr[z] \max_{w \in \mathcal{W}} \sum_{y \in Y} \Pr[y \mid z]\, g_Y(w, y) \\
=& V_{g_Y}(\pi_Y, Y \to Z).
\end{aligned}
$$

□

In fact, it is interesting to consider the situation where $g_Y(w, y) = \sum_{x \in X} \Pr[x \mid y]\, g_X(w, X)$ for all $w \in \mathcal{W}$ and $y \in Y$. Operationally, the gain function corresponds to the case where one guesses y first, and using his knowledge on the channel $X \to Y$ to invert y into x according to the channel matrix. Furthermore, if the equality above holds, then $V_{g_X}(\pi_X, X \to Z) = V_{g_Y}(\pi_Y, Y \to Z)$.

Combining our theorem with DPI proven by Alvim et al. [1], we can bound posterior g-vulnerability of a channel by that of any of its sub-channels. We formalise the statement in the following proposition.

Proposition 24. Let $X_1 \to \cdots \to X_n$ be a Markov chain. Let π_{X_1} be prior distribution on X_1 and $\pi_{X_i}[y] = \sum_{x \in X_1} \pi_{X_1}[x] \mathcal{C}_{X_1 \to X_i}[x, y]$ be prior distribution on X_i. Let $\mathcal{W}$ be the set of allowed guesses, $g_{X_1} : \mathcal{W} \times X_1 \to [0, 1]$ and $g_{X_i} : \mathcal{W} \times X_i \to [0, 1]$ be gain functions on X_1 and X_i respectively. If $g_{X_i}(w, y) \geq \sum_{x \in X_1} \Pr[x \mid y]\, g_{X_1}(w, X)$ for all $w \in \mathcal{W}, y \in X_i$, then

$$V_{g_X}(\pi_{X_1}, X_1 \to X_n) \leq V_{g_{X_i}}(\pi_{X_i}, X_i \to X_j),$$

for all $1 \leq i \leq j \leq n$.

Proof. It follows from theorem 23 that $V_{g_X}(\pi_{X_1}, X_1 \to X_j) \leq V_{g_{X_i}}(\pi_{X_i}, X_i \to X_j)$. By DPI in [1], $V_{g_X}(\pi_{X_1}, X_1 \to X_n) < V_{g_X}(\pi_{X_1}, X_1 \to X_j)$, and the desired result is proven. □

6 Future Work

In the future, we plan to apply our security notion to analyse security of constructions in the literature and our own. We will try to understand how leakage and prior knowledge affects security of schemes. In particular, we are interested in padding and how they affect efficiency and security. In addition to this line of work, we are interested in attacks on SSE schemes.

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] 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.

[3] 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.

[4] 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.

[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] 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.

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

[8] 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.

[9] 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.

[10] Seny Kamara, Charalampos Papamanthou, and Tom Roeder. Dynamic searchable symmetric encryption. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *ACM CCS 12: 19th Conference on Computer and Communications Security*, pages 965–976, Raleigh, NC, USA, October 16–18, 2012. ACM Press.

[11] Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. Generic attacks on secure outsourced databases. 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 1329–1340, Vienna, Austria, October 24–28, 2016. ACM Press.

[12] 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.

[13] Marie-Sarah Lacharité, Brice Minaud, and Kenneth G. Paterson. Improved reconstruction attacks on encrypted data using range query leakage. Cryptology ePrint Archive, Report 2017/701, 2017. `http://eprint.iacr.org/2017/701`.

[14] A.M. Mathai and P.N. Rathie. *Basic concepts in information theory and statistics: axiomatic foundations and applications.* A halsted press book. Wiley, 1975.

[15] 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.

[16] 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.

[17] Charles V. Wright and David Pouliot. Early detection and analysis of leakage abuse vulnerabilities. Cryptology ePrint Archive, Report 2017/1052, 2017. `http://eprint.iacr.org/2017/1052`.

[18] Yupeng Zhang, Jonathan Katz, and Charalampos Papamanthou. All your queries are belong to us: The power of file-injection attacks on searchable encryption. Cryptology ePrint Archive, Report 2016/172, 2016. `http://eprint.iacr.org/2016/172`.