Leakage Inversion: Towards Quantifying Privacy in Searchable Encryption

Evgenios M. Kornaropoulos
George Mason University, USA
evgenios@gmu.edu

Nathaniel Moyer
George Mason University, USA
nmoyer5@gmu.edu

Charalampos Papamanthou
Yale University, USA
charalampos.papamanthou@yale.edu

Alexandros Psomas
Purdue University, USA
apsomas@cs.purdue.edu

ABSTRACT

Searchable encryption (SE) provides cryptographic guarantees that a user can efficiently search over encrypted data while only disclosing patterns about the data, also known as *leakage*. Recently, the community has developed leakage-abuse attacks that shed light on what an attacker can infer about the underlying sensitive information using the aforementioned leakage. A glaring missing piece in this effort is the absence of a systematic and rigorous method that quantifies the privacy guarantees of SE.

In this work, we put forth the notion of *leakage inversion* that quantifies privacy in SE. Our insight is that the leakage is a function and, thus, one can define its inverse which corresponds to the collection of databases that reveal structurally equivalent patterns to the original plaintext database. We call this collection of databases the *reconstruction space* and we rigorously study its properties that impact the privacy of an SE scheme such as the entropy of the reconstruction space and the distance of its members from the original plaintext database. Leakage inversion allows for a foundational algorithmic analysis of the privacy offered by SE and we demonstrate this by defining closed-form expressions and lower/upper bounds on the properties of the reconstruction space for both keyword-based and range-based databases. We use leakage inversion in three scenarios: (*i*) we quantify the impact that auxiliary information, a typical cryptanalytic assumption, has to the overall privacy, (*ii*) we quantify how privacy is affected in case of restricting range schemes to respond to a limited number of queries, and (*iii*) we study the efficiency vs. privacy trade-off offered by proposed padding defenses. We use real-world databases in all three scenarios and we draw theoretically-grounded new insights about the interplay between leakage, attacks, defenses, and efficiency.

CCS CONCEPTS

• **Security and privacy** → *Management and querying of encrypted data*; *Privacy protections*; **Cryptanalysis and other attacks**.

KEYWORDS

Searchable Encryption; Privacy; Quantifying Metric; Encrypted Databases

CCS '22, November 7–11, 2022, Los Angeles, CA, USA

ACM ISBN 978-1-4503-9450-5/22/11.
https://doi.org/10.1145/3548606.3560593

ACM Reference Format:
Evgenios M. Kornaropoulos, Nathaniel Moyer, Charalampos Papamanthou, and Alexandros Psomas. 2022. Leakage Inversion: Towards Quantifying Privacy in Searchable Encryption. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS '22), November 7–11,2022, Los Angeles, CA, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3548606.3560593

1 INTRODUCTION

Searchable Encryption (SE) [55] is the most prominent cryptographic primitive for searching on encrypted data. The proposed SE schemes strike a balance between efficiency and security and scale to large databases. The work by Curtmola *et al.* [18] was the first cryptographic treatment of SE where schemes were guaranteed to reveal some formally-defined and superficially harmless information, known as a *leakage profile* (or simply leakage). The academic community has built an impressive body of work on SE that covers topics such as dynamic schemes [13, 40, 41, 50], schemes for geometric queries [7, 23, 24, 27], locality-aware schemes [5, 15, 21, 25], schemes optimized for solid-state drives [8], forward and/or backward private schemes [9, 11, 16, 17, 20, 26, 56], schemes for Boolean queries [14, 37], techniques for leakage suppression [3, 29, 39], as well as defenses that decrease the observed leakage [22, 38, 52, 54].

On the offensive end, in the last few years, we have witnessed a surge of results on leakage-abuse attacks [6, 28, 30–32, 36, 42, 44–48, 51, 51, 58, 59] where the goal is to reconstruct the underlying plaintext database (or plaintext query) given access to the leakage of queries. These findings have redefined our perception of an attacker with access to what was initially thought to be "harmless" leakage. In terms of the quality of reconstruction, we have seen attacks that achieve *exact* reconstruction [28, 30, 31, 42, 48] and attacks that achieve a reconstruction with *approximation guarantees* [30, 44, 45].

In this arms race, there is a pressing need to understand the level of privacy offered by a given leakage profile. This task is challenging because a leakage profile only tells us what is *revealed* for each processed query but it does not tell us what *can be inferred* from this piece of information. Therefore, the expression of leakage profile (as a mathematical formulation) is not as useful for assessing privacy. One would hope that at least a *comparative* approach between two leakage profiles would shed some light on the inner workings of leakage. Unfortunately, the most promising formal treatment for leakage comparison, by Bost and Fouque in [10], has very limited applicability. The authors show that a leakage profile Λ leaks less than another leakage profile Λ' if the output of Λ is simulatable from the output of Λ'. In practice, simulatability is possible only

when Λ is a *subset* of Λ' in which case it is straightforward which leakage profile provides more privacy.

Given our insufficient understanding of leakage, academic works resort to evaluating the privacy of a leakage profile based on the outcome of known attacks. This introduces a cat-and-mouse game where slightly tweaked leakage profiles are hypothesized to be secure until a new customized leakage-abuse attack is proposed that reconstructs the plaintext data.

Leakage Inversion. In this work we quantify the privacy of searchable encryption schemes by introducing a rigorous, versatile, and widely applicable approach. Our starting point is to treat a leakage profile as a mathematical *function* that takes as an input a plaintext database DB and a sequence Q of queries and outputs structural characteristics about the plaintext data, i.e., the leaked information. Much like any mathematical function, one can define the *inversion* of the leakage function as the set of plaintext databases for which the sequence of queries Q reveals identical structural characteristics, i.e., the same exact pattern. In other words, the inversion of leakage gives a set of potential databases that are consistent, and therefore explain, the observed leakage. We refer to this set of databases $RS(\text{DB})$ as the *reconstruction space*. Fundamental properties of the reconstruction space, such as its entropy and its geometry, capture the privacy of the leakage from an SE scheme.

The power of leakage inversion is that (*i*) it is universal since it applies to different types of queries, i.e., keyword/range search, etc., as well as variations of leakage profiles, i.e., responses with padded volume, (*ii*) it permits a foundational *algorithmic* treatment for characterizing the structural properties of the reconstruction space, (*iii*) it allows for intuitive and interpretable notions of privacy such as the *entropy* of $RS(\text{DB})$ and the *expected/maximum distance* of a member of $RS(\text{DB})$ to the original plaintext DB, and (*iv*) it allows for a comparative analysis between leakage profiles, e.g., one can directly compare the privacy offered by two schemes by comparing the entropy of their reconstruction spaces.

Leakage inversion can be used not only for SE scheme comparison but also for comparing attacks by assessing the contribution of the auxiliary information towards a successful reconstruction.

Our Contributions. We make the following contributions:

- We define the notion of *reconstruction space* as the set of databases with structurally equivalent leakage with respect to a leakage profile. We define as *leakage inversion* a quantitative property of the reconstruction space that captures a notion of privacy. We put forth the definitions of leakage inversion via *entropy* which applies to both keyword and range search as well as leakage inversion via *bounded maximum distance* and via *expected distance* which apply to range search. These flavors of leakage inversion capture different dimensions of privacy such as the number of potential databases that explain the leakage as well as the distance of a random/furthest guess from the original database.
- We revisit the adversarial assumptions from the area of leakage-abuse attacks under the lens of leakage inversion. Our analysis quantifies the entropy reduction for varying types of auxiliary information and leakage. This application of leakage inversion on attacks quantifies how much of a head start these attacks gain from assuming access to auxiliary information.
- We apply leakage inversion to SE schemes for range queries. Our first finding is that for dense databases the reconstruction space of the standard quadratic scheme (which permits the query leakage of a quadratic number of queries) is identical to the reconstruction space of the more recent augmented binary tree scheme (which permits the query leakage of a linear number of queries). To the best of our knowledge, this is the first evidence that less leakage does not always lead to more privacy. We derive bounds on leakage inversion for the binary tree scheme and evaluate our findings on real-world databases. Our evaluation shows that in the case of the binary tree scheme, the distance between the true plaintext database DB and a random guess from $RS(\text{DB})$ is very close to the maximum distance between DB and a member of $RS(\text{DB})$.
- We apply leakage inversion to a defense mechanism from the SE literature called *padding*. An SE scheme with padding introduces a redundant retrieval of records to obscure the volumetric leakage of SE. We use leakage inversion to analyze the entropy of padding mechanisms from the literature and we apply our analysis to real-world databases. Our evaluation shows that there exist a sweet spot of parameterizing the studied defenses where increased padding does not introduce a significant increase in entropy.

2 PRELIMINARIES

Let $\mathbb{V}$ be the universe of values of a database attribute. We denote the size of the universe of values $N = |\mathbb{V}|$. In the context of keyword search, $\mathbb{V}$ is the universe of words; in the context of range search, $\mathbb{V}$ is the set of values of the attribute on which range queries operate. Let $\mathbb{I}$ be the universe of all possible identifiers of a database. Each identifier is uniquely associated with an encrypted record/file, thus, these terms are interchangeable. We denote the set of identifiers that appear in a given database as $I \subseteq \mathbb{I}$. A database is denoted as DB and is the collection of identifier-value pairs, i.e, $\text{DB} = \{(id, v) | id \in I, v \in \mathbb{V}\}$. Let $\mathbb{Q}$ be the universe of all possible queries on a database DB. In the context of keyword search, the universe of queries $\mathbb{Q}$ can be a subset (in case not all words are queryable) of $\mathbb{V}$. In the context of range search, the universe of queries $\mathbb{Q}$ consists of pairs of values from $\mathbb{V}$, i.e., the boundaries of the range, for which the first value is smaller or equal to the second value. We denote as n the number of identifiers from a database DB, i.e., $n = |I|$. We denote as v the number of values with at least one associated record in DB. We define as $\mathbb{DB}_n$ the universe of all possible databases with n documents over an attribute with N values. The notation $a \xleftarrow{\$} A$ denotes that a is sampled from the set A uniformly at random.

Searchable Encryption. In the following, we define the searchable encryption primitive. The term λ denotes the security parameter. The term PPT refers to a probabilistic and polynomial time algorithm. We only define *static* SE schemes, for the definition of the dynamic case we refer the reader to [41]. A static searchable encryption scheme consists of algorithms *Setup*, *Trpdr*, *Search*, and *Filter* that are executed between a client and a server.

- $(K, EDB) \leftarrow Setup(\lambda, DB)$: On input the security parameter λ and the plaintext database DB the algorithm outputs client's secret key K and an encrypted database *EDB* initialized with DB that is outsourced to the server.
- $t \leftarrow Trpdr(K, q)$: On input the client's secret key and the query $q \in \mathbb{Q}$ the algorithm outputs a token (i.e., a trapdoor) for query q.

- $R \leftarrow$*Search(t, EDB):* On input a token t from the client and an encrypted database *EDB* from the server the algorithm outputs a set of identifiers $R \subseteq \mathbb{I}$ to the client.
- $R' \leftarrow$*Filter(q, R):* On input a set of identifiers R and q the algorithm returns a set of identifiers $R' \subseteq R$. The algorithm runs locally on the client.

Leakage Profile. Following the notation presented in [39], we define as $\mathsf{qeq}(q_1, \ldots, q_t) \coloneqq M$ the *query equality pattern* that takes as an input queries $q_1, \ldots, q_t$ and outputs a binary matrix M such that $M[i, j] = 1$ if $q_i = q_j$ and $M[i, j] = 0$ otherwise. The above pattern is also known as the search-pattern leakage. Next, we define the *response identity pattern*, denoted as $\mathsf{rid}(q)$, as the set of identifiers in DB that are returned when query q is issued on DB. We note here that $\mathsf{rid}(q)$ refers to the exact answer of the query on the plaintext DB. There are response-revealing schemes, e.g., [23, 27], that generate false positive responses which means that the returned identifiers are a superset of $\mathsf{rid}(q)$. To capture this behavior that depends on a scheme Σ, we define the *scheme-specific response identity pattern*, denoted as $\mathsf{srid}(q)$, to be the set of identifiers of DB that are returned by $\Sigma.Search(t, EDB)$ algorithm where *EDB* is derived as $(K, EDB) \leftarrow \Sigma.Setup(\lambda, \mathsf{DB})$ and t is derived as $t \leftarrow \Sigma.Trpdr(K, q)$. For the simpler case where the SE scheme returns exact responses, e.g., scheme QD for ranges and scheme [13] for keyword search, the outputs of srid and rid are equal sets (assuming *EDB* is generated from DB via *Setup* of Σ).

Given a database DB we define $\mathsf{vol}(q)$ as the number of identifiers in DB that are returned when q is issued on DB. We call *response-hiding* the SE schemes that hide the response identity pattern (which records are retrieved) and only reveal $\mathsf{vol}(q)$ (how many records are retrieved or else the size of set $\mathsf{rid}(q)$). If a scheme is not response-hiding then we say it is *response-revealing*. Analogously to srid, we define the *scheme-specific volume pattern* which is denoted as $\mathsf{svol}(q)$ to be the size of the set of identifiers of srid. We call *padding* any mechanism that alters the exact set of records $\mathsf{rid}(q)$ that is retrieved by query q. The goal of padding is to obfuscate the volumetric information that an attacker observes. We define as $\mathsf{trlen}(\mathsf{DB}) \coloneqq \sum_{q \in \mathbb{Q}} \mathsf{vol}(q)$ the *total response length pattern*, i.e., the setup leakage. We define as *leakage profile* for the searchable encryption scheme the collection Λ of setup leakage $\mathcal{L}_{\mathsf{Setup}}$ and query leakage $\mathcal{L}_{\mathsf{Query}}$ is defined as: $\Lambda = \left(\mathcal{L}_{\mathsf{Setup}}, \mathcal{L}_{\mathsf{Query}}\right)$. Typical constructions in SSE, i.e., without any additional mitigation such as padding, response-hiding, false positive responses, have leakage profile $\Lambda = \big(\mathsf{trlen}, (\mathsf{qeq}, \mathsf{rid})\big)$.

Informally, an SE scheme with leakage Λ reveals nothing about the underlying plaintext DB other than its leakage functions. The formal security is captured by the real/ideal paradigm with games Real^{SE} and Ideal^{SE} (see [18, 56] for a detailed description).

Definition 1. *An SE scheme Σ=(Setup, Trpdr, Search, Filter) is* adaptively secure *with respect to leakage profile Λ, iff for any ppt adversary Adv issuing poly(λ) queries, there exists a stateful ppt simulator Sim and a negligible function negl(λ) such that:*

$$|\,\mathbf{Pr}[\mathit{Real}^{SE}_{\mathit{Adv},\Sigma}(\lambda) = 1] - \mathbf{Pr}[\mathit{Ideal}^{SE}_{\mathit{Adv},\mathit{Sim},\Lambda}(\lambda) = 1]| \leq \mathit{negl}(\lambda).$$

A response-revealing SE scheme is correct if for all queries, the set of identifiers returned in the scheme-specific pattern srid is a superset of (or equal to) the identifiers from pattern rid. The next definition allows schemes with false positive responses (but not false negatives) to be correct, e.g., [23, 27]. Algorithm *Filter* removes the false positive records (if any) and returns the exact response.

Definition 2. *An SE scheme Σ=(Setup, Trpdr, Search, Filter) is* correct *if for every $q \in \mathbb{Q}$ of DB and (K, EDB) derived from Σ.*Setup*, after the execution of $t \leftarrow \Sigma.\mathrm{Trpdr}(K, q)$ and $R \leftarrow \Sigma.\mathrm{Search}(t, EDB)$, the algorithm* $\mathrm{Filter}(q, R)$ *returns a set of identifiers R' that is equal to the set of identifiers from pattern $\mathsf{rid}(q)$.*

Categories of Databases. We differentiate between two types of databases. The first is a *keyword-based database* which is a database that allows keyword search on documents. Each document may contain multiple keywords therefore there may exist pairs with the same identifier *id* and different values, $(id, v), (id, v') \in \mathsf{DB}$. For the case of keyword-based databases, we define $\mathbb{Q}$ as the set of keywords that can be queried in DB. The second type is a *range-based database* which is a database that allows range-based search on records, e.g., range queries on the attribute AGE of a hospital patient. In this case, each record is associated with a single value, therefore, in DB, there exists at most one pair (id, v) for any given id. That is, a hospital patient can only have one value associated with the attribute AGE. To simplify our analysis we assume that the values of a range-based database are positive and non-zero integers. Our analysis can be adjusted to address other cases. We define the *span* of a query $q = (\alpha, \beta)$ as the number of values covered by q, i.e., $\beta - \alpha + 1$ values. For the case of range-based databases, given a value $v \in \mathbb{V}$ we define as the *reflection* of value v the point $v^R = N - v + 1$. For the case of range-based databases we define $\mathbb{Q}$ as the set of ranges with boundaries in $\mathbb{V}$. A *dense* range-based database has at least one record associated with each value of $\mathbb{V}$. If a range-based database is not dense then we say that it is *sparse*.

Distance in Range-Based Databases. From the leakage attack literature, it is known that without any significant prior knowledge (e.g., known data distribution) plaintext reconstructions are correct *up to reflection* [42]. Let a range-based database $\mathsf{DB} \in \mathbb{DB}$, we define as DB^R the database with the reflected values, i.e., $\mathsf{DB}^R = \{(id, v^R) |$ where $(id, v) \in \mathsf{DB}\}$. Thus, for two range-based databases DB_i and DB_j, their distance is defined as:

$$d(\mathsf{DB}_i, \mathsf{DB}_j) = \tfrac{1}{n} \min \left\{ \begin{array}{cc} \sum\limits_{\substack{(id,v) \in \mathsf{DB}_i \\ (id,v') \in \mathsf{DB}_j}} |v - v'|, & \sum\limits_{\substack{(id,v) \in \mathsf{DB}_i \\ (id,v') \in \mathsf{DB}_j^R}} |v - v'| \end{array} \right\}.$$

We assume that $\mathsf{DB}_i, \mathsf{DB}_j$ are defined on the same set of identifiers.

On Range-Based Schemes. Encrypted range schemes can be categorized as either exact response or approximate response [23, 27]. The approximate response designs allow false positives by serving only a subset of ranges. This design choice reduces both the storage overhead (compared to serving all possible ranges) and the leakage (since the server won't see the response of some ranges). We briefly describe both types of range schemes using the definitional framework of regular structured encryption schemes introduced by Kornaropoulos *et al.* (Definition 1 in [46]).

The range queries of $\mathbb{Q}$ can be partitioned into groups of queries with the same span. A scheme with exact responses (one where the set srid is equal to rid for all queries) answers without false positives all possible range queries and therefore it stores (in the

worst-case) a quadratic number of responses for queries across all spans. We call the above exact response scheme the *quadratic* range scheme, denoted as QD. On the other hand, range schemes with approximate responses only serve a set Q' of ranges where $Q' \subset \mathbb{Q}$. We call Q' the set of *allowable ranges*. When the client issues a range $q \in \mathbb{Q} - Q'$ outside the allowable ranges, the scheme maps q to a range $q' \in Q'$ such that the set of identifiers in the response for q' is a superset of the response for q, also called a *cover* of a range.

Range schemes with approximate responses from the literature form Q' by allowing only ranges that are powers of 2, i.e., span 2^i for $i \in [0, \log N]$. Furthermore, given a fixed span 2^i, the approximate schemes from the literature only include a limited number of ranges with a span 2^i. Specifically, the first scheme that we consider in this work is called the Binary Tree scheme [27], denoted as BT. For scheme BT, Q' contains the sequence of ranges $[1, 2^i], [2^i + 1, 2 \cdot 2^i], [2 \cdot 2^i + 1, 3 \cdot 2^i], \ldots$ for all $i \in [0, \log N]$. More concisely, BT builds the sequence of ranges of span 2^i by starting from $[1, 2^i]$ and increasing both boundaries by an additive factor of 2^i. This additive factor is called the *step* of the span. Intuitively, scheme BT can be seen as a binary tree where each leaf maps to a value of the attribute domain. Then, the set of allowable ranges Q' contains the nodes of the binary tree where each node u represents a range that spans across the leaves that u can reach (see Figure 1). The second scheme that we consider from the literature is the Augmented Binary Tree scheme [23], denoted as ABT, which is a binary tree augmented with nodes that are placed in-between every two consecutive (per level) internal nodes. More formally, ABT builds the sequence of ranges of span 2^i by starting from $[1, 2^i]$ and increasing both boundaries by an additive factor of 2^{i-1}. For example, for $N = 8$ the ranges/nodes in the BT scheme are $[1, 1], \ldots, [8, 8]$ of span 1, then $[1, 2], [3, 4], [5, 6], [7, 8]$ of span 2, then $[1, 4], [5, 8]$ of span 4, and $[1, 8]$ of span 8. The corresponding ABT scheme has all the above ranges/nodes plus the in-between ranges $[2, 3], [4, 5], [6, 7], [3, 6]$. The rid (resp. srid) pattern of the above schemes reveals only the retrieval of the allowable ranges.

In this work, we focus on schemes QD, BT, and ABT since they are considered standard by the community. We mention that one can construct different approximate response schemes, e.g., by parameterizing differently the span/step of the scheme, which requires leakage inversion analysis beyond the scope of this work.

3 LEAKAGE INVERSION

In this section, we introduce for the first time an approach for quantifying privacy in searchable encryption. On a high level we use a leakage profile Λ of an SE scheme on a given database DB to produce the set of all possible databases that may have produced a "structurally equivalent" leakage as the original DB. We formalize this concept for keyword search and range search and introduce the concept of *leakage inversion* which is a function that takes as an input a plaintext database DB as well as a leakage profile Λ of an SE scheme and outputs a numerical value that quantifies privacy based on its reconstruction space.

3.1 Limitations of Current Approaches

Why Can't We Compare Leakage Directly? Even though SE is becoming a mature cryptographic primitive [55], we still lack a rigorous approach for quantifying the privacy of an SE construction. For example, we still cannot give a satisfactory answer to the following fundamental question

"*Does leakage Λ provide more privacy than leakage Λ'?*"

The most promising approach is given by Bost and Fouque in [10]. Informally, the proposal in [10] suggests that leakage profile Λ leaks less than Λ' if the output of Λ is simulatable from the output of Λ'. The simplified version for the static SE reads:

Definition 3. *[10] Define two leakage profiles for a static SE scheme as $\Lambda = \left(\mathcal{L}_{Setup}, \mathcal{L}_{Query}\right)$ and $\Lambda' = \left(\mathcal{L}'_{Setup}, \mathcal{L}'_{Query}\right)$. We say that Λ leaks less than Λ', denoted by $\Lambda \preceq \Lambda'$, if and only if, there exists a pair of stateful polynomial-time algorithms $\mathcal{S} = (\mathcal{S}_{Setup}, \mathcal{S}_{Query})$, such that, for any database DB and sequence of queries $(q_1, \ldots, q_m)$,*

$$\mathcal{L}_{Setup}(DB) = \mathcal{S}_{Setup}\left(\mathcal{L}'_{Setup}(DB)\right)$$

$$\forall 1 \leq i \leq m, \mathcal{L}_{Query}(q_i) = \mathcal{S}_{Query}\left(\mathcal{L}'_{Query}(q_i)\right).$$

As noted in [10] the relation $\preceq$ is a partial order therefore *not all leakage profiles are comparable* under $\preceq$. We present two cases that highlight the fundamental shortcomings of using the $\preceq$ relation.

Case I: Inability to Capture Leakage Granularity. Consider the case where Λ describes the leakage of a response-hiding SE augmented by the following padding: pad the volume to the closest power of 2, an approach proposed in [22]. Next, consider the case where Λ' describes the leakage of a response-hiding SE with the following padding: pad the volume to the closest power of 3. Notice that each gap between consecutive powers of 3, i.e., padding in Λ', is significantly larger than its counterpart for consecutive powers of 2, i.e., padding in Λ. In other words, the volumes of Λ' are of more coarse-granularity (w.r.t. the plaintext volumes) than Λ. Nevertheless, the larger uncertainty provided by Λ' can not be captured by the framework of [10] since we cannot simulate one leakage from the other. For example, if we attempt to simulate Λ from Λ' we may observe a padded volume 3^4. Even though the simulator knows that its true volume is larger than 3^3 it is not possible to know if it is smaller than 2^5 (in which case the simulator should map it to 2^5) or larger than 2^5 (in which case it should be mapped to 2^6). If we attempt to simulate Λ' from Λ we may observe a padded volume 2^5 and even though we know that its true volume is larger than 2^4 it is not possible to know if it should be mapped to 3^3 or 3^4.

Case II: Less Leakage $\nRightarrow$ More Privacy. The shortcomings of Definition 3 also appear in the context of encrypted ranges. The fact that QD scheme reveals the leakage of a quadratic number of ranges whereas ABT scheme reveals the leakage of a linear number of ranges, implies that the attacker sees strictly less information in ABT deployments. Nevertheless, Definition 3 is not applicable in this case either since we cannot simulate one leakage from the other. Given our inability to quantify privacy in the context of SE, the community developed *heuristics* on how to compare leakages. So far, the rule of thumb is "less leakage means more privacy". Under this heuristic, one would think that ABT provides more privacy than QD since an attacker in ABT attempts to reconstruct with the leakage from a linear number of distinct queries whereas in QD there are $O(N^2)$ distinct queries. In Section 5, we show that this intuition is not always correct. Specifically, we show that the number of plaintext databases that "explain" the leakage from ABT, is identical

to the number of databases for QD in the case of dense databases. Therefore, less leakage doesn't always imply more privacy.

For completeness, we note that the work of Jurado *et al.* [34, 35] proposes quantifying information leakage for Order-Revealing Encryption (ORE) based on the framework of Quantitative Information Flow [2]. On a high level, QIF applied to ORE quantifies privacy as the difference between the probability of correct reconstruction before revealing the ordering of ORE ciphertexts and after. Proposed SE schemes for keyword-search and range-search do not reveal the ordering by ciphertext alone (as opposed to ORE). Thus, the above results are not directly applicable to the more general case of SE.

3.2 Quantifying Privacy by Analyzing the Reconstruction Space

In the following, we introduce the technical tools to capture the intuition that several databases can generate the observed leakage of a given instantiated *EDB*. We define the retrieval matrix *RM* which describes which identifiers are retrieved for each query according to the leakage from the algorithms of an SE scheme. The retrieval matrix is a concept that can be applied to different schemes and classes of queries, even beyond keyword search and range search, since it only specifies which records are retrieved for each query.

Definition 4. *(**Retrieval Matrix**) Let DB be a database, let $\mathbb{Q}$ be the query universe of DB, let $I \in \mathbb{I}$ be the set of identifiers that appear in DB, and let EDB be derived by running algorithm* Setup *of SE scheme Σ with leakage profile Λ. We define as a **retrieval matrix** RM (with respect to profile Λ and DB) the binary matrix with $|\mathbb{Q}|$ rows and $|I|$ columns such that for each $q \in \mathbb{Q}$, cell $RM[q, id]$ takes value 1 if $id \in srid(q, EDB)$ and value 0 otherwise.*

We emphasize that an attacker only observes query tokens and responses, and does not know which token corresponds to which plaintext query (i.e., the label of a row of *RM*). Therefore, to the eyes of the attacker, all plaintext databases that produce the *RM* of DB (up to a permutation of rows) are a valid candidate reconstruction, e.g., see Figure 1. Based on the above insight, we define the notion of structurally equivalent leakages and provide a definition of a reconstruction space that applies to keywords and ranges.

Definition 5. *(**Structurally Equivalent Leakage**) Let DB be a database, let Σ be an SE scheme with leakage profile Λ. Let DB$'$ be a database different from DB. Let RM (resp. RM$'$) be the retrieval matrix of DB (resp. DB$'$) under scheme Λ. Let $\hat{Q}_{DB} \subseteq \mathbb{Q}$ be the set of queries in RM such that if $q \in \hat{Q}_{DB}$, then the row vector $RM[q, *]$ has at least one cell with value 1. Let $\hat{Q}_{DB'} \subseteq \mathbb{Q}$ be the set of queries in RM$'$ such that if $q \in \hat{Q}_{DB'}$ then vector $RM'[q, *]$ has at least one cell with value 1. We say that DB and DB$'$ have **structurally equivalent leakage** under Λ if there exists a bijection $\phi : \hat{Q}_{DB'} \rightarrow \hat{Q}_{DB}$ such that for every $q \in \hat{Q}_{DB'}$, we have that $RM'[q, id] = RM[\phi(q), id]$ for all $id \in I$.*

We highlight two characteristics of the definition of structural equivalence. First, the bijection ϕ, which can be thought of as a relabeling of queries, operates on the set of queries that return at least one record. If one allows the bijection to extend to empty queries, i.e., queries from $\mathbb{Q} - \hat{Q}_{\text{DB}}$, then we allow relabeling between queries that return no records. These empty queries play no role in the content or the retrieval operations of the DB at hand. Relabeling empty queries artificially augments the set of databases that have structurally equivalent leakage with DB which may convey a false sense of security. Second, the definition only considers the bijection on rows but not on columns. Relabeling columns would be equivalent to swapping content between records which, indeed, generates distinct databases. However, in a realistic scenario an adversary observes the query leakage of a given DB under an instantiated scheme Σ. Relabeling columns would result in databases that conflict with the observed leakage. To illustrate this point, consider the case where Definition 5 allows relabeling of columns of *RM*. Under such a definition, one could generate a keyword database DB$''$ that is structurally equivalent to the upper-left DB of Figure 1 by swapping the content between id_1 and id_3 as well as the content between id_2 and id_4. The resulting database DB$''$ wouldn't explain the observed leakage of the true DB since id_1 in DB contains three words (and is retrieved by 3 queries) while id_1 in DB$''$ contains two words (and is retrieved by 2 queries).

Definition 6. *(**Reconstruction Space**) Let DB be a database and let Σ be an SE scheme with leakage profile Λ. We define the **reconstruction space**, denoted by $RS(DB)$, as the collection of databases that have structurally equivalent leakage to DB under Λ.*

The term reconstruction space was introduced by Kornaropoulos *et al.* [44] in the context of encrypted k-Nearest Neighbor queries. Later works [28, 47] used a similar notion for two-dimensional queries. These works [28, 44, 47] used a similar concept to $RS(\text{DB})$ to explain the inherent limitations of geometric leakage attacks, i.e., the impossibility of distinguishing between symmetries (see rotations/reflections) or identical Voronoi diagrams. In this work, we give the first definition of reconstruction space that applies to keyword search and (one-dimensional) range search but can also be extended to other queries and response-hiding schemes. More importantly, our work is the first to study the *structure* of the reconstruction space, e.g., the size, the entropy, and the maximum/expected distance, for the popular cases of keyword search and range search.

What Properties Capture Privacy? We propose two properties that capture privacy in SE, the first is a measure of uncertainty of $RS(\text{DB})$ (i.e., the entropy) and the second captures the geometry of $RS(\text{DB})$ (i.e., the maximum and the expected distance of a reconstruction from the original plaintext DB). *Shannon entropy* is the most used measure of uncertainty. In this context, it formalizes the uncertainty that an attacker faces in an attempt to choose a correct reconstruction among $RS(\text{DB})$. In several leakage-abuse attacks works [12, 33], the goal of the adversary is to recover the *exact* plaintext database that an SE operates on. In this scenario, the higher the entropy the harder it is for the attacker to guess the correct underlying plaintext. In the case where the attacker does not have any auxiliary information, all the members of $RS(\text{DB})$ are treated as equiprobable and the entropy reduces to the logarithm of the *size of the reconstruction space* $RS(\text{DB})$.

Additional Filtering based on Auxiliary Information. In case the attacker has access to auxiliary information, denoted as aux, then the reconstruction space can be further filtered to reflect the information exposed by aux. We differentiate between two categories of auxiliary information. The first category of aux causes the *removal* of a set of databases from $RS(\text{DB})$ even if it has structurally

equivalent leakage with the original DB. E.g., if aux gives away that the keyword "cost" is definitively in the plaintext database, then all the members of $RS(\mathrm{DB})$ that do not contain this keyword are removed from $RS(\mathrm{DB})$. The second category of aux causes an *adjustment of the probabilities* for each database in $RS(\mathrm{DB})$. E.g., if aux is the data distribution P, then the probability associated with a $\mathrm{DB}' \in RS(\mathrm{DB})$ is the probability that DB' is generated by P. In this work, we focus on the first category of auxiliary information.

For generality, we capture both the case with auxiliary information and the case without, with the term Γ which is a random variable (r.v.) over $RS(\mathrm{DB})$. In the case of $\mathrm{aux} = \perp$, r.v. Γ has a uniform probability distribution over $RS(\mathrm{DB})$. In the case of $\mathrm{aux} \neq \perp$, r.v. Γ has (a potentially nonuniform) probability distribution over $RS(\mathrm{DB})$ that respects the given auxiliary information (e.g., removed databases from $RS(\mathrm{DB})$ are assigned probability 0).

Definition 7. *Let DB be a database, let Σ be an SE scheme with leakage profile Λ. Let aux be the auxiliary information that an attacker has access to. Let Γ be a random variable with probability distribution over $RS(DB)$ which is computed as a function of aux. We define a* ***leakage inversion via entropy*** *as the function*

$$LeakInv_H(DB, \Lambda, aux) \triangleq H(\Gamma).$$

The Geometry of the Reconstruction Space. Besides exact reconstruction, we are interested in capturing privacy with respect to *approximate reconstruction* which appears mainly in the context of range-based databases [44–46]. In this scenario, not only the size but also the *geometry* of the reconstruction space matters in defining privacy. We propose the study of the maximum distance between the true plaintext DB and a member from $RS(\mathrm{DB})$. This privacy measure provides an *upper bound on the quality of the approximation* for an attacker reconstructing among $RS(\mathrm{DB})$.

Additionally, we study the expected distance between a randomly chosen database from $RS(\mathrm{DB})$ and the true plaintext DB. This measure describes how close (in expectation) a random member of $RS(\mathrm{DB})$ is to the original plaintext database. If the expected distance is small then even if the size of $RS(\mathrm{DB})$ is large, an attacker can accurately approximate the original DB by choosing a random reconstruction from $RS(\mathrm{DB})$. The term $\mathrm{supp}(\Gamma)$ refers to the *support* of the probability function of Γ.

Definition 8. *Let DB be a database with numerical values, let Σ be an SE scheme with leakage profile Λ. Let aux be the auxiliary information that an attacker has access to. Let Γ be a random variable with probability distribution over $RS(DB)$ which is computed as a function of aux. We define:*

- ***leakage inversion via bounded maximum distance*** *as:*

$$LeakInv_{MX}(DB, \Lambda, aux) \triangleq \max_{x \in supp(\Gamma)} d(x, DB),$$

- ***leakage inversion via expected distance*** *as:*

$$LeakInv_{XP}(DB, \Lambda, aux) \triangleq \mathrm{E}[d(x, DB) | x \xleftarrow{s} supp(\Gamma)].$$

4 LEAKAGE INVERSION ON CRYPTANALYTIC ASSUMPTIONS

In the infancy of SE, the community focused its defensive efforts on the assumption that adversaries would use query leakage to reconstruct the underlying plaintexts. This changed in 2012 when

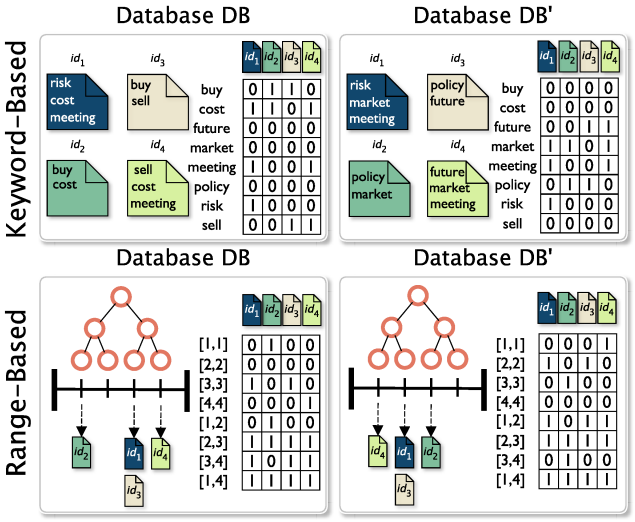

Figure 1: An illustration of databases with structurally equivalent leakages for the case of keyword-based (first row) and range-based (second row).

Islam *et al.* [33] demonstrated that an attacker with partial knowledge of DB and auxiliary information (i.e., knowledge of keyword frequency statistics and a number of known queries) could mount a query-recovery attack. Since then, researchers have developed a number of attacks [6, 12, 51] that leverage a variety of cryptanalytic assumptions. Most of the attacks focus on keyword query recovery but it is possible to re-purpose a successful query recovery attack to infer the content of the encrypted DB. In this section, we study the reconstruction of the contents of the encrypted records of DB.

When benchmarking leakage attacks [36] one typically compares the reconstruction accuracy across various datasets, but there is another dimension that is ignored. This is the role of cryptanalytic assumptions in the success of an attack. As mature as the cryptanalytic efforts are, we still do not know how much of an advantage these assumptions offer to an attacker. This is because we compare the results of attacks but not their starting points. It is hypothesized that not all types of auxiliary information are the same but we do not have the tools to quantify this intuition.

In this section, we focus on response-revealing keyword-based databases and use leakage inversion to quantify the *privacy reduction* from cryptanalytic assumptions. Specifically, we give closed-form expressions to calculate the entropy of the reconstruction space under common cryptanalytic assumptions. We consider cases of varying degrees of leakage and types of auxiliary information:

(1) *Setup leakage, no queries issued, no auxiliary information*
(2) *Leakage after observing all queries, no auxiliary information*
(3) *Leakage after observing all queries, known keywords*
(4) *Leakage after observing all queries, mapping from plaintext query-to-volume as auxiliary information*

Entropy from Setup Leakage. Setup leakage reveals the number of keyword-identifier pairs in DB, i.e., the trlen pattern. Notice that this number does not disclose the number of distinct documents or the number of unique keywords in DB. Therefore, to count the

number of possible databases that satisfy the setup leakage we must consider all valid numbers of documents and all valid numbers of keywords in each document.

We paint the picture of why it is impractical to calculate the entropy in this case by using a basic method from combinatorics. At one extreme, we have the case where each keyword-identifier pair belongs to a distinct document, i.e., we have as many documents as pairs $n = \mathsf{trlen}(\mathsf{DB})$. At the other extreme, we have the case where all keyword-identifier pairs belong to the same document, i.e., $n = 1$ with $\mathsf{trlen}(\mathsf{DB})$ distinct keywords. In between these two extremes, one can consider the number of documents as a variable x and we have to consider all possible ways that we can assign keyword-identifier pairs to x distinct documents. This question is a variation of the *divider method* from combinatorics that counts the permutations of $\mathsf{trlen}(\mathsf{DB}) + (x-1)$ identical balls and $x-1$ dividers. By applying the divider method to our problem we get $\sum_{x=1}^{\mathsf{trlen}(\mathsf{DB})} \binom{\mathsf{trlen}(\mathsf{DB})+(x-1)}{x-1}$ ways of assigning $\mathsf{trlen}(\mathsf{DB})$ keywords to x documents. For small values such as $\mathsf{trlen}(\mathsf{DB}) = 50$ we have over 10^{29} ways of assigning keywords to documents. Thus, it is infeasible to scale the above calculations to realistic datasets. (The Enron database, for example, has $\mathsf{trlen}(\mathsf{DB}) = 10^6$.) In the above analysis, leakage inversion confirms the intuition that the setup leakage is insignificant. Perhaps unsurprisingly, there exists no reconstruction attacks that rely solely on setup leakage.

Entropy from Query Leakage. In the following, we analyze the entropy when all keywords in the database have been queried, that is, the rid pattern is revealed across all queries. In contrast to the previous case, we note that once all queries have been processed, the number of documents, the number of distinct keywords that appear in DB, and the co-occurrence of tokens in documents are all known. The only unknown is the plaintext values of the keywords. Given that we assume no auxiliary information, any mapping from the keyword universe to observed tokens is possible, i.e., we cannot exclude mappings. It follows that every distinct assignment of ν (out of N) keywords to ν tokens constitutes a distinct plaintext database DB' that is a member of the reconstruction space $RS(\mathsf{DB})$.

Interestingly, there is an exception to the previous statement for which distinct mappings give the same database. This phenomenon occurs when we have a set S of tokens that always return the same documents from DB. That is, if the keyword "leakage" and the keyword "abuse" always appear together it does not matter which token refers to which word. An exact calculation of the entropy would treat all possible mappings of keywords to S as a single database. For simplicity, we approximate the entropy by overcounting the above cases (we indicate this by $\approx$). This approximation is supported by the observation that in most real datasets, including the ones in our experiments, there are only a few keywords that always appear together and, thus, their impact on entropy is small.

Theorem 1. *Let DB be a keyword-based database with a keyword universe of size N and ν distinct keywords that appear in at least one document. Let $\Lambda = (\mathit{trlen}, (\mathit{qeq}, \mathit{rid}))$ be the leakage profile of an SE scheme. Then, the leakage inversion with respect to entropy after observing all queries is $\mathit{LeakInv}_H(\mathit{DB}, \Lambda, \perp) \approx \log\left(\binom{N}{\nu}\right) + \log(\nu!)$.*

Proof Sketch. After all queries have been issued, ν tokens will have been observed. We count the number of possible databases in $RS(\mathsf{DB})$ in two steps, (i) by counting the number of ways we can choose ν keywords out of the set $\mathbb{V}$, and (ii) by counting the number of ways we can assign the chosen keywords to the ν observed tokens. There are $\binom{N}{\nu}$ ways of choosing ν active keywords out of a keyword universe of size N, and $\nu!$ ways of assigning ν keywords to ν observed tokens. Thus, the possible databases are $\binom{N}{\nu} \cdot \nu!$. Since all possible databases are equiprobable, the Shannon entropy of the reconstruction space is $\log\left(\binom{N}{\nu} \cdot \nu!\right) = \log\left(\binom{N}{\nu}\right) + \log(\nu!)$. □

We note that leakage attacks achieve at least some success with *partial* query leakage. We assume that *all queries* have been issued but one can devise a parameterized version of our approach where the entropy is computed based on the subset of observed queries.

Entropy from Query Leakage & **Known Keywords.** In the following, we analyze the entropy when all keywords in the database have been queried and the set of keywords that appear DB is given as auxiliary information. Unlike the previous case, the entropy grows strictly as a function of ν.

Theorem 2. *Let DB be a keyword-based database with ν distinct words that appear in a document at least once. Let auxiliary information $\mathit{aux} = \{v : \exists \mathit{id} \in I, (\mathit{id}, v) \in \mathit{DB}\}$. Let $\Lambda = (\mathit{trlen}, (\mathit{qeq}, \mathit{rid}))$ be the leakage profile of an SE scheme. Then, the leakage inversion with respect to entropy after observing all possible queries is $\mathit{LeakInv}_H(\mathit{DB}, \Lambda, \mathit{aux}) \approx \log(\nu!)$.*

Entropy from Query Leakage & **Volume Vector.** In the following, we analyze the entropy when all keywords in the database have been queried and there is some available auxiliary information. Specifically, the auxiliary information is defined as the mapping from a plaintext query to the number of documents in which it occurs, i.e., the volume $\mathsf{vol}(q)$ of the query. This mapping is called "volume vector" and is used as auxiliary information in the attack by Oya and Kerschbaum [51]. To illustrate why the entropy is reduced in this scenario, suppose there is only one token t that returns exactly 10 documents when queried. Then, given access to the volume vector for DB, there exists only one plaintext keyword that explains the observed volume leakage of token t. We say that the above mapping *respects* the volume vector. In general, to count the size of the reconstruction space, and in turn calculate the entropy, it is enough to consider the databases that result from all possible mappings from keywords to tokens that respect the volume vector. A comparison between the next theorem and Theorem 2 shows a significant reduction in entropy given auxiliary information.

Theorem 3. *Let DB be a keyword-based database and let $\Lambda = (\mathit{trlen}, (\mathit{qeq}, \mathit{rid}))$ be the leakage profile of an SE scheme. Let the collection of sets of keywords with the same volume, i.e., $\{G_i = \{v : \mathit{vol}(v) = i\}\}$ for all $i \in [1, ..., \nu]\}$, be the auxiliary information aux. Then, the leakage inversion with respect to entropy after observing all possible queries is $\mathit{LeakInv}_H(\mathit{DB}, \Lambda, \{G_i\}_{i=1}^{\nu}) = \sum_{i=1}^{\nu} \log(|G_i|!)$.*

Proof Sketch. Recall that the execution of all queries means that the volume of every token has been observed. This means that for every keyword, there is at least one token with the same volume. If some number of keywords share a volume, however, then every bijection from those keywords to the set of tokens with the same volume must be counted as a unique, structurally equivalent database. Thus, for every set of keywords G_i with the same volume i,

we count the possible assignments between those keywords and the equal-volume tokens as $|G_i|!$, yielding a reconstruction space of size $\prod_{i=1}^{\nu} |G_i|!$. Each of these databases is equiprobable, so the logarithm of the size of the reconstruction space is $\sum_{i=1}^{n} \log(|G_i|!)$. □

		Keyword-Based Databases					
	Cryptanalytic Assumptions	Enron		Apache		Ubuntu	
		# Keywords		# Keywords		# Keywords	
		500	5,000	500	5,000	500	5,000
$\text{Leakinv}_{\text{H}}$	Leakage: All Queries Aux: $\perp$	7,969	79,424	8,246	82,283	10,084	100,830
	Leakage: All Queries Aux: Known Keywords	3,767	54,232	3,767	54,232	3,767	54,232
	Leakage: All Queries Aux: Volume Vector	71	13,467	25	11,195	1	1,110
	Number of Records n	29,461		50,531		26,360,716	
	Keyword Universe	63,031		92,454		1,179,077	

Table 1: Leakage inversion via entropy on cryptanalytic assumptions. The entropy reduction quantifies to what degree the auxiliary information helps in plaintext reconstruction.

4.1 Evaluation

Methodology. We test our findings on the entropy reduction from cryptanalytic assumptions using three datasets, all of which come from real-world communications between individuals. The Enron [43] and Apache [4] email corpora consist of around 30,000 and 50,000 emails, respectively, and feature prominently in the attack literature [6, 12, 19, 33]. We also use the Ubuntu chat corpus, first introduced to the SE community in 2016, which consists of around 26,000,000 IRC messages from an Ubuntu development channel. We treat each message as a document. This results in a document count for Ubuntu that is about 5 times larger than Apache and 9 times larger than Enron. The size of the dataset, as well as the nature of the messages (anonymous chat concerning a technical topic) contribute to a high number of unique words in Ubuntu: about 1.2 million, which is approximately 10 and 15 times larger than Apache and Enron, respectively. Following the methodology of the attack literature, we restrict our analysis to "vocabularies" that consist of the 500 and 5,000 most frequent words, where a word's frequency is the number of documents containing that word. These numbers reflect the typical range of vocabulary sizes used in the leakage-abuse attack literature [12].

Experimental Results. Table 1 summarizes the results of our experiments. Unsurprisingly, we observe decreasing entropy as greater leakage/auxiliary information is provided. Interestingly, though, the entropy decreases at different rates for different datasets. Moreover, we observe that one dataset can have greater entropy than another under one set of assumptions, but less entropy under another. E.g., Ubuntu has the highest entropy in the "Aux: $\perp$" case, but the lowest entropy in the "Aux: Volume Vector" case, for both small and large vocabularies. The high entropy in the first case is due to Ubuntu's large number of unique words, while the low entropy in the second case is due to its high number of frequent words with unique volumes. This finding highlights the fact that entropy is not strictly a function of the leakage, the auxiliary information, or the database alone. Rather, it is determined by all three.

The adversarial assumptions in our experiments come directly from the attack literature. The case of "Aux: $\perp$" reflects the baseline scenario, against which no attack is proposed so far. In this scenario, the sensitive data is not known to anybody but the user, i.e., no publicly accessible plaintexts or statistics about the data in the form of auxiliary information. The case of "Aux: Known Keywords", which decreases the entropy by roughly 50% in our experiments, reflects the assumption that all attack implementations make: that the attacker knows which plaintext keywords are in the database, but does not know the mapping from encrypted queries to plaintext keywords [6, 12, 19, 33, 51, 53]. The case of "Aux: Volume Vector", which is also a common assumption in many attacks [6, 12, 51], further reduces the entropy by roughly 99% in small-vocabularies and around 80-99% in large-vocabularies.

Discussion. Recent work by Kamara *et al.* [36] rightly points to the need for a better comparative understanding of attacks. We believe that this change requires a shift away from an empirical approach for leakage comparison, and towards a theoretically-grounded metric for measuring privacy. Quantifying privacy typically assists defense mechanisms– indeed, this is the case with the next two sections of this work. In this section, however, we demonstrated that cryptanalytic efforts can also benefit from quantifying privacy. Typically attacks treat the algorithmic technique and the cryptanalytic assumption as an inseparable pair. Here, we showed that we can isolate the contribution of the auxiliary information towards a successful attack.

5 COMPARING ENCRYPTED RANGE SCHEMES VIA LEAKAGE INVERSION

In this section, we use leakage inversion to quantify the privacy provided by known response-revealing encrypted range schemes. Starting with entropy-based comparison, our first finding shows that in dense databases the entropy of $RS(\mathsf{DB})$ for the quadratic scheme QD is constant and identical to the entropy of the augmented binary tree ABT. This is a rather surprising finding since QD reveals the query leakage of $O(N^2)$ distinct queries while ABT reveals the query leakage of $O(N)$ distinct queries. On the other hand, the binary tree scheme BT allows $O(N)$ distinct range queries and results in a significantly larger reconstruction space. In terms of leakage inversion with respect to maximum/expected distance, we focus on the binary tree scheme BT and we derive lower and upper bounds on the above privacy measures. Specifically, for the case of expected distance, we derive bounds as closed-form expressions (see Theorem 7) while for maximum bounded distance we propose lower and upper bounds as solutions to a combinatorial optimization problem (see Theorem 6).

5.1 Entropy Comparison

For the case of encrypted range schemes, one can view each distinct allowable range query as a *constraint* on the potential databases that explain the observed leakage. To illustrate this point, suppose that we have an SE scheme for ranges that returns the entire database, no matter what the query is. Thus, *any* reconstruction is plausible as long as it respects trlen leakage. If we were to increase the number of distinct responses (as opposed to returning the entire DB) then we would decrease the number of plausible databases that explain the observed leakage. Using this rationale, one might assume that

fewer constraints imply larger reconstruction space which, in turn, implies more privacy.

The above simple observation can be applied to the state of the art range schemes. Scheme ABT, which allows $3N - \log N - 2$ distinct range queries, imposes significantly fewer constraints than QD which allows $\binom{N}{2} + N$ queries. Thus, a first attempt to compare the two (without using leakage inversion) would result in hypothesizing that ABT provides significantly more privacy than QD since it imposes fewer constraints. However, the following results show that for dense databases ABT and QD provide the same level of privacy under leakage inversion via entropy. In particular, there are only two databases that explain the observed leakage of ABT: one is the true plaintext DB and the other one is its reflection DB^R. We refer the reader to the full version for the proofs of this section.

Theorem 4. *Let DB be a dense range-based database and Λ_{ABT} be the leakage of the ABT scheme. Then, the leakage inversion with respect to entropy is $\mathit{LeakInv}_H(DB, \Lambda_{ABT}, \perp) = 1$.*

Scheme QD "constrains" the reconstruction space even further than ABT, since it allows $O(N^2)$ distinct ranges, which gives an upper bound on QD: $\mathsf{LeakInv}_{\mathsf{H}}(\text{DB}, \Lambda_{\text{ABT}}, \perp) \geq \mathsf{LeakInv}_{\mathsf{H}}(\text{DB}, \Lambda_{\text{QD}}, \perp)$. Since DB and DB^R are always part of the reconstruction space, we get a lower bound on QD: $\mathsf{LeakInv}_{\mathsf{H}}(\text{DB}, \Lambda_{\text{QD}}, \perp) \geq 1$.

Corollary 1. *Let DB be a dense range-based database and Λ_{QD} be the leakage of the QD scheme. Then, the leakage inversion with respect to entropy is $\mathit{LeakInv}_H(DB, \Lambda_{QD}, \perp) = 1$.*

On the other hand, scheme BT which allows $2N - 1$ distinct range queries, i.e., only $N - \log N$ less than ABT, has entropy that is linear to the number of values associated with a record.

Theorem 5. *Let DB be a range-based database and Λ_{BT} be the leakage of the BT scheme. Let v be the number of values with at least one associated record in DB. Then, the leakage inversion with respect to entropy is: $v - 1 \leq \mathit{LeakInv}_H(DB, \Lambda_{BT}, \perp) \leq \min\{v \log N, N - 1\}$.*

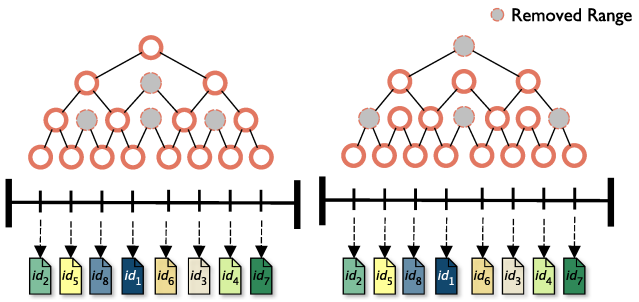

Figure 2: Illustrating why less leakage does not always lead to more privacy. We removed four ranges from each scheme to reduce leakage and used the same underlying DB. The scheme on the left has reconstruction space with entropy 7, i.e., reflections of each subtree. The scheme on the right has reconstruction space with entropy 1, i.e., DB and DB^R.

Discussion. Given the above analysis, a natural question is: why less leakage results in the *same* privacy in the case of ABT vs. QD but less leakage results in *more* privacy in the case of ABT vs. BT? Our findings show that we should not focus on the "quantity" of leakage that is *removed* but rather on the "quality" of leakage that *remains*, i.e., how damaging is the remaining leakage. An illustration of this point is presented in Figure 2, where we remove exactly four range queries (denoted in gray) from two hypothetical range schemes. Even though the same number of ranges are removed, the left-hand side has entropy 7 after the removal. This is because the ordering of the leaves of a sub-tree (normal ordering or a reflection of its leaves) introduces one bit of entropy to the reconstruction space. On the other hand, the right-hand side scheme has entropy 1 (i.e., 2 reconstructions). This is because the only permutation that gives the same set of responses is the reflection of the entire tree.

5.2 Leakage Inversion on Binary Tree Scheme

In the following we present results on the leakage inversion with respect to bounded maximum distance and expected distance for scheme BT. The next theorem provides a lower and an upper bound on $\mathsf{LeakInv}_{\mathsf{MX}}(\text{DB}, \Lambda_{\text{BT}}, \perp)$. Each bound in Theorem 6 is a solution to a variation of the so-called *assignment problem* where a set of tasks are assigned to agents so that the toal cost is minimized. In our case, each task represents a set of records where members share the same (unknown) value and each agent represents a plaintext value v. Assigning a set to a value v is interpreted as: all the records with an identifier from the set have the same plaintext value v. In terms of time complexity, both of the following optimization problems can be over-approximated with linear programming (see the full version for a precise formulation).

Theorem 6. *Let DB be a range-based database and Λ_{BT} be the leakage of scheme BT. The leakage inversion with respect to maximum distance, i.e., $\mathit{LeakInv}_{MX}(DB, \Lambda_{BT}, \perp)$, is lower bounded by:*

$$d(DB, x^*), \text{ where } x^* = \arg\max_{x \in supp(\Gamma)} \sum_{\substack{(id,v') \in x \\ (id,v) \in DB}} \min\{|v' - v|, |v' - v^R|\}$$

and upper bounded by:

$$\frac{1}{2n} \max_{x \in supp(\Gamma)} \sum_{\substack{(id,v') \in x \\ (id,v) \in DB}} |v' - v| + |v' - v^R|.$$

The next theorem provides a lower and an upper bound on $\mathsf{LeakInv}_{\mathsf{XP}}$. We resort to bounding (as opposed to exact calculation) the quantity of interest due to the difficulty of deriving an analytical expression for the expectation of a multivariate distribution of random variable $|\sum_i X_i|$, where X_i are random variables. To calculate the bounds for any given DB one has to plug in the terms v_i and $\text{vol}(v_i)$ from DB and calculate the closed-form expressions.

Theorem 7. *Let DB be a range-based database and Λ_{BT} be the leakage of the BT scheme. Let $v_1, \ldots, v_v$ be the set of unique values from DB in ascending order. Suppose that N is even, then the leakage inversion with respect to expected distance $\mathit{LeakInv}_{XP}(DB, \Lambda_{BT}, \perp)$ is lower bounded by:*

$$\frac{1}{2n}\left(2 \sum_{j=1}^{N} \zeta_j \cdot vol(v_j) - \sum_{i=1}^{v} \frac{vol(v_i)}{N} \cdot \left(2\mu_i^- \cdot (N - 2\mu_i^- + 1)\right)\right)$$

and upper bounded by $\frac{1}{n} \sum_{i=1}^{v} \zeta_i \cdot vol(v_i)$, where $\mu_i^- = \min\{v_i, v_i^R\}$, $\mu_i^+ = \max\{v_i, v_i^R\}$, $\zeta_i = \left(\frac{v_i(v_i-1)}{2N} + \frac{v_i^R(v_i^R-1)}{2N}\right)$.

5.3 Evaluation

In this evaluation, we use real-world datasets MIMIC-T4 [49], UK-Salaries [57], and HCUP-AGE [1] and ask:

- What is the error between a randomly chosen reconstruction from $RS(\mathsf{DB})$ and the true plaintext DB?
- What is the maximum possible error between the true plaintext DB and a member from $RS(\mathsf{DB})$?
- What is the gap between the maximum error and the error from a randomly chosen reconstruction?
- How does an empirical computation of $\mathsf{LeakInv}_{\mathsf{MX}}$ (resp. $\mathsf{LeakInv}_{\mathsf{XP}}$) compare to the lower and upper bound?

Table 2 presents the lower/upper bounds from Theorems 5, 6, and 7 on real data from previous leakage-abuse attacks. The gray rows show the empirical computation of the maximum error between the true plaintext DB and a member from $RS(\mathsf{DB})$ as well as the expected error between a randomly chosen member of $RS(\mathsf{DB})$ and DB. To calculate the empirical values we sampled 10^5 members of $RS(\mathsf{DB})$ and computed the maximum error (for the first case) and the mean and standard deviation of the error (for the second case).

The results on the (white) rows in Table 2 provide the theoretical guarantees for the reconstruction error which answers the first two questions posed at the beginning of the evaluation. The experiments show that the lower and upper bounds of $\mathsf{LeakInv}_{\mathsf{MX}}$ are close. In both cases, any minor discrepancy between the empirical value and the theoretical bound comes from sub-sampling. It is worth differentiating here between a random reconstruction (which is a random assignment of records to values) and a *randomly sampled reconstruction from* $RS(DB)$ (which is a reconstruction where records with the same unknown value are assigned as a group to a value while also conforming to the responses revealed by Λ_{BT}).

To compare $\mathsf{LeakInv}_{\mathsf{XP}}$ and $\mathsf{LeakInv}_{\mathsf{MX}}$, start by noticing that the number of databases in $RS(\mathsf{DB})$ with respect to Λ_{BT} is larger than $2^{\nu-1}$ (see Theorem 5). For example, in MIMIC-T4 (which is almost dense) there are 2^{74} databases in $RS(\mathsf{DB})$. Given such a large size of $RS(\mathsf{DB})$, the fact that the expected error $\mathsf{LeakInv}_{\mathsf{XP}}$ is so close to the maximum error $\mathsf{LeakInv}_{\mathsf{MX}}$ means that the number of databases with close-to-maximum error vastly outnumber the databases with small error. Note that this does not mean that the absolute number of low-error databases is small.

In general, these findings shed light on the *structure of the reconstruction space*. For the tested data, the expected error is as large as the maximum error for the case of BT. This is the first formal analysis of the privacy offered by an SE scheme that does not rely on an attack performance, e.g., [46], but rather is based on a theoretical understanding of the reconstruction space.

		Range-Based Databases		
		MIMIC-T4	UK-Salaries	HCUP-AGE
$\mathsf{LeakInv}_{\mathsf{MX}}$	Upper Bound	34	198	35
	Lower Bound	27	163	34
	Empirical	31.5	190	32
$\mathsf{LeakInv}_{\mathsf{XP}}$	Upper Bound	27.5	148	29
	Lower Bound	19.7	122	21
	Empirical $\parallel \sigma$	23.4 $\parallel$ 3	117 $\parallel$ 25	28 $\parallel$ 1
$\mathsf{LeakInv}_{\mathsf{H}}$	Upper Bound	73	78	92
	Lower Bound	58	8	90
Size of Universe of Values N		74	398	93
Number of Records n		8,058	536	6,772,133

Table 2: Application of leakage inversion via bounded maximum distance (denoted as $\mathsf{LeakInv}_{\mathsf{MX}}$), leakage inversion via expected distance (denoted as $\mathsf{LeakInv}_{\mathsf{XP}}$), and leakage inversion via entropy (denoted as $\mathsf{LeakInv}_{\mathsf{H}}$) on range-based datasets. The bounds are calculated from Theorems 5, 6, and 7. The gray rows show the empirical measurement (and its standard deviation σ) based on 10^5 samples from $RS(\mathbf{DB})$.

6 COMPARING PADDING MITIGATIONS VIA LEAKAGE INVERSION

In this section, we apply leakage inversion to three padding mitigations on response-hiding SE for keyword search. We emphasize that we do not propose new mitigations but rather study the privacy of *already proposed* padding approaches. Our contribution is the application of a theoretically grounded approach, i.e., leakage inversion, for quantifying the privacy of each approach. Using leakage inversion we can study questions such as:

"How much more privacy do we get if we accept a larger communication overhead from padding?"

Leakage inversion allows us to not only compare different padding methods but also choose the parameterization that strikes a good *balance between efficiency and privacy* for given padding, which, in turn, results in efficient SE deployments with quantifiable privacy.

En route to cryptographic definitions for padding, we introduce a new modular leakage profile called "fos" that captures generalizations of previous proposals, which may be of independent interest.

Padding the Signature of Response-Hiding Schemes. On a high level, most padding schemes associate additional fake records with each value. This way, when an encrypted query is received, the scheme returns a superset of the desired response. Given that response-hiding schemes hide the identity of the retrieved records, the leakage boils down to a mapping from tokens to padded volumes. More formally, we define the *signature* of a (padded or non-padded) database $\sigma(\mathsf{DB})$ as the sequence of pairs $(v_i, \mathrm{vol}(v_i))$ for $i \in [N]$. Padding mitigations augment the original database DB to get DB′ for which the adversary observes the *padded* signature $\sigma(\mathsf{DB}')$.

Reconstruction Space Over Signatures. From an adversarial perspective, an attacker that observes the leakage of a padded response-hiding SE may need to first recover the signature of the non-padded database before attempting to reconstruct the original plaintext. In some cases, the non-padded signature is the connecting link between the padded leakage observations and the auxiliary information (common in all leakage attacks [12, 33, 51] on keyword-databases). For example, the most recent attack by Oya and Kerschbaum [51] considers the case where the auxiliary information is the mapping from *plaintext keywords* to non-padded volumes (as opposed to the mapping from tokens to padded volumes which is revealed by leakage Λ). This auxiliary information is called "volume vector of keywords" and is used in part by other attacks [12]. One way to illustrate the connection between signature recovery and privacy is to consider an attacker that guesses correctly the non-padded volume of a keyword, e.g., token t appears in 22032

real documents. In this case, the attacker can use the auxiliary information to unambiguously identify the plaintext of t which must be the only entry in the volume vector with volume 22032.

From a theoretical perspective, as the community expands its SE-hardening efforts by applying multiple mitigations, it is essential to understand the *individual effect* of each mitigation to the overall privacy. Towards this goal, we want to study the effect of padding alone on privacy which translates to quantifying the uncertainty between the padded signature and the non-padded signature.

In this section, we will re-define the reconstruction space so as to capture the privacy increase offered by padding in the response-hiding setting. Specifically, the reconstruction space will comprise all the *non-padded signatures* (as opposed to plaintext databases) that could have resulted in the observed padded signature.

6.1 Padding Approaches

On a high level, we consider padding approaches that group together a number of values $v \in \text{DB}$ to form set P_i. Instead of revealing different information for each individual value from P_i, the considered padding approaches reveal the same information for *all* members of P_i. For example, a padding approach may reveal the *sum of all the volumes* implemented by returning all records associated with values from P_i as a single logical unit. We note that the approach of grouping values appeared in prior works (e.g., see Section 4.2 in [12], Section 6 in [31], Section 6 in [38]). This approach can be generalized, e.g., instead of revealing the sum of volumes in P_i , one may choose to reveal another function of the volumes in P_i. In this work, we develop a common definitional framework to express proposed padding variations and their generalizations.

Notation. The analyzed padding approaches partition values v s.t. $(id, v) \in \text{DB}$ into sets of the same size m. After the partition of values, all the pairs $(id, v') \in \text{DB}$ are also considered part of the partition set P_i where value v' belongs, i.e., $v' \in P_i$. As a result, even though each partition set contains the same number of values m, the total volume of each partition may significantly differ depending on the number of pairs (id, v') associated with v'. For simplicity, we assume that the number ν of values that appear in DB is a multiple of m. We write $\mathcal{P} = (P_1, P_2, \ldots, P_{\nu/m})$ for a partition of the ν values into ν/m non-overlapping sets. That is, the union of the sets, $\cup_{i\in[\nu/m]} P_i$, is equal to the set of values in DB. We denote with $\mathcal{P}(v)$ the partition set of $\mathcal{P}$ that contains v. To simplify notation, we write $\max(P_i, j)$ for the *volume* of the value with the j-th largest volume in a set P_i. We also use $\max(\text{DB}, j)$ for the *volume* of the value with the j-th largest volume in the entire set of values in DB. For the largest volume we omit the second input and simply write $\max(P_i) = \max(P_i, 1)$ and $\max(\text{DB}) = \max(\text{DB}, 1)$. We write $\text{sum}(P_i) = \sum_{v_j \in P_i} \text{vol}(v_j)$ for the total volume of all values in P_i.

Rounding Padding. In this approach the volume of each value v is "rounded" to the closest power of a constant c. For $c = 2$, the padded volume becomes $2^{\lceil \log \text{vol}(v) \rceil}$. The rationale behind this previously proposed padding [22] is that the adversary observes only a set of predetermined padded volumes, i.e., the powers of the chosen c, and as a result, can only observe a coarse-grain volume leakage. Rounding is inline with the partitioning framework since each Rounding partition contains a single value.

Alignment Padding. In this approach, the volume of each value is increased to the largest volume of its partition set. Alignment is a tunable generalization of a proposed padding approach [52] that pads *all* values to the maximum volume of the database which corresponds to the case of $m = N$ in Alignment.

Bucketing Padding. This previously proposed padding approach [38] groups values together and treats them as a single logical unit, which we will call a "bucket" (or bin). When one of the values is requested then the padded SE scheme returns all the records associated with the values of the bucket.

On Partitioning Strategies. A common characteristic of the above paddings is that they all partition the values of the database. If the partitioning strategy is decided based on the volume of each value then the SE scheme leaks *additional information* due to the partitioning strategy. To illustrate this point, consider a database where $v_1, \ldots, v_4$ have volumes $50, 47, 49, 48$. Next, we fix $m = 2$ and choose Alignment padding which only reveals the maximum volume of each partition set. Suppose that we partition by grouping values with large volumes first, then we get P_1 which contains volumes $\{50, 49\}$ and P_2 with volumes $\{48, 47\}$. If the partitioning strategy is known to the attacker, then (s)he infers that the smallest volume of P_1 is at least 48. This is because the largest volume of P_2 is 48, thus P_1's volumes must be at least as large. On the contrary, if a random partitioning resulted in the same P_1 and P_2 then the inference becomes: One of the volumes of P_1 is 50 and the other volume can be anything from 1 to 50. Thus, in this work, we only consider the random partitioning strategy which leaks less.

More formally, we define the following parameterizable leakage profile that applies to all padding algorithms.

Definition 9. *(**Function-Over-Subset Pattern**) Let $\mathcal{P}$ be a partition of values of a DB into disjoint sets with m values each. Let f be a (possibly randomized) function with multivariate input and a single-number output. We define as* function-over-subset pattern*, or fos pattern, the function family that takes as input a DB, the partition $\mathcal{P}$, the function f, a value v, and outputs:*

$$fos(DB, \mathcal{P}, f, v) = f\left(\left\{vol(v') : v' \in \mathcal{P}(v)\right\}\right).$$

For the case of Alignment the function f of fos is $\max(\cdot)$. For the case of Bucketing the function f of fos is $\text{sum}(\cdot)$. For the case of Rounding, for which each partition set contains a single value, the function f of fos for the chosen rounding constant c is $c^{\lceil \log_c(\cdot) \rceil}$.

In this section we re-define reconstruction space so that it contains all possible non-padded signatures that explain the revealed padded signature under a given padding algorithm. More formally, let Alg be a padding algorithm, let m be the number of values per partition, and let DB′ be the padded database from running padding $\text{Alg}(m, \text{DB})$. We define $\text{RS}_{\text{Alg}}(\text{DB})$ to be the *reconstruction space* with respect to a given signature $\sigma^* \leftarrow \text{Alg}(m, \text{DB})$:

$$\text{RS}_{\text{Alg}}(\text{DB}) = \Big\{\sigma(\text{DB}^{\text{npad}}) : (\text{DB}^{\text{pad}}, \mathcal{P}) \leftarrow \text{Alg}(m, \text{DB}^{\text{npad}}), \sigma(\text{DB}^{\text{pad}}) = \sigma^*, \text{DB}^{\text{npad}} \in \mathbb{DB}_n\Big\}.$$

6.2 Entropy Comparison

Following in the footsteps of Section 3.2, we define the leakage inversion for the reconstruction space $\text{RS}_{\text{Alg}}(\text{DB})$. With the term Λ_{Alg} we denote the leakage profile when the database is padded with algorithm Alg and a randomly chosen partition $\mathcal{P}$. With the term $\Lambda^{\mathcal{P}}_{\text{Alg}}$ we denote the leakage under a fixed partition $\mathcal{P}$.

Definition 10. *Let DB be a database, let SE be a response-hiding searchable encryption scheme with leakage profile Λ that uses padding algorithm Alg. Let Γ be a random variable with probability distribution over $RS_{Alg}(DB)$. We define as* ***leakage inversion via entropy*** *with respect to padding the function*

$$LeakInv_H(DB, \Lambda_{Alg}) \triangleq H(\Gamma).$$

Notice that in this definition we do not consider the auxiliary information, therefore, all members of $\mathsf{RS}_{\mathsf{Alg}}(\mathsf{DB})$ are equiprobable.

Entropy of Padding via Rounding. The fos pattern reveals the closest power of c for each volume.

Theorem 8. *Let $\sigma(DB)$ be the signature of the database DB. Let Rounding be the padding approach applied to DB. Let c be the constant used by Rounding to pad volume $vol(v)$ to $c^{\lceil \log_c(vol(v)) \rceil}$. The leakage inversion with respect to entropy is:*

$$LeakInv_H(DB, \Lambda_{Round}) = \sum_{i=1}^{v} (\lceil \log_c(vol(v)) \rceil - 1) \log c.$$

Entropy of Padding via Alignment. The fos pattern reveals the maximum volume within each partition P_i, denoted as $\max(P_i)$. From fos, the attacker infers that among the m volumes of the partition, there are at least 1 and at most m volumes with value $\max(P_i)$. The remaining volumes can take any non-zero value that is less than $\max(P_i)$. As for deriving a general entropy bound that works for any partition, in the next theorem we prove that the partition $\mathcal{P}'$ that maximizes the $\mathsf{LeakInv}_{\mathsf{H}}(\mathsf{DB}, \Lambda_{\mathsf{Align}})$ is the one where the top v/m volumes are assigned to distinct partitions. Additionally, we prove that the partition $\mathcal{P}''$ that minimizes the $\mathsf{LeakInv}_{\mathsf{H}}(\mathsf{DB}, \Lambda_{\mathsf{Align}})$ is the one where the largest volumes are grouped together.

Theorem 9. *Let $\sigma(DB)$ be the signature of the database DB. Let Alignment be the padding approach applied to DB using a given partition $\mathcal{P}$. The leakage inversion with respect to entropy is:*

$$LeakInv_H(DB, \Lambda^{\mathcal{P}}_{Align}) = \sum_{i=1}^{v/m} \log\left(\sum_{j=1}^{m} \binom{m}{j} (\max(P_i) - 1)^{m-j}\right).$$

For any partition, the leakage inversion $LeakInv_H(DB, \Lambda_{Align})$ is lower bounded by

$$\sum_{i=0}^{v/m-1} \log\left(\sum_{j=1}^{m} \binom{m}{j} (\max(DB, im+1) - 1)^{m-j}\right)$$

and upper bounded by $\log\left(\prod_{i=1}^{v/m} \sum_{j=1}^{m} \binom{m}{j} (\max(DB, i) - 1)^{m-j}\right)$, and these bounds are tight.

Entropy of Padding via Bucketing. The fos pattern reveals the sum of the volumes within each partition P_i, denoted as $\mathrm{sum}(P_i)$. Given this observation, the attacker infers that the m volumes of the partition must add up to $\mathrm{sum}(P_i)$.

Theorem 10. *Let $\sigma(DB)$ be the signature of the database DB. Let Bucketing be the padding approach applied to DB using a given partition $\mathcal{P}$. The leakage inversion with respect to entropy is:*

$$LeakInv_H(DB, \Lambda^{\mathcal{P}}_{Bucket}) = \sum_{i=1}^{v/m} \log\binom{\mathrm{sum}(P_i) - 1}{m-1}.$$

To derive a lower bound, the next theorem proves that the minimum entropy for padding via bucketing is given by the partition that groups the largest volumes together, e.g., the largest m have total volume $\sum_{j=1}^{m} \max(\mathsf{DB}, j)$, the next batch of m largest volumes have total volume $\sum_{j=1}^{m} \max(\mathsf{DB}, m+j)$ etc.

Theorem 11. *Let $\sigma(DB)$ be the signature of the database DB. Let Bucketing be the padding approach applied to DB. The leakage inversion with respect to entropy under any partition is lower bounded by the following expression (and the lower bound is tight):*

$$\sum_{i=0}^{v/m-1} \log\binom{\sum_{j=1}^{m} \max(DB, im+j) - 1}{m-1} \leq LeakInv_H(DB, \Lambda_{Bucket}).$$

For the case where each partition has more than two values, i.e., $m > 2$, we show that if there exists a partition where each set has equal total volume, then this partition maximizes the entropy.

Theorem 12. *Let $\sigma(DB)$ be the signature of the database DB and $m > 2$. If there exists a partition $\mathcal{P}^*$ such that $\mathrm{sum}(P_i^*) = \mathrm{sum}(P_j^*)$, for all sets P_i^*, P_j^* in $\mathcal{P}^*$, then the leakage inversion under Bucketing is at most*

$$LeakInv_H(DB, \Lambda_{Bucket}) \leq (v/m) \cdot \log\binom{\mathrm{sum}(DB)/(v/m) - 1}{m-1}.$$

Unfortunately, as we show next, deciding whether such a partition exists is an NP-hard problem. We leave it as an open problem whether one can efficiently find partitions that are approximately balanced (under some definition of "approximately").

Theorem 13. *It is NP-hard to decide if there exists a partition $\mathcal{P}^*$ for the values of $\sigma(DB)$, such that $\mathrm{sum}(P_i^*) = \mathrm{sum}(P_j^*)$, for all sets P_i^*, P_j^* in $\mathcal{P}^*$, i.e. a partition with maximum possible reconstruction space size for Bucketing.*

The next theorem provides a necessary condition for which the entropy of Alignment is larger than the entropy of Bucketing.

Theorem 14. *For every partition $\mathcal{P}$ that satisfies $\mathrm{sum}(P_i) \leq \frac{m-1}{e} \max(P_i)$ for all P_i, the following relation holds between the leakage inversion by Alignment and the leakage inversion by Bucketing:*

$$LeakInv_H(DB, \Lambda^{\mathcal{P}}_{Align}) \geq LeakInv_H(DB, \Lambda^{\mathcal{P}}_{Bucket}).$$

6.3 Evaluation

In this evaluation, we quantify the privacy provided by different padding parameterizations. We test our analysis on the Enron and Apache datasets. For padding via Rounding, we evaluate different c values which, as discussed, are used to pad $\mathrm{vol}(v)$ to $c^{\lceil \log_c(\mathrm{vol}(v)) \rceil}$. For padding via Bucketing and Alignment, we evaluate how the entropy changes for a different number of *values per partition*, i.e., m ranges from 5 to 1000. The communication overhead is measured as the average (across all values of DB) ratio of padded volume over non-padded volume.

Figure 3 presents our experiments. For the plots on Bucketing and Alignment, each solid line connects boxplots of entropy (resp. communication overhead) over 20 random partitions using the chosen m value. The derived measurements were concentrated and as a result, the quartiles are not visible. The shaded blue regions denote the upper and lower bounds from the theoretical range of

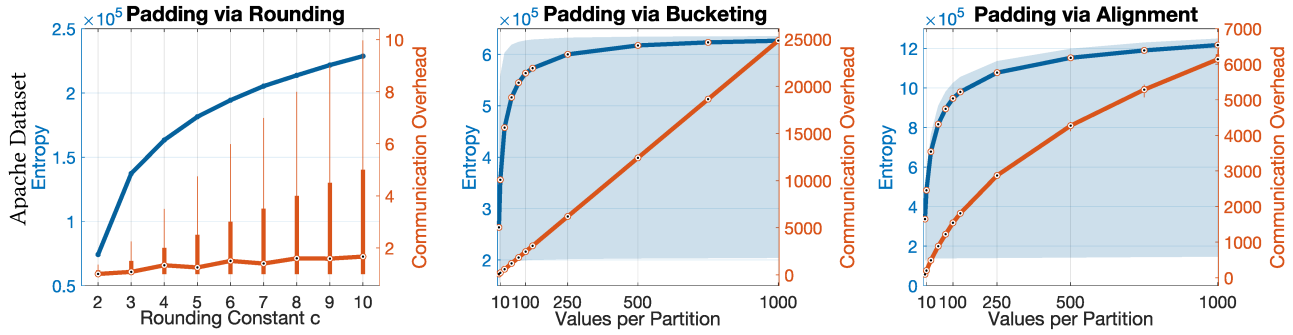

Figure 3: Analysis of the efficiency vs. privacy trade-off across padding approaches and their parameterizations. Each shaded region corresponds to the theoretical lower & upper bounds of the entropy from Theorems 9, 11, and 12. The solid lines correspond to the empirical entropy (resp. communication overhead) across several runs.

entropy from our analysis in Theorems 9, 11, and 12. For the plots on Rounding, since every partition contains exactly one value, there is no need for multiple runs. The entropy is calculated using Theorem 8. On the contrary, the communication overhead varies a lot which is why we present a more detailed view. Specifically, for each c, we show the distribution (across all values) of the ratio between padded over non-padded volumes.

A first finding is that the entropy is closer to the upper bound than the lower bound. This is explained by the fact that the partition that gives the lower bound has a structure that is not easy to find in practice, i.e., it groups the largest volumes together. Another finding is that in Apache, Alignment has 1.3× to 1.9× larger entropy than Bucketing while Bucketing has 1.4×-3.5× larger entropy than Rounding. This illustrates the power of leakage inversion which can be used to compare the privacy offered by different defenses.

The most practical finding is that there exists a *threshold* number of "values per partition" after which (1) there is little privacy gain, and (2) there is a significant increase in the communication overhead. The diminishing returns on privacy holds for both Alignment and Bucketing. For example, in the Apache dataset, if we increase the values per partition of Bucketing from $m = 100$ to $m = 1,000$, we observe an 11% increase in entropy and a 900% increase in communication overhead.

7 FUTURE DIRECTIONS

In the following, we provide a (non-exhaustive) list of open problems inspired by the newly introduced concept of leakage inversion.

Leakage Inversion on Richer Auxiliary Information. In Section 4, we treat only cases where *exact* auxiliary information is provided, but we hope that future efforts in the area will extend our analysis to include *approximate* information, such as partial volume knowledge and approximate keyword co-occurrence data.

Another interesting direction is to calculate the entropy for the case where the auxiliary information is defined as the *plaintext co-occurrence matrix*. This is a very common cryptanalytic assumption and it benefits the community to understand exactly how damaging this information can be in the hands of an adversary.

Another practical direction is to develop a similar analysis to the one in this work, but with the additional assumption that the attacker has knowledge of some auxiliary information about the *data distribution*. The auxiliary information can be as simple as a parameter about the underlying data distribution. This piece of knowledge would allow the adversary to assign a probability to each member of the reconstruction space and re-calculate the entropy.

Leakage Inversion on Padding Variations. In this work, we developed a formal understanding of padding with respect to the functions "max" and "sum" over the set of returned volumes. One can extend our analysis to other functions and show how they compare to padding via alignment and padding via bucketing.

The padding approaches that we analyze in this paper do not add any probabilistic noise to the final overall volume of the padding. Recent proposals consider differentially private leakage with the addition of Laplacian noise [52]. An interesting open problem is to derive a formal analysis of this noisy padding approach, and show sufficient conditions for which it outperforms the competition. It is unclear whether the addition of noise, as described in [52] provides significant gains in terms of entropy increase.

Conclusion. In this work, we took the first step toward quantifying privacy metrics for searchable encryption using the concept of leakage inversion. We presented a diverse set of scenarios where leakage inversion can provide new insights. Leakage inversion has the power to become a valuable and rigorous tool for future constructions and attacks in searchable encryption. New designs can use leakage inversion to compare their entropy to the entropy of well-established schemes. New cryptanalytic efforts can establish the power of their algorithmic approach by providing an entropic analysis of their underlying assumptions.

ACKNOWLEDGEMENTS

Kornaropoulos is supported in part by the NSF award CNS-2154732, a Commonwealth Cyber Initiative (CCI) award, and the Meta Security Research Award. Papamanthou is supported in part by awards from Protocol Labs, VMware, J.P.Morgan, the ACE-PAVE grant from the Algorand Foundation as well as NSF awards CNS-1652259 and CNS-2154732. Psomas is supported in part by an NSF CAREER award CCF-2144208, a Google Research Scholar Award, and the MEGA-ACE grant from the Algorand Foundation.

REFERENCES

[1] Agency for Healthcare Research & Quality, "Healthcare Cost and Utilization Project (HCUP) Nationwide Inpatient Sample (NIS)," www.hcup-us.ahrq.gov/nisoverview.jsp, 2009.

[2] M. Alvim, K. Chatzikokolakis, A. McIver, C. Morgan, C. Palamidessi, and G. Smith, *The Science of Quantitative Information Flow*, ser. Information Security and Cryptography. Springer International Publishing, 2020.

[3] M. Ando and M. George, "On the Cost of Suppressing Volume for Encrypted Multi-maps," *Proc. Priv. Enhancing Technol.*, vol. 2022, no. 4, pp. 44–65, 2022.

[4] Apache.org, "Java-user@lucene.apache.org, past," lists.apache.org/list.html?java-user@lucene.apache.org, 2005-2011.

[5] G. Asharov, M. Naor, G. Segev, and I. Shahaf, "Searchable Symmetric Encryption: Optimal Locality in Linear Space via Two-Dimensional Balanced Allocations," in *Proc. of the 48th ACM STOC*, 2016, pp. 1101–1114.

[6] L. Blackstone, S. Kamara, and T. Moataz, "Revisiting Leakage Abuse Attacks," in *Proc. of the 27th NDSS*, 2020.

[7] A. Boldyreva and T. Tang, "Privacy-Preserving Approximate k-Nearest-Neighbors Search that Hides Access, Query and Volume Patterns," *Proc. Priv. Enhancing Technol.*, vol. 2021, no. 4, pp. 549–574, 2021.

[8] A. Bossuat, R. Bost, P. Fouque, B. Minaud, and M. Reichle, "SSE and SSD: Page-Efficient Searchable Symmetric Encryption," in *Proc. of the 41st Annual International Cryptology Conference, CRYPTO 2021*, ser. Lecture Notes in Computer Science, vol. 12827. Springer, 2021, pp. 157–184.

[9] R. Bost, "$\sum o\varphi o\varsigma$: Forward Secure Searchable Encryption," in *Proc. of the 23rd ACM CCS*, 2016, pp. 1143–1154.

[10] R. Bost and P. Fouque, "Security-Efficiency Tradeoffs in Searchable Encryption," *Proc. Priv. Enhancing Technol.*, vol. 2019, no. 4, pp. 132–151, 2019.

[11] R. Bost, B. Minaud, and O. Ohrimenko, "Forward and Backward Private Searchable Encryption from Constrained Cryptographic Primitives," in *Proc. of the 24th ACM CCS*, 2017, pp. 1465–1482.

[12] D. Cash, P. Grubbs, J. Perry, and T. Ristenpart, "Leakage-Abuse Attacks Against Searchable Encryption," in *Proc. of the 22nd ACM CCS*, 2015, pp. 668–679.

[13] D. Cash, J. Jaeger, S. Jarecki, C. S. Jutla, H. Krawczyk, M. Rosu, and M. Steiner, "Dynamic Searchable Encryption in Very-Large Databases: Data Structures and Implementation," in *Proc. of the 21st NDSS*, 2014.

[14] D. Cash, S. Jarecki, C. S. Jutla, H. Krawczyk, M. Rosu, and M. Steiner, "Highly-Scalable Searchable Symmetric Encryption with Support for Boolean Queries," in *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference*, ser. Lecture Notes in Computer Science, vol. 8042. Springer, 2013, pp. 353–373.

[15] D. Cash and S. Tessaro, "The Locality of Searchable Symmetric Encryption," in *Proc. of IACR - EUROCRYPT*, 2014, pp. 351–368.

[16] J. G. Chamani, D. Papadopoulos, M. Karbasforushan, and I. Demertzis, "Dynamic Searchable Encryption with Optimal Search in the Presence of Deletions," in *Proc. of the 31st USENIX Security Symposium*, 2022, pp. 2425–2442.

[17] J. G. Chamani, D. Papadopoulos, C. Papamanthou, and R. Jalili, "New Constructions for Forward and Backward Private Symmetric Searchable Encryption," in *Proc. of 25th ACM CCS*, 2018, pp. 1038–1055.

[18] R. Curtmola, J. A. Garay, S. Kamara, and R. Ostrovsky, "Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions," in *Proc. of the 13th ACM CCS*, 2006, pp. 79–88.

[19] M. Damie, F. Hahn, and A. Peter, "A Highly Accurate Query-Recovery Attack against Searchable Encryption using Non-Indexed Documents," in *30th USENIX Security Symposium*, 2021, pp. 143–160.

[20] I. Demertzis, J. G. Chamani, D. Papadopoulos, and C. Papamanthou, "Dynamic Searchable Encryption with Small Client Storage," in *Proc. of the 27th NDSS*, 2020.

[21] I. Demertzis, D. Papadopoulos, and C. Papamanthou, "Searchable encryption with optimal locality: Achieving sublogarithmic read efficiency," in *Proc. of the 38th CRYPTO*, 2018, pp. 371–406.

[22] I. Demertzis, D. Papadopoulos, C. Papamanthou, and S. Shintre, "SEAL: Attack Mitigation for Encrypted Databases via Adjustable Leakage," in *29th USENIX Security Symposium*, 2020, pp. 2433–2450.

[23] I. Demertzis, S. Papadopoulos, O. Papapetrou, A. Deligiannakis, and M. Garofalakis, "Practical Private Range Search Revisited," in *Proc. of ACM SIGMOD*, 2016, pp. 185–198.

[24] I. Demertzis, S. Papadopoulos, O. Papapetrou, A. Deligiannakis, M. N. Garofalakis, and C. Papamanthou, "Practical Private Range Search in Depth," *ACM Trans. Database Syst.*, vol. 43, no. 1, pp. 2:1–2:52, 2018.

[25] I. Demertzis and C. Papamanthou, "Fast Searchable Encryption With Tunable Locality," in *Proc. of ACM SIGMOD*, 2017, pp. 1053–1067.

[26] M. Etemad, A. Küpçü, C. Papamanthou, and D. Evans, "Efficient Dynamic Searchable Encryption with Forward Privacy," *Proc. Priv. Enhancing Technol.*, vol. 2018, no. 1, pp. 5–20, 2018.

[27] S. Faber, S. Jarecki, H. Krawczyk, Q. Nguyen, M. Rosu, and M. Steiner, "Rich Queries on Encrypted Data: Beyond Exact Matches," in *Proc. of the 20th ESORICS*, 2015, pp. 123–145.

[28] F. Falzon, E. A. Markatou, Akshima, D. Cash, A. Rivkin, J. Stern, and R. Tamassia, "Full Database Reconstruction in Two Dimensions," in *Proc of the 27th ACM CCS*, 2020.

[29] M. George, S. Kamara, and T. Moataz, "Structured Encryption and Dynamic Leakage Suppression," in *Proc. of IACR - EUROCRYPT*, 2021, pp. 370–396.

[30] P. Grubbs, M. Lacharité, B. Minaud, and K. G. Paterson, "Learning to Reconstruct: Statistical Learning Theory and Encrypted Database Attacks," in *Proc. of the 40th IEEE S&P*, 2019, pp. 496–512.

[31] ——, "Pump up the Volume: Practical Database Reconstruction from Volume Leakage on Range Queries," in *Proc. of the 25th ACM CCS*, 2018, pp. 315–331.

[32] Z. Gui, O. Johnson, and B. Warinschi, "Encrypted Databases: New Volume Attacks against Range Queries," in *Proc. of the 26th ACM CCS*, 2019, pp. 361–378.

[33] M. S. Islam, M. Kuzu, and M. Kantarcioglu, "Access Pattern Disclosure on Searchable Encryption: Ramification, Attack and Mitigation," in *Proc. of the 19th NDSS*, 2012.

[34] M. Jurado, C. Palamidessi, and G. Smith, "A Formal Information-Theoretic Leakage Analysis of Order-Revealing Encryption," in *Proc. of the 34th IEEE Computer Security Foundations Symposium – CSF*, 2021, pp. 1–16.

[35] M. Jurado and G. Smith, "Quantifying Information Leakage of Deterministic Encryption," in *Proceedings of the ACM SIGSAC Conference on Cloud Computing Security Workshop*, 2019, pp. 129–139.

[36] S. Kamara, A. Kati, T. Moataz, T. Schneider, A. Treiber, and M. Yonli, "SoK: Cryptanalysis of Encrypted Search with LEAKER - A Framework for LEakage Attack Evaluation on Real-World Data," in *Proc. of the 7th IEEE European Symposium on Security and Privacy (EuroS&P)*, 2022, pp. 90–108.

[37] S. Kamara and T. Moataz, "Boolean Searchable Symmetric Encryption with Worst-Case Sub-linear Complexity," in *Proc. of IACR - EUROCRYPT*, 2017, pp. 94–124.

[38] ——, "Computationally Volume-Hiding Structured Encryption," in *Proc. of IACR - EUROCRYPT*, 2019, pp. 183–213.

[39] S. Kamara, T. Moataz, and O. Ohrimenko, "Structured Encryption and Leakage Suppression," in *Proc. of the 38th CRYPTO*, 2018, pp. 339–370.

[40] S. Kamara and C. Papamanthou, "Parallel and Dynamic Searchable Symmetric Encryption," in *Proc. of the 17th International Conference in Financial Cryptography and Data Security – FC*, 2013, pp. 258–274.

[41] S. Kamara, C. Papamanthou, and T. Roeder, "Dynamic Searchable Symmetric Encryption," in *Proc. of the 19th ACM CCS*, 2012, pp. 965–976.

[42] G. Kellaris, G. Kollios, K. Nissim, and A. O'Neill, "Generic Attacks on Secure Outsourced Databases," in *Proc. of the 23rd ACM CCS*, 2016, pp. 1329–1340.

[43] B. Klimt and Y. Yang, "Introducing the Enron Corpus," First Conference on Email and Anti Spam (CEAS), 2004.

[44] E. M. Kornaropoulos, C. Papamanthou, and R. Tamassia, "Data Recovery on Encrypted Databases With k-Nearest Neighbor Query Leakage," in *Proc. of the 40th IEEE S&P*, 2019.

[45] ——, "The State of the Uniform: Attacks on Encrypted Databases Beyond the Uniform Query Distribution," in *Proc. of the 41th IEEE S&P*, 2020.

[46] ——, "Response-Hiding Encrypted Ranges: Revisiting Security via Parametrized Leakage-Abuse Attacks," in *Proc. of the 42nd IEEE S&P*, 2021.

[47] E. A. Markatou, F. Falzon, R. Tamassia, and W. Schor, "Reconstructing with Less: Leakage Abuse Attacks in Two Dimensions," in *Proc of the 28th ACM CCS*, 2021, pp. 2243–2261.

[48] E. A. Markatou and R. Tamassia, "Full Database Reconstruction with Access and Search Pattern Leakage," in *Proc. of the 22nd ISC*, 2019.

[49] Medical Information Mart for Intensive Care, "MIMIC-III," mimic.mit.edu, 2001-2012.

[50] B. Minaud and M. Reichle, "Dynamic Local Searchable Symmetric Encryption," *arXiv–CoRR*, vol. abs/2201.05006, 2022.

[51] S. Oya and F. Kerschbaum, "Hiding the Access Pattern is Not Enough: Exploiting Search Pattern Leakage in Searchable Encryption," in *Proc. of the 30th USENIX Security Symposium*, 2021, pp. 127–142.

[52] S. Patel, G. Persiano, K. Yeo, and M. Yung, "Mitigating Leakage in Secure Cloud-Hosted Data Structures: Volume-Hiding for Multi-Maps via Hashing," in *Proc. of the 26th ACM CCS*, 2019, pp. 79–93.

[53] D. Pouliot and C. V. Wright, "The Shadow Nemesis: Inference Attacks on Efficiently Deployable, Efficiently Searchable Encryption," in *Proc. of the 23rd ACM CCS*, 2016, pp. 1341–1352.

[54] Z. Shang, S. Oya, A. Peter, and F. Kerschbaum, "Obfuscated Access and Search Patterns in Searchable Encryption," in *Proc. of the 28th NDSS*, 2021.

[55] D. X. Song, D. A. Wagner, and A. Perrig, "Practical Techniques for Searches on Encrypted Data," in *Proc. of the 21st IEEE S&P*, 2000, pp. 44–55.

[56] E. Stefanov, C. Papamanthou, and E. Shi, "Practical Dynamic Searchable Encryption with Small Leakage," in *Proc. of the 21st NDSS*, 2014.

[57] United Kingdom Government Legal Department, "Organogram of Staff Roles & Salaries," www.data.gov.uk/dataset/34d08a53-6b96-4fb6-b043-627e2b25840d/organogram-of-staff-roles-salaries, 2018.

[58] S. Wang, R. Poddar, J. Lu, and R. A. Popa, "Practical Volume-Based Attacks on Encrypted Databases," in *Proc. of the 5th IEEE EuroS&P*, 2020.

[59] Y. Zhang, J. Katz, and C. Papamanthou, "All Your Queries Are Belong to Us: The Power of File-Injection Attacks on Searchable Encryption," in *Proc. of the 25th USENIX Security*, 2016, pp. 707–720.