Response-Hiding Encrypted Ranges: Revisiting Security via Parametrized Leakage-Abuse Attacks

Evgenios M. Kornaropoulos
UC Berkeley
evgenios@berkeley.edu

Charalampos Papamanthou
University of Maryland
cpap@umd.edu

Roberto Tamassia
Brown University
rt@cs.brown.edu

***Abstract*—Despite a growing body of work on leakage-abuse attacks for encrypted databases, attacks on practical *response-hiding* constructions are yet to appear. Response-hiding constructions are superior in that they *nullify access-pattern based attacks* by revealing only the search token and the result size of each query. Response-hiding schemes are vulnerable to existing volume attacks, which are, however, based on strong assumptions such as the uniform query assumption or the dense database assumption. More crucially, these attacks only apply to schemes that cannot be deployed in practice (ones with quadratic storage and increased leakage) while practical response-hiding schemes (Demertzis et al. [SIGMOD'16] and Faber et al. [ESORICS'15]) have linear storage and less leakage. Due to these shortcomings, the value of existing volume attacks on response-hiding schemes is unclear.**

In this work, we close the aforementioned gap by introducing a parametrized leakage-abuse attack that applies to *practical response-hiding structured encryption schemes*. The use of non-parametric estimation techniques makes our attack *agnostic* to both the data and the query distribution. At the very core of our technique lies the newly defined concept of a *counting function with respect to a range scheme*. We propose a two-phase framework to approximate the counting function for any range scheme. By simply switching one counting function for another, i.e., the so-called "parameter" of our modular attack, an adversary can attack different encrypted range schemes. We propose a constrained optimization formulation for the attack algorithm that is based on the counting functions. We demonstrate the effectiveness of our leakage-abuse attack on synthetic and real-world data under various scenarios.

I. Introduction

The notion of *searchable encryption*, introduced by Song-Wagner-Perrig in [43], proposes cryptographic schemes in which a client encrypts a privacy-sensitive data collection and outsources this resulting encrypted database to a server that efficiently answers search queries without ever decrypting the database. Since then, there has been a surge of research on this subject addressing issues such as improved definitions [11], dynamic constructions [30], [44], forward and backward privacy [5], [6], [9], [12], and locality of encrypted records [3], [13], [16]. For an overview of the area, see the survey by Fuller *et al.* [20]. In this work, we are interested in the general definitional framework called *Structured Encryption* (STE) introduced by Chase and Kamara [10] and, more specifically, schemes that support encrypted range queries [8], [15], [18].

To balance efficiency and privacy, STE schemes reveal some information about the query and its corresponding response. This information is called *leakage profile*. These schemes cryptographically guarantee that nothing more is revealed beyond what the designer allowed via the leakage profile. Several works analyze how an adversary can reconstruct the plaintext data from this observed leakage. Based on the specific information revealed to the adversary when a response is returned to a database query, three main types of leakage have been investigated: *volumetric leakage* reveals the size (number of records) of the response; *access-pattern leakage* reveals identifiers (typically ciphertexts) uniquely associated with the records of the response; and *search-pattern leakage* reveals an identifier (typically output by a keyed pseudo-random function), called search token, uniquely associated with the query. Correspondingly, there are three main categories of leakage-abuse attacks: those based solely on access-pattern leakage (see, e.g., [21], [31], [32], [35]), those based on both access- and search-pattern leakage (see. e.g., [34], [38]), and those based on volumetric leakage (see, e.g., [23], [25], [31]).

Recently proposed *response-hiding schemes* [2], [28], [29] *nullify all the access-pattern based attacks* by precomputing responses to a set of canonical queries and creating a fresh copy of an encrypted record for every precomputed response that returns it. The set of canonical queries is selected at setup time in such a way that for any query q, there exists a canonical query q' such that the response to q is a subset of the response to q'. In a response-hiding scheme, the adversary can not infer whether two different responses have overlapping records, thus making reconstruction harder. Therefore, the only hope for the adversary to reconstruct the plaintext data of a response-hiding scheme is to rely on volumetric leakage. However, even though the proposed volume-based attacks [23], [25], [31] shed light on how to exploit volume under specific setups, unfortunately all of them have significant limitations which we detail below.

A. Limitations of Known Volumetric Attacks

Limitation I: Uniform Queries. The first volume-based attack was presented by Kellaris-Kollios-Nissim-O'Neal [31] and it assumes that the encrypted queries are issued uniformly at random. As mentioned in previous works, the *uniformity assumption is unrealistic* since it implies that the probability that the client queries the entire domain of the database is the same as the probability that the most popular record is queried.

Limitation II: Dense Databases. The work by Grubbs *et al.* [23] is not based on the uniformity assumption but it assumes that the database is dense, i.e., there is at least one record for every possible value of the plaintext domain. The *density assumption* can only capture heavily populated databases with small domains. Also, even in the small domain

TABLE I
COMPARISON OF OUR ATTACK WITH PREVIOUS ATTACKS FOR RANGE QUERIES ON DATABASES ENCRYPTED WITH STRUCTURED ENCRYPTION SCHEMES

Value Reconstruction Attack Algorithms	Applies to Response-Hiding Range Schemes	Applies to non-Quadratic Range Schemes	**Assumptions**			**Exploited Leakage**		
			Query Distribution	Dense Database	Known Data Distribution	Volume Leakage	Access-Pattern Leakage	Search-Pattern Leakage
KKNO [31] ACCESSPATTERNBASED	-	-	Uniform	-	-	-	●	-
LMP [35] FULLRECONSTRUCTION	-	-	Agnostic	●	-	-	●	-
GLMP [21] GENERALIZEDKKNO	-	-	Uniform	-	-	-	●	-
GLMP [21] AOR to ADR	-	-	Known	-	●	-	●	-
KPT [34] AGNOSTICRECONSTRUCTION	-	-	Agnostic	-	-	-	●	●
KKNO [31] VOLUMEBASED	●	-	Uniform	-	-	●	-	-
GLMP [23] GETELEMVOLUMES	●	-	Agnostic	●	-	●	-	-
GJW [25] EXTENDLEFTRIGHT	●	-	Agnostic	●	-	●	-	-
This Work	●	●	**Agnostic**	-	-	●	-	●

of attribute "length of hospital stay" in an experiment from [23], only 0.01% of the tested historical datasets satisfy the density assumption. The work by Gui *et al.* [25] presents several variations and improvements of the attack in [23] but all of them depend on the density assumption as well. As enlightening as the above techniques are, it is not possible to extend them to non-dense databases[1] without additional assumptions or rich auxiliary information, e.g., known data distribution.

Limitation III: Multiple Reconstructions. Given a leakage profile, there is a case that multiple plaintext databases explain the observed leakage and it is *impossible to distinguish* which one is the client's plaintext. This phenomenon was first observed by Kellaris *et al.* [31] and their proposal is to pick an *arbitrary reconstruction* among the many. Other works propose to produce *all possible reconstructions* [23], [25], or even abort and fail, but this approach is hard to follow in practice because as the size of the database grows the number of possible reconstructions might grow exponentially. Even though there is an indication that some real-world datasets have unique reconstructions, e.g., see [25], this observation is 1) based on a specific dataset and 2) based on the leakage analysis of the quadratic scheme. Deployment of STE schemes with less leakage [15], [18] does not reveal to the adversary enough structure of the plaintext and as a result it *always admits multiple reconstructions* that explain the observed leakage.

Limitation IV: Quadratic Storage. In addition to the uniformity and density assumption, all attacks on encrypted range queries have an even more crucial limitation: They only apply to schemes unlikely to be deployed in practice, i.e., those with quadratic overhead, the so-called *quadratic schemes*. At a high-level, the quadratic scheme returns the exact response for every encrypted query and as a result, the server needs to store an encrypted multimap of quadratic size. Followup works by Demertzis *et al.* [15] and Faber *et al.* [18] propose practical encrypted range constructions with *linear storage overhead* at the expense of introducing false positive responses. A fortunate byproduct of this storage efficiency is that these schemes allow only a restricted number of range queries and as a result, they have *significant less leakage than the quadratic scheme*. Neither the volumetric nor the access-pattern based attacks can be applied to the above two practical schemes.

Problem Statement. Previous research, summarized in Table I, left open the problem of whether state-of-the-art response-hiding schemes are vulnerable to leakage-abuse attacks. This work specifically addresses this open question:

"Can the adversary approximate the plaintext of a response-hiding STE *scheme without relying on unrealistic assumptions such as uniform query distribution, database density, unique database reconstruction, or scheme with quadratic overhead?"*

We answer this question in the affirmative by proposing new attacks on state-of-the-art response-hiding schemes from [15] and [18] as well as proposing a *parametrized leakage-abuse attack framework* that can be easily adjusted and applied to a wide family of current and future encrypted range STE schemes without any of the above limiting assumptions.

B. Our Contributions

Our work makes the following contributions:

1) We introduce two new notions so as to rigorously describe the generality of our parametrized attack. The first notion is a family of STE schemes that we call *regular* STE *schemes* for range queries (Definition 1 in Section IV). All the proposed STE schemes, such as the quadratic scheme and practical schemes introduced in [15], [18], can be reframed as regular STE schemes. The second notion is a function that outputs the number of canonical ranges that return a fixed response with respect to a regular scheme, called *query counting function* (Definition 2 in Section V). There is an intertwined connection between leakage-abuse attacks and query counting functions as we show in Section V. We propose a two-phase framework to rigorously approximate the query counting function of *any regular scheme* in Section VI.
2) We present in Section VII a *parametrized leakage-abuse attack* for response-hiding range schemes. This is the first attack that *applies to practical schemes* with non-quadratic storage overhead. Our attack is based on search-pattern and volume leakage, both of which are standard in response-hiding schemes. Armed with the powerful abstraction of counting functions, our technique can be easily adjusted to *attack any regular range scheme*, including the schemes from [15] and [18]. The parameter in our setup is the closed-form expression of the counting function with

[1] Suppose that each value has an associated counter that counts the number of records with this value, e.g., $(1, 0, 4, 2, 0, \ldots, 0)$ means that the there is one record with the 1st value etc. The attacks [23], [25] reconstruct the relative order of the non-zero counters, i.e., $1 \rightarrow 4 \rightarrow 2$. They can not infer how long are the "in-between zeros" and, therefore, can not recover non-dense DBs.

respect to the regular scheme under attack. Our attack is also *agnostic* to the query and data distribution, a property we achieve by using non-parametric estimation techniques. Finally, our range attack is the first one that addresses the phenomenon of multiple reconstructions by generating a set of *candidate reconstructions* and choosing one that minimizes the error on average.

3) We conduct an experimental evaluation to assess the quality of our reconstruction attack under different setups. We analyze the quality of the volumetric profile estimation under several query distributions and domain densities. We perform experiments to demonstrate how the quality of the final reconstruction is affected by (i) the volumetric profile estimation, (ii) the domain density, and (iii) the number of candidate reconstructions. As shown in our experiments in Section VIII, attacks that do not address the multiple reconstruction phenomenon can output a reconstruction with error that is up to 7× larger. We evaluate our technique in both synthetic and real-world databases and observe that in multiple setups, our technique outperforms a powerful attacker that has access to the data distribution.

II. Related Work

Attacks Based on Access-Pattern. Kellaris *et al.* [31] are the first that introduce leakage-abuse attacks for geometric queries. They exploit access-pattern leakage and assume uniform query distribution. Lacharité *et al.* [35] explore the case of a dense database and derive attacks requiring fewer queries than [31]. Grubbs *et al.* [21] present several attack scenarios that assume the query distribution is uniform or known to the adversary. They also present the AOR attack which achieves approximate *order reconstruction*, as opposed to *value reconstruction*, without any strong assumptions. Work by Markatou and Tamassia [38] assumes that the adversary observes search pattern leakage from all possible queries and presents an efficient value reconstruction method. The first attack that overcomes both the uniformity and density assumptions is the agnostic attack by Kornaropoulos *et al.* [34] which relies on both search- and access-pattern leakage. Kornaropoulos *et al.* [33] present a leakage-abuse attack on k-NN queries, which is the first attack that rigorously formulate and exploit the structure of the reconstruction space. Recent work by Falzon *et al.* [19] presents the first leakage-abuse attack on 2D range queries and proves inherent information theoretic limitations to the reconstruction. None of the above attacks apply to response-hiding range schemes.

Other Leakage-Abuse Attacks. There are two major lines of work outside the area of attacks on geometric queries, e.g., ranges and k-NN. The first line of research proposes attacks on single-keyword search under strong assumptions such as known plaintext [4], [7], [26] or the ability of the attacker to inject carefully crafted inputs in the plaintext data of the client [4], [45], [46]. The majority of these attacks focus on recovering the privacy-sensitive query of the client as opposed to recovering the plaintext data of the encrypted database. The other line of research proposes attacks on property-preserving encryption schemes [17], [24], [40] which are cryptographic constructions that have significantly more leakage than STE schemes. None of the above attacks apply to response-hiding range schemes.

Mitigations. Another interesting line of work [28], [42] mitigates the volume leakage by always returning the maximum number of records among all possible queries, denoted with l. The goal of these techniques is optimizing the storage efficiency while always returning the maximum number of records $O(l)$. Unfortunately, these mitigations are designed with the single-keyword search in mind and can not be applied to range queries due to the fact that the maximum number of records for the case of range queries is the entire database. Thus, applying them would incur $O(n)$ communication complexity per query. Several mitigation techniques for access pattern leakage from range queries are used in [39], including batching queries and issuing fictitious queries to introduce noise. Their mitigations do not apply to response-hiding schemes which is the focus of this paper. A recent defense method based on frequency-smoothing [22] is designed for encrypted key-value stores and does not address the mitigation of leakage attacks on encrypted ranges, e.g. [31], [34]. In concurrent work, Demertzis *et al.* [14] present SEAL, a framework for encrypted databases with improved security via a *light* use of ORAM and padding. It is open whether our attack applies in such modified settings.

ORAM. Other related work investigates the limits of the efficiency of Oblivious RAM (ORAM) a much stronger primitive than structured encryption STE. A series of strong lower bounds for ORAM by Larsen *et al.* [37] as well as Oblivious Data Structures by Jacob *et al.* [27] and Oblivious k-NN by Larsen *et al.* [36] shows that it is not possible to achieve stronger access-pattern privacy than STE with the same efficiency. Recent work by Patel *et al.* [41] shows that even hiding part of the search-pattern leakage of encrypted multimaps incurs an $\Omega(\log n)$ lower bound.

III. Preliminaries

In the context of this paper, a *database DB* is a collection of n records $(id_i, val(id_i))$, $i \in [1, n]$, where id_i is a unique identifier and $val(id_i)$, or simply v_i, is a value from the universe $[\alpha, \beta]$ for given constants α and β. We assume that values $v_1, \ldots, v_n$ are sorted in nondecreasing order, i.e., $v_i \leq v_{i+1}$. The term $d(v, v') = |v' - v|$ denotes the distance between two values. We assume integer values and denote with $N = \beta - \alpha + 1$ the size of the plaintext universe. We define the *length* L_i between two consecutive values as $L_i = d(v_{i-1}, v_i)$, $\forall i \in [2, n]$. For the two extreme cases we define L_1, L_{n+1} as $L_1 = d(\alpha - 1, v_1)$ and $L_{n+1} = d(v_n, \beta + 1)$. We define as *domain density* of the database the percentage of unique values from the universe that are assigned to records. A range query consists of two values $x \leq y$ and its response is the set of identifiers of DB with values within interval $[x, y]$. We define as *span* of a query $[x, y]$ the number of values covered by the range, i.e., $y - x + 1$. In a structured encryption scheme (STE) for DB, we use the term *query* to refer to the plaintext query and the term *search token* to refer to the encrypted object that the client sends to the server to query the encrypted multimap (EMM) of the STE scheme. We define *access-pattern leakage* as the set of encrypted records that are retrieved as part of the response to a token. We define *search-pattern leakage* the server's ability to observe whether two tokens were generated from the same plaintext query. To the best of our knowledge, all STE schemes leak the search-pattern [20].

The *response-hiding* design for an STE [2], [28], [29] hides overlaps between different queries and reveals only the size of the answer of each query, or an upper bound on the size. At setup, a response-hiding scheme selects a set of *canonical queries*, precomputes the corresponding responses, and freshly encrypts the records in such responses. Given a client query, a precomputed response for a canonical query whose range includes that of the client query is returned, which may result in *false positives* (records in the answer to the canonical query but not to the client query) that must be filtered out by the client. None of the published access-pattern based attacks (e.g., [21], [31], [34], [35]) can be applied to response-hiding schemes.

In the selection of canonical queries for which responses are precomputed, a response-hiding designer faces a trade-off between two types of performance drawbacks: (1) *space overhead* due to storing multiple encryptions of the same record; and (2) *communication overhead* due to false positives. The quadratic scheme selects all possible responses to queries as canonical ones. Thus, it incurs $O(n^2)$ space overhead, which is impractical, but no communication overhead. Conversely, there are schemes that select $O(n)$ canonical queries [15], [18] and have $O(n)$ space overhead at the expense of doubling the span of the original query in the worst case.

Threat Model. In this work we consider the threat model where the adversary is the honest-but-curious server that stores the encrypted database and observes a series of encrypted range queries issued by the client. In this setting the attacker has no knowledge about the query distribution, data distribution, or access to any auxiliary information about them. The attacker can not issue any queries or inject/remove plaintext data. The goal of the attacker is to *reconstruct* the plaintext values stored in the database using the query leakage that stems from the response-hiding structured encryption scheme for range queries. We elaborate on the assumptions of our attack in the following.

Assumptions. Our assumptions are as follows:

- *Static Database.* No updates, i.e., addition, deletions, take place once the database is encrypted.
- *Fixed Query Distribution.* We assume that the adversary issues independent and identically distributed (i.i.d.) queries with respect to a fixed query distribution. We emphasize that our adversary does not know any information about the family or the parameters of the query distribution.
- *One-dimensional Data Values.* We do not address encrypted databases for high-dimensional data.
- *Known Setup.* We assume that the adversary knows the number of encrypted values n, the size of the universe of values N and the endpoints of the universe α, β.
- *Response-Hiding Scheme.* We assume that the client deployed a response-hiding scheme so as to protect against known access-pattern based leakage-abuse attacks.
- *Practical Structured Encryption Scheme.* We assume that the client deployed a practical STE scheme, e.g., see [15], [18], that does not impose a quadratic storage overhead. Our attacks apply to any scheme that allows a *restricted number of range queries* to be queried, we denote this family of schemes as *regular schemes* and we formally define them in Section IV.

Roadmap. Our proposed leakage-abuse attack applies to an entire family of range schemes inspired by the design of [15], [18]. We introduce this family in Section IV under the name *regular range schemes*. Section V sets the foundations of our parameterized leakage-abuse attack by introducing a powerful abstraction based on the new notion of *counting functions*. Armed with the above abstraction, Section VI presents the technical results for counting functions that allows us to simply switch between different counting functions and apply the same technique to *any* regular scheme. Finally, Section VII uses the above results together with (A) a new application of non-parametric estimators (Section VII-A) and (B) a constrained optimization formulation (Section VII-B) to assemble the final leakage-abuse attack.

IV. Regular Range Schemes

Our leakage analysis is motivated by the state-of-the-art schemes for encrypted range queries BT (binary tree) by Faber *et al.* [18] and ABT (augmented binary tree) by Demertzis *et al.* [15]. The response-hiding adaptation of these two schemes is not vulnerable to any leakage-abuse attack from the literature and thus are considered to be secure. In this work, we do not tailor our analysis to the specific leakage of these schemes but rather introduce a *general framework of leakage analysis for all range schemes that follow the same design principles*.

Scheme BT can be seen as a full binary tree over the domain of plaintexts where every node represents a range query that spans a consecutive subset of leaves. Figure 1 shows a binary tree where each node of the tree is denoted with a gray interval. Scheme ABT is a binary tree augmented with nodes that are placed in-between two consecutive internal nodes, e.g., see the red intervals in Figure 1 for the additional nodes.

A common characteristic of the BT and ABT schemes is that they *do not store all possible range queries* in the encrypted multimap, i.e., EMM. Instead, at setup time, the scheme generates *a subset of range queries* of span 2^j for $j \leq \log(N)$ and stores their encrypted answers. Specifically, for a given span, the EMM stores a "regular" progression of ranges spaced from each other by an *additive step*. E.g., for span 2^2 and step 2 we have ranges $[1, 4], [3, 6], [5, 8], \ldots$. The stored ranges and their responses are later used to answer an arbitrary user query by selecting a stored range whose span covers the span of the user query and returning the answer to such a query. Note that some instances of the scheme have an inherent communication overhead as they may return additional elements not present in the range queried by the user, i.e., false positives. Such extra elements can be easily filtered out by the user. We capture and generalize the "regularity" property of the above schemes with the notion of a (T, f)-*regular scheme* in Definition 1.

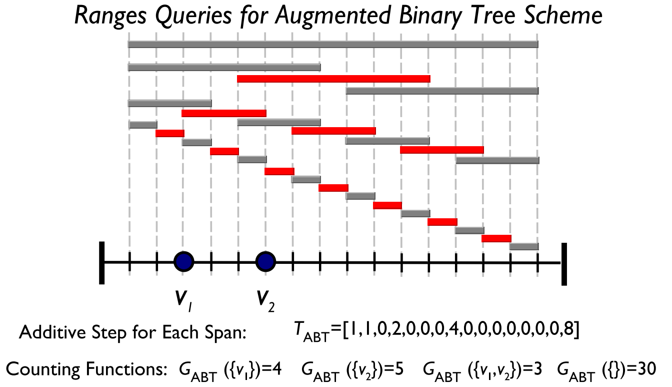
Ranges Queries for Augmented Binary Tree Scheme
V1
V2
Additive Step for Each Span: T_ABT=[1,1,0,2,0,0,0,4,0,0,0,0,0,0,0,8]
Counting Functions: G_ABT ({v1})=4 G_ABT ({v2})=5 G_ABT ({v1,v2})=3 G_ABT ({})=30

Fig. 1. Canonical range queries for schemes BT (gray intervals) and ABT (gray & red intervals) for $N = 16$. In both schemes, canonical ranges have spans that are powers of two. Also shown is a *DB* with values, $v_1 = 3$ and $v_2 = 6$. In ABT, a client query for range $[1, 5]$ returns response $\{v_1, v_2\}$ for canonical range $[1, 8]$. This response includes v_2 as a false positive.

Defining Regular Schemes. We consider a broad class of STE schemes where the *choice of stored ranges is deterministic and data-independent*. These schemes are parameterized by (i) a sequence of spans and (ii) associated step values. The scheme precomputes and stores in encrypted multimap EMM the answers to a set of queries that depends only on the above parameters and the size of the database universe, N.

Definition 1. *A* ***regular structured encryption scheme*** *for range queries over a database with universe size N comprises the following components:*

- *A sequence T with N nonnegative integer entries, denoted in vector notation as $T = (T[1], T[2], \ldots, T[N])$, where $T[s]$ denotes the step for ranges of span s.*
- *An encrypted multimap* EMM *precomputes and stores responses to canonical queries with ranges*
$$[k \cdot T[s] + 1, k \cdot T[s] + s]$$
for $s = 1, \ldots, N$, $T[s] > 0$, and $k = 0, \ldots, \left\lfloor \frac{N-s}{T[s]} \right\rfloor$.
- *A function f mapping an arbitrary database range query to a canonical range stored in* EMM.

Such a scheme is also referred to as ***(T, f)-regular*** *or, when function f is clear from the context,* ***T-regular***. *We call* ***weight*** *of T the number of positive entries denoted as* ***$weight(T)$***.

Schemes with sublinear weight are considered practical. Typically, function f maps a client query q to the canonical query of shortest length whose span covers q.

Remark 1. *The quadratic scheme, QD, that stores all possible ranges and the more efficient schemes BT [18] and ABT [15] are examples of (T, f)-regular schemes:*

- *Scheme QD has $T[s] = 1$ for $s = 1, \ldots, N$, hence $weight(T) = N$.*
- *Scheme BT has $T[s] = s$ when s is a power of 2 and $T[s] = 0$ otherwise, hence $weight(T)$ is $O(\log N)$.*
- *Scheme ABT has $T[1] = 1$, $T[s] = s/2$ when s is a power of 2 and $s > 1$, and $T[s] = 0$ otherwise, hence $weight(T)$ is $O(\log N)$.*

In all three schemes, f maps query $[\alpha, \beta]$ to the shortest stored query covering interval $[\alpha, \beta]$.

An illustration of the canonical queries of schemes BT and ABT is shown in Figure 1. We now introduce a new scheme called BASE for range queries as an *intermediate step for our analysis* of schemes BT [18] and ABT [15]. Like these two schemes, BASE only considers spans that are powers of two but it uses a *step equal to one* for all such spans.

Remark 2. BASE *is a (T, f)-regular scheme such that*

- *$T[s] = 1$ when s is a power of 2 and $T[s] = 0$ otherwise, hence $weight(T)$ is $O(\log N)$.*
- *f maps range $[x, y]$ to the shortest canonical query range covering $[x, y]$. In case of a tie, it maps to the range that starts at x if it exists, else to the range that ends at y.*

We note that in schemes BT and ABT, there is a unique canonical range that covers a given range $[\alpha, \beta]$ and has the shortest span, hence the simple definition of function f in Remark 1. On the contrary, in scheme BASE, there can be multiple canonical ranges with shortest span that cover $[\alpha, \beta]$, hence the need for the tie-breaking rule in the definition of function f in Remark 2.

V. Leakage Attacks from Counting Functions

In this section we formalize the notion of *query counting function*, or, simply, *counting function*, and show how it was used without being formalized in previous works to develop reconstruction attacks against the quadratic scheme QD [31], [34]. In our work, counting functions serve as a *powerful abstraction* that enables a parametrized framework for attacks by disentangling the derivation of counting functions (Section VI) from the architecture of the attack that uses counting functions as a blackbox (Section VII).

At a high-level, the counting function $C(r, s)$ outputs the number of canonical range queries of span s that return response r, e.g., in Figure 1 we have $C_{\text{ABT}}(\{v_1\}, 2) = 2$ due to queries $[2, 3], [3, 4]$. The global counting function $G(r)$ outputs the number of *all* canonical range queries that return response r, e.g., in Figure 1 we have $G_{\text{ABT}}(\{v_1\}) = 4$ due to queries $[3, 3], [2, 3], [3, 4], [1, 4]$. The outputs of the global counting function for Figure 1 are $G_{\text{ABT}}(\emptyset) = 30$, $G_{\text{ABT}}(\{v_1\}) = 4$, $G_{\text{ABT}}(\{v_2\}) = 5$, and $G_{\text{ABT}}(\{v_1, v_2\}) = 3$.

Definition 2. *Let* ANY *be a regular structured encryption scheme for range queries over a database with universe size N. The* ***query counting function of scheme*** **ANY**, *denoted $C_{\text{ANY}}(r, s)$, takes as input a response r to a query on the database (i.e., a sequence of consecutive values) and a span s, and outputs the number of canonical queries of* ANY *of span s whose response is r. The* ***global query counting function of scheme*** **ANY**, *denoted $G_{\text{ANY}}(r)$, takes as input a response r and outputs the number of canonical queries of* ANY *(*of any span*) whose response is r. We have:*
$$G_{\text{ANY}}(r) = \sum\nolimits_{s=1}^{N} C_{\text{ANY}}(r, s).$$

Note the in the above definition, the canonical queries contributing to the count must have a response equal exactly to r, i.e., yielding no false positives.

We start with the straightforward counting function of the quadratic scheme QD which is the target of all previous leakage-abuse attacks [21], [23], [31], [34], [35].

Counting for the Quadratic Scheme. The quadratic response-hiding scheme QD has the largest storage overhead, i.e., quadratic space, because the set of canonical range queries comprises all possible $\binom{N}{2} + N$ ranges. Since the scheme pre-computes and stores the response for every possible client query, the scheme returns no false positives and has no communication overhead. Recalling the definition of length between two consecutive database values, $L_i = d(v_{i-1}, v_i)$, an interesting property of the quadratic scheme is that the number of queries that return a specific response, e.g., $r = \{v_2, v_3, v_4, v_5\}$, depends only on the lengths between (1) the smallest value of r and its previous value, e.g., $L_2 = d(v_1, v_2)$, and (2) the largest value of r and its next value, e.g., $L_6 = d(v_5, v_6)$. More formally, the global query counting function for QD is:

$$G_{\mathrm{QD}}\left(\{v_i, \ldots, v_{i+k}\}\right) = L_i \cdot L_{i+k+1}.$$

There are two main factors that make the quadratic scheme QD a convenient option for leakage-abuse attacks. First, *the counting function is simple*, it depends only on two lengths. Second, overall QD *leaks significantly more information*, i.e., from $O(N^2)$ canonical ranges, than practical schemes [15], [18], i.e., from $O(N)$ canonical ranges. Thus, the leakage of QD reveals more about the geometric structure of the plaintexts.

Abstraction of Attacks via Counting Functions. The volumetric attack for QD [31] (unknowingly) uses the notion of counting functions. Let θ_i be the number of all queries of QD that return a response of volume i. We define as the *volumetric profile* of QD the vector $(\theta_0, \ldots, \theta_n)$. The notion of volumetric profile can be extended to other schemes, e.g., BT and ABT, where each entry θ_i counts the number of distinct *canonical* queries with volume i. Kellaris *et al.* [31] define and solve the system of equations on the left hand-side of the next figure. We can abstract their approach by swapping each product of lengths with the corresponding counting function.

EQUATIONS FROM [31]	COUNTING FUNCTION ABSTRACTION
$\sum_{i=1}^{n} L_i \cdot L_{i+1} = \theta_1$	$\sum_{i=1}^{n} G_{\mathrm{QD}}\left(\{v_i\}\right) = \theta_1$
$\sum_{i=1}^{n-1} L_i \cdot L_{i+2} = \theta_2$	$\sum_{i=1}^{n-1} G_{\mathrm{QD}}\left(\{v_i, v_{i+1}\}\right) = \theta_2$
$\vdots$	$\vdots$
$L_1 \cdot L_{n+1} = \theta_n$	$G_{\mathrm{QD}}\left(\{v_1, \ldots, v_n\}\right) = \theta_n$

Given the above abstraction one can simply plug in the counting function of a *different regular scheme* and derive a reconstruction by solving the system of equations. Even though this approach works in theory, in practice there are some important challenges to overcome. In particular: (i) There is no known closed-formula for the counting functions of practical range schemes [15], [18]; (ii) It is not realistic to assume our attacker has access to the exact values of θ_i; (iii) The counting functions might have a cumbersome expression that does not allow an analytical solution to the system of equations. We address all of the above as follows: (i) We introduce counting functions for [15], [18] in Section VI; (ii) We estimate θ_i based on search-pattern leakage in Section VII-A; (iii) We introduce an optimization formulation that can be used to approximate the solution of the system in Section VII-B.

VI. Counting for Practical Regular Schemes

The next challenge for our leakage analysis is to answer the following question about counting functions:

How many canonical range queries *of a regular scheme return a given response?*

In order to perform a leakage analysis on the BT and ABT schemes [15], [18] we develop *new insights* about the counting functions of *practical* regular schemes. We present a general approach in this section which, can be used to understand the vulnerability of current and future regular range schemes.

A Two-Phase Framework for Leakage Analysis. We follow a two-phase approach in our analysis. In the first phase, in Section VI-A, we derive *exact* formulas for the counting functions of T-regular schemes with a unit step in their canonical ranges, i.e., $T[s] \leq 1$ for each span s. Example of such schemes are the QD and the BASE scheme. In the second phase, in Section VI-B, we use the results of the first phase to *approximate* the counting functions of T-regular schemes where $T[s]$ takes arbitrary nonzero values.

Specifically, the approximation of counting functions for a general T-regular scheme, Any, where *not all the values* of T_{ANY} are 0 or 1, is obtained as follows. We build from Any a modified scheme, Step1-Any, by replacing each nonzero step of T_{ANY} with step 1. Next, we derive closed-formulas for the counting functions of Step1-Any and we approximate the counting functions of Any from the corresponding functions of Step1-Any. Note that even though in this work we are primarily interested in spans that are powers of two, as they are used in practical response-hiding schemes [15], [18], one can apply our two-phase framework to *any regular scheme*.

A. Exact Counting for Regular Schemes with Step One

The intuition of our approach is described using the Base scheme but the lemmas and theorems are written in their general form, i.e., for any regular scheme Step1-Any with 0/1 entries in T. It is clear that for $r = \{v_i, \ldots, v_{i+k}\}$ in case the span 2^j is smaller than the distance $d(v_i, v_{i+k}) = \sum_{t=1}^{k} L_{i+t}$, then no query of span 2^j can return r. Similarly, the span must be less than $d(v_{i-1}, v_{i+k+1})$. Thus, we only consider spans 2^j s.t.:

$$\left\lceil \log\left(\sum_{t=1}^{k} L_{i+t}\right)\right\rceil \leq j \leq \left\lfloor \log\left(\sum_{t=0}^{k+1} L_{i+t}\right)\right\rfloor.$$

We illustrate the intuition with a running example where $r = \{v_2, v_3\}$ and $s = 2^3$, see Figure 2. Since we are interested in counting how many ranges of span s return r, we can ignore all ranges that touch locations before v_1 and after v_4. The goal, in this toy example, is to count how many times an interval of span $s = 2^3$ can be "displaced" in the area from v_1 to v_4 while satisfying the requirement that it covers exactly r. Depending on the instantiation of the underlying distances L_2, L_3, L_4 we have *four distinct cases* for the formula of the counting function C_{BASE}. The following lemmas provide a sufficient condition for each of the four distinct formulas of the counting function. For compactness of our formulas, we use the empty sum property

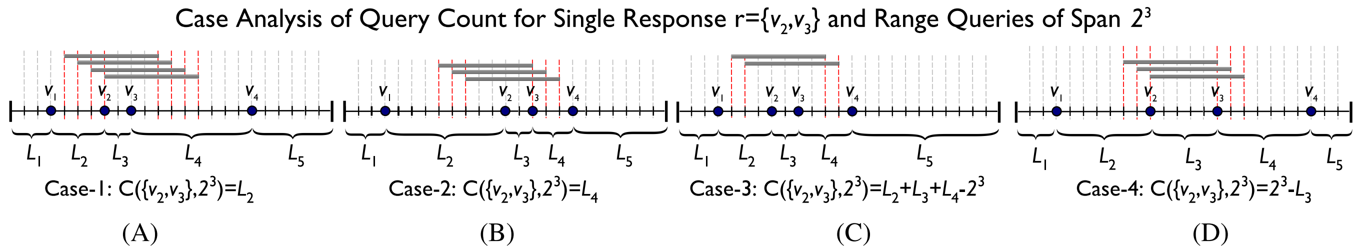
Case Analysis of Query Count for Single Response r={v2,v3} and Range Queries of Span 2^3
v1
v2
v3
v4
L1
L2
L3
L4
L5
Case-1: C({v2,v3},2^3)=L2
Case-2: C({v2,v3},2^3)=L4
Case-3: C({v2,v3},2^3)=L2+L3+L4-2^3
Case-4: C({v2,v3},2^3)=2^3-L3
(A)
(B)
(C)
(D)

Fig. 2. Illustration of the four distinct cases for the counting function C with input the response $r = \{v_2, v_3\}$ and the span $s = 2^3$. In case-1 the grey interval of span s iterates through the entire leftmost length L_2 of response r. In case-2 the interval iterates through the rightmost length L_4. In case-3 the grey interval is confined by the two neighboring values v_1 and v_4, i.e., the distance from v_1 to v_4 is not large enough to iterate through the neither the leftmost nor the rightmost length. In case-4 the interval confined by the extreme values of r, i.e., the span is not large enough to iterate through neither L_2 nor L_4 length.

where for $x > y$ we have $\sum_{i=x}^{y} f(i) = 0$. The proofs can be found in the Appendix.

Case-1: Iterating the Leftmost Length. The lower-boundary of the range of fixed span *iterates through all* the possible lower-boundary locations, e.g., locations of L_2 in Figure 2-(A).

Lemma 1. *Let* STEP1-ANY *be a T-regular scheme where the entries of T are 0 or 1. Let s be a span such that $T[s]$=1 and $r = \{v_i, \ldots, v_{i+k}\}$ be a response ($i \in [1, n]$, $k \in [0, n]$). If :*

$$\sum_{t=0}^{k} L_{i+t} \leq s < \sum_{t=1}^{k+1} L_{i+t},$$

then the counting function of scheme STEP1-ANY *for span s and response r is $C_{\text{STEP1-ANY}}(r, s) = L_i$.*

Case-2: Iterating the Rightmost Length. The upper-boundary of the range *iterates through all* the possible upper-boundary locations, e.g., locations of L_4 in Figure 2-(B).

Lemma 2. *Let* STEP1-ANY *be a T-regular scheme where the entries of T are 0 or 1. Let s be a span such that $T[s]$=1 and $r = \{v_i, \ldots, v_{i+k}\}$ be a response ($i \in [1, n]$, $k \in [0, n]$). If :*

$$\sum_{t=1}^{k+1} L_{i+t} \leq s < \sum_{t=0}^{k} L_{i+t},$$

then the counting function of scheme STEP1-ANY *for span s and response r is $C_{\text{STEP1-ANY}}(r, s) = L_{i+k+1}$.*

Case-3: Confined by the Neighboring Values of r. The boundaries *can not iterate through either the leftmost or the rightmost length* because they "bump" on the neighboring plaintexts that are not in r, e.g., see v_1, v_4 in Figure 2-(C).

Lemma 3. *Let* STEP1-ANY *be a T-regular scheme where the entries of T are 0 or 1. Let s be a span such that $T[s]$=1 and $r = \{v_i, \ldots, v_{i+k}\}$ be a response ($i \in [1, n]$, $k \in [0, n]$). If :*

$$\max\left\{\sum_{t=0}^{k} L_{i+t}, \sum_{t=1}^{k+1} L_{i+t}\right\} \leq s < \sum_{t=0}^{k+1} L_{i+t},$$

then the counting function of scheme STEP1-ANY *for span s and response r is $C_{\text{STEP1-ANY}}(r, s) = \sum_{t=0}^{k+1} L_{i+t} - s$.*

Case-4: Confined by the Values of r. The boundaries of the range *can not iterate through either the leftmost or the rightmost length* because they "bump" on the extreme plaintext values that define r, see v_2, v_3 in Figure 2-(D).

Lemma 4. *Let* STEP1-ANY *be a T-regular scheme where the entries of T are 0 or 1. Let s be a span such that $T[s]$=1 and $r = \{v_i, \ldots, v_{i+k}\}$ be a response ($i \in [1, n]$, $k \in [0, n]$). If :*

$$\sum_{t=1}^{k} L_{i+t} < s < \min\left\{\sum_{t=0}^{k} L_{i+t}, \sum_{t=1}^{k+1} L_{i+t}\right\},$$

then the counting function of scheme STEP1-ANY *for span s and response r is $C_{\text{STEP1-ANY}}(r, s) = s - \sum_{t=1}^{k} L_{i+t}$.*

How to Overcome Case Analysis. Given the above four lemmas the attacker may want to check the relation between the values of s and the corresponding lengths $L_i, \ldots, L_{i+k+1}$ and decide which is the correct counting function among the four cases. The problem is that *the attacker does not know the values of the lengths $L_i, \ldots, L_{i+k+1}$* because they are derived from the unknown plaintexts, therefore there is no way to know which one of the four cases applies. The next theorem overcomes the case analysis by presenting an expression that *provides the correct counting function regardless of the case.*

Theorem 1. *Let* STEP1-ANY *be a T-regular scheme where the entries of T are 0 or 1. Let s be a span such that $T[s] = 1$ and let $r = \{v_i, \ldots, v_{i+k}\}$ be a response ($i \in [1, n]$, $k \in [0, n]$). The counting function $C_{\text{STEP1-ANY}}(r, s)$ is defined as*

$$\min\left\{L_i, L_{i+k+1}, \sum_{t=0}^{k+1} L_{i+t} - s, s - \sum_{t=1}^{k} L_{i+t}\right\},$$

whenever $\sum_{t=1}^{k} L_{i+t} < s < \sum_{t=0}^{k+1} L_{i+t}$ and is 0 otherwise.

The above two-case expression for $C_{\text{STEP1-ANY}}(r, s)$ can be reframed as a single closed-form expression:

$$\max\left\{0, \min\left\{L_i, L_{i+k+1}, \sum_{t=0}^{k+1} L_{i+t} - s, s - \sum_{t=1}^{k} L_{i+t}\right\}\right\},$$

since whenever $s \notin (\sum_{t=1}^{k} L_{i+t}, \sum_{t=0}^{k+1} L_{i+t})$ the min expression in the theorem is a negative number and therefore the new max/min formula holds for arbitrary spans s.

Corollary 1. *Let* STEP1-ANY *be a T-regular scheme where the entries of T are 0 or 1. Let r be a response $r = \{v_i, \ldots, v_{i+k}\}$, where $i \in [1, n]$, $k \in [0, n]$. The global query counting function $G_{\text{STEP1-ANY}}(r)$ is given by*

$$G_{\text{STEP1-ANY}}(r) = \sum_{s=1}^{N} C_{\text{STEP1-ANY}}(r, s).$$

Remark 3. *Theorem 1 and Corollary 1 hold for scheme* BASE.

B. Approximate Counting for All Regular Schemes

Having established the first phase of our framework, where we derive exact formulas for counting functions of regular schemes with step one, we now show how to use this result to *approximate* the schemes with step values greater than one.

Let ANY be any T_{ANY}-regular scheme for which the entries of T_{ANY} can have step values greater than 1. Let STEP1-ANY be a $T_{\text{STEP1-ANY}}$-regular scheme for which $T_{\text{STEP1-ANY}}$ has value 1 in all positions that T_{ANY} has a nonzero step, and step 0 elsewhere. Denoting with $\lfloor \cdot \rceil$ the rounding operation, we propose to *approximate* the counting functions as:

$$C_{\text{ANY}}(r,s) \approx \widetilde{C}_{\text{ANY}}(r,s) = \left\lfloor \frac{C_{\text{STEP1-ANY}}(r,s)}{T_{\text{ANY}}[s]} \right\rceil, \quad (1)$$

for $s \in \{1,\ldots,N\}$, $T_{\text{ANY}}[s] > 0$. Similarly, the global counting function can be approximated as:

$$G_{\text{ANY}}(r) \approx \widetilde{G}_{\text{ANY}}(r) = \sum_{\substack{s\in\{1,\ldots,N\}\\ T_{\text{ANY}}[s]>0}} \left\lfloor \frac{C_{\text{STEP1-ANY}}(r,s)}{T_{\text{ANY}}[s]} \right\rceil. \quad (2)$$

The theorem below provides rigorous guarantees for the approximations of the query counting functions given by Equations 1–2 for a general regular scheme.

Theorem 2. *Let* ANY *be a regular response-hiding structured encryption scheme. The approximations of the counting functions of* ANY *given by Equations 1–2 are bounded as follows:*

$$\left| C_{\text{ANY}}(r,s) - \widetilde{C}_{\text{ANY}}(r,s) \right| \leq 1, \textit{ for } s \geq 1,\ T_{\text{ANY}}[s] > 0$$

$$\left| G_{\text{ANY}}(r) - \widetilde{G}_{\text{ANY}}(r) \right| \leq \textit{weight}(T_{\text{ANY}}).$$

The approximation guarantees of Theorem 2 hold for any regular scheme. However, they are especially meaningful for schemes like BT and ABT that achieve efficient storage overhead by using a linear number of canonical ranges and allowing for false positives in the query answers.

Corollary 2. *Given a database with universe size N, Equations 1–2 yield approximations of the global query counting functions for the response-hiding scheme BT [18] and* ABT *[15] bounded by* $\log N$.

The above exposition focuses on non-empty responses. For the case of empty responses, i.e., volume is equal to 0, the formula is presented as the first term of the loss function in Figure 3. On a high-level, for a fixed span s and the case where the entries of T are 0 or 1 the counting function has the form $\sum_{i=1}^{n+1} L_i - s$. In case T has entries larger than 1 we approximate by dividing with the corresponding additive step.

VII. Parametrized Leakage-Abuse Attacks

Overview of the Attack. The first building block of the attack is presented in Section VII-A where we show how to apply non-parametric estimation techniques, similar to [34], so as to estimate the volumetric profile of practical response-hiding schemes without knowledge of the query or data distributions. The second building block is presented in Section VII-B where we use the closed-formulas of counting functions derived in Section VI, as well as the estimation results from Section VII-A, to formulate an optimization problem that outputs a reconstruction matching the estimated volumetric profile. The next building block of our attack, presented in Section VII-C, proposes a strategy to output a solution that takes advantage of the structure of the reconstruction space. To achieve this, we sample the reconstruction space by repeatedly solving the proposed optimization problem and picking the most "central" reconstruction among the observed ones. Finally, Section VII-D combines the above components into our overall attack method, which we call a "parametrized attack" because it applies to *any regular response-hiding STE scheme*, where the parameter is the counting function of the scheme. By simply substituting one counting function for another, an adversary can attack a variety of response-hiding STE schemes.

A. Estimating the Number of (Unseen) Queries of Fixed Volume

Our approach is inspired by the techniques introduced by Kornaropoulos *et al.* [34]. Their attacks [34] are designed for schemes where every query of the client reveals the pair (x, r) where x is the search token and r is the response set of identifiers. By defining random variable X whose values are all possible tokens of the scheme and random variable R whose values are all the possible responses of a range query with respect to DB, the authors of [34] observe that the pair (x, r) can be seen as a *sample from the conditional probability distribution* $p_{\mathsf{X}|\mathsf{R}}(\mathsf{X} = x|\mathsf{R} = r)$. Given a multiset of token-response pairs, the attack from [34] partitions the multiset *with respect to response* r and applies a support-size estimation technique on each partition set. The output of this process is a collection of estimations, each of which estimates how many tokens exist that return response r. The generality of the approach by Kornaropoulos *et al.* [34] comes from the fact that the estimation techniques are non-parametric and as a result make *no assumption about the query or data distribution.*

In this work we put forth the application of support-size estimation techniques for the *estimation of the number of unseen tokens that return a fixed volume*. Let X be the random variable whose possible values are the search tokens of a response-hiding scheme and let Z be a random variable whose values come from the set $[0,\ldots,n]$ and describe the volume of an issued query. Then, the pair (x, z) can be seen as a *sample from the conditional probability distribution* $p_{\mathsf{X}|\mathsf{Z}}(\mathsf{X} = x|\mathsf{Z} = z)$. Therefore, the estimation of the support size of $p_{\mathsf{X}|\mathsf{Z}}(\mathsf{X} = x|\mathsf{Z} = z)$ translates to an estimation of the total number of queries θ_z with response of volume z.

Adjusting the Estimations. Our new technique outlined above derives the estimation of the support-size of each conditional probability distribution independently *without considering the result of the other estimations.* Let $Q_{\text{ANY}}(N)$ denote the total number of distinct canonical queries for scheme ANY. A desired property that is overlooked so far is that the *sum of all the estimated support-sizes must be equal to the total number of canonical ranges* of the scheme, i.e., $Q_{\text{ANY}}(N) = \sum_{i=0}^{n} \theta_i$. Notice that the total number of range queries for a regular scheme can be computed from the vector T and the size of the domain N. Let us assume for simplicity that N is a power of two.

Loss Functions for Attacks on Response-Hiding Range Schemes BASE, BT, ABT:

$$\textsc{Loss}_{\textsc{Base}}\left(\{L_i\}_{i=1}^{n+1}\right) = w_0 \left(\left(\sum_{i=1}^{n+1} \sum_{l=0}^{\lfloor \log(N) \rfloor} \max\left\{0, L_i - 2^l\right\} \right) - \hat{\theta}_0 \right)^2 + \sum_{k=0}^{n-1} w_{k+1} \left(\left(\sum_{i=1}^{n-k} \sum_{l=0}^{\lfloor \log(N) \rfloor} \max\left\{0, \min\left\{L_i, L_{i+k+1}, \sum_{t=0}^{k+1} L_{i+t} - 2^l, 2^l - \sum_{t=1}^{k} L_{i+t}\right\}\right\} \right) - \hat{\theta}_{k+1} \right)^2$$

$$\textsc{Loss}_{\textsc{BT}}\left(\{L_i\}_{i=1}^{n+1}\right) = w_0 \left(\left(\sum_{i=1}^{n+1} \sum_{l=0}^{\lfloor \log(N) \rfloor} \max\left\{0, \left\lfloor \frac{L_i - 2^l}{2^l} \right\rceil \right\} \right) - \hat{\theta}_0 \right)^2 + \sum_{k=0}^{n-1} w_{k+1} \left(\left(\sum_{i=1}^{n-k} \sum_{l=0}^{\lfloor \log(N) \rfloor} \max\left\{0, \left\lfloor \frac{\min\left\{L_i, L_{i+k+1}, \sum_{t=0}^{k+1} L_{i+t} - 2^l, 2^l - \sum_{t=1}^{k} L_{i+t}\right\}}{2^l} \right\rceil \right\} \right) - \hat{\theta}_{k+1} \right)^2$$

$$\textsc{Loss}_{\textsc{ABT}}\left(\{L_i\}_{i=1}^{n+1}\right) = w_0 \left(\left(\sum_{i=1}^{n+1} \sum_{l=0}^{\lfloor \log(N) \rfloor} \max\left\{0, \left\lfloor \frac{L_i - 2^l}{\max\{1, 2^{l-1}\}} \right\rceil \right\} \right) - \hat{\theta}_0 \right)^2 + \sum_{k=0}^{n-1} w_{k+1} \left(\left(\sum_{i=1}^{n-k} \sum_{l=0}^{\lfloor \log(N) \rfloor} \max\left\{0, \left\lfloor \frac{\min\left\{L_i, L_{i+k+1}, \sum_{t=0}^{k+1} L_{i+t} - 2^l, 2^l - \sum_{t=1}^{k} L_{i+t}\right\}}{\max\{1, 2^{l-1}\}} \right\rceil \right\} \right) - \hat{\theta}_{k+1} \right)^2$$

Fig. 3. The loss functions for attacking the response-hiding schemes BASE, ABT, and BT. The terms $\{\hat{\theta}_i\}_{i=0}^n, \{w_i\}_{i=0}^n$ are initialized in the volumetric profile estimation phase of the attack and are considered constants when solving a minimization problem with one of the above loss functions.

For the three schemes we have:

$$Q_{\textsc{Base}}(N) = \sum\nolimits_{i=0}^{\log N} (N - 2^i + 1)$$
$$Q_{\text{ABT}}(N) = 2(2N-1) - \log N - N,\ Q_{\text{BT}}(N) = 2N - 1. \tag{3}$$

In our experiments we observed that in a lot of cases the output of the estimation is *an underestimate of the true support-size*, i.e., $Q_{\text{ANY}}(N) > \sum_{i=0}^n \hat{\theta}_i$. Because of this, the overall estimated volumetric profile $(\hat{\theta}_0, \ldots, \hat{\theta}_n)$ might be far from the true volumetric profile. To deal with this we propose to use a probabilistic approach to adjust each estimations so as the sum of the adjusted estimations is equal to the total number of queries $Q_{\text{ANY}}(N)$. Our strategy for the adjustment is to *respect the distribution of the derived estimations*, and as a result, estimations with larger $\hat{\theta}_v$ will increase with higher probability. Specifically, we (probabilistically) generate a vector of "synthetic" frequencies of volumes that is added to the original vector of estimations $(\hat{\theta}_0, \ldots, \hat{\theta}_n)$ so that the resulting entries sum to $Q_{\text{ANY}}(N)$. First, we define a discrete probability distribution where the sample space is the set of possible volumes and the probability of sampling volume $v \in \{0, \ldots, n\}$ is $(\hat{\theta}_v + 1)/(n + 1 + \sum_{i=0}^n \hat{\theta}_i)$. Then, we generate $Q_{\text{ANY}}(N) - \sum_{i=0}^n \hat{\theta}_i$ samples and for every sampled volume we increment the corresponding $(\hat{\theta}_0, \ldots, \hat{\theta}_n)$ position. Notice that we do not give zero probability to unobserved volumes. A similar technique can be applied in case the sum of the estimations is an *overestimate* of the number of queries. In that case we *subtract* the frequency of the sampled volumes from $(\hat{\theta}_0, \ldots, \hat{\theta}_n)$ and verify that the adjusted estimation has nonnegative entries.

B. Reconstructions that Match The Volumetric Profile

We propose a process so as to generate a reconstruction that matches the (estimated) volumetric profile as close as possible.

First Approach: System of Equations. Previous volumetric attacks addressed the case of the quadratic scheme [23], [31] under the uniform query distribution assumption and no search-pattern leakage. As mentioned in Section V a natural approach is to extend their technique to practical response-hiding schemes [15], [18]. In this case the attacker can solve a system of equations with unknowns $L_1, \ldots, L_{n+1}$ so as to derive the distance between consecutive plaintexts. The work by Kellaris *et al.* [31] proposed to solve the system for QD by factoring polynomials with integer coefficients. Armed with the closed-form expression of counting functions from Section VI we attempt to swap one counting function for another as proposed in the abstraction of Section V.

EQUATIONS BASE

$$\sum_{i=1}^{n} \sum_{l=0}^{\lfloor \log(N) \rfloor} \max\left\{0, \min\left\{L_i, L_{i+1}, \sum_{t=0}^{1} L_{i+t} - 2^l, 2^l\right\}\right\} = \theta_1$$

$$\sum_{i=1}^{n-1} \sum_{l=0}^{\lfloor \log(N) \rfloor} \max\left\{0, \min\left\{L_i, L_{i+2}, \sum_{t=0}^{2} L_{i+t} - 2^l, 2^l - L_{i+1}\right\}\right\} = \theta_2$$

$$\vdots$$

$$\sum_{l=0}^{\lfloor \log(N) \rfloor} \max\left\{0, \min\left\{L_1, L_{n+1}, \sum_{t=0}^{n} L_{1+t} - 2^l, 2^l - \sum_{t=1}^{n-1} L_{1+t}\right\}\right\} = \theta_n$$

Unfortunately, the complexity of the above system of equations for the BASE scheme makes the previous approach of factoring polynomials [31] not applicable to the more sophisticated schemes. Therefore, it is clear that the techniques proposed for exact reconstruction on volumetric attacks for the QD scheme *can not be extended* and we need a different approach to address practical schemes.

Proposed Approach: Constrained Optimization. To address the roadblocks of the previous approach [31] we propose a constrained optimization formulation. The intuition is that we require the reconstruction to *match as close as possible the estimated volumetric profile*. By applying the support-size estimation techniques on the conditional probability distributions $p_{\mathsf{T}|\mathsf{V}}$ we derive an estimation of the total number of queries that return i records, i.e., $\{\hat{\theta}_i\}_{i=0}^n$. We define as unknown variables the lengths between n consecutive plaintexts $\{L_i\}_{i=1}^{n+1}$ and the goal is to output lengths such that the *counting functions* (which themselves take $\{L_i\}_{i=1}^{n+1}$ as input) result in volumes that match the estimated volumetric profile $\{\hat{\theta}_i\}_{i=0}^n$. We define as hard constraints, conditions that must be satisfied, i.e., the requirement that all lengths must be non-negative as well as the hard constraint that their sum must be N. Since the volumetric profile in hand is not exact, but rather an estimate, we deal differently with the goal of finding lengths that give responses with volume close to the estimated volumetric profile. Specifically, we introduce soft constraints that appear in the loss function so as to *penalize the deviation from the estimated* *profile* $\hat{\theta}_i$ following a squared error formulation. Finally, since the number of samples used to derive each estimate $\{\hat{\theta}_i\}_{i=0}^n$ differs we weight the contribution of each error term in the objective function with the multiplicative weights $\{w_i\}_{i=0}^n$. For our experiments we choose as weight w_i of the error term for $\hat{\theta}_i$ to be the normalized frequency of the queries that returned volume i. The general formula for the loss function of scheme ANY is given in the following:

$$\textsc{Loss}_{\textsc{Any}} \triangleq w_0 \left(G_{\textsc{Any}}(\emptyset) - \hat{\theta}_0\right)^2 + \sum_{j=1}^{n} w_j \left(\left(\sum_{\forall r : |r| = j} G_{\textsc{Any}}(r) \right) - \hat{\theta}_j \right)^2 .$$

In case the scheme under attack has entries in T_{ANY} beyond 0/1 then we use the approximation for $G_{\text{ANY}}(r)$ that is defined in Equation 1–2. The analytical expressions of the loss function for schemes BASE, ABT, and BT are depicted in Figure 3.

C. Dealing with Multiple Reconstructions

Depending on the leakage of the corresponding STE there might be multiple plaintext reconstructions that result in the same observed leakage. This is not a limitation of a specific attack algorithm but rather an *intrinsic characteristic* of the leakage from some STE constructions. This issue was first discovered by Kellaris *et al.* [31] where the factorization of polynomials for QD does not necessarily have a unique factorization and has resurfaced in followup works [23], [33]. The set of all valid reconstructions that generate the observed leakage is called *reconstruction space* and was first defined by Kornaropoulos *et al.* [33] in the context of k-NN queries. Unfortunately, none of the range attacks in the literature uses insights about the reconstruction space and as a result these techniques either arbitrarily pick one reconstruction or they fail. We propose a technique to produce a reconstruction space informed output, much like the paradigm of the attack in [33].

As a first step, we run the constrained optimization problem m times with different starting points so as to generate multiple reconstructions out_i. These *candidate reconstructions* can be seen as samples from the reconstruction space. Given that our adversary has no prior knowledge about the data distribution, the adversary treats all the members of the reconstruction space as equally likely to be the plaintext DB under attack. Therefore, our approach is to choose the reconstruction that is as close as possible to the rest of the candidate reconstructions on average, for a notion of "closeness". Specifically, for each reconstruction out_i we compute the average MAE (mean absolute error) between out_i and all other out_j, $\forall j \in [1, m]$ such that $j \neq i$, and we refer to this quantity as the *score* s_i of out_i. The score s_i serves as a measure of closeness between out_i and the rest of the candidate reconstructions. For the score we chose the average MAE, as opposed to the maximum MAE, so as to be more robust to outliers. Among the m reconstructions we pick the reconstruction out_k with the minimum score, i.e., $k = \arg\min_i s_i$. As we experimentally show in the next section, the *maximum MAE* among the derived reconstruction samples, which maps to the worst-case error by an "unlucky" pick that may occur by the previous attacks [23], [31], might be up to 7× larger than the MAE of our approach. Thus, a reconstruction space informed output significantly improves the quality of the output reconstruction.

Algorithm 1: AGNOSTIC-PARAMETRIZED-ATTACK

```
Input: Parameter $T_{\text{ANY}}$ for the regular STE scheme ANY; Multiset
    $D = \{(t_1, V_1), \ldots, (t_z, V_z)\}$ of observed search tokens and
    corresponding volumes for scheme ANY; Endpoints $\alpha$ and $\beta$ of
    the domain universe with size $N = \beta - \alpha + 1$; Number $m$ of
    candidate reconstructions
Output: Approximate reconstruction of the database plaintext values
    $out^* = (\hat{v}_1, \ldots, \hat{v}_n)$
// Estimate the Number of Queries per Volume
1  for $i \in [0, n]$ do
2      Let $D_i$ be the mulitset of all pairs $(t_j, V_j)$ in $D$ with
       volume $V_j = i$;
3      Let weight $w_i = |D_i|^2$;
4      Run Algorithm SUPPORT-SIZE-ESTIMATOR [34] on multiset $D_i$ of
       search tokens to output the $i$-th entry of the volumetric profile $\hat{\theta}_i$;
5  end
// Adjust the Estimations
6  if $|Q_{\text{ANY}}(N) - \sum_{i=0}^{n} \hat{\theta}_i| > 0$ then
7      Construct probability distribution $pdf = (p_0, \ldots, p_n)$ such that
       $p_i = (\hat{\theta}_i + 1)/(n + 1 + \sum_{j=0}^{n} \hat{\theta}_j)$ ;
8      Pick $|Q_{\text{ANY}}(N) - \sum_{i=0}^{n} \hat{\theta}_i|$ samples from distribution $pdf$ ;
9      Add/Subtract the number of occurrences of each sampled value to
       $(\hat{\theta}_0, \ldots, \hat{\theta}_n)$ depending on the sign of $Q_{\text{ANY}}(N) - \sum_{i=0}^{n} \hat{\theta}_i$;
10 end
// Approximate Counting Functions if Needed
11 if not all entries of $T_{\text{ANY}}$ are 0/1 then
12     Use the approximations presented in Equations 1–2 for the
       formulation of the loss function $\textsc{Loss}_{\text{ANY}}$ ;
13 end
// Derive $m$ Candidate Reconstructions
14 for $j \in [1, m]$ do
       // Compute Candidate Reconstruction $out_j$
15     Pick a random initial point $L_{init} = \{L'_i\}_{i=1}^{n+1}$ such that $L'_i \geq 0$
       and $\sum_{i=1}^{n+1} L'_i = N$;
16     Solve the constrained optimization with initial point $L_{init}$:
           $L^{(j)} = \arg\min_{L_i} \textsc{Loss}_{\text{ANY}} \left(\{L_i\}_{i=1}^{n+1}\right)$
               $s.t.\ L_i \geq 0, \forall i \in [1, n+1]$
               $\sum_{i=1}^{n+1} L_i = N$
       Define candidate reconstruction $out_j = (\hat{v}_1, \ldots, \hat{v}_n)$, where
       $\hat{v}_i = \alpha + \sum_{k=1}^{i} L_k^{(j)}$;
17 end
// Select among the Candidate Reconstructions
18 Define $s_j = \frac{1}{m-1} \sum_{i=1}^{m} \frac{1}{n} \min\{|out_i - out_j|, |out_i - \text{Flip}(out_j)|\}$,
   where $j \in [1, m]$ and Flip($x$) outputs the values of sequence $x$ in
   reverse order;
19 return $out^* = out_k$, where $k = \arg\min_j s_j$;
```

D. The Attack Algorithm

Algorithm 1 combines the building blocks described in Sections VII-A to VII-C. Lines 1-5 deploy a support size estimators for each conditional probability distribution to derive the estimated volumetric profile. If the sum of estimated queries is smaller/larger than the total number of distinct canonical range queries $Q_{\text{ANY}}(N)$, Lines 6-10 probabilistically adjust the estimated frequencies. If the T_{ANY}-regular scheme under attack has non 0/1 entries in T_{ANY}, then Lines 11-13 use an approximate counting function. Lines 14-17 solve the constrained optimization problem using the counting functions and the estimations for the regular scheme that is passed as

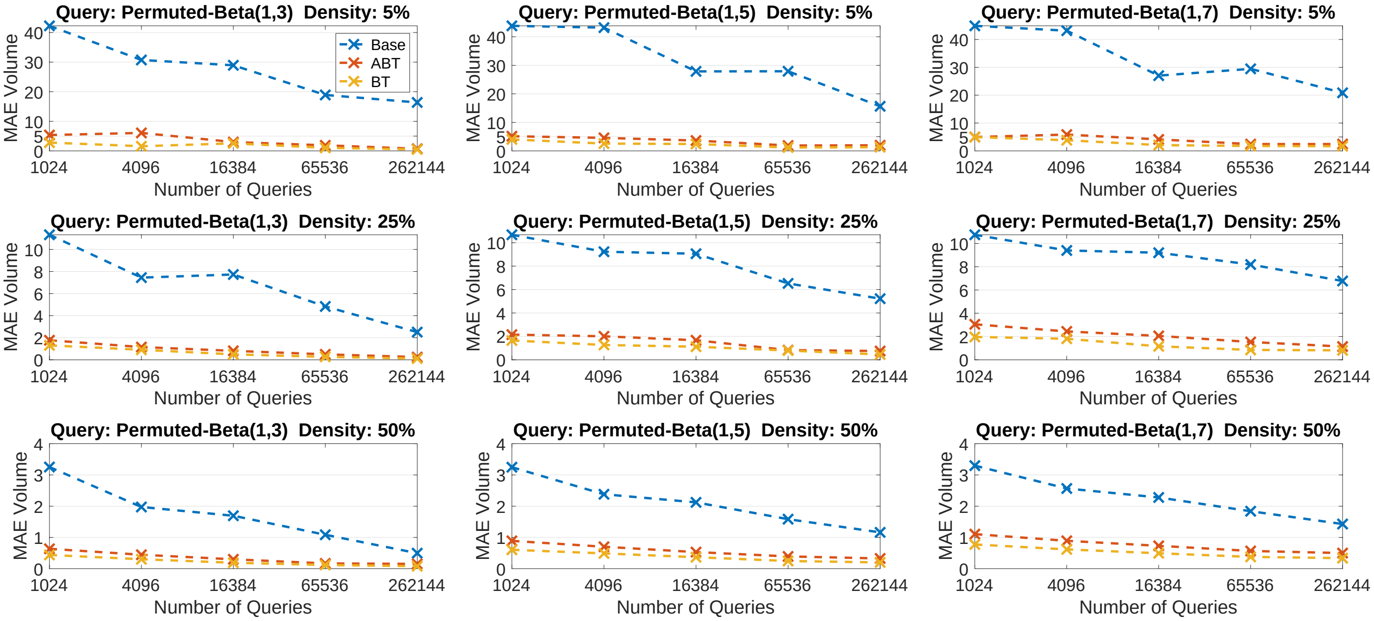

Fig. 4. Evaluation of the volumetric profile estimation. The Y-axis represents the Mean Absolute Error (MAE) between the estimated volumetric profile and the original. The X-axis represents the number of queries used for the estimation. Plots on the same column are produced with the same query distribution, whereas plots on the same row are produced with the same data density. Each experiment compares the accuracy of the estimation for the same set of queries for three different practical response-hiding schemes, BASE, ABT, and BT.

a parameter. Finally, Lines 18-19 return a reconstruction that performs well on average with respect to the m derived samples from the reconstruction space.

The approximation of the volumetric profile requires the evaluation of closed-form polynomial formulas for the jackknife estimators which can be found in the appendix of [34]. The optimization component of our attack is more challenging to analyze with traditional time complexity standards since it is unclear how to theoretically analyze the convergence of the iterative methods for the objective functions depicted in Figure 3. In practice, all the experiments conducted in this work terminated in less than a minute in a typical laptop setup.

VIII. Evaluation

We have conducted experiments to assess the practical performance of our attack based on the following factors:

- **Quality of the Volumetric Profile Estimation.** The first step of the attack estimates the volumetric profile (Section VII-A) and the rest of the attack crucially depends on the quality of this estimation. Indeed, an inaccurate estimation will lead to processing a reconstruction space that might be vastly different from the true reconstruction space associated with the original plaintext data.
- **Quality of the Minimization Solution.** The next phase of the attack (Section VII-B) uses the estimated volumetric profile to generate candidate reconstructions that match the volumetric profile as close as possible. To achieve this task, the attacker solves a constrained optimization problem. The quality of the overall reconstruction depends on the ability of the solver to minimize the objective function. Non-optimal solutions imply that the output reconstruction may not approximate the (estimated) volumetric profile to a satisfactory degree.
- **Structure of the Reconstruction Space.** The pairwise relations between the candidate reconstructions that satisfy the volumetric profile plays a significant role in the quality of the final reconstruction. The structure of the reconstruction space can be such, that all databases are "far" from each other, which implies that it may be challenging to output a reconstruction that is simultaneously close to *all* databases from the reconstruction space.

We present experiments and metrics that shed light to the above practical challenges. The evaluation in Section VIII-A focuses exclusively on the quality of the estimation of the volumetric profile. Section VIII-B presents an evaluation of the minimization under exact and estimated volumetric profiles for different data densities. Finally, Section VIII-C evaluates the proposed attack on hospital data from the HCUP dataset [1].

A. Approximating the Volumetric Profile

In Figure 4 we evaluate the quality of the estimation of the volumetric profile. We consider a domain of size $N = 2^{10}$ which is larger of the typical domain used in previous works, i.e., in [23], [25], [31] the authors chose $N = 365$.

For the data generation we follow the approach from [34] and deploy a PermutedBeta distribution. Specifically, the beta distribution is defined under the continuous interval [0, 1], which we discretized into N segments of equal length. The probability mass of each segment is equal to the aggregate mass associated with the segment. After the discretization step, we permute the probability masses so as to minimize the predictability of the probability mass given its location. For the shape parameters we chose $\alpha = 1$ and $\beta = 5$, i.e., $\mathsf{PermutedBeta}(1, 5)$. The rationale behind this choice of parameters is to benchmark how the estimation performs when there is controlled concentration, i.e. through different β shape parameter, but no obvious structure in the data/query probability distribution, i.e., achieved via the permutation step. The generated data, which may include multiple records with the same value, are sampled so as to test three different data densities $\{5\%, 25\%, 50\%\}$. We deploy three *query distributions* that progress in ascending order with respect to their concentration: $\mathsf{PermutedBeta}(1, 3)$, $\mathsf{PermutedBeta}(1, 5)$, $\mathsf{PermutedBeta}(1, 7)$, see the Figure 7 in [34] for an illustration of the beta parametrizations. Note that the estimators that construct the volumetric profile are *agnostic* and they do not know anything about the above query distributions. The number of sampled queries takes value from the set $\{1024, 4096, 16384, 65536, 262144\}$. We measure the quality of the estimation by computing the mean absolute error (MAE) between the original volumetric profile and the estimated volumetric profile. We test the quality of the estimation for the practical response-hiding schemes BASE, ABT, and BT. The number of canonical ranges is different for each scheme, i.e., $Q_{\text{BASE}}(2^{10}) = 9228$, $Q_{\text{ABT}}(2^{10}) = 3060$, and $Q_{\text{BT}}(2^{10}) = 2047$ (see Equation 3).

As expected, the error of the estimation in Figure 4 decreases significantly as the attacker observes more queries. In most cases the MAE reduces by half between the smallest and the largest number of tested queries. Interestingly, the volumetric profile of schemes ABT and BT is estimated more accurately than the profile of BASE. This phenomenon can be explained by the fact that the overall number of canonical ranges in ABT and BT is smaller than in BASE, which implies that the frequency is concentrated in a smaller subset and as a result the estimators provide more accurate results. Another interesting observation is that the sparser the density of the database the harder it is to approximate the volumetric profile.

B. Evaluation on Synthetic Data

In this experiment we measure the performance of all the phases of the attack on synthetic data. For comparison we start by presenting a benchmark attack, the so-called Oracle Attack, which is based on oracle access to the data distribution that is unrealistic to find in most scenarios. We emphasize here that there is no other leakage abuse attack in the literature that applies to ABT and BT. In Appendix X-A we evaluate the quality of the approximation of the counting functions.

Oracle Attack: An Unfair Comparison. For comparison purposes we define the following attack that is based on unrealistic adversarial knowledge: we assume that the attacker has oracle access to the *exact data distribution* of the plaintext. Knowing the data distribution implies that the adversary knows not only the attribute on which queries are executed (e.g., age), but also the context of the data. Context is important because the same attribute can have different distributions in different databases. E.g., attribute age is distributed differently in the following databases: employees of a company, students of a university, retirees of a state pension fund, and airline passengers. The "Oracle Attack" derives n samples with respect to the data distribution and outputs the result as a reconstruction.

Table II presents the MAE of the oracle attack (averaged over 1000 runs) on the same plaintext data. The oracle attack does not use any query leakage and its reconstruction is performed based solely on the oracle's output. We emphasize that the oracle attack is extremely accurate in the following scenarios: (i) the probability mass is concentrated in a few values, not necessarily neighboring (ii) the probability mass is accumulated in neighboring values, in which case incorrect reconstruction is still in a close proximity to the plaintext, and (iii) the database contains a large number of records, in which case the oracle is queried so much that its output captures accurately the shape of the distribution and, consequently, the database. Overall, in this (unfair) comparison, our approach has a major disadvantage because it has *no knowledge of the data distribution*. Thus, one would expect that our proposed attack is always inferior to the powerful oracle attack that operates under different assumptions. Nevertheless, our agnostic attack outperforms the oracle attack in several of the tested setups.

Evaluation of the Leakage-Abuse Attack. For the main experiment of this subsection, we evaluate the performance of Algorithm 1 under a wide variety of setups and present the results in Table II. The plaintext domain is $N = 1024$ and the data is generated according to distribution $\mathsf{PermutedBeta}(1, 5)$. We generate a single plaintext database with no multiplicities for each of the three domain densities 1%, 5%, 10%. We study two metrics to assess the quality of the reconstruction and test whether it can perform as good as the benchmark attack. At a high-level, the goals of this experiment are the following:

- Examine the variability of candidate reconstructions by measuring the quality metric MAE MaxPair.
- Study whether an increase in the number of candidate reconstructions returns a more "central" reconstruction and therefore a more robust output.
- Compare the quality of the proposed attack with the (unrealistic) adversary of the Oracle Attack.

Quality Metrics. The first metric, denoted MAE Plaintext, measures the mean absolute error between the reconstruction and the plaintext database. We note here that given a plaintext database DB we measure the MAE of reconstruction out^* as the minimum MAE among the pair of (DB, out^*) and the pair $(DB, \text{Flip}(out^*))$, where the $\text{Flip}(\cdot)$ reverses the order of the elements of the vector. This approach is common among attack evaluations because even if the adversary recovers correctly the pairwise distances between plaintexts it is not possible to infer whether the correct ordering is out^* or $\text{Flip}(out^*)$. The second metric, denoted MAE MaxPair, measures the maximum MAE between pairs of candidate reconstructions. This metric shows the structure of the reconstruction space, i.e., if the MAE MaxPair is large then this indicates a "spread out" reconstruction space. We emphasize that a large MAE MaxPair is an intrinsic characteristic of the reconstruction space and is not a flaw of the attack algorithm. To test the effect of

		Scheme	Candidate Reconstr. = 3				Candidate Reconstr. = 10				Candidate Reconstr. = 100				Candidate Reconstr.= 500				Oracle Attack MAE Plaintext
			MAE Plaintext		MAE MaxPair		MAE Plaintext		MAE MaxPair		MAE Plaintext		MAE MaxPair		MAE Plaintext		MAE MaxPair		
			Exc	Est	Exc	Est	Exc	Est	Exc	Est	Exc	Est	Exc	Est	Exc	Est	Exc	Est	
Domain Density	1%	BASE	85.0	59.0	104.5	93.5	70.8	80.2	118.9	144.6	58.6	83.5	144.6	189.6	42.8	62.7	188.8	247.6	92.1
		ABT	113.9	73.1	101.4	96.5	100.9	78.3	133.2	174.5	96.3	72.9	213.7	223.2	91.1	68.1	225.6	248.4	
		BT	105.6	107.6	168.8	72.5	103.9	76.0	148.8	178.3	101.0	81.2	232.7	195.6	98.7	83.4	260.9	249.1	
	5%	BASE	26.2	19.3	30.9	14.4	29.9	24.7	48.2	46.8	29.9	26.8	69.0	56.0	24.8	27.9	78.4	61.4	42.4
		ABT	32.4	33.0	26.5	38.2	30.6	31.2	67.8	54.2	29.6	30.7	78.9	87.1	35.2	28.3	91.5	99.7	
		BT	33.7	27.6	26.9	31.4	26.0	29.3	48.2	31.3	27.1	27.9	82.1	57.1	30.5	30.6	92.1	61.3	
	10%	BASE	15.0	15.1	16.5	12.7	19.2	13.8	22.9	27.5	12.2	12.8	46.8	35.7	12.2	12.7	50.0	48.4	24.1
		ABT	15.7	14.0	33.6	42.7	15.6	12.9	35.4	32.0	12.8	14.1	54.1	54.3	15.3	15.0	66.5	62.3	
		BT	13.7	15.8	17.6	27.6	17.1	15.1	44.4	37.7	13.7	13.1	52.6	55.1	11.5	14.5	66.9	52.8	

TABLE II

Performance of our attack for various data densities and numbers of candidate reconstructions. The domain size is $N = 1024$. Table entries show the MAE between reconstructed and plaintext values (MAE Plaintext) and the maximum MAE between pairs of candidate reconstructions (MAE MaxPair). Gray columns (Est) present the output of Algorithm 1. To understand the role of the volumetric profile estimation, we also present in white columns (Exc) the same attack but with the exact volumetric profile.

candidate reconstructions we deploy the attack algorithm for $\{3, 10, 100, 500\}$ candidate reconstructions and study its impact to MAE Plaintext and MAE MaxPair.

Setup. For the constraint optimization problem we use function fmincon from MATLAB that deploys an interior-point algorithm. In the majority of our experiments, the loss function was trapped in local minima. To deal with this phenomenon, we perform 10^3 random restarts for the computation of *each candidate reconstruction* and we choose as a candidate the plaintext database that has the smallest observed loss.

To study the *effect of the volumetric profile estimation* in the overall performance we present two variations of Algorithm 1: The first follows the attack exactly as it is described in Algorithm 1 and the volumetric profile is estimated based on $|Q| = 3072$ queries generated with a (different from the plaintext) $\mathsf{PermutedBeta}(1, 5)$ distribution; while the second variation skips Lines 1-10 and uses directly the exact volumetric profile in the remaining Lines 11-19, i.e., no estimation takes place. We note here that given the observed results of the volumetric profile estimation from Section VIII-A, we expect that different query distributions from the tested one, e.g., $\mathsf{PermutedBeta}(1, 7)$, would present almost identical behavior. The case of exact volumetric profile is denoted with "Exc" and the case of estimated profile with "Est" in Table II.

Results. First, notice that there is a discrepancy between the two metrics. MAE MaxPair can be seen as a worst-case performance, up to $4.5\times$ larger than our attack. This worst-case error is possible if the attacker picks an arbitrary reconstruction among the many, much like previous approaches. To better understand this discrepancy we have to analyze the trends. The first interesting observation is that MAE MaxPair increases significantly, up to $3.8\times$, as we generate more candidate reconstructions. This fact holds across all domain densities and all schemes. Thus, a larger number of candidate reconstructions paints a more accurate picture of the structure of the reconstruction space. Another interesting observation is that MAE Plaintext decreases, in most cases, as we generate more candidate reconstructions. This is because our strategy to pick a "central" reconstruction performs better as we explore the reconstruction space with multiple candidates. These observations show the importance of designing *reconstruction space informed attacks*. The case of exact volumetric profile is presented to factor out one of the sources of error, i.e., the error from the volume profile estimation, and thus shed light into how the structure of the reconstruction space affects the quality of the output of the attack. Interestingly, the more realistic case of estimated volumetric profiles follows the behavior of the exact profile case. We chose $|Q| = 3072$ number of queries to show the performance of our attack. Table II shows that the performance of the "Est" is relatively close to the ideal case "Exc". Finally, the MAE Plaintext is *consistently smaller* than the Oracle Attack, in some cases it is as low as half the error.

C. Evaluation on Hospital Data

For this experiment we use real hospital datasets obtained from the US government Healthcare Cost and Utilization Project (HCUP) Nationwide Inpatient Sample (NIS) from year 2009 [1]. This dataset is used in previous leakage-abuse attacks [19], [23], [25], [31].

We chose the attributes with the largest domain size. Attribute AGE records the age in years of each patient and has values from 1 to 91. Attribute AGEDAY records the age in days of infants and has values from 1 to 364. To estimate the volumetric profile we issued $3 \cdot 10^4$ range queries with respect to the $\mathsf{PermutedBeta}(1, 5)$ distribution over all range queries. We considered 10 candidate reconstructions, each of which minimized the objective function after 100 random restarts.

Toward testing the attack under a fixed domain density, we randomly picked database records (with multiplicities) until we reached a fixed domain density. Note that the resulting number of records n depends on the data distribution. The resulting n for attribute AGE was $5, 9, 28, 100$ and for attribute AGEDAY was $19, 38, 118, 310$ for the domain densities 5%, 10%, 25%, and 50%. To build the oracle for the "Oracle Attack" we applied a kernel density estimator on all data of the attribute using function fitdist from MATLAB, a non-parametric technique for deriving the probability density function from data.

Table III presents a comparison between the proposed attack from Algorithm 1 and the "Oracle Attack" with respect to the MAE Plaintext quality measure. For domain densities 5% and 10%, the proposed attack outperforms the oracle attack, which operates under different and strong assumptions (access to an oracle from the data distribution). As it is expected, as the number of records increases in skewed distributions the oracle attack converges to the plaintext database and therefore

		Scheme	Attribute AGE		Attribute AGEDAY	
			Algo. 1 Attack	Oracle Attack	Algo. 1 Attack	Oracle Attack
Domain Density	5%	Base	2.8	12.4	28.0	43.1
		ABT	2.1		28.5	
		BT	3.6		26.5	
	10%	Base	5.9	8.4	17.3	42.6
		ABT	6.0		17.9	
		BT	7.1		20.0	
	25%	Base	7.9	4.8	48.4	20.1
		ABT	8.0		49.2	
		BT	7.8		47.0	
	50%	Base	11.0	3.0	57.6	11.0
		ABT	11.1		42.0	
		BT	11.2		57.7	

TABLE III
Performance of our attack for various data densities on attributes from hospital data of HCUP [1]. The quality is measured as the mean absolute error (MAE Plaintext).

outperforms Algorithm 1. Nevertheless, for the case of attribute AGE, not as severely-skewed as AGEDAY, the proposed attack has relatively small error even for domain density 25%, 50%.

IX. Conclusion

We present the first leakage-abuse attack on *practical response-hiding structured encryption schemes*, i.e., those with non-quadratic storage overhead. Our attack is parametrized in the sense that it can be applied to a wide variety of encrypted range schemes by simply switching the expression of the so-called counting function which acts as a parameter. Our technique allows us to reassess the security and even compare different encrypted range schemes based on the output of our parametrized attack. Overall, although response-hiding schemes are more secure than standard structured encryption schemes, our results show that they are still vulnerable to leakage-abuse attacks based on search-pattern and volumetric leakage.

Acknowledgments

We thank our shepherd Rahul Chatterjee and the anonymous reviewers for their detailed and constructive feedback. This work was supported in part by the National Science Foundation (NSF), by the National Institute of Standards and Technology (NIST), and by the Center for Long-Term Cybersecurity (CLTC) at UC Berkeley. The first author underwent the HCUP Data Use Agreement training. We did not attempt to deanonymize the HCUP data, nor are our attacks designed to deanonymize.

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] G. Amjad, S. Kamara, and T. Moataz, "Breach-Resistant Structured Encryption," *PoPETs*, vol. 2019, no. 1, 2019.

[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 STOC*, 2016.

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

[5] R. Bost, "$\sum o\varphi o\varsigma$: Forward Secure Searchable encryption," in *Proc. of the 23rd ACM CCS*, 2016.

[6] 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.

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

[8] 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.

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

[10] M. Chase and S. Kamara, "Structured Encryption and Controlled Disclosure," in *Proc. of the 16th ASIACRYPT*, 2010.

[11] 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.

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

[13] I. Demertzis, D. Papadopoulos, and C. Papamanthou, "Searchable Encryption with Optimal Locality: Achieving Sublogarithmic Read Efficiency," in *Proc. of the 38th CRYPTO*, 2018.

[14] I. Demertzis, D. Papadopoulos, C. Papamanthou, and S. Shintre, "SEAL: Attack Mitigation for Encrypted Databases via Adjustable Leakage," in *Proc. of the 29th USENIX Security*, 2020.

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

[16] I. Demertzis and C. Papamanthou, "Fast Searchable Encryption With Tunable Locality," in *Proc. of ACM SIGMOD*, 2017.

[17] F. B. Durak, T. M. DuBuisson, and D. Cash, "What Else is Revealed by Order-Revealing Encryption?" in *Proc. of the 23rd ACM CCS*, 2016.

[18] 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.

[19] 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.

[20] B. Fuller, M. Varia, A. Yerukhimovich, E. Shen, A. Hamlin, V. Gadepally, R. Shay, J. D. Mitchell, and R. K. Cunningham, "SoK: Cryptographically Protected Database Search," in *Proc. of the 38th IEEE S&P*, 2017.

[21] 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.

[22] P. Grubbs, A. Khandelwal, M. Lacharité, L. Brown, L. Li, R. Agarwal, and T. Ristenpart, "pancake: Frequency Smoothing for Encrypted Data Stores," in *Proc. of the 29th USENIX Security*, 2020.

[23] P. Grubbs, M. Lacharité, B. Minaud, and K. G. Paterson, "Pump up the Volume: Practical Database Reconstruction from Volume Leakage on Range Queries," in *Proc. of the 25th ACM CCS*, 2018.

[24] P. Grubbs, K. Sekniqi, V. Bindschaedler, M. Naveed, and T. Ristenpart, "Leakage-Abuse Attacks Against Order-Revealing Encryption," in *Proc. of the 38th IEEE S&P*, 2017.

[25] Z. Gui, O. Johnson, and B. Warinschi, "Encrypted Databases: New Volume Attacks Against Range Queries," in *Proc. of the 26th ACM CCS*, 2019.

[26] 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.

[27] R. Jacob, K. G. Larsen, and J. B. Nielsen, "Lower Bounds for Oblivious Data Structures," in *Proc. of the 30th ACM-SIAM SODA*, 2019.

[28] S. Kamara and T. Moataz, "Computationally Volume-Hiding Structured Encryption," in *Proc. of 38th EUROCRYPT*, 2019.

[29] S. Kamara, T. Moataz, and O. Ohrimenko, "Structured Encryption and Leakage Suppression," in *Proc. of the 38th CRYPTO*, vol. 10991, 2018.

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

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

[32] E. M. Kornaropoulos, "Information Leakage in Encrypted Systems Through an Algorithmic Lens," Ph.D. dissertation, Department of Computer Science, Brown University, 2019.

[33] 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.

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

[35] 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.

[36] K. G. Larsen, T. Malkin, O. Weinstein, and K. Yeo, “Lower Bounds for Oblivious Near-Neighbor Search,” in *Proc. of the 31st SODA*, 2020.

[37] K. G. Larsen and J. B. Nielsen, “Yes, There is an Oblivious RAM Lower Bound!” in *Proc. of the 38th CRYPTO*, 2018.

[38] E. A. Markatou and R. Tamassia, “Full Database Reconstruction with Access and Search Pattern Leakage,” in *Proc. Int. Conf. on Information Security (ISC)*, 2019.

[39] ——, “Mitigation techniques for attacks on 1-dimensional databases that support range queries,” in *Proc. Int. Conf. on Information Security (ISC)*, 2019.

[40] M. Naveed, S. Kamara, and C. V. Wright, “Inference Attacks on Property-Preserving Encrypted Databases,” in *Proc. of the 22nd ACM CCS*, 2015.

[41] S. Patel, G. Persiano, and K. Yeo, “Lower Bounds for Encrypted Multi-Maps and Searchable Encryption in the Leakage Cell Probe Model,” in *Proc. of the 40th CRYPTO*, 2020.

[42] 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.

[43] 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.

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

[45] 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.

[46] 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.

X. Appendix

A. Counting Function Approximation Performance

In this experiment, we evaluate the quality of the approximation of the global counting function $\widetilde{G}_{\text{ANY}}(r)$ presented in Equation 2. In particular, we fix a database and we evaluate the quality of the outputs of the counting functions introduced in previous sections. We use the same plaintext data (database) over domain $N = 1024$ as the data used for Table II. For domain density 1%, 5%, and 10%, the number of possible responses are 79, 1486, and 5996, respectively, as indicated in Table IV. To assess the error we measure (i) the number of responses for which the approximation of the counting function is not equal to the exact counting, indicated as “# Errors” in Table IV, as well as (ii) the maximum error among all the responses, , indicated as “Max” in Table IV. Since the counting function for scheme Base is exact, see Remark 3, the output has no errors. The quality of the approximation is similar for schemes ABT and BT. The number of errors is relatively low with respect to the growth of the number of possible responses, i.e., less than 3.5% of the responses for 5996 responses. Another interesting observation is that the maximum error among all responses is at most 3 which is relatively low compared to the (pessimistic) upper bound for $N = 1024$ which is $\mathit{weight}(T_{\text{ABT}}) = \mathit{weight}(T_{\text{BT}}) = \log(1024) = 10$ according to Theorem 2.

		Scheme	Volume Profile via Counting Approx.	
			# Errors	Max
Domain Density	1%	BASE	0/79	0
		ABT	19/79	2
		BT	17/79	2
	5%	BASE	0/1486	0
		ABT	82/1486	2
		BT	73/1486	2
	10%	BASE	0/5996	0
		ABT	209/5996	3
		BT	177/5996	3

TABLE IV
Performance of the counting function approximation, $N = 1024$

B. Proof of Lemma 1

The fact that $s \geq \sum_{t=0}^{k} L_{i+t}$ can be rehashed as $s \geq d(v_{i-1}, v_{i+k})$. Therefore, if we fix the lower-boundary at the *leftmost possible location* so as to return r, i.e., location $v_{i-1}+1$, then the span s is large enough to *cover all the desired values* so as to return r. Notice that the Basis scheme contains all possible ranges of span s, see Remark 2. Thus, the question boils down to how many times can we “advance” the leftmost lower-boundary towards the right before we: (A) either reach position v_i with the lower-boundary, or (B) reach position $v_{i+k+1} - 1$ with the upper-boundary, or (C) both reach position $v_i - 1$ with the lower and $v_{i+k+1} - 1$ with the upper-boundary. All of the above cases imply that we can not advance the lower-boundary anymore, therefore, there are no more range queries of span s to count. In case (A), one can increment the lower-boundary until it reaches the rightmost possible lower-boundary location, i.e. location v_i, and there are still a few positions in L_{i+k+1} that can not be considered as an upper-boundary. Going back to the facts, we know that s is strictly less than $\sum_{t=1}^{k+1} L_{i+t}$ which means that $s < d(v_i, v_{i+k+1})$. Therefore, there is at least one location to the left of v_{i+k+1} that can not be claimed as an upper-boundary. This fact implies case (A) therefore we can iterate through the entire L_i and $C_{\textsc{Step1-Any}}(r, s) = L_i$. It is clear that cases (B) and (C) can not hold since if they were true we would have $s = d(v_i, v_{i+k+1})$ but we know from our facts that $s < d(v_i, v_{i+k+1})$.

C. Proof of Lemma 2

The fact that $s \geq \sum_{t=1}^{k+1} L_{i+t}$ can be rehashed as $s \geq d(v_i, v_{i+k+1})$. Therefore, if we fix the upper-boundary at the *rightmost possible location* so as to return r, i.e., location $v_{i+k+1} - 1$, then the span s is large enough to *cover all the desired values* so as to return r. The question boils down to how many times can we “advance” the rightmost upper-boundary towards the left before we: (A) either reach position v_{i+k} with the upper-boundary, or (B) reach position $v_{i-1} - 1$ with the lower-boundary, or (C) both reach position v_{i+k} with the upper and $v_{i-1} - 1$ with the lower-boundary. All of the above cases imply that we can not advance the upper-boundary anymore, therefore, there are no more range queries of span s to count. In case (A), one can decrease the upper-boundary until it reaches the leftmost possible upper-boundary location, i.e. location

v_{i+k}, and there are still a few positions in L_i that can not be considered as a lower-boundary. Going back to the facts, we know that s is strictly less than $\sum_{t=0}^{k} L_{i+t}$ which means that $s < d(v_{i-1}, v_{i+k})$. Therefore, there is at least one location to the right of v_{i-1} that can not be claimed as a lower-boundary. The assumption follows the condition of case (A) therefore we can iterate through the entire L_{i+k+1} and $C_{\text{STEP1-ANY}}(r, s) = L_{i+k+1}$. It is clear that cases (B) and (C) can not hold since if they were true we would have $s = d(v_{i-1}, v_{i+k})$ but we know from our facts that $s < d(v_{i-1}, v_{i+k})$.

D. *Proof of Lemma 3*

Similarly to the proof of Lemma 1, the fact that $s \geq \sum_{t=0}^{k} L_{i+t}$ can be rehashed as $s \geq d(v_{i-1}, v_{i+k})$. Therefore, if we fix the lower-boundary at the *leftmost possible location* so as to return r, i.e., location $v_{i-1} + 1$, then the span s is large enough to *cover all the desired values* so as to return r. Thus, the question boils down to how many times can we "advance" the leftmost lower-boundary towards the right before one of the following three cases happen: (A) either reach position v_i with the lower-boundary, (B) or reach position $v_{i+k+1} - 1$ with the upper-boundary, (C) or both reach position $v_i - 1$ with the lower and $v_{i+k+1} - 1$ with the upper-boundary at the same time. All of the above cases imply that we can not advance the lower-boundary anymore, therefore, there are no more range queries of span s to count. The proof differentiates from the one of Lemma 1 in the remaining.

If case (A) is true then one can increment the lower-boundary until it reaches the rightmost possible lower-boundary location, i.e. location v_i, and there is at least one empty position in L_{i+k+1} that can not be considered as an upper-boundary. This can not be true because it implies that $s < \sum_{t=1}^{k+1} L_{i+t}$, i.e., $s < d(v_i, v_{i+k+1})$, but from the facts we know that $s \geq \sum_{t=1}^{k+1} L_{i+t}$ therefore case (A) contradicts the facts. This means that with the current facts, the lower-boundary can not bump into v_i first before the upper-boundary bumps into v_{i+k+1}.

Switching focus to case (B), in this case one can increment the lower boundary until the corresponding upper-boundary reaches the rightmost possible upper-boundary location, i.e., location $v_{i+k+1} - 1$. For this case to hold it must be the case that $s > \sum_{t=1}^{k+1} L_{i+t}$ which can be rehashed as $s > d(v_i, v_{i+k+1})$. From the facts of this lemma we know that, $s \geq \max\left\{\sum_{t=0}^{k} L_{i+t}, \sum_{t=1}^{k+1} L_{i+t}\right\}$, which subsumes the condition for (B) to hold. Thus, when $s > \sum_{t=1}^{k+1} L_{i+t}$ we are in case (B) and we have $C_{\text{STEP1-ANY}}(r, s)(r, s) = \sum_{t=0}^{k+1} L_{i+t} - s$.

Switching focus to case (C), if both events take place simultaneously, i.e., lower-boundary bumps onto v_i and upper-boundary bumps onto v_{i+k+1}, then we must have:

$$L_i = \sum_{t=1}^{k+1} L_{i+t} - s \Rightarrow s = \sum_{t=1}^{k+1} L_{i+t} - L_i \tag{4}$$

From the facts of this lemma we know that $s \geq \max\left\{\sum_{t=0}^{k} L_{i+t}, \sum_{t=1}^{k+1} L_{i+t}\right\}$ which proves that equation (4) can not be true. This means that with the current facts, we can not have the case where the lower-boundary bumps into v_i and the upper-boundary bumps into v_{i+k+1} at the same time.

E. *Proof of Lemma 4*

If we place the upper-boundary of the range at its leftmost possible upper-boundary location, i.e., v_{i+k}, then the span s is large enough to cover all the desired values so as to return s since we know from the facts that $s < \sum_{t=0}^{k} L_{i+t}$, which can be rehashed as $s < d(v_{i-1}, v_{i+k})$, as well as $\sum_{t=1}^{k} L_{i+t} < s$, which can be rehashed as $d(v_i, v_{i+k}) < s$. Therefore, the question boils down to how many times can we "advance" the upper-boundary towards the right before one of the following three cases happen: (A) either reach position v_i with the lower-boundary, (B) or reach position $v_{i+k+1} - 1$ with the upper-boundary, (C) or both reach position $v_i - 1$ with the lower and $v_{i+k+1} - 1$ with the upper-boundary at the same time.

If case (A) is true then one can increment the lower-boundary until it reaches the rightmost possible lower-boundary location, i.e. location v_i, and there is at least one empty position in L_{i+k+1} that can not be considered as an upper-boundary. For this to happen the following condition must be true $s < \sum_{t=1}^{k+1} L_{i+t}$. From the facts we know that $s < \min\left\{\sum_{t=0}^{k} L_{i+t}, \sum_{t=1}^{k+1} L_{i+t}\right\}$ which subsumes the condition $s < \sum_{t=1}^{k+1} L_{i+t}$. Therefore case (A) is possible given our facts, in which case the counting function is $C_{\text{STEP1-ANY}}(r, s) = s - \sum_{t=1}^{k} L_{i+t}$.

Moving on to case (B), in order for this even to take place we must have $\sum_{t=1}^{k+1} L_{i+t} \geq s$ but from the fact we know that $s < \sum_{t=1}^{k+1} L_{i+t}$. Therefore neither case (B) nor case (C) can be true.

F. *Proof of Theorem 1*

Let X be $X = \sum_{t=0}^{k} L_{i+t}$ and Y be $Y = \sum_{t=1}^{k+1} L_{i+t}$. It is easy to see that if $\sum_{t=1}^{k} L_{i+t} < s < \sum_{t=0}^{k+1} L_{i+t}$ then only one of the following four cases is true:

1) $X \leq s < Y$, in which case Lemma 1 holds,
2) $Y \leq s < X$, in which case Lemma 2 holds,
3) $\max\{X, Y\} \leq s$, in which case Lemma 3 holds,
4) $s < \min\{X, Y\}$, in which case Lemma 4 holds.

To prove the theorem it is enough to prove the following:

- If Case-1 is true, where $X \leq s < Y$, then,
$$L_i \leq \min\left\{L_{i+k+1}, \sum_{t=0}^{k+1} L_{i+t} - s, s - \sum_{t=1}^{k} L_{i+t}\right\}$$
,
- If Case-2 is true, where $Y \leq s < X$, then,
$$L_{i+k+1} \leq \min\left\{L_i, \sum_{t=0}^{k+1} L_{i+t} - s, s - \sum_{t=1}^{k} L_{i+t}\right\}$$
,
- If Case-3 is true, where $\max\{X, Y\} \leq s$, then,
$$\sum_{t=0}^{k+1} L_{i+r} - s \leq \min\left\{L_i, L_{i+k+1}, s - \sum_{t=1}^{k} L_{i+t}\right\}$$
,
- If Case-4 is true, where $s < \min\{X, Y\}$, then,
$$s - \sum_{t=1}^{k} L_{i+t} \leq \min\left\{L_i, L_{i+k+1}, \sum_{t=0}^{k+1} L_{i+t} - s\right\}.$$

We proceed by proving the above four cases.

Analysis of Case $X \leq s < Y$. From the assumptions:

$$X < Y \Rightarrow \sum_{t=0}^{k} L_{i+t} < \sum_{t=1}^{k+1} L_{i+t} \Rightarrow L_i < L_{i+k+1}. \qquad (5)$$

Additionally, from the assumptions,

$$s < Y \Rightarrow s + L_i < \sum_{t=1}^{k+1} L_{i+t} + L_i \Rightarrow L_i < \sum_{t=0}^{k+1} L_{i+t} - s \quad (6)$$

From the assumptions,

$$X \leq s \Rightarrow 0 \leq s - \sum_{t=0}^{k} L_{i+t} \Rightarrow L_i \leq s - \sum_{t=1}^{k} L_{i+t} \qquad (7)$$

From Equations (5), (6), (7) we conclude that:

$$L_i \leq \min\left\{ L_{i+k+1}, \sum_{t=0}^{k+1} L_{i+t} - s, s - \sum_{t=1}^{k} L_{i+t} \right\}.$$

Analysis of Case $Y \leq s < X$. From the assumptions:

$$Y < X \Rightarrow \sum_{t=1}^{k+1} L_{i+t} < \sum_{t=0}^{k} L_{i+t} \Rightarrow L_{i+k+1} < L_i. \qquad (8)$$

Additionally, from the assumptions,

$$s < X \Rightarrow s + L_{i+k+1} < \sum_{t=0}^{k+1} L_{i+t} \Rightarrow L_{i+k+1} < \sum_{t=0}^{k+1} L_{i+t} - s \qquad (9)$$

From the assumptions,

$$Y \leq s \Rightarrow 0 \leq s - \sum_{t=1}^{k+1} L_{i+t} \Rightarrow L_{i+k+1} \leq s - \sum_{t=1}^{k} L_{i+t} \quad (10)$$

From Equations (8), (9), (10) we conclude that:

$$L_{i+k+1} \leq \min\left\{ L_i, \sum_{t=0}^{k+1} L_{i+t} - s, s - \sum_{t=1}^{k} L_{i+t} \right\}.$$

Analysis of Case $\max\{X, Y\} \leq s$. From the assumptions:

$$Y \leq s \Rightarrow \sum_{t=1}^{k+1} L_{i+t} - s \leq 0 \Rightarrow \sum_{t=0}^{k+1} L_{i+t} - s \leq L_i \qquad (11)$$

Additionally, from the assumptions,

$$X \leq s \Rightarrow \sum_{t=0}^{k} L_{i+t} - s \leq 0 \Rightarrow \sum_{t=0}^{k+1} L_{i+t} - s \leq L_{i+k+1} \quad (12)$$

By summing equations (11), (12) we get:

$$\begin{aligned} &2\left(\sum_{t=0}^{k+1} L_{i+t} - s \right) \leq L_i + L_{i+k+1} \\ &\quad\Rightarrow \sum_{t=0}^{k+1} L_{i+t} - s \leq L_i + L_{i+k+1} - \sum_{t=0}^{k+1} L_{i+t} + s \\ &\quad\Rightarrow \sum_{t=0}^{k+1} L_{i+t} - s \leq s - \sum_{t=1}^{k} L_{i+t} \qquad (13) \end{aligned}$$

From Equations (11), (12), (13) we conclude that:

$$\sum_{t=0}^{k+1} L_{i+r} - s \leq \min\left\{ L_i, L_{i+k+1}, s - \sum_{t=1}^{k} L_{i+t} \right\}.$$

Analysis of Case $s < \min\{X, Y\}$. From the assumptions:

$$s < X \Rightarrow s - \sum_{t=0}^{k} L_{i+t} < 0 \Rightarrow s - \sum_{t=1}^{k} L_{i+t} < L_i \qquad (14)$$

Additionally, from the assumptions,

$$s < Y \Rightarrow s - \sum_{t=1}^{k+1} L_{i+t} < 0 \Rightarrow s - \sum_{t=1}^{k+1} L_{i+t} < L_{i+k+1} \quad (15)$$

By summing equations (14), (15) we get:

$$\begin{aligned} &2\left(s - \sum_{t=1}^{k} L_{i+t} \right) \leq L_i + L_{i+k+1} \\ &\quad\Rightarrow s - \sum_{t=1}^{k} L_{i+t} \leq L_i + L_{i+k+1} - s + \sum_{t=1}^{k} L_{i+t} \\ &\quad\Rightarrow s - \sum_{t=1}^{k} L_{i+t} \leq \sum_{t=0}^{k+1} L_{i+t} - s \qquad (16) \end{aligned}$$

From Equations (14), (15), (16) we conclude that:

$$s - \sum_{t=1}^{k} L_{i+t} \leq \min\left\{ L_i, L_{i+k+1}, \sum_{t=0}^{k+1} L_{i+t} - s \right\}.$$

For the last part of this proof we want to show that if s is out of the limits imposed in Theorem 1, i.e., $\sum_{t=1}^{k} L_{i+t} < s < \sum_{t=0}^{k+1} L_{i+t}$,then the expression with the minimum value is negative. In this case the corresponding span does not contribute in the counting. It is enough to show that:

- if $s < \sum_{t=1}^{k} L_{i+t}$ then $\min\left\{ L_i, L_{i+k+1}, \sum_{t=0}^{k+1} L_{i+t} - s, s - \sum_{t=1}^{k} L_{i+t} \right\} < 0$
- if $\sum_{t=0}^{k+1} L_{i+t} < s$ then $\min\left\{ L_i, L_{i+k+1}, \sum_{t=0}^{k+1} L_{i+t} - s, s - \sum_{t=1}^{k} L_{i+t} \right\} < 0$

Starting with the first case, we have:

$$s < \sum_{t=1}^{k} L_{i+t} \Rightarrow s - \sum_{t=1}^{k} L_{i+t} < 0,$$

therefore we have that,

$$\min\left\{ L_i, L_{i+k+1}, \sum_{t=0}^{k+1} L_{i+t} - s, s - \sum_{t=1}^{k} L_{i+t} \right\} < 0$$

so the first item is proved.

For the second case, we have:

$$\sum_{t=0}^{k+1} L_{i+t} < s \Rightarrow \sum_{t=0}^{k+1} L_{i+t} - s < 0,$$

therefore we have that,

$$\min\left\{ L_i, L_{i+k+1}, \sum_{t=0}^{k+1} L_{i+t} - s, s - \sum_{t=1}^{k} L_{i+t} \right\} < 0$$

so the second item is proved.

G. *Proof of Theorem 2*

Another way to express the output of $C_{\text{ANY}}(r, s)$ is to define an interval δ of all possible lower-boundaries of a range and count the number of locations such that if one places a range query of span s with a starting point among the locations of δ then the response would be r. Without loss of generality we

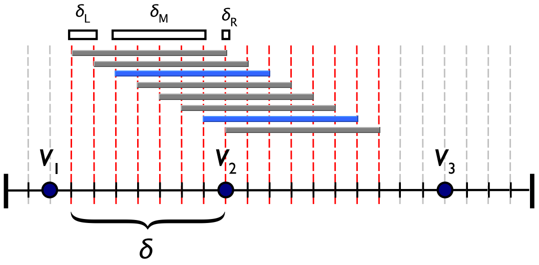

Fig. 5. An illustration of the structure of interval δ for the case of BASE and ABT. Grey intervals represent the range queries of BASE that have span $s = 2^3$ and response $r = \{v_2\}$, while blue intervals represent the range queries of ABT that have span $s = 2^3$ and response $r = \{v_2\}$. Note that the additive step for this span is $T_{\text{ABT}}[2^3] = 2^2$.

proceed with the proof using the interval δ for our arguments. Notice that not all of the location that are covered by interval δ are lower-boundaries of a canonical range of span s from the scheme ANY due to the fact that $T_{\text{Any}}[s] > 1$. E.g., the lower-boundaries that correspond to canonical ranges of scheme ANY are marked with blue in Figure 5. We further partition δ into three segments $\delta_L, \delta_M, \delta_R$ such that:

- δ_M: the interval with start-point the location that coincides with the lower-boundary of the leftmost canonical range of ANY within δ. The end-point of interval δ_M is the location that coincides with the lower-boundary of the rightmost canonical range of ANY within δ.
- δ_L: the interval with start-point the leftmost location of δ that does not belong to δ_M. The end-point of interval δ_L is the previous location from the starting point of δ_M.
- δ_R: the interval with start-point the leftmost location of δ that does not belong to δ_M or δ_L. The end-point of interval δ_L is the rightmost location of δ.

Let $|\delta|$ denote the width of the interval δ, then we have $C_{\text{Step1-Any}}(r,s) = |\delta_L| + |\delta_M| + |\delta_R|$. Notice that by construction the width $|\delta_M|$ is a multiple of $T_{\text{Any}}[s]$, i.e., let us define $|\delta_M| = k \cdot T_{\text{Any}}[s]$ for an integer k. Additionally, for the other two intervals of the partition we have $|\delta_L| < T_{\text{Any}}[s]$ and $|\delta_R| < T_{\text{Any}}[s]$. Therefore:

$$\frac{C_{\text{Step1-Any}}(r,s)}{T_{\text{Any}}[s]} = \frac{|\delta_L| + |\delta_M| + |\delta_R|}{T_{\text{Any}}[s]} = k+\frac{|\delta_L|}{T_{\text{Any}}[s]}+\frac{|\delta_R|}{T_{\text{Any}}[s]}$$

Given that $C_{\text{Any}}(r,s) = k$ we proceed with case analysis with respect to $|\delta_L|$ and $|\delta_R|$:

- *Case* $|\delta_L| = 0, |\delta_R| \neq 0$: then

$$\left\lfloor \frac{C_{\text{Step1-Any}}(r,s)}{T_{\text{Any}}[s]} \right\rfloor = \left\lfloor k + \frac{|\delta_R|}{T_{\text{Any}}[s]} \right\rfloor = k = C_{\text{Any}}(r,s) \tag{17}$$

- *Case* $|\delta_L| \neq 0, |\delta_R| = 0$: then

$$\left\lfloor \frac{C_{\text{Step1-Any}}(r,s)}{T_{\text{Any}}[s]} \right\rfloor = \left\lfloor k + \frac{|\delta_L|}{T_{\text{Any}}[s]} \right\rfloor = k = C_{\text{Any}}(r,s) \tag{18}$$

- *Case* $|\delta_L| = 0, |\delta_R| = 0$: then

$$\left\lfloor \frac{C_{\text{Step1-Any}}(r,s)}{T_{\text{Any}}[s]} \right\rfloor = \frac{C_{\text{Step1-Any}}(r,s)}{T_{\text{Any}}[s]} = k = C_{\text{Any}}(r,s) \tag{19}$$

- *Case* $|\delta_L| \neq 0, |\delta_R| \neq 0$: then

$$\left\lfloor \frac{C_{\text{Step1-Any}}(r,s)}{T_{\text{Any}}[s]} \right\rfloor = \left\lfloor k + \frac{|\delta_L|}{T_{\text{Any}}[s]} + \frac{|\delta_R|}{T_{\text{Any}}[s]} \right\rfloor \tag{20}$$

if $\frac{|\delta_L|}{T_{\text{Any}}[s]} + \frac{|\delta_R|}{T_{\text{Any}}[s]} \geq T_{\text{Any}}[s]$ then:

$$(20) \Rightarrow \left\lfloor k + \frac{|\delta_L|}{T_{\text{Any}}[s]} + \frac{|\delta_R|}{T_{\text{Any}}[s]} \right\rfloor = k{+}1 = C_{\text{Any}}(r,s){+}1 \tag{21}$$

if $\frac{|\delta_L|}{T_{\text{Any}}[s]} + \frac{|\delta_R|}{T_{\text{Any}}[s]} < T_{\text{Any}}[s]$ then:

$$(20) \Rightarrow \left\lfloor k + \frac{|\delta_L|}{T_{\text{Any}}[s]} + \frac{|\delta_R|}{T_{\text{Any}}[s]} \right\rfloor = k = C_{\text{Any}}(r,s) \tag{22}$$

From equations (17),(18),(19),(21),(22) we get:

$$\left| C_{\text{Any}}(r,s) - \left\lfloor \frac{C_{\text{Step1-Any}}(r,s)}{T_{\text{Any}}[s]} \right\rfloor \right| \leq 1. \tag{23}$$

Additionally the case analyzed in equation (21) proves that the approximation is tight. For the last part of the proof we will show that:

$$\left| G_{\text{Any}}(r) - \sum_{s=1}^{N} \max\left\{0, \left\lfloor \frac{C_{\text{Step1-Any}}(r,s)}{T_{\text{Any}}[s]} \right\rfloor \right\} \right| \leq \mathit{weight}(T_{\text{Any}}).$$

We start by analyzing the following term:

$$\sum_{s=1}^{N} \max\left\{0, \left\lfloor \frac{C_{\text{Step1-Any}}(r,s)}{T_{\text{Any}}[s]} \right\rfloor \right\}$$

Using the result derived from equation (23), let x be the number of spans among the spans $s = 0, \ldots, N$ such that $\left| C_{\text{Any}}(r,s) - \left\lfloor \frac{C_{\text{Step1-Any}}(r,s)}{T_{\text{Any}}[s]} \right\rfloor \right| = 1$. Notice that $0 \leq x \leq \mathit{weight}(T_{\text{Any}})$. Due to equation (23) we have $\mathit{weight}(T_{\text{Any}}) - x$ terms in the summation for which $\left| C_{\text{Any}}(r,s) - \left\lfloor \frac{C_{\text{Step1-Any}}(r,s)}{T_{\text{Any}}[s]} \right\rfloor \right| = 0$. Therefore

$$\begin{aligned} \sum_{s=1}^{N} \max\left\{0, \left\lfloor \frac{C_{\text{Step1-Any}}(r,s)}{T_{\text{Any}}[s]} \right\rfloor \right\} &= \sum_{s=1}^{N} (G_{\text{Any}}(r,s)) + x \\ &= G_{\text{Any}}(r) + x. \end{aligned}$$

So overall we have the following approximation:

$$\left| G_{\text{Any}}(r) - \sum_{s=1}^{N} \max\left\{0, \left\lfloor \frac{C_{\text{Step1-Any}}(r,s)}{T_{\text{Any}}[s]} \right\rfloor \right\} \right| \leq x \leq \mathit{weight}(T_{\text{Any}}).$$