Pushing the Limits of Frequency Analysis in Leakage Abuse Attacks

Nathaniel Moyer
nmoyer5@gmu.edu
George Mason University
USA

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

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

ABSTRACT

Searchable encryption (SE) is the most scalable cryptographic primitive for searching on encrypted data. Typical SE constructions often allow access-pattern leakage, revealing which encrypted records are retrieved in the server's responses. All the known generic cryptanalyses assume either that the queries are issued uniformly at random or that the attacker observes the search-pattern leakage. It remains unclear what can be reconstructed when using only the access-pattern leakage and knowledge of the query distribution.

In this work, we focus on a cryptanalytic technique called *frequency analysis* in the context of leakage-abuse attacks on schemes that support encrypted range queries. Frequency analysis matches the frequency of retrieval of an encrypted record with a plaintext value based on its probability of retrieval that follows from the knowledge of the query distribution. We generalize this underexplored cryptanalytic technique and introduce a generic attack framework called LAMa (from Leakage-Abuse via Matching) that works even on high-dimensional encrypted data. We identify a parameterization of LAMa that brings frequency analysis to its limit– that is, we prove that there is no additional frequency matching that an attacker can perform to refine the result. Given the above result, we identify query distributions that make frequency analysis challenging for the attacker and, thus, can act as a mitigation mechanism. Finally, we implement and benchmark LAMa and reconstruct, for the first time, plaintext data from encrypted range queries spanning up to four dimensions.

1 INTRODUCTION

Searchable Encryption [39] is a cryptographic primitive that enables searching encrypted data efficiently by revealing information about the pattern of querying/accessing, known as a *leakage profile*. The first SE scheme was introduced by Curtmola *et al.* [9]. Since then, the community has produced research covering topics such as dynamic schemes [7, 24, 25, 36], geometric queries [6, 11, 12, 14], locality-aware schemes [3, 8, 10, 13], leakage suppression [2, 17, 23], and quantifying the privacy of SE constructions [5, 27]. Recently, there has been a surge in leakage-abuse attacks aimed at reconstructing plaintext databases or queries using typical leakage profiles [4, 15, 18–22, 26, 28–30, 33, 34, 37, 40, 41]. In this work, we focus on schemes with access-pattern leakage, which allow the adversary to observe which encrypted records are retrieved as part of a response to an encrypted query. Previous attacks exploiting access-pattern leakage have relied either on additional leakage (e.g. the search-pattern) or on the assumption that the clients queries come from a specific known distribution (e.g. uniform). Here, we present the first rigorous treatment of frequency analysis, a cryptanalytic technique that only relies on (*i*) knowledge of the query distribution and (*ii*) access pattern. Our results contextualize earlier efforts, and analyze the full reconstructive power of frequency analysis for range schemes.

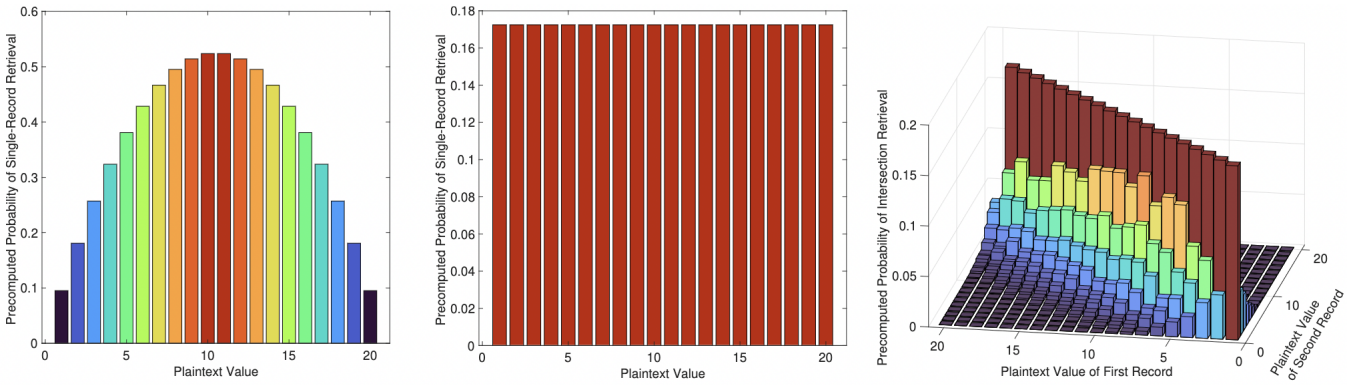
An Overview of the First Frequency Analysis. In the context of range queries, Kellaris, Kollios, Nissim, and O'Neal proposed the first leakage-abuse attack [26]. The adversarial strategy employed relied heavily on *frequency matching*, i.e., the attacker matches the observed frequency of accessing encrypted records (derived from access-pattern leakage) to the theoretical probability of accessing a plaintext value (derived from knowledge of the query distribution). Specifically, the adversary is assumed to: (*i*) know that each query issued has the same probability of appearance, the so-called *uniformity assumption*, (*ii*) observe the access-pattern leakage. Given their knowledge of the query distribution, the adversary can compute the probability of accessing a plaintext value v.

As a first step, the adversary tries to match the observed frequency of an encrypted record id to the theoretical frequency of a plaintext value. Under the uniformity assumption, each encrypted record will match to exactly two plaintext candidates: its true value and its "reflection." This phenomenon is illustrated in Figure 1-(A), where the plaintext domain ranges from 1 to 20 on the X-axis, while the Y-axis represents the retrieval probability. The plot's concave and symmetric shape, resulting from the uniformity assumption, ensures that only two plaintext values share the same retrieval probability. Consequently, the attacker identifies two candidate values for each of the n encrypted records, leading to $O(2^n)$ plausible reconstructions.

As a next step, the attack from [26] identifies the record id^* with the lowest frequency, which under a uniform query distribution indicates an extreme value (either the min or max). The attacker then commits to one of these two candidate values for id^*. Next, for each encrypted record id, the attacker checks how often id appears *together* with id^* in responses. This frequency analysis of simultaneous retrievals helps discard one of the two candidates for id, narrowing down plausible database reconstructions from $O(2^n)$ to just 2. If the guess for id^*'s extremity is correct, the reconstruction is accurate; otherwise, the true database is its reflection.

Limitations of Current Approaches. In the following, we present several limitations of the previous approaches:

Previous Frequency Analysis Attacks Are Customized For Uniform Query Distributions. All cryptanalytic techniques [18, 26, 31] that use frequency analysis in one-dimensional plaintext data assume that the queries are issued uniformly at random. We emphasize that the uniformity assumption shapes the retrieval probability of plaintexts "favorably" for the attacker, as seen in Figure 1-(A). Without this assumption, attacks can yield arbitrarily bad reconstructions. To illustrate this point, if the query distribution is such that the



(A) (B) (C)

Figure 1: `Subfigure (A)`: The probability of retrieving a record with a fixed plaintext value (denoted on X-axis) when queries come from the uniform query distribution. `Subfigure (B)`: An example of a query distribution for which every record is retrieved with the same probability regardless of its plaintext value. `Subfigure (C)`: The probability of retrieving a pair of records together as part of a response for the same query distribution as Subfigure (B).

probability of accessing a plaintext is as shown in Figure 1-(B), each encrypted record could have as many candidate reconstructions as there are plaintext values. However, this limitation does not preclude other cryptanalytic methods from achieving accurate reconstructions. For example, using the query distribution from Figure 1-(B) to calculate the probability that two distinct records are accessed together as part of the same response, the attacker gets Figure 1-(C) where each bar has a *distinct height*, indicating a unique candidate reconstruction for each observed frequency. This illustrates that a more nuanced frequency analysis can address previous challenges, underscoring the need for further exploration of frequency analysis' limitations.

Previous Approaches Need Search-Pattern Leakage. All prior cryptanalytic approaches that do not assume the query distribution is uniform must observe the *search-pattern leakage*, i.e., the attacker's ability to determine if encrypted queries are being repeated. For the case of one-dimensional ranges, the work by Kornaropoulos *et al.* [29] is the first attack that overcomes the uniformity assumption by using the search pattern. For the case of two-dimensional ranges, the works by Markatou *et al.* [33], and Falzon *et al.* [15] crucially rely on the search-pattern leakage to infer how many distinct ranges return the same response. All of the above works overcome the uniformity assumption by relying on additional leakage (which recent efforts show that it is possible to suppress [23]) and thus it is still an open question how attackers can overcome uniformity by using only access-pattern leakage.

Existing High-Dimension Attacks Make Different Assumptions or Achieve Approximate Reconstruction. Only two previous attacks, [32] and [35], operate in more than two dimensions. The attack proposed in [32] critically relies on constructions with a non-standard leakage profile. Specifically, [32] is tailored to constructions like [16] that map each plaintext query to multiple encrypted queries, thereby exposing the interrelation among tokens. It also relies on the assumption of search-pattern leakage made by the attacks mentioned above.

The attack presented in [35] succeeds at *approximate* reconstruction in dimensions beyond 2 and assumes only access-pattern leakage. By contrast, our attack makes the additional assumption that the adversary knows the query distribution of the client but targets exact reconstruction. As we explain in Section 4, our proposed attack only outputs reconstructions whose leakage is indistinguishable from that of the true database. We see [35] as a viable attack complementary to our own, but distinct in that it uses very different techniques and does not consider adversarial knowledge of the query distribution, on which the frequency analysis is focused. A promising direction for future work may involve a hybrid approach that combines the leakage amplification techniques of [35] with frequency analysis.

Our Contributions. We make the following contributions:

- *The Foundations of Frequency Analysis for Arbitrary Dimensions.* This is the first work that focuses on frequency analysis and pushes this cryptanalytic technique to its limits (in §4). We only assume access-pattern leakage and knowledge of the query distribution (which can differ from uniform). We formalize the basic ideas of frequency analysis (in §3) and generalize prior attempts, meaning any previous and future frequency analysis attack can be cast using this unifying framework, which allows direct comparison between approaches.
- *A Cryptanalytic Framework that Automates Frequency Analysis.* In §3, we present our framework Leakage-Abuse via Matching, or simply LAMa , that abstracts and streamlines the operations to execute frequency analysis. For the first time, we draw interesting connections with Satisfiability Modulo Theories (SMT) solvers, which we use to efficiently identify database reconstructions that satisfy the observed frequencies of retrieval. LAMa supports frequency analysis for any arbitrary conditional retrieval, e.g., frequency of retrieving id_a but not id_b. In §6 we implement and evaluate a LAMa prototype on real-world healthcare datasets.
- *A Parameterization of Frequency Analysis That Maximizes the Reconstructive Power.* In §4, we present a parametrization of LAMa so that no other (generic) frequency analysis could produce a

more accurate reconstruction. To support that, we prove that our parameterization allows LAMa to output only databases with the same *response distribution* as the true database, representing the optimal outcome achievable through frequency analysis.

- *Thwarting Frequency Analysis by Adjusting the Query Distribution.* In §5 we show how the defender can "flatten" the probability of retrievals so as to (partially) mitigate frequency analysis. We also show that it is impossible to adjust the query distribution so that nothing is revealed (unless queries with probability 0 are allowed).

Overall, our results are consequential for both attackers and defenders in the context of leakage-abuse attacks. For attackers, we show that frequency analysis can effectively reconstruct data when the query distribution is known, *even in high dimensions* and with only access-pattern leakage. For defenders, we show that controlling the query distribution can greatly increase the number of equally plausible reconstructions, effectively increasing adversarial uncertainty.

2 BACKGROUND AND PRELIMINARIES

Notation. For any integer y, let $[y]$ denote the set $\{1, 2, \ldots, y\}$. Let $[y]^k$ denote the k-fold Cartesian product, i.e., $\{1, 2, \ldots, y\} \times \ldots \times \{1, 2, \ldots, y\}$ for k sets. For any integer N, if $a = (a_1, \ldots, a_k)$ and $b = (b_1, \ldots, b_k)$ are points in the *k-dimensional plaintext domain* $\mathcal{V} = [N]^k$, and $a_i \leq b_i$ for all dimensions $i \in [k]$, then we say that b *dominates* a (or equivalently, a is dominated by b), denoted $a \preceq b$. To be consistent with previous works, we refer to points in $\mathcal{V} = [N]^k$ as values, even though they are k-dimensional vectors. We define the *distance* between two values $a, b \in \mathcal{V}$, denoted $dist(a, b)$, as the L^1 (or Manhattan distance): $dist(a, b) = \sum_{i=1}^{k} |a_i - b_i|$.

Structured Encryption for Range Queries. Let $\mathcal{V} = [N]^k$ be the *domain of values*, where N and k are positive integers. Let $\mathcal{I}$ be the set of identifiers of the database used to uniquely identify encrypted records. A *database* $\mathsf{DB} = \{(\mathsf{id}, v) \mid \mathsf{id} \in \mathcal{I}, v \in [N]^k\}$ is a collection of identifier-value pairs. A dimension of $\mathcal{V}$ can be seen as a database attribute, e.g., "AGE" and each identifier as an encrypted medical file of a patient. We use *record* and *identifier* interchangeably, as each encrypted record has a unique identifier $\mathsf{id} \in \mathcal{I}$. We denote the value v of id as $\mathsf{DB}(\mathsf{id})$.

A *structured encryption scheme for range queries (R-STE)* is a primitive for encrypted search. An R-STE scheme allows the client to encrypt and outsource DB to a server and perform queries. A range query q in $\mathcal{V}$ can be seen as a *hyperrectangle* in $[N]^k$. That is, a query $q = [a, b]$ is defined by two vertices of the corresponding hyperrectangle, i.e., the vertex $a \in \mathcal{V}$ that is dominated by all other vertices of the hyperrectangle and the vertex $b \in \mathcal{V}$ that dominates all other vertices of the hyperrectangle. The universe of all queries (with respect to $[N]^k$) is denoted as $\mathcal{Q}$. We say that a query $q = [a, b]$ *covers* value v if $a \preceq v \preceq b$.

Query Phase. In the query phase of an R-STE, the client issues an encrypted range query to the server. The server then responds with the identifiers whose values lie in the range specified by the query. The set of returned identifiers is called a *response*, denoted rsp, and the universe of responses $\mathcal{R}$ is the power set of identifiers $\mathcal{R} = \mathcal{P}(\mathcal{I})$. Some schemes support update operations and are called *dynamic*. Those that do not are called *static*.

Algorithms. We define a static R-STE scheme consisting of the following algorithms: Setup, which takes the security parameter λ and DB and outputs the secret key sk to the client and the encrypted database EDB to the server; Trpdr, which takes the secret key sk and the query q from the client and outputs a token (i.e., a trapdoor) for query q to the client; and Search, which takes a token t from the client and encrypted database EDB from the server and outputs a set of identifiers $\mathsf{rsp} \subseteq \mathcal{I}$ to the client.

An R-STE scheme is correct if every response contains the identifiers whose value is covered by the query. In this work, we focus on the R-STE scheme that contains *only* the identifiers needed for the scheme to be correct, i.e., no false positives. More formally:

Definition 1. *Let* $\Sigma = (\mathsf{Setup}, \mathsf{Trpdr}, \mathsf{Search})$ *be an R-STE scheme and let DB be database over* $\mathcal{I}$ *and* $\mathcal{V}$. *We say that* Σ *is* correct *if, for every* $q = (a, b)$ *in* $\mathcal{Q}$, *after the execution of* $(\mathsf{sk}, EDB) \leftarrow \Sigma.\mathsf{Setup}(\lambda, DB)$, $t \leftarrow \Sigma.\mathsf{Trpdr}(\mathsf{sk}, q)$, *and* $rsp \leftarrow \Sigma.\mathsf{Search}(t, EDB)$, *the following holds for rsp:* $rsp = \{id \mid a \preceq DB(id) \preceq b\}$.

Notice that in our analysis, every identifier is associated with exactly one value. For simplicity, we make one further assumption: that every value $v \in \mathcal{V}$ is associated with at most one identifier $\mathsf{id} \in \mathcal{I}$. We note that this is a standard simplifying assumption, and there are several ways [29] to extend our analysis to the general case.

Leakage Profile. The information revealed to the server while running R-STE algorithms is defined as a set of functions over the plaintext data called *leakage functions*. Taken together, these functions make up the *leakage profile* Λ of a scheme, and are typically categorized as either *setup leakage* $\mathcal{L}_{Setup}$ or *query leakage* $\mathcal{L}_{Query}$, where $\Lambda = (\mathcal{L}_{Setup}, \mathcal{L}_{Query})$.

Following the notation in [23], we define three leakage functions relevant to our analysis. The *total response-length pattern* trlen takes DB and outputs the total number of identifiers in all responses returned for queries $q \in \mathcal{Q}$. The *response-identity pattern* $\mathsf{rid}(q)$ (often called the *access pattern*) reveals, for each execution of Search, the identifiers contained in the response. The *query-equality pattern* qeq (often called the *search pattern*) takes an array of queries $[q_1, \ldots, q_M]$ and outputs an $M \times M$ binary matrix where $M[i, j] = 1$ if $q_i = q_j$, and $M[i, j] = 0$ otherwise. A common leakage profile, both with respect to earlier constructions [11, 14] and cryptanalytic efforts [29], for R-STE schemes is $\Lambda = \{\mathsf{trlen}, (\mathsf{qeq}, \mathsf{rid})\}$.

In this work, we focus on a less revealing leakage profile (i.e., a more challenging scenario for the attacker), $\Lambda = \{\mathsf{trlen}, \mathsf{rid}\}$, to demonstrate the effectiveness of frequency analysis even when search pattern is suppressed, as in [23]. We use a Real/Ideal security game to define adaptive security:

Definition 2. *Let* $\Sigma = (\mathsf{Setup}, \mathsf{Trpdr}, \mathsf{Search})$ *be an R-STE scheme and let DB be a database over* $\mathcal{I}$ *and* $\mathcal{V}$. Σ *is* adaptively secure *with respect to leakage profile* Λ *if for any probabalistic polynomial time (ppt) adversary Adv issuing poly(*λ*) queries, there exists a stateful ppt simulator Sim and a negligible function negl(*λ*) such that*

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

Query and Response Distributions. Following Kellaris *et al.* [26], we model client queries as i.i.d. samples from a distribution on the universe of queries $\mathcal{Q}$. We call this distribution the *query distribution*, denoted QD. Formally, let X be a random variable over $\mathcal{Q}$ that follows the distribution QD and denotes a query issued by the client. We denote the probability that $X = q$ as $\Pr[X = q] = \Pr_{QD}[q]$.

We may omit the subscript QD for brevity if it is clear from the context. Interestingly, by fixing a distribution on the universe of queries $\mathcal{Q}$, we fix a distribution on the universe of responses $\mathcal{R}$ for a given DB, that we call the *response distribution*, denoted RD. Intuitively, the probability that a response rsp will be returned is equal to the sum of probabilities of all queries that return rsp. The response distribution is thus a function of both the database and the query distribution. Let $\mathcal{Q}(\text{rsp}, \text{DB})$ denote the set of queries with response rsp in database DB. Formally, let Y be a random variable over the responses $\mathcal{R}$ that follow RD. We denote the probability that $Y = \text{rsp}$ as

$$\Pr[Y = \text{rsp}] = \Pr_{RD}[\text{rsp}] = \sum_{q \in \mathcal{Q}(\text{rsp},\text{DB})} \Pr_{QD}[q]$$

where again, we drop the subscript RD when clear from context. The last sum indicates that the probability that a response rsp will be returned by a query sampled from QD is the sum of the probabilities of all queries that return rsp.

Adversarial Goal. Let DB be a database over the identifiers $\mathcal{I}$ and domain $[N]^k$, and associated with query distribution QD, and let Σ be an instance of the R-STE scheme defined above. The adversary that we consider in this work is the server in Σ, who attempts to learn DB by observing the access-pattern leakage. As is common in the leakage cryptanalysis literature [15, 26, 29], we assume that the server knows the domain $[N]^k$ and universe of queries $\mathcal{Q}$. More importantly, we assume that the server knows the query distribution QD, much like Kellaris *et al.* [26].

3 LAMa: A CRYPTANALYSIS FRAMEWORK FOR ARBITRARY DIMENSIONS

We introduce a new framework called **L**eakage-**A**buse via **Ma**tching, or simply LAMa , for performing database reconstruction using frequency-matching analysis. LAMa consists of the components Selector, Translator, and Solver, where each one has well-defined and synergistic input and output. The abstraction of LAMa permits a range of instantiations that can capture past and future leakage-abuse attacks based on frequency analysis.

At a high level, Selector specifies the left-hand expressions of frequency probability pairs, i.e. ex_L in pair $\text{fp} = (\text{ex}_L, \text{ex}_R)$. In the proposed framework LAMa we *only consider matching pairs*. The reason behind this design choice is efficiency. We deem it more efficient to identify all table entries with precomputed probabilities that match an observed frequency than to identify all the entries that do not match the aforementioned frequency. After this, Translator does the following: (*i*) identifies values that form a matching pair given what Selector chose, and (*ii*) translates the matching pairs into a logical formula. Lastly, Solver takes the logical formula, solves the satisfiability instance, and outputs a database consistent with the matching pairs.

We use the following running example throughout this section. Let the domain be $\mathcal{V} = [5]$ and the records/value assignments be $\text{DB}(\text{id}_a) = 3$ and $\text{DB}(\text{id}_b) = 4$. The query distribution QD is: Queries $[2, 4]$, $[3, 4]$, $[4, 4]$ have probability $\frac{1}{42}$. Queries $[1, 2]$, $[1, 4]$, $[2, 3]$, $[2, 5]$, $[3, 3]$, $[4, 5]$ have probability $\frac{2}{42}$. Queries $[1, 3]$, $[3, 5]$ have probability $\frac{3}{42}$. Queries $[1, 1]$, $[5, 5]$ have probability $\frac{5}{42}$. Query $[2, 2]$ has probability $\frac{4}{42}$ and query $[1, 5]$ has probability $\frac{7}{42}$.

Selector : Choosing Record-Retrieval Events. The Selector component is responsible for determining which record-retrieval set expressions will be used for the attack. In this way, Selector has a large impact on the runtime and accuracy of the attack. If the set expressions it chooses are too few, or too small, the resulting reconstruction may fall outside the set $IRD_{QD}(\text{DB})$. If the set expressions are too many, or too large, the runtime of the reconstruction algorithm may dramatically increase. Rather than explicitly generating matching pairs, Selector outputs only the left-hand-side expressions (i.e., ex_L in our definition of frequency probability pair) as a collection $\text{EX}_L = \{\text{ex}_{L,1}, \text{ex}_{L,2}, \ldots\}$. In our running example, suppose that Selector chooses two expressions: $\text{EX}_L = \{\text{ex}_{L,1}, \text{ex}_{L,2}\} = \{e_{\text{id}_a}, e_{\text{id}_a} \cap e_{\text{id}_b}\}$.

Translator: One Formula from All Matching Pairs. Translator finds matching pairs and then "translates" them into a logical formula that constrains the value-to-record assignments. To do this, Translator finds all right-hand expressions that form a matching pair for each left-hand expression given by Selector. That is, for every $\text{ex}_{L,i}$ in EX_L, Translator finds every expression $\text{ex}^j_{R,i}$ (where j iterates over all matching pairs for $\text{ex}_{L,i}$) such that $f(\text{ex}_{L,i}) = \Pr[\text{ex}^j_{R,i}]$.

In our example, Translator would find, for the expression $\text{ex}_{L,1} = e_{\text{id}_a}$, the two expressions $\text{ex}^1_{R,1} = e_2$ and $\text{ex}^2_{R,1} = e_3$, denoted more generally as $\text{EX}_{R,i} = \{\text{ex}^1_{R,i}, \text{ex}^2_{R,i}, \ldots\}$. For the expression $\text{ex}_{L,2} = e_{\text{id}_a} \cap e_{\text{id}_b}$, Translator would find the two expressions $\text{ex}^1_{R,2} = (e_3 \cap e_4)$ and $\text{ex}^2_{R,2} = (e_4 \cap e_3)$. In Figure 2, the bottom section shows the frequency (from $\mathcal{L}_{Query}$) of each expression generated by the Selector being computed. The middle section shows how the precomputed probabilities are used to find right-hand expressions that match the initial left-hand expressions. Once all the matching pairs have been found, Translator uses them to generate a logical formula C. The purpose of C is to constrain the assignment of values to records which will take place in Solver. Therefore, C consists of assignment statements of the form $\text{id} = v$. We drop the $\text{DB}(\text{id})$ notation to emphasize that we are not making claims about the true value of id in DB.

In order to ensure that every C-satisfying reconstruction output by Solver is consistent with all matching pairs, the statements in C are joined by logical operators (depicted in the top section of Figure 2). In the following, each logical variable can have at most one value assigned to it, e.g., id_a cannot be both 3 and 4. The logical formula is constructed based on three observations:

(A) A Matching Pair May Imply Multiple Simultaneous Assignments. In our running example, the equality $f(e_{\text{id}_a} \cap e_{\text{id}_b}) = \Pr[e_4 \cap e_3]$ implies the simultaneous assignments $\text{id}_a = 4$ and $\text{id}_b = 3$. Thus, in C, these assignments are connected with the logical AND to form a single statement $(\text{id}_a = 4 \wedge \text{id}_b = 3)$. Generally, Translator uses the $\wedge$ operator to bind assignments from the same matching pair.

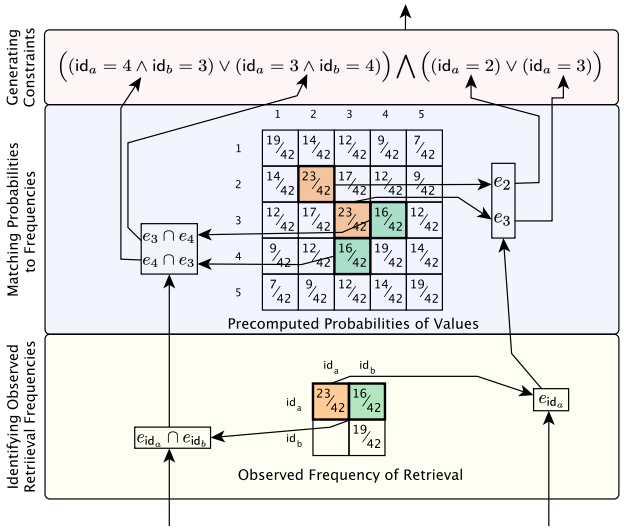


Figure 2: Internal view of Translator identifying matching pairs and generating the formula passed to the Solver.

(B) Matching Pairs With the Same Left-Hand Expression Imply Mutually Exclusive Assignments. Staying with our example, notice that the results $f(e_{\mathrm{id}_a} \cap e_{\mathrm{id}_b}) = \Pr[e_4 \cap e_3]$ and $f(e_{\mathrm{id}_a} \cap e_{\mathrm{id}_b}) = \Pr[e_3 \cap e_4]$ imply the mutually exclusive statements $(\mathrm{id}_a = 4 \wedge \mathrm{id}_b = 3)$ and $(\mathrm{id}_a = 3 \wedge \mathrm{id}_b = 4)$. In C, they are joined by the logical OR operator $\vee$, yielding $(\mathrm{id}_a = 4 \wedge \mathrm{id}_b = 3) \vee (\mathrm{id}_a = 3 \wedge \mathrm{id}_b = 4)$. Where we use the fact that a logical variable can have at most one value and, thus, at most, one of the OR-ed expressions can be true. More generally, all statements from matching pairs with identical left-hand expressions are joined by the $\vee$ operator.

(C) The True Database Satisfies All Matching Pairs Simultaneously. In our example, at least one assignment statement coming from the expression $\mathsf{ex}_{L,1}$ must correspond to the true underlying plaintext DB, and the same is true for the assignment statements coming from $\mathsf{ex}_{L,2}$. To reduce the search space of reconstructions, Translator requires all logical expressions from (A) and (B) to hold simultaneously, i.e., we join them with the $\wedge$ operator.

In our example, the following formula C is output by Translator:

$$C = \Big((\mathrm{id}_a = 4 \wedge \mathrm{id}_b = 3) \vee (\mathrm{id}_a = 3 \wedge \mathrm{id}_b = 4)\Big) \bigwedge \Big((\mathrm{id}_a = 2) \vee (\mathrm{id}_a = 3)\Big).$$

The following concisely expresses C as described above:

$$C = \left(\bigwedge_{\mathsf{ex}_{L,i} \in \mathsf{EX}_L} \bigvee_{\mathsf{ex}^j_{R,i} \in \mathsf{EX}_{R,i}} \bigwedge_{\substack{e_{\mathsf{id}} \text{ from} \\ \mathsf{fp}=(\mathsf{ex}_{L,i}, \mathsf{ex}^j_{R,i})}} \mathrm{id} = g_{\mathsf{fp}}(\mathrm{id}) \right),$$

where we highlight that g_{fp} is the value-to-record assignment with respect to the specific fp matching pair.

Solver : Reconstruction as Constraint-Satisfaction. Solver takes the formula C output by Translator and finds an assignment of values to identifiers that satisfies C. Notice that any assignment that satisfies C will have exactly one value per record. Furthermore, any assignment that satisfies C will satisfy all matching pairs output by Selector, and one of these assignments is guaranteed to be the true database DB. In our example, there is only one assignment that satisfies the constraints, that is $(\mathrm{id}_a, \mathrm{id}_b) = (3, 4)$.

In our experiments in Section 6, we use a constraint solver, but any method for finding a C-satisfying assignment will do. It is possible to make Solver output more assignments by having it repeatedly find a satisfying assignment and then add a constraint that explicitly forbids the new assignment. In this way, multiple C-satisfying assignments can be found, one of which is guaranteed to be the true database DB.

4 THE MATCHING PAIRS THAT BRING FREQUENCY ANALYSIS TO ITS LIMIT

In this section, we introduce an instantiation[1] of LAMa from Section 3 that works in high dimensions and outputs databases *exclusively* from the set of identical-response-distribution databases $IRD_{QD}(\mathrm{DB})$ which, as we covered in 3.2, is the best outcome a frequency analysis attack can hope for.

Specifically, we show that for any database, domain, and query distribution, there always exists a collection of left-hand expressions of matching pairs, denoted as $T_\cap$, such that (*i*) it only uses intersection operations – see (Q1) in 3.2, (*ii*) the corresponding matching pairs implied by $T_\cap$ are *sufficient* to reconstruct databases exclusively from $IRD_{QD}(\mathrm{DB})$, and (*iii*) the *size of the largest expression in $T_\cap$ grows linearly with the dimension k* of the plaintext domain – see (Q3) in 3.2.

The last characteristic is a rather surprising finding given the almost universal applicability of the so-called "curse of dimensionality" in different contexts, i.e., the phenomenon where the complexity of a task increases exponentially to the number of dimensions.

In terms of the universality of this finding, we emphasize that the choice of left-hand expressions in $T_\cap$ is agnostic as to which query distribution is operating on the DB, i.e., it holds under all query distributions and databases. Additionally, the proof of the sufficiency of $T_\cap$ serves as an upper-bound on the number of matching pairs required to reconstruct any database, see (Q2) in 3.2.

The Instantiation with $\mathbf{T}_\cap$. Informally, the collection in $T_\cap$ contains the following set expressions: for every $i \in [2k]$, all possible $\binom{n}{i}$ subsets of records of DB that are connected under intersection (the ordering is arbitrary). If we were to list these expressions we would have the following: for $i = 1$ we get $e_{\mathrm{id}_1}, \ldots, e_{\mathrm{id}_n}$, for $i = 2$ we get the family of pairs $e_{\mathrm{id}_1 \cap \mathrm{id}_2}, e_{\mathrm{id}_1 \cap \mathrm{id}_3} \ldots, e_{\mathrm{id}_1 \cap \mathrm{id}_n}, e_{\mathrm{id}_2 \cap \mathrm{id}_3}, \ldots, e_{\mathrm{id}_{n-1} \cap \mathrm{id}_n}$, for $i = 3$ the family of triplets $e_{\mathrm{id}_1 \cap \mathrm{id}_2 \cap \mathrm{id}_3}, \ldots$ all the way to the family of $2k$-tuples. More formally, for a set of records $\mathcal{I}$ and domain $\mathcal{V} = [N]^k$, the collection $T_\cap$ is defined as:

$$T_\cap = \left\{ \bigcup_{i_1 \le n} e_{\mathrm{id}_{i_1}}, \ldots, \bigcup_{i_1 < \ldots < i_{2k} \le n} e_{\mathrm{id}_{i_1}} \cap \ldots \cap e_{\mathrm{id}_{i_{2k}}} \right\}$$

Next, we characterize all the databases that result from an instantiation of LAMa where Selector uses $T_\cap$.

[1] A naive approach would have the Selector component select every possible matching test, which quickly becomes computationally infeasible, even for small plaintext domains.

Definition 3. *Let QD be a query distribution that operates on DB. Let DB′ be the output of LAMa (where the frequency of retrievals is taken in the limit) in which the Selector uses $T_\cap$ as its EX_L. A database DB′ is called* $T_\cap$-passing *if and only if the value-to-record assignment of DB′ implies a collection of ex_R for $T_\cap$ such that all resulting frequency-probability pairs (with respect to the values of DB′) are also matching pairs in the limit.*

This definition labels a database as $T_\cap$-passing if it can be given as an output by LAMa when Selector uses $T_\cap$ (with respect to some true database DB). We note that in the following, we make reference to ex_L expressions that correspond to different databases. To disambiguate, we will use a superscript, i.e., the notation ex_L^{DB} indicates that this left-hand expression refers to retrievals from database DB.

In the following, we point out how a $T_\cap$-passing database relates to the true database. Recall that the frequency of a subset S of records retrieved simultaneously in a response is defined relative to a *fixed* true database DB. Therefore, if one finds a $T_\cap$-passing database DB′ that is different than DB, it follows that the subset of records S is retrieved simultaneously in a response with the same frequency in both DB and DB′, even though the assigned plaintext values in DB′ are different from DB. More formally, for all $(\text{ex}_L, \text{ex}_R) \in T_\cap$ we have $f(\text{ex}_L^{\text{DB}}) = f(\text{ex}_L^{\text{DB}'})$. If we expand the above relation by using the expressions of $T_\cap$ (which consists of intersections of events), we get:

$$f\left(\bigcap_{\text{id}\in S} e_{\text{id}}^{\text{DB}}\right) = f\left(\bigcap_{\text{id}\in S} e_{\text{id}}^{\text{DB}'}\right), \text{for all } S \subseteq \mathcal{I} : |S| \leq 2k. \quad (2)$$

Equation 2 will be used in the proof of Theorem 4.1.

All Reconstructions via $\mathbf{T_\cap}$ Are In $\mathbf{IRD_{QD}(DB)}$. In the following, we show that the response distributions of DB and *any* $T_\cap$-passing database DB′ are the same. This is a surprising finding, as it shows that a global property of a candidate reconstruction database (i.e., matching the response distribution of the original) can be guaranteed by ensuring that a set of local properties holds (matching the frequencies of simultaneous retrieval for certain subsets of records). Furthermore, by discovering a database that satisfies these local properties, one effectively generates a database with a response distribution identical to the true one, i.e., one from $IRD_{QD}(\text{DB})$. Since the attacker in this setting cannot prioritize over members from $IRD_{QD}(\text{DB})$, this means that no parameterization can do better than $T_\cap$.

As a first step, we give a lemma concerning the geometry of queries in $\mathcal{V} = [N]^k$.

Lemma 1. *For any set of values V in $\mathcal{V} = [N]^k$, there exists a subset $V^* \subseteq V$ of size at most $2k$, such that any query covering all values in V^* covers all values in V.*

Proof. To see that this is true, notice that for any set of values V in $[N]^k$, there is, for each attribute/dimension, at least one minimal value and at least one maximal value.

If, for each dimension, we define V^* as the set such that for the i-th dimension (where $1 \leq i \leq k$), we include in V^* the point from V that is minimal and the point from V that is maximal with respect to the i-th dimension, we have at most $|V^*| = 2k$ distinct points. Geometrically, each of these $2k$ values resides on the face of a hyper-rectangle in the k-dimensional space where all values of V are either internal to the hyper-rectangle or on its face. Any query covering the $2k$ values of V^* will cover all values in V. □

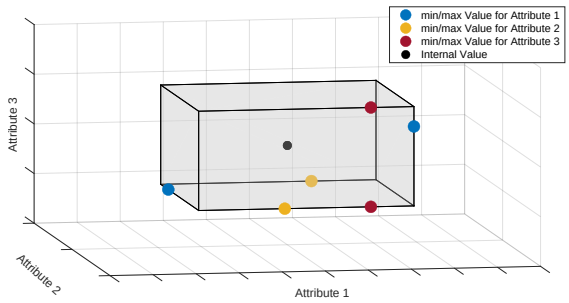


Figure 3: Values of a DB in k=3 dimensions. Red/blue/yellow/black points comprise the associated values of records that are part of rsp. Queries that cover the $2k = 6$ min/max "colored" values from rsp with respect to Attribute-1, Attribute-2, and Attribute-3 *must* cover all records of rsp with internal values.

Figure 3 illustrates Lemma 1 in the context of record retrieval: in any response rsp with V associated values, there is always a set V^* of $2k$ or fewer values from V such that every query covering V^* also covers all other records in V.

Theorem 4.1. *Let DB and DB′ be databases over $\mathcal{I}$ and $\mathcal{V} = [N]^k$. Let QD be the query distribution that issues queries for DB and DB′, and let RD, RD′ be the respective response distributions of DB and DB′, both induced by QD. If DB′ is $T_\cap$-passing, then RD = RD′.*

Proof. Assume, for the sake of contradiction, that $RD \neq RD'$. Since the response distributions differ, then there must be at least one response $\text{rsp} \in \mathcal{R}$ whose probability in RD is different from its probability in RD', i.e., $\Pr_{RD}[\text{rsp}] \neq \Pr_{RD'}[\text{rsp}]$. Let rsp_{lrg} be the response with the largest number of records whose probability differs in RD and RD'. Notice that rsp_{lrg} cannot be the empty response, since if RD and RD' differ on the empty rsp_{lrg}, then they must differ on at least one other response rsp′, which will be non-empty. In that case, $|\text{rsp}'| > |\text{rsp}_{\text{lrg}}|$ which contradicts the fact that rsp_{lrg} has the largest number of retrieved records.

The set of values in $\mathcal{V}$ associated with the records in rsp_{lrg} are referred to as V. Assume w.l.o.g. that the probability of rsp_{lrg} in RD is greater than its probability in RD', that is:

$$\Pr_{RD}[\text{rsp}_{\text{lrg}}] > \Pr_{RD'}[\text{rsp}_{\text{lrg}}]. \quad (3)$$

By Lemma 1, there exists a subset of records $\text{rsp}^*_{\text{lrg}} \subseteq \text{rsp}_{\text{lrg}}$ whose values comprise a set V^* such that $V^* \subseteq V$ and $|V^*| \leq 2k$. Specifically, set V^* contains the minimal and maximal values (among the choices in rsp_{lrg}) for each dimension. Given that, DB′ is $T_\cap$-passing, from Equation (2), we have that the records associated with the values in V^* must have the same probability of simultaneous retrieval

in RD and RD', that is:

$$f\left(\bigcap_{\mathsf{id}\in\mathsf{rsp}^*_{\mathsf{lrg}}} e^{\mathsf{DB}}_{\mathsf{id}}\right) = f\left(\bigcap_{\mathsf{id}\in\mathsf{rsp}^*_{\mathsf{lrg}}} e^{\mathsf{DB}'}_{\mathsf{id}}\right). \tag{4}$$

According to a generalization of relation (2), each frequency of Equation (4) can be written as a sum of the probability of all responses that contain all records from $\mathsf{rsp}^*_{\mathsf{lrg}}$.

Notice that the response $\mathsf{rsp}_{\mathsf{lrg}}$ will contribute to both the expansion of the left and right frequency terms of (4). But given the inequality $\Pr_{RD}[\mathsf{rsp}_{\mathsf{lrg}}] > \Pr_{RD'}[\mathsf{rsp}_{\mathsf{lrg}}]$, the contribution of term $\Pr_{RD}[\mathsf{rsp}_{\mathsf{lrg}}]$ is larger towards $f\left(\bigcap_{\mathsf{id}\in\mathsf{rsp}^*_{\mathsf{lrg}}} e^{\mathsf{DB}}_{\mathsf{id}}\right)$ than its counterpart $\Pr_{RD'}[\mathsf{rsp}_{\mathsf{lrg}}]$ towards $f\left(\bigcap_{\mathsf{id}\in\mathsf{rsp}^*_{\mathsf{lrg}}} e^{\mathsf{DB}'}_{\mathsf{id}}\right)$. But since the two frequencies must be equal according to (4), there must be a different response, call it rsp'', containing all the records in $\mathsf{rsp}^*_{\mathsf{lrg}}$, such that rsp'' has a higher probability in RD' than RD. We proceed with case analysis:

- *Case where rsp'' has larger size than $\mathsf{rsp}_{\mathsf{lrg}}$.* This contradicts the assumption that $\mathsf{rsp}_{\mathsf{lrg}}$ is the largest response whose probability differs in RD and RD'.
- *Case where rsp'' has smaller size than $\mathsf{rsp}_{\mathsf{lrg}}$.* Recall that for rsp'' to contribute towards $f\left(\bigcap_{\mathsf{id}\in\mathsf{rsp}^*_{\mathsf{lrg}}} e^{\mathsf{DB}'}_{\mathsf{id}}\right)$, it has to contain all records from $\mathsf{rsp}^*_{\mathsf{lrg}}$. From Lemma 1, all responses that contain $\mathsf{rsp}^*_{\mathsf{lrg}}$ must also contain the internal values of $\mathsf{rsp}_{\mathsf{lrg}}$, thus, there can not be a response containing $\mathsf{rsp}^*_{\mathsf{lrg}}$ with smaller size than $\mathsf{rsp}_{\mathsf{lrg}}$. We can dismiss this case.
- *Case where rsp'' and $\mathsf{rsp}_{\mathsf{lrg}}$ have the same size.* Recall that $\mathsf{rsp}^*_{\mathsf{lrg}}$ is the subset of records in $\mathsf{rsp}_{\mathsf{lrg}}$ which are maximal/minimal among $\mathsf{rsp}_{\mathsf{lrg}}$. If rsp'' contains these records, it must also contain the other records of $\mathsf{rsp}_{\mathsf{lrg}}$, since they are internal, which means that $\mathsf{rsp}'' = \mathsf{rsp}_{\mathsf{lrg}}$. If $\mathsf{rsp}'' = \mathsf{rsp}_{\mathsf{lrg}}$, then recall that we already assumed[2] from 3 that $\Pr_{RD}[\mathsf{rsp}_{\mathsf{lrg}}] > \Pr_{RD'}[\mathsf{rsp}_{\mathsf{lrg}}]$ which means that it is not possible to have $\Pr_{RD}[\mathsf{rsp}''_{\mathsf{lrg}}] < \Pr_{RD'}[\mathsf{rsp}''_{\mathsf{lrg}}]$ which is what needed to balance out the sums and get 4 to hold. Thus, no response containing $\mathsf{rsp}^*_{\mathsf{lrg}}$ exists which has greater probability in RD' than RD, i.e., $RD = RD'$. □

We emphasize here that our analysis holds for all databases and query distributions (even in high dimensions). Beyond our findings, which apply universally to all query distributions, we conjecture that it is possible to devise matching tests tailored to specific query distributions. Such a finding would permit a smaller set of matching pairs, e.g., one such example is the work of Kellaris *et al.* [26] for one-dimensional data and uniform query distribution. We leave this question as an open problem for future work.

$\mathbf{T}_\cap$ Gives a Tight Upper Bound. Recall that $T_\cap$ represents the set of expressions that use only the intersection operator $\cap$, with the largest expressions in $T_\cap$ containing up to $2k$ terms. This result, therefore, establishes $2k$ as an upper bound on the size of intersection-based expressions required so that LAMa outputs only databases within $IRD_{QD}(\mathrm{DB})$. To show that this upper bound is tight, we compare the parameterization $T_\cap$ to the parameterization $T'_\cap$, which differs only in that $T'_\cap$ excludes expressions of size exactly $2k$. In doing so, we find that for any dimension k, there exists a case where a database DB' (distinct from the true database DB) is $T'_\cap$-passing but not $T_\cap$-passing. This confirms that expressions of size $2k$ provide a more accurate reconstruction than those limited to size $2k-1$.

We give a constructive proof for the next Theorem that shows how, for any k, an appropriate DB, DB', N, and QD can be found, which will satisfy Theorem 4.2. The proof works by finding two disjoint sets of values, V and V', each of size $2k$. The values in V will be assigned to records under DB, and the values in V' will be assigned to records under DB'.

Theorem 4.2. *For any dimension k and $N \geq 6$, there exists a domain $\mathcal{V} = [N]^k$, a database DB, and a query distribution QD such that, given the query leakage from DB, at least one database DB' is $T'_\cap$-passing but not $T_\cap$-passing (in the limit).*

Proof. Let k and N be positive integers where $N \geq 6$. Let DB and DB' be databases over the set of records $\mathcal{I} = \mathsf{id}_1, \ldots, \mathsf{id}_{2k}$ and over the domain $\mathcal{V} = N^k$. Let QD be the uniform distribution over $\boldsymbol{Q}$, the query universe for domain $\mathcal{V}$. For ease of exposition, we represent the query distribution as an assignment of weights to queries: each query q_i has a positive integer weight w_i, and the probability of query q_i is given by its weight divided by the sum of all query weights, i.e. $\frac{w_i}{\sum_{j\in[|\boldsymbol{Q}|]} w_j}$. We initialize the distribution to be uniform by requiring that $w_i = \alpha$ for all $i \in [|\boldsymbol{Q}|]$, where α is a positive integer.

Next, we define two disjoint sets of values, $S = \{s_1, \ldots, s_{2k}\} \subset \mathcal{V}$ and $S' = \{s'_1, \ldots, s'_{2k}\} \subset \mathcal{V}$. Intuitively, S will be the values assigned to the records in database DB, and S' will be the values assigned to the records in database DB', so that $\mathrm{DB}(\mathsf{id}_i) = s_i$ and $\mathrm{DB}'(\mathsf{id}_i) = s'_i$.

Now we define the values in S and S', thereby defining the values of records in each database. For $i \in [n]$, let s_i be the k-vector with 1 on all dimensions, except for dimension $\lceil \frac{i}{2} \rceil$, on which it is 1 if i is odd and 3 if i is even. For $i \in [n]$, let s'_i be the k-vector with $N-1$ on all dimensions, except for dimension $\lceil \frac{i}{2} \rceil$, on which it is N if i is odd and $N-2$ if i is even. For every value $s_i \in S$, we say that it has a corresponding value $s'_i \in S'$, and that every subset $\{s_a, s_b, \ldots, s_c\} \subset S$ has a corresponding subset $\{s'_a, s'_b, \ldots, s'_c\} \subset S'$. Note that both databases have the following nice property: for $i \in [2k]$, each set of i records in DB (resp. DB') has a minimum-bounding query that covers no other records in DB (resp. DB').

Next, we adjust the query distribution by altering the weights of queries, according to the following procedure: For $i \in [2k]$, if i is odd, **decrease** the weight of the MBQ of every i-sized subset of S by δ, and if i is even, **increase** the weight of the MBQ of every i-sized subset of S by δ.

At the end of this procedure, the following holds:

(1) The probability of simultaneously querying all values in S, is δ greater than the probability of simultaneously querying all values in S', i.e. $\Pr[S] > \Pr[S']$.

[2] We note here that if one changes the inequality of 3 to go the opposite direction, this last case of the case analysis will again reach a contradiction; this time because we can not find a rsp'' such that $\Pr_{RD}[\mathsf{rsp}''_{\mathsf{lrg}}] > \Pr_{RD'}[\mathsf{rsp}''_{\mathsf{lrg}}]$.

(2) The probability of simultaneously querying any strict subset of S is equal to the probability of simultaneously querying its corresponding subset in S', i.e. $\Pr[S_*] = \Pr[S'_*]$ for all strict non-empty subsets $S_* \subset S$.

To see that the first statement is true, notice that the set of queries that cover S and the set of queries that cover S' begin with a uniform weighting, and that the only such query altered by the procedure is the MBQ of S. Crucially, this query does not cover S' since, by construction, every value in S' dominates every value in S, which means that their MBQ's do not cover any of the same values.

To see that the second statement is true, consider a strict, non-empty subset $S_* \subset S$. For each subset of S containing S_*, the corresponding MBQ is increased by δ if the subset's size is even, and decreased by δ if its size is odd. For any i between $|S_*|$ and $|S|$, there are $\binom{|S|-|S_*|}{i}$ subsets of size i that contain S. Thus, the total weight added to queries covering S_* is given by

$$\delta \sum_{i=0}^{|S|-|S_*|} \binom{|S| - |S_*|}{i} (-1)^i$$

which equals zero:

$$\begin{aligned} 0 &= \delta(1-1)^{|S|-|S_*|} \\ &= \delta \sum_{i=0}^{|S|-|S_*|} \binom{|S| - |S_*|}{i} 1^{|S|-|S_*|-i}(-1)^i \\ &= \delta \sum_{i=0}^{|S|-|S_*|} \binom{|S| - |S_*|}{i} (-1)^i. \end{aligned}$$

Thus, for all strict subsets $S_* \subset S$, it holds that $\Pr[S_*] = \Pr[S'_*]$.

To complete the proof, we assume that DB is the real database, over which the leakage occurs. The database DB′ **will not** be $T_\cap$-passing, since DB and DB′ differ on $f(e_1 \cap e_2 \cap \ldots \cap e_{2k})$, which will be checked by $T_\cap$'s $2k$-sized expression. However, DB′ **will** be $T'_\cap$-passing, since every set containing fewer than $2k$ records will have the same frequency of being simultaneously queried in DB and DB′, and $T'_\cap$ has no expressions larger than $2k-1$. □

5 TWEAKING QUERY DISTRIBUTIONS TO FLATTEN FREQUENCY OF RETRIEVALS

Having shown that a database DB with query distribution QD can be reconstructed up to $IRD_{QD}(\text{DB})$, we ask:

"*What can a defender do to mitigate the reconstructive power of frequency analysis?*"

A natural approach is to attempt to increase the size of $IRD_{QD}(\text{DB})$, which in turn introduces more "uncertainty" to the attacker since $IRD_{QD}(\text{DB})$ contains plausible reconstructions under this attack setting. To accomplish this increase, one has to change either DB, QD, or both. Altering the data is not ideal since this may undermine the correctness of the *R-STE* scheme. Therefore, the defender's alternative recourse lies in the choice of the query distribution QD.

In this section, we study how a defender can "tweak" the query distribution to increase the number of plausible reconstructions for an attacker mounting a frequency analysis attack. Any change in the query distribution will translate to an updated response distribution RD, affecting the frequency of retrievals.

Desired Properties of Query Distributions. A first approach would be to tailor the query distribution to the underlying DB, but such a strategy would *directly leak* information about DB since, in this setting, we assume that QD is known to the attacker. Thus, we only study query distributions that are *independent of the database DB they operate on.* This way, our (universal) analysis holds regardless of which database is queried. In particular, we focus on *expressive query distributions*, i.e., distributions where every query $q \in \mathcal{Q}$ has a non-zero probability of being issued. We avoid non-expressive distributions for the same reason that we avoid altering the data: forbidding queries degrades the functionality of the scheme.

First, we examine the limitations of increasing $IRD_{QD}(\text{DB})$ through the choice of query distribution. We then explore whether selecting an appropriate QD can ensure a meaningful privacy property for the databases within $IRD_{QD}(\text{DB})$. As an affirmative answer, we show an expressive distribution $\widetilde{QD}$ which guarantees, for any database DB, a corresponding $IRD_{\widetilde{QD}}(\text{DB})$ containing all databases with the *same pairwise* L_1 *distances* as those of DB.

5.1 Flattening the Frequency Across Record-Retrieval Events

For the defender, an ideal query distribution QD^* would imply an $IRD_{QD^*}(\text{DB})$ that consists of every possible database over $\mathcal{I}, \mathcal{V}$. In such a case, an attacker using frequency analysis would have no advantage over a random guess from the set of all possible databases over $\mathcal{I}, \mathcal{V}$.

Recall from Theorem 4.1 that any two databases for which Equation (2) holds must have the same response distribution. It follows that if we could construct a query distribution QD^* under which: all values are queried with probability p^*_1, all intersections of pairs of values are queried with probability p^*_2,..., all intersections of $2k$-sets of values are queried with probability p^*_{2k}, then all databases operating under QD^* would have the same response distribution. In this case, the attacker can identify which of the $p^*_1, \ldots, p^*_{2k}$ probabilities is being processed but cannot infer anything about the underlying plaintext values because all possible geometries of plaintexts in a subset S of values gives exactly the same probability $p^*_{|S|}$.

Algorithm 1: Flatten Probability of Single Values

Data: Input QD is seen as a dictionary that maps queries $q \in \mathcal{Q}$ to weights $QD[q] = w_q$

. Define v_{mx} as $v_{\text{mx}} = \arg\max_{v \in \mathcal{V}} \Pr[e_v]$ and call $\Pr[e_{v_{\text{mx}}}]$ as p^*_1;
Find the sum s_{mx} of the weights of queries covering v_{mx};
for *every value* v_i *in* $\mathcal{V}$ **do**
    Find the sum of weights s_i of queries covering v_i;
    $QD[[v_i, v_i]] = QD[[v_i, v_i]] + (s_{\text{mx}} - s_i)$; // $\Pr[e_{v_i}] = p^*_1$
end
return QD

As a warm-up, we show in Algorithm 1 how this can be done for 1-tuples of records by only adjusting the probabilities of 1-tuples of values in any input distribution QD. Our approach will impose a "minimal" change in the input QD by increasing the probabilities of just the smallest queries (those that cover only a single value).

For simplicity of the exposition, we assume that every query q is associated with a weight w_q (which is a natural number) and that the probability of this query q is given by normalizing its weight divided by the sum of all query weights.

Intuitively, the proposed algorithm works by first finding a value $v \in \mathcal{V}$ with the highest probability $\Pr[e_v]$ denoted as p_1^*. Then, for each other value v', we increase the weight of the query $[v', v']$ so that $\Pr[e_{v'}] = p_1^*$. When the process ends, $\Pr[e_v] = \Pr[e_{v'}]$ will hold for all $v, v' \in \mathcal{V}$.

Theorem 5.1. *Let QD be a query distribution over $\mathcal{Q}$. Let QD' be the output of Algorithm 1 with input QD, then we have:*

$$\Pr_{QD'}[e_v] = \Pr_{QD'}[e_{v'}] \text{ for all } v \in \mathcal{V}.$$

Proof. First, Algorithm 1 identifies a value v_{mx} with highest probability, and computes the total weight s_{mx} of queries covering it. In lines 2 through 5 it increases, for each value v_i, the weight of query $[v_i, v_i]$, until the total weight of queries covering v_i is equal to s_{mx}. Since the weight w_{q_i} of query $q = [v_i, v_i]$ contributes only to the probability of v_i, increasing w_{q_i} does not affect the probabilities of other values. At the end of the process, although the total sum of query weights has increased, the total weights of queries covering any value v is s_{mx}. Thus, after normalizing, we have that $\Pr_{QD}[e_v] = \Pr_{QD}[e_{v'}]$ for all $v, v' \in \mathcal{V}$. □

The effect of the "frequency-flattening" from Algorithm 1 is that an adversary who performs only matching tests of size one will be unable to refine their reconstruction beyond the set of all databases over $\mathcal{I}, \mathcal{V}$. If one applies the KKNO attack [26] for the case where the queries are issued by the output distribution of Algorithm 1, the attack is neutralized since size-1 matching tests cannot find an ordering and, thus, can not identify an "anchor-point".

Unfortunately, our findings show that there is no way to do the same frequency-flattening for all intersections of sets of size greater than 1, which means that there is no hope of constructing the ideal (for the defender) QD^* that was discussed in this section. In fact, as we show in the following theorem, there is *no expressive query distribution* in which every pair of values of distance d have the same probability as every pair of distance d' if $d \neq d'$.

Theorem 5.2. *Let QD be a query distribution over the universe of queries $\mathcal{Q}$ and domain $\mathcal{V} = [N]^k, N > 2$, such that every query $q \in \mathcal{Q}$ has non-zero probability. For every pair of values v, v' in $\mathcal{V}$ with L^1-distance $dist(v, v') = d$, there exists a pair of values v, v'' with L^1-distance $dist(v, v'') \neq d$ such that $\Pr[e_v \cap e_{v'}] \neq \Pr[e_v \cap e_{v''}]$.*

Proof. Let $\mathcal{V} = [N]^k$ be the domain of values, where $N > 2$. Let t_1 be a pair of values in $\mathcal{V}$ such that $dist(t_1) = d$. We will show that there is always another pair t_2 of values in $\mathcal{V}$ with distance $dist(t_2) \neq d$ such that either: (*i*) every query covering t_1 covers t_2 or (*ii*) every query covering t_2 covers t_1.

For any pair $t_1 = \Big(v = (v_1, \ldots, v_k), v' = (v'_1, \ldots, v'_k)\Big)$, we construct another pair $t_2 = (v, v'')$, where v'' is equal to v' in all but one dimension; let the differing dimension be j, then the value of v''_j is (*i*) $v_j + 1$ in case $v_j < N$, or (*ii*) $v_j - 1$ in case $v_j = N$. Notice that we can always apply the above transformation from v to v'' regardless of the number of dimensions and the choice of v. Since v' is altered by 1 on a single dimension, and v stays the same, the distances of t_1 and t_2 must be different. Furthermore, if $dist(t_2) > dist(t_1)$, then every query covering t_2 covers t_1, and if $dist(t_2) < dist(t_1)$, then every query covering t_1 covers t_2. Since all queries have non-zero probability, and one of the two pairs is covered by a strict subset of the queries covering the other, the pairs must have a different probability, i.e., $\Pr[e_v \cap e_{v'}] \neq \Pr[e_v \cap e_{v''}]$. □

This result effectively dashes any hopes of constructing an ideal distribution QD^* that is both *expressive* and *database-agnostic*. It shows that we cannot flatten the frequency of retrieval of pairs of records in a data-agnostic way, let alone larger tuples of records. This throws us back on the question of how to effectively alter $IRD_{QD}(\mathrm{DB})$ in a way that is consistent across databases.

5.2 Flattening Retrieval Frequency Across Pairs of Records with Equidistant Values

Fortunately, Theorem 5.2 does not preclude the possibility of an expressive query distribution under which all value pairs with the *same distance* are plausible plaintext value assignments. Such a distribution can in fact be constructed using an iterative version of Algorithm 1. The following algorithm outputs, for any expressive query distribution QD, a new distribution $\widehat{QD}$, under which all value pairs of distance d have the same probability of being queried, for $d = 0, 1, \ldots, k(N-1)$ (i.e. the maximum L^1 distance in the domain $\mathcal{V} = [N]^k$). Note that a "pair" of values with distance 0 is just the same value twice, e.g., (v, v).

Algorithm 2: Flatten Frequency of Equidistant Pairs

Data: Input QD is a dictionary that maps queries to natural number weights $QD[q] = w_q$

1 . **for** $d = k(N-1), k(N-1)-1, \ldots, 0$ **do**
2 Find the pair of distance d values $t_{\mathsf{mx}} = (v, v')$ with the highest probability $\Pr[e_v \cap e_{v'}]$ among distance d pairs;
3 Find the sum s_{mx} of weights of queries covering t_{mx};
4 **for** *every pair of values $t = (v, v')$ with distance d* **do**
5 Find the sum of weights s_t of all queries covering t;
6 Find the query $q = [a, b]$ that covers t and a is as large as possible while b is as small as possible, i.e., the minimum bounding query;
7 $QD[q] = QD[q] + (s_{\mathsf{mx}} - s_t)$;
8 **end**
9 **end**
10 **return** QD

Theorem 5.3. *Let QD be an expressive query distribution over $\mathcal{Q}$. For the query distribution $\widehat{QD}$ output by Algorithm 2 on input QD, the following holds for all pairs $(v, v'), (v'', v''')$ where $dist(v, v') = dist(v'', v''')$:*

$$\Pr_{\widehat{QD}}[e_v \cap e'_v] = \Pr_{\widehat{QD}}[e_{v''} \cap e_{v'''}]$$

Proof. We first establish a few facts about the geometry of queries over $\mathcal{V} = [N]^k$. Recall that we define the size of a query $q = [a, b]$ as the L_1 distance between a and b, $dist(a, b)$.

(1) For every pair of values v, v', there is a smallest query q covering v, v', called the *minimum bounding query*, and the size of q is equal to $dist(v, v')$.
(2) A query $q = [a, b]$ such that $dist(a, b) = d$ does not cover any pairs of values of distance greater than d.

Algorithm 2 iterates over all pairs of values in order of decreasing distance d. For each d, it finds the pair t_{mx} of distance d with the greatest probability. It then increases, for each pair of distance d, the weight of its minimum bounding query. At the end of each iteration d, the pairs of distance d are guaranteed to have the same probability. Furthermore, in each iteration d, Algorithm 2 only alters the weights of queries with endpoints with distance d. Thus, it only alters the probabilities of pairs of values with distance d or less. Since every iteration d ends with d-distance pairs having the same probability, and future iterations $d-1, \ldots$ will not affect the probability of d-distant pairs, it holds that Algorithm 2 terminates by outputting a distribution QD in which all distance-d pairs have the same probability, for $d \in \{0, 1, \ldots, k(N-1)\}$. □

An illustration of the effect of $\widehat{QD}$ on the intersection of retrievals is presented in Figure 4. This toy example is in the two-dimensional domain $\mathcal{V} = [4] \times [4]$. The Y-axes of the plots indicate the possible underlying plaintext values of the first record of the intersection, while the X-axes indicate the possible underlying plaintext values of the second record. For example, the pair of values $v = (1, 3)$ and $v' = (1, 1)$ has the same probability of simultaneous retrieval (indicated with a unique color filling) as any other pair of distance 2, i.e., pair $(1, 3)$ and $(3, 3)$, pair $(2, 4)$ and $(2, 2)$ etc. Thus, by observing a frequency of simultaneous retrieval of a pair, the attacker can discover the distance of the underlying plaintext values but has no advantage over discovering the true values (since all placements of v and v' with a fixed distance have the same frequency). More generally, the effect of the "frequency-flattening" mechanism from Algorithm 2 is the following: for any database DB, the set $IRD_{\widehat{QD}}(\mathsf{DB})$ consists of databases DB′ over $\mathcal{I}, \mathcal{V}$ such that every pair of records in DB′ has the same L_1 distance as the same pair of records in DB. Informally, we can say that an attacker, faced with any database under query distribution $\widehat{QD}$ can infer, at most, the distance between each pair of records.

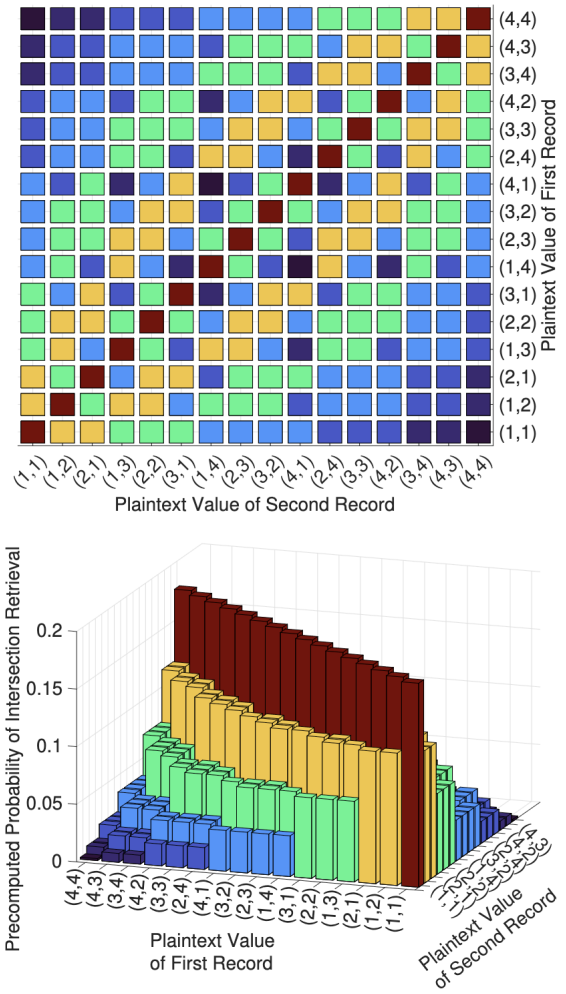


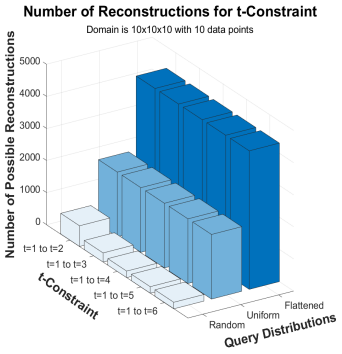
Figure 4: An illustration of how the query distribution by Algorithm 2 affects the frequency of retrieval of pairs (in $\mathcal{V}$=[4] × [4]). All pairs (v, v') that have a fixed L^1-distance, have the same probability of retrieval $\Pr_{\widehat{QD}}[e_v \cap e'_v]$.

6 EMPIRICAL EVALUATION OF LAMa

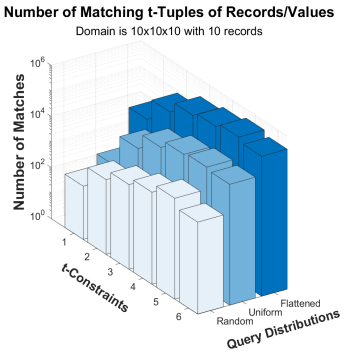
Evaluation Setting. We evaluate the LAMa framework using the HCUP [1] data set. We sample subsets of HCUP records to create databases of various domain sizes, discretizing and scaling the data as needed to accommodate the domain. Specifically, we use the `GAPICC`, `APICC`, `WIX` and `GAF` attributes, which represent, respectively, the hospital-specific all-payer inpatient cost-to-charge ratio, the group average cost-to-charge ratio, the geographical wage index, and the geographic adjustment factor, across hospitals in the 2009 dataset. In order to better observe the effects of dimensionality on reconstruction, we consider three domain sizes with roughly equivalent total numbers of values: a one-dimensional domain with $N = 1000$, a two-dimensional domain with $N = 32$, a three dimensional domain with $N = 10$, and a four dimensional domain with $N = 6$. Because we wish to have the solver find *all* reconstructions for a given set of constraints, we deliberately keep the number of records capped at 10.

All experiments are written in Python using the CP-SAT solver from Google's OR-TOOLS library [38], on a 32 core cluster with 16 GB of RAM per core, with parallelization enabled. We run experiments on three distinct query distributions. In the Uniform distribution, every query is issued with equal probability. In the Random distribution, every query is given a random weight from $[1, 2, \ldots 10]$, and then normalized. Finally, the Flattened distribution comes from applying Algorithm 2 to the uniform distribution, as described in Section 5, such that equidistant pairs of values always have the same probability of being queried.

We implement LAMa using the $T_\cap$ parameterization described in Section 4, in which every possible intersection-based matching test of t-tuples is performed for $t = 1$ to $t = 2k$. For each t value, we identify the successful frequency-value matches across all t-tuples



(a) The number of possible reconstructions decreases as t increases.



(b) The number of possible reconstructions after each t-Constraint.

		QD	Number of Reconstructions After t-Constraints					
			t = 1	t = 2	t = 3	t = 4	t = 5	t = 6
Plaintext Dimensions	$k=1$	U	512	2	-	-	-	-
		R	12	1	-	-	-	-
		F	$\sim 2^{23}$	38	-	-	-	-
	$k=2$	U	1,064	32	32	32	-	-
		R	188	96	88	88	-	-
		F	$\sim 2^{23}$	156	156	156	-	-
	$k=3$	U	86,456	2,016	2,016	2,016	2,016	2,016
		R	12,202	434	284	244	244	244
		F	$\sim 2^{23}$	4,320	4,320	4,320	4,320	4,320
	$k=4$	U	756,224	6,436	6,346	6,346	6,346	6,346
		R	86,312	1,048	682	362	362	362
		F	$\sim 2^{23}$	12,044	12,044	12,044	12,044	12,044

(c) Number of reconstructions by distribution, U = Uniform, R = Random, F = Flattened. We omit $t = 7$ and $t = 8$ as reconstruction counts do not change.

Figure 5: Evaluation of the LAMa frequency analysis framework on hospital data from HCUP across different query distributions and plaintext dimensions, using the $T_\cap$ parameterization.

of records and add these matches as constraints to the Solver. E.g., in the two dimensional space, if record id_1 has the same frequency as values $(1,1)$ and $(2,2)$, we add the constraint $\mathsf{id}_1 = (1,1) \vee \mathsf{id}_1 = (2,2)$ to the Solver. For brevity, we refer to this process for a particular t value as a t-constraint.

Optimizations. To avoid overloading the constraint-solver (and scaling our codebase), we use the results of prior matching tests at each t-constraint. That is, rather than naively adding a constraint for every value t-tuple whose frequency matches a record t-tuple, we first check whether the value t-tuple has been ruled out by the round of $t-1$-tuple tests. For example, assume that the t-constraint for $t = 1$ finds that $(3,3)$ is not a candidate value for record id_1 or id_2. When considering the t-constraint for $t = 2$ and considering the pair $\mathsf{id}_1, \mathsf{id}_2$, we can ignore any pairs of values that contain $(3,3)$ since, even if they match frequencies, they cannot lead to a valid reconstruction. To accomplish this for t values beyond 1 and 2, we run a small instance of the constraint-solver whenever we are processing a t-tuple of records and give it, as constraints, all the results of the $t-1$ tests. We then have the solver compute all satisfying assignments for the record tuple. This allows us to avoid adding as constraints values that have already been ruled out by past t-constraints.

6.1 Experimental Results

Number of Reconstructions. First, we consider the number of possible reconstructions after each t-constraint, i.e., the total number of valid solutions output by the solver when given the results of a t-constraint. Figures 5a and 5b concern the 3-dimensional frequency analysis attack scenario for ten encrypted records. Figure 5a confirms that the number of reconstructions decreases as t increases across all query distributions. Note that, for visual clarity, we omit the reconstructions for $t = 1$ since the Flattened distribution has $\sim 2^{23}$ reconstructions.

We observe in Figure 5a that for both the Flattened and Uniform distributions, the smallest set of possible reconstructions is reached after just the $t = 2$ constraint (see also Table 5c), while in the Random distribution, the reconstruction space continues to shrink until $t = 4$, at which point it also reaches $IRD_{\text{Uniform}}(\text{DB})$. As our analysis in the previous section showed, the size of the reconstruction space for Flattened remains *much larger* than either of the other query distributions.

Number of Matches Generated by t-Constraints. Next, we evaluate the number of value/record matches after t-constraints. The number of matches increases the computational burden on the solver, since every match is added to a logical constraint, which the solver must satisfy in its solution. Figure 5b depicts the increase in matches as higher t-constraints are performed. Note that each t-constraint requires finding matches for $\binom{n}{t}$ t-tuples of records, where n is the number of records. The increase in matches owes to this fast-growing expression. As evidence of this, note that the number of matches goes *down* after $t = 5$, since $\binom{10}{6} < \binom{10}{5}$. While this blowup is an unavoidable consequence of using an exhaustive parameterization like $T_\cap$, it is mitigated by the fact that we discard matches that do not agree with already considered $(t-1)$-constraint results.

Dimensionality & Number of Reconstructions. The number of possible reconstructions across distributions and dimensions is shown in Table 5c. We note that Random is relatively unaffected by the change in dimensionality but that the number of possible reconstructions in both Uniform and Flattened increases with dimensionality for the same t values. This is due to the fact that the latter distributions have a high degree of symmetry, i.e., for every subset of records in the true database, there are multiple reflected tuples with the same frequency. Higher dimensionality leads to a greater number of these reflections, causing a greater number of possible reconstructions.

Intuitively, under Uniform, every unique combination of reflections constitutes a possible reconstruction. The Flattened distribution enjoys the same reconstructions-via-reflection, but also gains additional reconstructions: any database that can be obtained by shifting all records in the true database will also be a plausible reconstruction. The Random distribution does not gain reconstructions via reflection, and thus has many fewer plausible reconstructions.

7 CONCLUSION

In conclusion, our work demonstrates that the frequency analysis technique for encrypted range schemes can be captured formally using the LAMa framework and can reconstruct databases in high dimensions and for arbitrary query distributions using only access-pattern leakage. We also prove that there exists a parameterization of LAMa that is guaranteed to give the adversary maximal reconstructive power in our setting (i.e. $T_\cap$). On the defensive side, by leveraging our newly acquired formal understanding of this threat model, we propose a query distribution that is data-agnostic such that an adversary only learns the Manhattan distances between pairs of records.

This work establishes a rigorous paradigm with which the community can assess the adversarial strength of various cryptanalytic techniques in the area of leakage-abuse attacks. Moreover, we hope that such a treatment will inspire defenses, like the one in Section 5, that work independently of which database is deployed.

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. 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.

[3] 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.

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

[5] A. Boldyreva, Z. Gui, and B. Warinschi, "Understanding Leakage in Searchable Encryption: a Quantitative Approach," *Proc. Priv. Enhancing Technol.*, vol. 2024, pp. 503–524, 2024.

[6] 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.

[7] 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.

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

[9] 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.

[10] 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.

[11] 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.

[12] 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.

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

[14] 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.

[15] 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.

[16] F. Falzon, E. A. Markatou, Z. Espiritu, and R. Tamassia, "Range search over encrypted multi-attribute data," *Proc. VLDB Endow.*, vol. 16, no. 4, pp. 587–600, 2022. [Online]. Available: https://www.vldb.org/pvldb/vol16/p587-falzon.pdf

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

[18] 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.

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

[20] 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.

[21] S. Kamara, A. Kati, T. Moataz, J. DeMaria, A. Park, and A. Treiber, "MAPLE: markov process leakage attacks on encrypted search," *Proc. Priv. Enhancing Technol.*, vol. 2024, no. 1, pp. 430–446, 2024.

[22] 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.

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

[24] 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.

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

[26] 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.

[27] E. M. Kornaropoulos, N. Moyer, C. Papamanthou, and A. Psomas, "Leakage inversion: Towards quantifying privacy in searchable encryption," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022.* ACM, 2022, pp. 1829–1842.

[28] 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.

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

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

[31] M. S. Lacharité, B. Minaud, and K. G. Paterson, "Improved Reconstruction Attacks on Encrypted Data Using Range Query Leakage," in *Proc. of the 39th IEEE S&P*, 2018, pp. 1–18.

[32] E. A. Markatou, F. Falzon, Z. Espiritu, and R. Tamassia, "Attacks on encrypted response-hiding range search schemes in multiple dimensions," *Proc. Priv. Enhancing Technol.*, vol. 2023, no. 4, pp. 204–223, 2023. [Online]. Available: https://doi.org/10.56553/popets-2023-0106

[33] 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.

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

[35] ——, "Reconstructing with even less: Amplifying leakage and drawing graphs," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, CCS 2024, Salt Lake City, UT, USA, October 14-18, 2024*, B. Luo, X. Liao, J. Xu, E. Kirda, and D. Lie, Eds. ACM, 2024, pp. 4777–4791. [Online]. Available: https://doi.org/10.1145/3658644.3670313

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

[37] 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.

[38] L. Perron and F. Didier, "CP-SAT," Google.

[39] 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.

[40] 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.

[41] 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.

8 APPENDIX

8.1 Prior Attacks via Frequency-Matching

The reconstruction attack of Kellaris *et al.* [26] can be re-framed as an application of our frequency-matching framework. Their attack assumes (*i*) an adversary who observes access-pattern leakage, (*ii*) a one-dimensional value domain $[N]$, and (*iii*) a uniform query distribution.

As a first step of the attack, the adversary iterates over all $v \in [N]$ to identify candidate values for each record $\mathsf{id} \in \mathcal{I}$ by performing a matching test $f(e_{\mathsf{id}}) \stackrel{?}{=} \Pr[e_v]$. Since the query distribution is uniform, every record is guaranteed (in the limit) to have exactly

Plaintext Dimensions		QD	Number of Matches per t-test							
			t = 1	t = 2	t = 3	t = 4	t = 5	t = 6	t = 7	t = 8
	$k=1$	U	20	90	-	-	-	-	-	-
		R	14	68	-	-	-	-	-	-
		F	10,000	51,108	-	-	-	-	-	-
	$k=2$	U	76	476	1,640	3,344	-	-	-	-
		R	16	72	136	252	-	-	-	-
		F	10,240	85,202	212,542	485,204	-	-	-	-
	$k=3$	U	336	6,264	26,496	54,600	66,864	54,600	-	-
		R	22	84	144	208	256	208	-	-
		F	10,000	74,924	212,458	305,564	448,238	305,564	-	-
	$k=4$	U	336	6,264	26,496	54,600	66,864	54,600	26,496	6,264
		R	122	844	2,080	4,040	9,026	4,040	2,080	844
		F	10,000	74,924	212,458	305,564	448,238	305,564	448,238	305,564

Table 1: Number of matches under different distributions U = **Uniform**, R = **Random**, F = **Flattened**

two candidate values: its true value $v = \mathsf{DB}(\mathsf{id})$ and its "reflection" $v' = N - (v-1)$, both of which have the same probability $\Pr[e_v] = \Pr[e_{v'}]$. At this point, the attacker has two interpretations per value and approximately 2^n candidate database-reconstructions (each of which with n records) that satisfy the matching tests deployed so far. To provide perspective, if no leakage is used for reconstruction, any database among all the N^n possible ones is valid, so reducing it to 2^n is an improvement. However, the attacker can perform additional matching tests to further reduce the number of candidate databases.

A naive next step would be to use the observed frequencies of all the possible pairs of the form $\{e_{\mathsf{id}} \cap e_{\mathsf{id}'}\}$, that would be $\binom{n}{2}$ pairs in total, to form a quadratic number of matching tests. Instead, the attack presented in [26] takes a more clever approach, which reduces the number of matching tests to *linear* based on the unique setting provided by the uniform query distribution. Specifically, the attacker identifies the record, e.g., id^*, with the smallest frequency $\mathsf{id}^* = \arg\min_{\mathsf{id}} f(\mathsf{id})$, which, due to the uniform query distribution, corresponds to one with an extreme value (either the smallest or the largest). Then, the attacker commits to one of the two values for id^* that passed the matching test, e.g., v^*. The clever insight is the following: even though relying solely on tests of the form $f(e_{\mathsf{id}}) \stackrel{?}{=} \Pr[e_v]$ one cannot break the tie between values v and $v' = N - (v-1)$ for id; if we consider the tests

$$t_v = \left(\{e_{\mathsf{id}^*} \cap e_{\mathsf{id}}\}, \{e_{v^*} \cap e_v\}\right) \text{ and } \quad t_{v'} = \left(\{e_{\mathsf{id}^*} \cap e_{\mathsf{id}}\}, \{e_{v^*} \cap e_{v'}\}\right), \tag{5}$$

the attacker can infer the value of id based on how frequently id appears *together* with id^* in a response. That is, only one of the t_v and $t_{v'}$ matching tests will pass, and its subscript would be the assigned value for id. Thus, the extreme value of id^* acts as an "anchor point" to reduce the number of matching tests from quadratic to linear. That is, for every $\mathsf{id} \in \mathcal{I} \setminus \mathsf{id}^*$, their attack forms two matching tests like (5) that use the anchor point and the two tied values of id. The attacker then outputs the reconstructed database and its reflection, with the guarantee that one of them is correct.

Unfortunately, the above clever optimization for reducing matching tests cannot be generalized to arbitrary query distributions beyond uniform.

Additional Data

Effects of Dimensionality on Number of Matches. The number of matching tests across distributions and dimensions is shown in Table 1. We note that the Random distribution is relatively unaffected by change in dimensionality, but that both Uniform and Flattened distributions have higher numbers of matches as dimensionality increases. This is due to the fact that the latter distributions have a high degree of symmetry. Thus, for every tuple of records in the true database, there are multiple reflected tuples with the same frequency. Higher dimensionality leads to a greater number of these reflections, causing a greater number of matches. This means that two domains with the same number of values may require vastly different amounts of computation when attempting to reconstruct a database therein.

Comparison with Prior Results In [33], Markatou *et al.* give an algorithm that takes a response multiset $\mathsf{RM}(\mathsf{DB})$ of the true database DB and outputs a compact encoding of the set E_{DB}, which contains every database DB' where $\mathsf{RM}(\mathsf{DB}') = \mathsf{RM}(\mathsf{DB})$. They note that the adversary can then sample uniformly from E_{DB}, or use knowledge of the data distribution to "prune" E_{DB} before sampling. By comparison, our algorithm outputs a database from the set $IRD_{QD}(\mathsf{DB})$, which contains every database DB' whose response distribution RD' is the same as that of DB, i.e. RD, under the query distribution QD. We show that $E_{\mathsf{DB}} \subseteq IRD_{QD}(\mathsf{DB})$

Theorem 8.1. $E_{DB} \subseteq IRD_{QD}(DB)$

Proof. First we show that no database can be a member of E_{DB} without also being a member of $IRD_{QD}(\mathsf{DB})$. □