Leakage Perturbation is Not Enough: Breaking Structured Encryption Using Simulated Annealing

Zichen Gui
University of Bristol

Kenneth G. Paterson
ETH Zürich

Sikhar Patranabis
ETH Zürich

June 9, 2021

Abstract

Structured encryption (STE) is a form of database encryption that enables searching directly over symmetrically encrypted "structured databases". STE is known to be vulnerable to *leakage-abuse attacks* that allow data/query reconstruction given only some auxiliary information about the original database. Many existing countermeasures against leakage-abuse attacks perturb the leakage from STE schemes so as to render the attacks infeasible in practice.

We present the first leakage-abuse attacks that achieve practically efficient and highly scalable query reconstruction against state-of-the-art STE schemes with perturbed leakage profiles while relying only no noisy co-occurrence pattern leakage and without making strong assumptions on the auxiliary information available to the adversary. Our attacks subvert the query privacy guarantees of STE schemes with differentially private access patterns (Chen *et al.*, INFOCOM'18) and STE schemes built in a naturally efficient manner from volume-hiding encrypted multi-maps (Kamara and Moataz, Eurocrypt'19 and Patel *et al.*, CCS'19).

Many existing leakage-abuse attacks only work in a strong *known-data* model where the auxiliary information available to the adversary is either an exact replica of or a "noise-free" subset of the target database. Our attacks are the first to work in a weaker and more realistic *inference* model where the auxiliary information available to the adversary is sampled independently from but statistically close to the target database. Compared to (a handful of) existing inference attacks, our attacks make significantly relaxed assumptions about the nature of auxiliary information available to the adversary.

Technically, our attacks exploit insufficiencies in existing leakage-perturbation techniques as well as novel observations surrounding inevitable *system-wide leakage* from efficient realizations of STE. We model the attacks as optimization problems with carefully designed objective functions that are maximized via simulated annealing. We demonstrate the practical effectiveness of our attacks via extensive experimentation over real-world databases. Our attacks achieve up to 90% query reconstruction against STE implementations using recommended security parameters, with 5x greater scalability than any existing attack exploiting access pattern leakage.

1 Introduction

Database Encryption. Database encryption is the study of cryptographic techniques that allow efficient query processing over encrypted databases without the need to decrypt them first. Database encryption is a key enabler for secure storage-as-a-service, wherein clients can securely outsource the storage and processing of large databases to (potentially untrusted) third party servers.

Efficiency, Leakage, Expressiveness. There exists a wide spectrum of cryptographic techniques for database encryption that offer varying tradeoffs between efficiency, information leakage, and query expressiveness. Solutions such as fully-homomorphic encryption (FHE) [19], functional encryption (FE) [8] and oblivious RAM (ORAM) [21] offer high security and query expressiveness, but are currently too inefficient in practice. Solutions such as deterministic encryption [3], order-preserving encryption [7] and, more generally, property-preserving encryption [46, 44], offer high practical efficiency and query expressiveness, but are vulnerable to attacks due to information leakage [40].

Structured Symmetric Encryption. Structured symmetric encryption (STE), introduced by Chase and Kamara in [13], is a special sub-class of database encryption that aims to support a specific class of search queries over symmetrically encrypted "structured databases". A simple example of functionality enabled by structured encryption is the following: given an encrypted document collection in which each document is tagged with keywords, find the set of all documents tagged with a given keyword w. The most well-known sub-class of structured encryption is searchable symmetric encryption (SSE) [52, 20, 15, 12, 11] that supports searching over document collections and relational databases. Most existing constructions of SSE are built from encrypted multi-maps [31, 33, 45] – an abstract STE for efficient encrypted query processing. In this paper, we focus primarily on SSE/STE for *static* document collections. This has historically received the most attention.

At a high level, a static STE scheme consists of two protocols – a *setup* protocol and a *search* protocol – executed between a client (owning a structured database) and a server (modeled as an adversary). The setup protocol allows the client to encrypt and offload the structured database to the server, such that the encrypted database can later be queried using the search protocol. In the search protocol, the client typically generates a *query token* and sends this to the server; the server then uses this token to process the query and returns the corresponding result.

Leakage. The term "leakage" is popularly used in the STE literature to denote any information that the server learns about either the database itself or the queries made by the client. For any STE scheme, leakage can be of two kinds - *setup leakage* – information learnt by the server from the encrypted database it receives at setup (i.e., prior to any query execution), and *query leakage* – information learnt by the server from the query token and the interaction between the query token and the encrypted database. Informally, an STE scheme is "more" secure if it incurs "less" leakage. Ideally, an STE scheme should be leakage-free, but this comes at huge performance overheads [52, 20]. In practice, all efficient STE schemes incur some setup and query leakage [15, 12, 11, 31, 32, 37].

Leakage Cryptanalysis. A natural question to ask is: *how do we analyze the impact of leakage on the real-world security of STE?* Unfortunately, there currently exist no formal metrics that can universally categorize leakage into "benign" (meaning that it does not impact security) or "critical" (meaning that it adversely impacts security). The practice commonly adopted in the STE literature is to perform *leakage cryptanalysis*. This involves developing concrete cryptanalytic attacks that exploit the leakage to subvert some security guarantee (such as data/query privacy) of the STE scheme. If a leakage profile stands up to such cryptanalysis over a period of time, it can then be considered benign.

Starting with the seminal work of Islam et al. [30], leakage cryptanalysis has been studied extensively in the context of STE [9, 54, 34, 35, 23, 24, 27, 5]. Generally, any leakage cryptanalysis of STE is characterized by the following factors:

- **Attack Model.** The adversary may be *snapshot* (with access to only the encrypted database) or *persistent* (with access to both the encrypted database and the "history" of query operations). Quite naturally, snapshot adversaries are less powerful as compared to persistent adversaries and are typically easier to thwart (see [25, 2, 5] for more details).
- **Attack Target.** The adversary may target either *data recovery* or *query reconstruction* (or both).
- **Attack Assumptions.** The adversary may have access to some *auxiliary data*. In a *known-data* attack, the adversary knows a subset of entries in the original database. In an *inference* attack the adversary has a set of entries that are distributed statistically close to the entries in the database (but not necessarily an actual subset of the data). In a *known-query* attack, the adversary knows a subset of the queries made by the client.
- **Attack Nature.** The adversary may either passively observe the leakage (referred to as *leakage-abuse attacks*) or actively create leakage by tampering with the client's database (referred to as *injection attacks*).

In this paper, we focus on persistent adversaries against STE for document collections. We consider adversaries targeting query reconstruction via inference attacks.

Leakage Types. An important characteristic of any leakage-abuse attack is the leakage type (also known as "leakage profile") that it exploits. The commonly studied leakage profiles with respect to STE for document collections are:

- **Response Length Pattern.** For a given query on a keyword w, the response length pattern leakage reveals the size of the query response set, i.e., number of documents containing w.
- **Co-occurrence Pattern.** For a pair of queries over keywords w_i and w_j, the co-occurrence leakage reveals the number of documents containing *both* w_i and w_j.
- **Search Pattern.** For a pair of queries over keywords w_i and w_j, the search pattern leakage reveals whether w_i and w_j are identical.

- **Access Pattern.** For a given query on a keyword w, the access pattern leakage reveals the set of (potentially randomized) identifiers pertaining to documents containing w.

Known-Data Attacks on STE. We summarize here the most well-known leakage-abuse attacks on STE for document collections. All of these attacks are known-data and exploit one or more of the aforementioned leakage profiles.

- **The IKK Attack.** The first attack to be proposed against STE for document collections was the *IKK attack* due to Islam et al. [30] that exploited co-occurrence leakage. The IKK attack was presented as an inference-style attack, but was shown to be ineffective in this form in practice by Cash et al. [9]. Hence, this attack should be classified as a known-data attack. The attack also requires known queries.
- **The Count Attack.** The Count attack was proposed by Cash et al. [9, 10] as an improved known-data attack against STE for document collections. It does not rely on known queries. It also has a higher query recovery rate. The Count attack can be instantiated in the setting where only a fraction of the documents are known to the adversary. However, experiments [9, 10, 5] have shown that the attack is only effective when over 70% of the documents are known.
- **Newer Attacks.** In a recent work, Blackstone et al. [5] introduced new known-data attacks that are significantly better than the Count attack and its variants. These attacks require no knowledge about the client's queries and perform well even when as little as 5% of the documents are known to the adversary. These attacks exploit a variety of leakage patterns. In certain cases, they exploit the (exact) response length pattern leakage pattern from the actual documents. In other cases, they exploit the full (and exact) access pattern leakage pattern. These attacks are known-data attacks: the adversary needs to know a subset of the target database.

Inference Attacks on STE. As already mentioned, known-data attacks require the auxiliary data available to the adversary to be either identical to or a subset of the target database. Inference attacks, on the other hand, make the weaker assumption that the adversary has access to auxiliary data that is *independent* of but still statistically close to the target database. This is a more realistic assumption, and has motivated the investigation of inference-based leakage-abuse attacks on STE. We summarize two such attack approaches below.

- **Graph Matching Attacks.** Pouliot and Wright [48] proposed an improved version of the IKK attack that exploits co-occurrence pattern leakage and relies on graph matching algorithms. The attack was shown to be effective against ShadowCrypt [29] and Mimesis Aegis [38], albeit under the strong assumption that the auxiliary and target datasets are *highly correlated*, i.e., the leakage is only *weakly perturbed*. As shown in [42], the attack efficiency degrades significantly when the leakage perturbation is high (which is expected to be the case in practice).
- **The SAP Attack.** In a recent preprint [42], Oya and Kerschbaum proposed a maximum likelihood estimation-based leakage-abuse attack against STE with perturbed leakage profiles, including the schemes in [14, 45, 16]. This attack, called the SAP attack, simultaneously exploits the (exact) search pattern leakage and (potentially noisy)

response length pattern leakage. However, as Oya and Kerschbaum clarify in [42], the SAP attack crucially relies on the availability of auxiliary information about query frequency patterns, which is seemingly a very strong assumption in practice.

Efficiency and Scalability. It appears that nearly all of the existing leakage-abuse attacks on STE for document collections fail to scale in a robust and efficient manner when attempting query-recovery over large query histories and/or in the presence of noisy auxiliary data. As an illustration, the analysis due to Blackstone et al. in [5] are based on the Enron database [53] for only 150 queries. The SAP attack[42] has a query recovery rate of only 22% when the auxiliary information available to the adversary has a large number of keywords, indicating a lack of robustness against high levels of noise in the leakage. However, for a leakage-abuse attack to be truly practical, it should ideally maintain efficiency when scaling to larger query histories while also being robust to the presence of high noise levels in the auxiliary data. It is currently open to design such leakage abuse attacks.

Countermeasures. In recent years, researchers have attempted to design STE schemes that specifically counter leakage-abuse attacks. Examples of such countermeasures include volume-hiding EMMs [33, 45] and STE schemes with differentially private access-patterns [14]. At a high level, these schemes aim to "perturb" the co-occurrence leakage and the access-pattern leakage from the STE scheme. Since the leakage-abuse attacks in [30, 9, 10, 5] rely on the "exact" knowledge of various leakage profiles, they either do not work or are practically infeasible against such perturbed leakage profiles. The only known attack to achieve query recovery against STE schemes with perturbed leakage profiles is the SAP attack [42]. However, this attack relies on *exact* search pattern leakage in addition to noisy access patterns. Also, as already mentioned, it makes strong assumptions on the availability of auxiliary information about query frequency patterns.

Our Contribution. In this paper, we ask the following question:

Can we design efficient and scalable inference-style leakage-abuse attacks on state-of-the-art STE schemes with perturbed leakage profiles that only rely on (noisy) co-occurrence pattern leakage and avoid strong assumptions on the availability of auxiliary information about query frequency patterns?

We answer this question in the affirmative. We propose and experimentally demonstrate the first inference-style leakage-abuse attacks that achieve efficient and highly scalable query reconstruction against STE schemes with perturbed leakage profiles while relying only on noisy co-occurrence pattern leakage and while assuming no auxiliary information about query frequency patterns. We target the following STE schemes:

- STE schemes for document collections with differentially private access patterns, such as the scheme proposed by Chen *et al.* [14].
- STE schemes for document collections that are built from various instantiations of volume-hiding encrypted multi-maps (EMMs), including those proposed by Kamara and Moataz [33], and by Patel *et al.* [45].

At a high level, our attacks exploit subtle gaps that exist between the security guarantees

Attack	Attack Assumption	Leakage Exploited	Additional Requirements	Perturbed Leakage?
IKK [30]	Known-data	Co-occurrence pattern	Known queries	No
Count [9]	Known-data	Co-occurrence pattern	Known queries*	No
BKM20 [5]	Known-data	Co-occurrence or access pattern	–	No
Graph Matching Attacks [48]	Inference	Co-occurrence pattern	–	Yes**
SAP [42]***	Inference	Search pattern and response length	Query frequency patterns (auxiliary information)	Yes
GPPW20 [28]****	Inference	Co-occurrence pattern	–	Yes
This work	Inference	Co-occurrence pattern	–	Yes

Table 1: Comparison of existing passive and persistent query reconstruction attacks based on co-occurrence and/or access pattern leakage. *Count attack does not need known queries if the entire database is known by the attacker; known queries are only helpful when only part of the database is known by the attacker. ** The attack targets in [48] have weakly perturbed leakage. *** The SAP attack makes strong assumptions on the availability of auxiliary information about query frequency patterns. **** Gui *et al.* [28] only presented a system-wide leakage based cryptanalysis of their own construction called SWiSSSE while making stronger assumptions on the auxiliary leakage available to the adversary as compared to our attacks. They also did not propose any attacks on differentially private access patterns.

that would be expected for the aforementioned STE schemes and the actual security guarantees that they achieve when used to design functioning STE systems over large databases. In certain cases (such as for STE with differentially private access patterns), these gaps arise from insufficiencies in the security guarantees provided by perturbed leakage profiles over real-world databases. In other cases (such as for volume-hiding EMMs), these gaps arise from the fact that the proposed countermeasures can only be applied (in a scalable and efficient manner) to a small sub-component of the overall STE system and fail to mask additional *system-wide leakage* that inevitably arises. This system-wide leakage can in turn be abused for query reconstruction.

Our attacks use standard simulated annealing techniques to converge on a solution assigning each query to a keyword. We use an objective function that is derived from the likelihood of observing a given assignment of keywords to queries, given the leakage profile and the auxiliary data as prior information. Maximising the objective function (as simulated annealing attempts to do) then corresponds to maximising the log likelihood of the solution. Thus the simulated annealing, if it works, will produce "good" solutions in which many keywords in the solution are correctly assigned to queries. This approach requires careful mathematical analysis to derive the likelihood functions for each targeted scheme and to efficiently implement their evaluation on large sets of queries and leakage. We demonstrate the practical effectiveness of our attacks via extensive experimentation using the Enron email corpus [53] as the target database.

Table 1 presents a summary of our attacks and compares them against existing leakage-abuse attacks. We expand further on our key observations and attack techniques below.

1.1 Attacks on Differentially Private Access Pattern Leakage

We present the first inference-style leakage-abuse attack based solely on noisy co-occurrence pattern leakage on an STE scheme with differentially private access patterns. We target

the scheme proposed by Chen *et al.* [14] (referred to as **DPAP-SE** henceforth). Our key contributions include: (a) developing a rigorous mathematical model for the "noisy" co-occurrence leakage patterns incurred by **DPAP-SE**, and (b) designing new attacks that can reconstruct queries given only this model.

Our attacks subvert the query privacy guarantees of an implementation of **DPAP-SE** for the *same* parameter set that the authors of [14] advocate using to counter leakage-abuse attacks. While it is possible to degrade our attack's efficiency by altering the parameter set, this also greatly reduces the practical efficiency of the resulting scheme. In the end, the query privacy of any reasonably efficient instantiation of **DPAP-SE** is broken by our attacks.

Implications of Our Attacks. Our attacks do not invalidate the original security claims of **DPAP-SE**, but indicate these security claims (and the corresponding usage of differential privacy techniques) are, in fact, insufficient in practice. **DPAP-SE** does resist naïve adaptations of existing leakage-abuse attacks [30, 9, 10, 5] that rely on the *exact* leakage such as response length patterns, co-occurrence patterns and access patterns. This is precisely the claim made by the authors of [14], and our attacks do not invalidate these claims. Rather, we demonstrate the possibility of stronger attacks that *do not* rely on such exact leakage and, hence, bypass the limitations of existing leakage-abuse attacks.

1.2 Attacks on System-Wide Leakage

We make critical observations surrounding "system-wide" leakage incurred by STE schemes in practice and their impact on query privacy. We illustrate, both theoretically as well as via practical experiments, that system-wide leakage can be exploited to launch stronger leakage-abuse attacks against state-of-the-art STE schemes for document collections built from volume-hiding EMMs, including the **PRT-EMM** schme due to Kamara and Moataz [33], as well as the **FP-EMM** and **DP-EMM** schemes due to Patel *et al.* [45].

We note here that our attacks do not subvert the security guarantees of volume-hiding EMMs themselves. Rather, they exploit the gap between the "ideal" security guarantees achieved by volume-hiding EMMs operating in isolation, and the "real-world" security guarantees that efficient STE implementations built from volume-hiding EMMs actually achieve in practice.

Volume-Hiding EMMs vs End-to-End STE Systems. Our attacks stem from the observation that while volume-hiding EMMs are, in isolation, resistant to leakage-abuse attacks, the same is not true for (efficient and scalable) end-to-end STE systems for document collections built from volume-hiding EMMs. State-of-the-art STE schemes for document collections (e.g., [15, 12, 11, 31, 37])) only use EMMs as a sub-component to realize a *encrypted search index* and associated operations on that index. However, EMMs no longer appear in the query processing step where the client *actually* fetches the encrypted documents matching a query.

We note here that this approach of using (volume-hiding) EMMs as a sub-component is, in fact, a direct and natural instantiation of the *structure-only* approach to designing STE schemes introduced and formalized by Chase and Kamara in [13]. In this approach, the designer separates the actual data items to be encrypted from the structures (e.g., the search index) designed to search over these data items, and uses dedicated STE techniques (e.g., EMMs) to only encrypt these data structures. Almost all state-of-the-art STE schemes today use this approach because it naturally yields the most modular and efficient constructions in practice.

In theory, one could apply volume-hiding EMMs to encrypt the whole database, and not just the search index. However, this is prohibitively expensive for large databases. For example, our experiments show that for the Enron email database [53], there is a $36\times$ storage overhead incurred by applying volume-hiding EMMs to encrypt whole database (see discussion in Appendix D.1 for more details). In addition, existing volume hiding EMMs typically incur additional computational and communication overheads during query execution due to their usage of padding techniques; while this cost is manageable when querying the search index alone, it blows up to impractical proportions if applied directly to encrypted document retrieval.

To summarize, with currently available techniques, we can have either efficient and scalable STE schemes where volume-hiding EMMs protect only a sub-component of the overall system, or we can have STE schemes protected end-to-end using volume-hiding EMMs that are inefficient in practice. As a result, *efficient* state-of-the art STE schemes built from volume-hiding EMMs will inevitably incur additional leakage during encrypted document retrieval.

System-Wide Leakage. We refer to the leakage incurred during encrypted document retrieval as *system-wide leakage* since it only appears in our analysis when we take a system-wide view of STE (as opposed to focussing on individual components e.g., the encrypted search index). In particular, existing security definitions of volume-hiding EMMs and, more generally, STE for document collections, fail to capture system-wide leakage. In this paper, we demonstrate that the careful leakage mitigation for the encrypted search index is, in effect, undermined by the inevitable system-wide leakage that arises from the STE system as a whole.

Modeling and Attacking System-Wide Leakage. In a prior work, Gui *et al.* [28] discussed system-wide leakage in STE schemes, and its impact on the security guarantees of STE schemes. However, they did not specifically focus on the impact of system-wide leakage on volume-hiding EMMs and their usage in STE schemes; they only presented a system-wide leakage based cryptanalysis of their own construction called SWiSSSE. In this paper, we refine and extend the analysis of [28] to develop full-fledged query reconstruction attacks against state-of-the-art STE schemes built from different variants of volume-hiding EMMs, including **PRT-EMM** [33], as well as **FP-EMM** and **DP-EMM** [45]. Our attacks work for volume-hiding EMM implementations using the same parameters and the same design/implementation choices advocated in [33] and [45].

We show how to model the noisy co-occurrence leakage pattern from STE schemes using volume-hiding EMMs as a function of the original database, the keyword queries and system-wide leakage. To our knowledge, this is the first attempt to formally model the co-occurrence leakage pattern of an STE scheme in the presence of system-wide leakage. We then use these models to develop new inference-style leakage-abuse attacks that can reconstruct queries from a combination of: (a) noisy co-occurrence leakage from volume-hiding EMMs and (b) additional system-wide leakage from document recovery. Beyond query reconstruction, our attacks are also capable of (approximate) database reconstruction in certain cases (see Section 4.4 for more details).

It is important to reiterate that our targets here are STEs built from the volume-hiding EMMs proposed in [33, 45]. One might be concerned that, since [33, 45] did not propose actual STEs but only EMMs, it is misleading to target specific STEs built from their EMMs. We argue that (volume-hiding) EMMs on their own are not very useful, that the authors of [33, 45] are clear that STEs are the main application domain for their EMMs, and that the constructions of STEs we target are the natural, efficient ones that one obtains when using EMMs as a starting point and when following the *structure-only* approach introduced and formalized in [13].

Implications of Our Attacks. Our attacks essentially underline the fact that existing proofs of security for (volume-hiding) EMMs, and more generally STE scheme built from volume-hiding EMMs via the *structure-only* approach proposed in [13], ignore system-wide leakage, and thus fail to account for potential attacks abusing this leakage. We believe that this motivates a more fundamental change of perspective with respect to STE design and leakage-mitigation. While the existing approach of focusing on the leakage arising from specific sub-components (such as the encrypted search index) of STE schemes facilitates modular design and analysis, it also brings the risk of incomplete leakage analysis from the system as a whole.

1.3 Inference Attacks: Techniques and Evaluation

A leakage-abuse attack is an *inference* attack when the adversary only has access to auxiliary data that is *independent* but statistically "close" to the target database. As mentioned earlier, the existing leakage-abuse attacks [30, 9, 10, 5] do not work as inference attacks as they necessarily rely on stronger models of auxiliary data. We make the first concrete progress towards practical realizations of inference-style attacks based on noisy co-occurrence pattern leakage. All of our leakage-abuse attacks on differentially private access patterns and system-wide leakage are inference attacks.

Our Techniques. We achieve inference-style leakage-abuse attacks by using a combination of statistical modeling and simulated annealing. The core technical idea is quite simple and natural. We model the attack as an optimization problem, where the objective function is the statistical likelihood of observing a given assignment of keywords to queries, given the observed leakage and auxiliary data as prior information. We then maximise the objective function using simulated annealing (we developed our own implementation of simulated

annealing for speed and flexibility, but one could use any off-the-shelf implementation here). The bigger hurdle lies in identifying and mathematically modeling the (potentially "noisy") leakage information available to the adversary as a function of the auxiliary data and a given keyword assignment, and how to transform the resulting model into an appropriately structured input to the simulated annealing algorithm.

Experimental Evaluation. In Section 5, we present extensive experimental evaluations to validate the practicality of our proposed attacks. Rather surprisingly, our apparently simple approach yields extremely powerful leakage-abuse attacks against state-of-the-art STE schemes equipped with leakage-suppressing countermeasure techniques. Our experiments show that our attacks achieve high success rate with reasonable practical efficiency even if: (a) the target STE schemes use aggressive security parameters (beyond those advocated by the authors of these schemes); (b) the keyword universe of the auxiliary information is significantly larger than the set of queried keywords (existing attacks); and (c) the auxiliary information available to the adversary is very "noisy".

As in prior work on leakage-abuse attacks (notably [5]), we use the Enron email corpus [53] as the target database for our evaluations. We use uniformly distributed keyword queries to evaluate our attacks. This is exactly as used in previous attacks [30, 9, 5]. We opt to split the overall database into two halves: a selection from one half is used to form the auxiliary data available to the adversary, while the other half is used as the target database for attack evaluation. We present additional discussion justifying this choice in Section 5.

Our attacks achieve over 75% query reconstruction rate in most of the settings we have tested. These include settings where the target STE schemes use aggressive security parameters even beyond those originally proposed, as well as settings where the auxiliary information available to the adversary is significantly perturbed (with upto 3000 keywords). As a comparison, for similar security and noise parameters, the SAP attack [42] achieves only 22% query reconstruction rate; we achieve $3.5x$ better query recovery rate while relying on a weaker leakage profile and while making more realistic assumptions about auxiliary data. In certain cases, our attacks achieve more than 90% query reconstruction rate when the parameters for the target STE schemes are identical to those originally proposed. Finally, our attacks achieve $5\times$ greater scalability than any existing leakage-abuse attack.

1.4 Other Related Work

Naveed *et al.* [40] proposed the first inference attacks against property-preserving encryption (PPE), and more specifically against CryptDB [6], using techniques such as ℓ_2-optimization and frequency analysis. Their attacks showcased vulnerabilities in well-known sub-classes of PPE such as deterministic encryption [3] and order-preserving encryption [6]. Other well-known inference attacks on PPE include the graph-based attack due to Grubbs *et al.* [26], the maximum likelihood estimation-based attack due to Lacharité and Paterson [36], and the Bayesian inference-based attack due to Bindschaedler *et al.* [4]. Our attacks share some similarities with the latter two attacks [36, 4], albeit for a different class of target schemes.

2 Preliminaries

2.1 Abstract Data Types

An abstract data type is a collection of data objects and a set of operations defined on those objects. For instance, set with an operation to initialise a set and the common set operations is an abstract data type. Operations on abstract data types can be broadly categorised into two groups: static operations which do not change the data objects; and dynamic operations which may change the data objects. Our attacks in this paper focus on abstract data types with static operations only.

2.2 Syntax of Static Structured Encryption

Let $\mathcal{T}$ be an abstract data type supporting query operation **Query**. Then, a private-key structured encryption scheme Σ for $\mathcal{T}$ is a tuple $\Sigma = (\mathbf{Setup}, \mathbf{Query}_e)$ where:

- **Setup** is the setup algorithm which takes as input some data **D** of structure $\mathcal{T}$, and outputs a secret key sk and some encrypted data **ED**.
- $\mathbf{Query}_e$ is the query protocol between the client and server. The client takes as input a secret key sk and a query q, and the server takes as input some encrypted data **ED**; after the interaction between the client and server, the client obtains a response rsp.

Correctness. We say that scheme Σ is correct if for all data **D** and all queries q, an execution of the query protocol $\mathbf{Query}_e$ on the encrypted data $\mathbf{ED} \leftarrow \mathbf{Setup}(sk, \mathbf{D})$ yields the same response as an execution of query operation **Query** on data **D** and query q.

2.3 Security of Structured Encryption

Unlike some other primitives, structured encryption needs to leak some information to be efficient. Hence, its security is parametrised by *leakage*, that is, an upper bound on the information of the data and queries for which an attacker can learn from the **Setup** algorithm and subsequent queries. Formally, security of structured encryption can be defined as follows.

Definition 1 (CQA2-security)**.** Let $\Sigma = (\mathbf{Setup}, \mathbf{Query}_e)$ be a private-key structured encryption scheme for abstract data type $\mathcal{T}$. Consider the following probabilistic experiments between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:

- $\mathbf{Real}_{\Sigma,\mathcal{A}}(k)$: the adversary $\mathcal{A}$ generates data **D** and sends it to the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ runs the **Setup** algorithm to generate a secret key sk and some encrypted data **ED**. The encrypted data **ED** is sent to the adversary. After that, the adversary picks a polynomial number of queries adaptively and send them to the challenger. The challenger and adversary executes the $\mathbf{Query}_e$ protocol on the queries where the

challenger plays the client and the adversary plays the server. Finally, the adversary outputs a bit b that is output by the experiment.

- $\mathbf{Ideal}_{\Sigma,\mathcal{A},\mathcal{S}}(k)$: the adversary $\mathcal{A}$ generates data $\mathbf{D}$ and $\mathcal{L}_{\mathbf{Setup}}(\mathbf{D})$ is sent to the simulator $\mathcal{S}$. The simulator $\mathcal{S}$ generates encrypted data $\mathbf{ED}$ using the leakage and sends it back to the adversary. The adversary picks a polynomial number of queries $\mathsf{q}_1, \ldots, \mathsf{q}_l$ adaptively and $\mathcal{L}_{\mathbf{Query}_e}(\mathsf{q}_1, \mathbf{D}), \ldots, \mathcal{L}_{\mathbf{Query}_e}(\mathsf{q}_l, \mathbf{D})$ is sent to the simulator. The simulator and adversary executes the $\mathbf{Query}_e$ protocol on the queries where the simulator plays the client and the adversary plays the server. Finally, the adversary outputs a bit b that is output by the experiment.

We say that Σ is $(\mathcal{L}_{\mathbf{Setup}}, \mathcal{L}_{\mathbf{Query}_e})$-secure against adaptive chosen-query attacks if for all probabilistic polynomial-time (PPT) adversaries $\mathcal{A}$, there exists a PPT simulator $\mathcal{S}$ such that

$$|\mathbf{Pr}\left[\mathbf{Real}_{\Sigma,\mathcal{A}}(k) = 1\right] - \mathbf{Pr}\left[\mathbf{Ideal}_{\Sigma,\mathcal{A},\mathcal{S}}(k) = 1\right]| \leq \mathtt{negl}(k).$$

2.4 Abstract Data Type in our Attacks

For the rest of our paper, we use database for searchable encryption as target abstract data type. A database $\mathbf{DB}$ consists of a set of documents d_i, each associated to a set of keywords $\mathrm{kw}(\mathsf{d}_i)$, so $\mathbf{DB} = \{(\mathsf{d}_i, \mathrm{kw}(\mathsf{d}_i))\}$. It supports keyword search queries. For a keyword search query q on keyword $\mathrm{kw}(\mathsf{q})$, the set of documents containing the keyword $\{\mathsf{d}_i \mid \mathrm{kw}(\mathsf{q}) \in \mathrm{kw}(\mathsf{d}_i)\}$ is returned. To emphasis that we are only considering keyword search queries, we denote the query protocol as $\mathbf{Srch}$.

Multi-maps have a different abstract data type to database for searchable encryption, but it can be transformed to look like the latter. Specifically, instead of representing the data type as a set of key-value pairs $\left\{(\mathsf{key}_i, \overrightarrow{\mathsf{v}_i})\right\}$, it can be represented as a set of value-key pairs $\left\{(\mathsf{v}_i, \overrightarrow{\mathsf{key}_i})\right\}$. The latter representation takes the same shape as a database for searchable encryption and has the same functionality. Indeed, encrypted multi-maps [31, 33, 45] are proposed to be used as searchable encryption schemes. As a result, an attack against a certain leakage from a searchable encryption scheme is applicable directly to an encrypted multi-map scheme with the same leakage.

3 Formal Description of Query Reconstruction Attacks using Co-Occurrence Leakage

This section establishes the leakage we are targeting in the paper and the attack setting.

Access-pattern Leakage. Access-pattern leakage refers to the information leakage associated to document retrieval. For instance, in a naïve searchable encryption scheme, the $\mathbf{Srch}$ protocol may return the exact set of encrypted documents matching the queried key-

word, hence, leaking the information that the given set of documents contains the queried keyword. We use an example to demonstrate how it is represented below.

If scheme Σ leaks the "exact" access pattern and nothing else, we can write the leakage of a query q on document collection $\mathbf{DB}$ as

$$\mathcal{L}_{\mathbf{Srch}}(\mathsf{q}, \mathbf{DB}) = \{i \mid \mathrm{kw}(\mathsf{q}) \in \mathrm{kw}(\mathsf{d}_i), (\mathsf{d}_i, \mathrm{kw}(\mathsf{d}_i)) \in \mathbf{DB}\},$$

where $\mathrm{kw}(\mathsf{q})$ denotes the keyword associated to query q. We note that although the leakage here is represented by the document identifiers, it is equivalent to an encrypted document based representation. We choose the former as the representation is more compact.

Co-Occurrence Leakage. Access-pattern leakage from different queries can be represented equivalently as a matrix, known as co-occurrence pattern. Consider a small document collection $\mathbf{DB}$ where

$$\mathbf{DB} = \{(\mathsf{d}_1, \{\mathrm{kw}_1, \mathrm{kw}_2, \mathrm{kw}_3\}), (\mathsf{d}_2, \{\mathrm{kw}_1, \mathrm{kw}_2\}), (\mathsf{d}_3, \{\mathrm{kw}_3\})\}.$$

Let q_i be a query on keyword kw_i. If the original access pattern is leaked, we know that

$$\mathcal{L}_{\mathbf{Srch}}(\mathsf{q}_1, \mathbf{DB}) = \mathcal{L}_{\mathbf{Srch}}(\mathsf{q}_2, \mathbf{DB}) = \{1, 2\}, \mathcal{L}_{\mathbf{Srch}}(\mathsf{q}_3, \mathbf{DB}) = \{1, 3\}.$$

This allows us to take intersections between the leakages as follows:

$$\begin{aligned} &\mathcal{L}_{\mathbf{Srch}}(\mathsf{q}_1, \mathbf{DB}) \cap \mathcal{L}_{\mathbf{Srch}}(\mathsf{q}_2, \mathbf{DB}) = \{1, 2\}, \\ &\mathcal{L}_{\mathbf{Srch}}(\mathsf{q}_\ell, \mathbf{DB}) \cap \mathcal{L}_{\mathbf{Srch}}(\mathsf{q}_3, \mathbf{DB}) = \{1\} \text{ for } \ell \in \{1, 2\}. \end{aligned}$$

The cardinality of the intersections can be very useful in an attack. For example, the co-occurrence pattern of the document collection above can be represented as a *co-occurrence matrix* $\bar{M}$:

$$\bar{M}(\mathsf{q}_1, \mathsf{q}_2, \mathsf{q}_3; \mathbf{DB}) = \begin{bmatrix} 2 & 2 & 1 \\ 2 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix},$$

where the i, j-th entry of the matrix is

$$\left|\mathcal{L}_{\mathbf{Srch}}(\mathsf{q}_i, \mathbf{DB}) \cap \mathcal{L}_{\mathbf{Srch}}(\mathsf{q}_j, \mathbf{DB})\right|.$$

If we know the underlying document collection perfectly, we can re-identify q_3 as a query on kw_3 as it is the only keyword that only shares one document with other keywords. This qualifies as a query reconstruction attack.

We note that a co-occurrence matrix contains strictly less information than the original access-pattern leakage as the information on intersections of more than two queries are removed. However, the co-occurrence matrix is often sufficient in attacks so it is used instead of the full leakage. We refer to the co-occurrence matrix as the co-occurrence leakage.

There are three complications to the representation of co-occurrence leakage in practice. Firstly, the schemes we consider in practice usually leak query equality pattern too. That

is, if $\mathrm{kw}(\mathsf{q}_i) = \mathrm{kw}(\mathsf{q}_j)$, the attacker knows that the two queries are for the same keyword. In terms of co-occurrence leakage, we use only one of the queries in the representation to simplify the problem. Secondly, the queries are unordered in practice. That means there is no standard representation of the leakage in terms of the known keywords. We use the convention that the i-th row and column of the co-occurrence matrix corresponds to the i-th non-repeating query in our representation. Finally, not all schemes leak the original access pattern and some schemes may even be randomised. In those cases, we need to use a suitable representation of the co-occurrence information, which may differ from what we have described above.

Auxiliary Information. Similar to a co-occurrence matrix, the auxiliary information the attacker receives can be represented as a co-occurrence matrix M. The co-occurrence matrix is indexed by the known keywords and typically contains full information on all keywords. In stronger attacks, M is assumed to be noisy in the sense that it is not generated directly from the target document collection. Instead, an auxiliary dataset is used for the purpose.

Let $\mathbf{DB} = \{(\mathsf{d}_i, \mathrm{kw}(\mathsf{d}_i))\}$ be an auxiliary document collection with keywords $\{\mathrm{kw}_1, \ldots, \mathrm{kw}_n\}$. In our attack, the i, j-th entry of M represents the empirical probability (derived from the auxiliary data $\mathbf{DB}$) of seeing kw_i and kw_j together in a document. It is computed as: $M_{i,j} = |\{\mathsf{d}_i \mid \mathrm{kw}_i \in \mathrm{kw}(\mathsf{d}_i) \wedge \mathrm{kw}_j \in \mathrm{kw}(\mathsf{d}_i)\}|/|\mathbf{DB}|$, where $|\mathbf{DB}|$ denotes the number of documents in the collection.

Attack Setting. Let Σ be a structured encryption scheme. Our attack exploits its co-occurrence leakage $\bar{M}(\cdot; \mathbf{DB})$ from retrieval of actual documents. If the scheme is an index-only one, we assume that the scheme used for document retrievals leaks the full access pattern induced by the index-only scheme. We abuse the notation $\bar{M}(\cdot; \mathbf{DB})$ to mean co-occurrence leakage from document retrieval, and the said leakage is used in our query reconstruction attacks.

We can describe a query reconstruction attack formally as follows. Let queries $\mathsf{q}_1, \ldots, \mathsf{q}_l$ be a sequence of queries on the document collection, so the attacker observes co-occurrence leakage $\bar{M}(\mathsf{q}_1, \ldots, \mathsf{q}_l; \mathbf{DB})$. Suppose that the attacker has access to some auxiliary information M. *The goal of the attacker is to recover $kw(\mathsf{q}_i)$ after observing the co-occurrence leakage $\bar{M}$ and knowing auxiliary information M.*

4 New Query Reconstruction Attacks using Co-Occurrence Leakage

4.1 Attack Overview

Attacks on Differentially Private Access Pattern Leakage. Differential privacy [18] is a statistical methodology to publish aggregated statistics without disclosing information on individuals. It is achieved by applying an obfuscation mechanism on a database, such that the chance of obtaining the resultant database is almost the same as if an indi-

vidual (or an entry) is removed before applying the obfuscation mechanism. This protects information of individuals as they are not 'important' in the final database.

This idea was extended to structured encryption as a mechanism to hide access-pattern leakage in many works [39, 14, 50]. In our paper, we show differentially private access pattern does not imply security against query reconstruction attacks using [14] as a case study.

Attacks on System-Wide Leakage. As discussed in the introduction, the most natural and efficient approach to use EMMs to build an end-to-end STE systems incurs *system-wide* leakage. In this paper, we exploit system-wide leakage in the form of co-occurrence leakage (from encrypted document retrievals) from end-to-end STE built for document collections from volume-hiding EMMs [33, 45].

4.2 Attack Targets and Co-occurrence Leakages

Searchable Encryption with Differentially-Private Access Pattern. Chen *et al.* proposed using differential privacy as a means to prevent leakage-abuse attacks that exploit access pattern leakages. They proposed an STE scheme for document collections where a differential privacy mechanism is used to obfuscate the plaintext database before building an encrypted database, such that a slight change in the real access pattern does not affect the obfuscated access pattern significantly. There are two key ingredients in their construction. Firstly, they used an erasure code [17] to split every document into m shards, each with size $\frac{1}{k}$ of the original document. The erasure code has the property that any k shards of a document can be used to reconstruct the original document. The client then picks two probabilities p and q, and does the following to each shard:

1. For any keyword that is originally in the shard, remove the keyword with probability $1 - p$.
2. For any keyword that is originally not in the shard, add it to the shard with probability q.

We refer to this scheme as **DPAP-SE**.

Intuitively, a smaller p and a larger q means more distortion to the co-occurrence information, and hence more "secure" against access-pattern leakage attacks. However, to ensure that enough shards are returned with high probability, p has to be decently large. Similarly, to control the communication overhead, q has to be small. In terms of the effect of the countermeasure on the co-occurrence leakage, it transforms query response lengths into noisy keyword response lengths on the shards; co-occurrence counts are now leaked as noisy co-occurrences on the shards. The derivation of the distribution of the co-occurrence leakage can be found in Appendix A.

Volume-hiding EMMs via Pseudo-random Transform. As a potential countermeasure to leakage-abuse attacks, Kamara and Moataz [33] introduced the concept of *volume-*

hiding encrypted multi-maps (EMMs) that hide response length patterns (and exact access patterns) while providing better search performance compared to naïve (or worst-case) padding. Kamara and Moataz proposed the first construction of volume-hiding EMMs based on an obfuscation mechanism called *pseudo-random transform*. Their idea is to pad or truncate the query response lengths of queries on a multi-map with a pseudo-random function as follows. Let key be a key for the multi-map and $F_{sk}(\cdot)$ be a pseudo-random function with key sk. The client computes: $n'_{\mathsf{key}} = \lambda + F_{sk}(\mathsf{key}||n_{\mathsf{key}})$, as the new query response length, where λ is a free parameter which the client can choose and n_{key} is the original query response length. These new query response lengths are used to build a multi-map on *document identifiers* as follows:

- If $n_{\mathsf{key}} \leq n'_{\mathsf{key}}$, add $\perp$ symbols in the multi-map on key key before encryption.
- If $n_{\mathsf{key}} > n'_{\mathsf{key}}$, truncate the multi-map on keyword key to the first n'_{key} entries.

The multi-map is then encrypted with an underlying encrypted multi-map scheme and uploaded to the server. We refer to this encrypted multi-map scheme as **PRT-EMM**.

It was not clear if the underlying encrypted multi-map scheme is an index-only one or not – our attack applies when it is, and we assume so based on our argument in the introduction.

We note that the original construction pads query responses with $\perp$ symbols if the real query response length is shorter. If the $\perp$ symbols are ignored in actual document retrieval, the attacker will be able to learn the true query response lengths when there is padding. In our attack, we assume the $\perp$ symbols are replaced by randomly picked indices, so the true query response lengths are not leaked. The derivation of the distribution of the co-occurrence leakage for **PRT-EMM** can be found in Appendix A.

Newer Constructions of Volume-hiding EMMs . Recently, Patel *et al.* proposed two volume-hiding EMM constructions in [45]. Both of the constructions use Cuckoo hashing [43] as the underlying data structure. Just like **PRT-EMM** [33], we assume that the constructions are index-only schemes.

The two schemes proposed by the authors are only different in terms of the padding mechanism on the query response lengths. The first scheme uses full padding, meaning that all query response lengths are padded to the maximum query response length. In terms of the hash table, this is done by querying additional addresses deterministically (generated by a pseudo-random function) for each key. We refer to this scheme as **FP-EMM**.

The second scheme uses differentially-private volume hiding as opposed to full padding. Let $2n_{\mathsf{key}}$ be the true query response length of a query on key key, where 2 comes from the fact that Cuckoo hashing uses two hash tables. Then the scheme pads the query response length to $2n_{\mathsf{key}} + n^* + \mathbf{Lap}_{sk}(2/\epsilon)$, where n^* is a parameter set by the client to offset the query response length in case the latter random variable is negative, and $\mathbf{Lap}_{sk}(\cdot)$ is a Laplace distribution with secret key sk as the seed. We refer to this scheme as **DP-EMM**.

The derivation of the distributions of the co-occurrence leakages for **FP-EMM** and **DP-EMM** can be found in Appendix A.

4.3 Attack Model

Attack Overview. Given the observed co-occurrence matrix $\bar{M}$ and auxiliary co-occurrence matrix M (which may not have the same dimensions), the goal of the adversary is to find an assignment P between the queries and the keywords such that the observed co-occurrence matrix fits the auxiliary information. In our formulation, the diagonal entries in the observed co-occurrence matrix are the query response lengths, and the off-diagonal entries are the number of documents accessed by two queries at the same time; the diagonal entries in the auxiliary co-occurrence matrix are probabilities such that a document contains the given keywords. For simplicity, we assume identical and independent distribution of the keywords, meaning that the true query response lengths can be modelled as binomial distributions. Furthermore, we assume that the off-diagonal entries in the auxiliary co-occurrence matrix specify the distribution of co-occurrences of keywords, and we use multinomial distributions to model the distributions. Given that there is randomness in the generation of leakage, we propose to use a likelihood function $\mathbf{L}\left[P \mid \bar{M}, M\right]$ to measure the fitness of the data. As the search space for the assignment is huge, a brute-force approach is impractical. We propose to use simulated annealing [1] to search for the most likely assignment. In the next section, we explain how simulated annealing works and outline the subroutines of the algorithm.

Simulated Annealing. We give a brief overview of simulated annealing [1] in this section. Simulated annealing is a probabilistic technique for searching for the global optimum of a given function. It is very similar to a greedy search algorithm – randomize the input of the function, in our case, that is the assignment P, recompute the score, and if the score is larger than before, the assignment is kept as the new solution, and it is discarded otherwise – except that a worse solution is accepted in simulated annealing if it is not *too bad*. This is to prevent the algorithm from sticking in a local optimum. More concretely, simulated annealing uses a *temperature* T which decreases per iteration and the differences between the current score of the target function and the previous best score maintained by the algorithm to compute an acceptance probability p, and with probability p the new solution is accepted. This probability is 1 if the new score is higher than the previous best, and less than 1 otherwise. For the same difference in the scores, a lower temperature T leads to a lower acceptance probability, which means simulated annealing acts more and more like a greedy search algorithm as the iterations go on.

Formally, simulated annealing consists of five subroutines, namely a function `InitPerm` to generate an initial assignment, a cooling scheme `Cooling`, a neighbourhood generation algorithm `Neighbour`, a function `Score` to compute the score and a function `AccptProb` to compute the acceptance probability. The syntax of the subroutines are defined below:

- `InitPerm`: takes as input an observed co-occurrence matrix $\bar{M}$ and a auxiliary co-occurrence matrix M, and outputs an assignment P.
- `Cooling`: takes as input a temperature T and the current iteration number i and outputs a new temperature T'.
- `Neighbour`: takes as input a assignment P, an observed co-occurrence matrix $\bar{M}$ and a auxiliary co-occurrence matrix M, and output a new assignment P'.

- `Score`: takes as input an observed co-occurrence matrix $\bar{M}$, a auxiliary co-occurrence matrix M and an assignment P, and output a score.
- `AccptProb`: takes as input a temperature T, a previous best score s and the new score s', and output a probability.

We are now ready to give an overview of simulated annealing. The algorithm begins with an initial temperature T_0 and a random assignment P. An initial score s is computed on this assignment P. Then, the algorithm computes a new temperature $T \leftarrow \texttt{Cooling}(T_0, 1)$, find a new assignment P' using the neighbourhood function $\texttt{Neighbour}(\cdot)$, and compute a new score s' with the score function $\texttt{Score}(\cdot)$. An acceptance probability is computed as $p \leftarrow \texttt{AccptProb}(T, s, s')$. A random number between 0 and 1 is generated and if the random number is less or equal to p, the new solution s' is accepted by the algorithm and kept as the new optimum solution. This process is repeated until the maximum number of iteration is reached. Pseudocode for our simulated annealing approach is presented in Algorithm 1.

Algorithm 1 Simulated Annealing

```
1:  procedure ATTACK(M̄, M, T_0, i_max)
2:      P ← InitPerm(M̄, M)
3:      T ← T_0
4:      s ← Score(M̄, M, P)
5:      for i ← 1, ..., i_max do
6:          T ← Cooling(T, i)
7:          P' ← Neighbour(P, M̄, M)
8:          s' ← Score(M̄, M, P')
9:          if AccptProb(T, s, s') > rand(0, 1) then
10:             P ← P'
11:             s ← s'
12:     return P
```

Application of Simulated Annealing to Query Reconstruction Attacks using Access-pattern Leakage. In this section, we specify the subroutines we used in our attacks. We used $T' \leftarrow 0.995T$ as our cooling scheme $\texttt{Cooling}(\cdot)$ and $p \leftarrow \exp(-\frac{s-s'}{T})$ as our function $\texttt{AccptProb}(\cdot)$ to compute the acceptance probability for all three leakages we have considered. We used the likelihood functions as the score functions $\texttt{Score}(\cdot)$, and detailed derivations of them are shown in Appendix B. We find the choices of $\texttt{InitPerm}(\cdot)$ and $\texttt{Neighbour}(\cdot)$ have a significant impact on the performance and effectiveness of our attacks. The subroutines presented in this section are the most effective variants we have found.

Initial Assignment Finding Subroutine $\texttt{InitPerm}(\cdot)$**.** An initial assignment finding subroutine $\texttt{InitPerm}(\cdot)$ is an efficient algorithm for guessing keywords/keys of the queries, so as to provide a starting point for the more expensive iterative steps later. For our attacks, only the query response lengths are used to avoid expensive computations. We observe that although the observed query response lengths are different from the true query response lengths for all of the schemes we target, these two are related. In particular, for **DPAP-SE** [14] and **DP-EMM** [45], we can compute the expected observed query response

lengths from the query response lengths in the auxiliary co-occurrence matrix, and match the queries to the keywords in the auxiliary co-occurrence matrix as well as we can. For **PRT-EMM** [33] and **FP-EMM** [45], the observed keyword frequencies are independent from the true keyword frequencies, .

Neighbourhood Generation Subroutine `Neighbour`$(\cdot)$**.** A neighbourhood generation subroutine generates new assignments for the attack. Although a uniformly randomly picked assignment works all the time, it may not be the most efficient choice. In particular, for **DPAP-EMM** [14] and **DP-EMM** [45], we know that if an observed query response length is too far from the expectation, the assignment is very unlikely, and can be safely discarded. This means the neighbourhood generation subroutines for the attacks on these two schemes can make use of this, and output a new assignment only if it is sound. The pseudocode for these subroutines can be found in Algorithms 2 and 3 in Appendix C. We note that these neighbourhood generation subroutines may prevent some correct assignments in the output of the attack if their observed query response lengths are too far from the expected query response lengths. By relaxing the bounds, we can make the chance of that happening arbitrarily small. However, the algorithm would then be less efficient as more iterations are required for a convergence. Hence, we see our choice of bounds as a trade-off between query recovery rate and attack efficiency.

For **PRT-EMM** [33] and **FP-EMM** [45], we have to use uniformly randomly picked assignments, since the observed query response lengths are independent from the true query response lengths. The pseudocode of this neighbourhood generation subroutine can be found in Algorithm 4 in Appendix C.

4.4 From Query Reconstruction to Database Reconstruction

In this section, we discuss how some of our query reconstruction attacks can be extended to database reconstruction attacks. Here, by database reconstruction, we mean recovery of keywords in the encrypted documents.

DPAP-SE [14]. As the documents are transformed into document shards, it is impossible to reconstruct the keywords in the documents. However, as access pattern on the shards is leaked by the scheme, we can use our query reconstruction attack to guess the keywords in the shards.

Recall that the scheme uses parameters p and q to control the number of real keywords and fake keywords in each shard. In our attack, we used $p = 0.89$ and $q = 0.045$, meaning that on average, 89% of the real keywords of a document are kept by its shards and 4.5% of the remaining keywords are introduced as fake keywords. This makes our attack a terrible database reconstruction attack as there are over 30 thousand keywords in Enron email corpus, and we are expected to see about 1300 fake keywords per shard.

These choices of parameters, however, make the scheme very inefficient in terms of query response time as 4.5% of the database needs to be retrieved on average per query just as noises. Practitioners are likely going to use more optimistic parameters such as $p = 0.89999$

and $q = 6.997 \times 10^{-6}$ as suggested as alternative parameters in the paper. In that case, each shard is expected to receive less than one fake keyword, so there is an overwhelming probability that the keywords we recover from the shards are real keywords. In general, what this means is that for small q, our attack is able to recover p fraction of the real keywords for each shard in expectation.

PRT-EMM [33] AND **FP-EMM** [45]. Our query reconstruction attack cannot be extended to a database reconstruction attack on **PRT-EMM** as the signal to noise ratio is too low. For a similar reason, our attack on **FP-EMM** cannot be extended to a database reconstruction attack.

DP-EMM [45]. Our attack on **DP-EMM** on the other hand, implies a database reconstruction attack for practical security parameters. Just as before, as access pattern is leaked in actual document retrievals, our query reconstruction attack recovers the keywords in the retrieved documents.

Recall that for **DP-EMM**, the user picks a parameter ϵ, and the query response length is padded to

$$2n_{\mathsf{key}} + n^* + \mathbf{Lap}_{sk}(2/\epsilon),$$

where n^* is a fixed constant to offset the latter Laplace random variable. For a large ϵ, n^* is small. For example, with $\epsilon = 0.2$ as suggested in the original paper, $n^* = 567$. The most frequent keywords have frequencies one or two orders of magnitudes higher than n^*, which means the later two terms contributes very little as noises. The major source of noise comes from the multiplicative factor of 2, which means our query reconstruction attacks recovers as many keywords for each document as the query reconstruction rate specifies, except that about half of the keywords we recover are fake ones due to the padded query response lengths.

5 Experimental Evaluation

5.1 Overview

EXPERIMENTAL DATA AND AUXILIARY INFORMATION. We use the Enron email corpus [53] as the target dataset for all of our attacks. A description of the dataset and our pre-processing step can be found in Appendix D. A major challenge for inference-style leakage-abuse attacks is deciding an appropriate model for evaluating their effectiveness in practice. Such a model should take into account both the distribution of queries as well as the distribution of the auxiliary information available to the adversary. Unfortunately, there do not exist concrete guidelines in the literature for how to construct such models; given this lack of precedence, we make certain assumptions that we believe are reasonable in practice.

QUERY DISTRIBUTION. We use uniformly distributed keyword queries to evaluate our

attacks. This is exactly as in previous attacks [30, 9, 5]. We note here that our attacks do not explicitly depend on the distribution of queries; hence a uniform distribution appears to be a reasonable choice.

Auxiliary Data Distribution. For the IKK attack, Islam *et al.* [30] proposed a method to model auxiliary information in an inference-style attack setting; their suggestion was to use an auxiliary co-occurrence pattern leakage obtained by adding Gaussian noise to the original co-occurrence pattern. However, this implicitly assumes a homogeneous distribution of keywords amongst the documents, which may not always be the case in practice. Instead, we opt to split the overall dataset into two halves: out of the 480000 documents in the dataset, half of the documents are used as the attack target and (a subset of) the other half of the documents are used to generate auxiliary information about the dataset. In total, we generate 10 different splits of the documents. For each split, we run 10 independent attacks with freshly generated observed co-occurrence matrices. We measure the fraction of correctly guessed keywords/keys and report the average over the 100 runs as the query recovery rate.

Keyword Extraction and Stemming. We extract keywords using the Natural language toolkit [49] in Python. The experimental results we present in the main body of the paper uses the keywords as they are. Since previous attacks [30, 9, 5] used stemming, we run additional experiments with stemming and study its effects on query reconstruction rate in Section 5.5.

Keyword and Query Selection. We use the $1000, 2000, 3000$ and 4000 most frequent keywords to build auxiliary co-occurrence matrices, and sample uniformly randomly without replacement from these most frequent keywords subsets of $250, 500, 750$ keywords as queried keywords. These queried keywords are used to build observed co-occurrence matrices. These observed and auxiliary co-occurrence matrices are then used as the inputs to our attacks. Further discussion on the choice of keywords for our attacks is presented in Section 6.

Security Parameter Selection for the Target Constructions. We use the security parameters suggested in the original papers to run our attacks. We also investigate how changes in the security parameters affect query reconstruction rates.

DPAP-SE. For **DPAP-SE** [14], the authors suggested $m = 6$ (the number of shards per document), $k = 2$ (a parameter of the erasure code which does not affect our attack), $p = 0.88703$ (the probability for which a keyword is kept in a shard) and $q = 0.04416$ as the parameters for the Enron [53] dataset. We used similar parameters where $m = 6, k = 2, p = 0.89$ and $q = 0.045$ in our experiments. A smaller q or a bigger q significantly reduces the efficiency of the construction so we opt to not run additional experiments with those parameters. Instead, we investigate how a smaller q affects query reconstruction rate. We use $q = 0.0045, 0.00045$ and 0.000045 as additional choices of parameters in our experiments.

PRT-EMM. Recall that **PRT-EMM** from [33] allows the client to pick a public parameter λ which controls the padded query response lengths as:

$$n'_{\mathsf{key}} = \lambda + F_{sk}(\mathsf{key}||n_{\mathsf{key}}).$$

The authors suggested to set λ between 0 and $0.25n_{max}$. We used $\lambda = 0$ and $0.25n_{max}$ in our experiments. In addition, we used $\lambda = 0.5n_{max}$ to see the effect of additional padding on query reconstruction rate.

FP-EMM and DP-EMM. **FP-EMM** from [45] does not have a tunable parameter and we run our attacks on the **FP-EMM** as it is. **DP-EMM** from [45] uses parameter ϵ to set query response volumes to:

$$2n_{\mathsf{key}} + n^* + \mathbf{Lap}_{sk}(2/\epsilon).$$

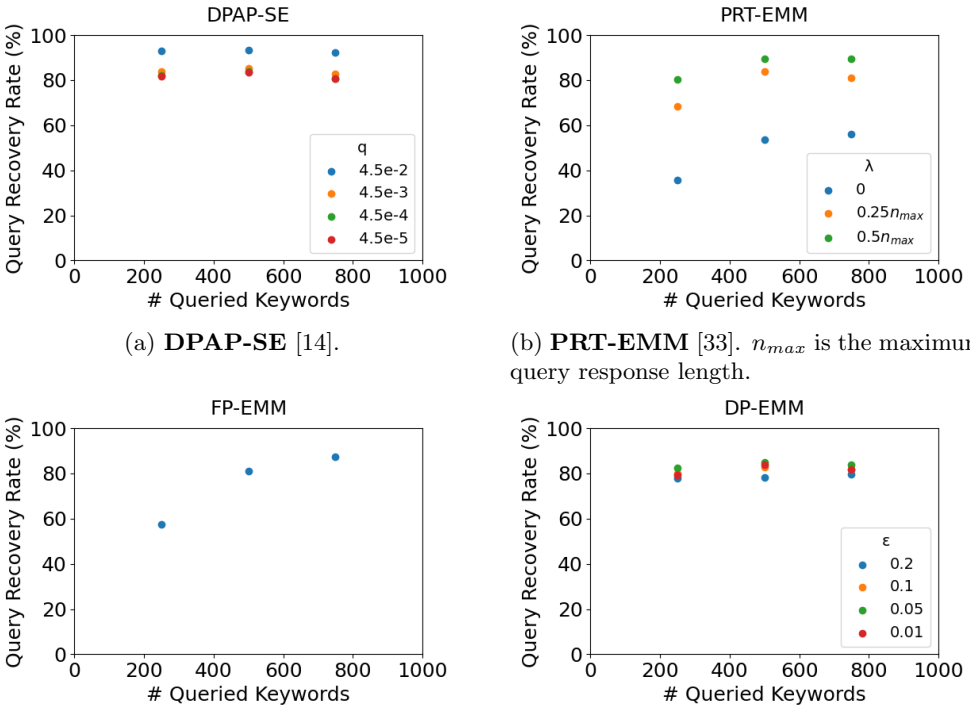
The authors suggested $\epsilon = 0.2$. In our attack, we use $\epsilon = 0.2$ just as suggested in the original paper. We also run experiments where ϵ is significantly smaller, ranging from 0.1 to 0.01.

Implementation. We implemented our attacks in C using GNU Scientific Library [22] for randomness generation and probability calculations. We used our custom code for simulated annealing for the best performance. We parallelized our implementation using OpenMP [41]. Our implementation is highly scalable. It takes less than one minute per run on the differentially-private schemes (**DPAP-SE** and **DP-EMM**) and no more than 6 minutes per run on the other schemes (**PRT-EMM** and **FP-EMM**) for all of our experimental settings, on a machine with an 8-core (16-thread) Sandy Bridge CPU clocked at 2.6 GHz.

Experiments. We present three sets of experiments on the target constructions in this section. In Section 5.2, we present the experimental results in basic settings, where the auxiliary co-occurrence matrix is built from all of the documents allocated for auxiliary information (50% of the total) using the 1000 most frequent keywords. We set the number of queried keywords to $250, 500$ or 750, and the security parameters are allowed to vary. In Section 5.3, we set the number of queried keywords to $250, 500$ or 750, and the security parameters to those suggested in the original papers, and vary the number of keywords used to build auxiliary information between 1000 and 4000. Just as before, all available documents are used in building auxiliary information. Finally, in Section 5.4, we use anywhere from 2.5% to 20% of the documents (5% to 40% of the 50% of documents allocated for auxiliary information) to build the auxiliary co-occurrence matrix, as a means to simulate auxiliary information with different levels of noise. The number of keywords used is set to 1000, and the number of queries is allowed to vary from 250 to 750. The security parameters are set to the ones recommended in the original papers.

5.2 Varying the Security Parameters of the Constructions

Differentially-private Access Pattern for Searchable Encryption. The results of our attacks are shown in Figure 1a. The attack is able recover over 80% of the queries



(a) **DPAP-SE** [14].

(b) **PRT-EMM** [33]. n_{max} is the maximum query response length.

(c) **FP-EMM** [45].

(d) **DP-EMM** [45].

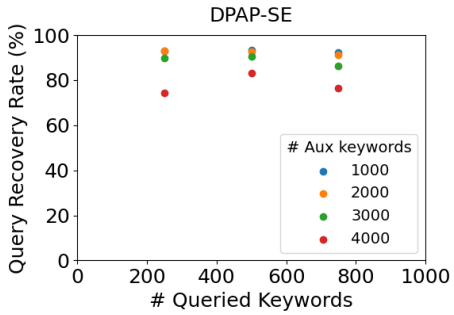
Figure 1: Experimental results with varying security parameters. The 1000 most frequent keywords are used in auxiliary information.

in all cases we have considered. The attack does not seem to perform worse with fewer observed queries. Interestingly, the attack performs better with a larger q. Intuitively, a larger q should generate more noise in the co-occurrence matrix, but it works in the favour of our attack. One possible explanation is that the auxiliary co-occurrence matrix is very different from the observed one, so our neighbourhood generation subroutine over-fits the assignments.

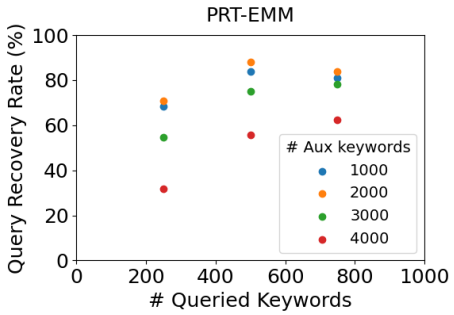
PRT-EMM. The experimental results on **PRT-EMM** are shown in Figure 1b. We observe an increasing query recovery rate with more queried keywords and larger λ. The attack performs significantly worse with $\lambda = 0$. This is likely due to removal of co-occurrence counts as the query response lengths are significantly shorter.

FP-EMM. The experimental results of our attack on **FP-EMM** [45] are shown in Figure 1c. As expected, the attack performs better with more queried keywords. The attack is able to recover over 80% of the queried keywords if over 500 keywords have been queried, suggesting that full padding is ineffective at adding noise to the co-occurrence leakage.

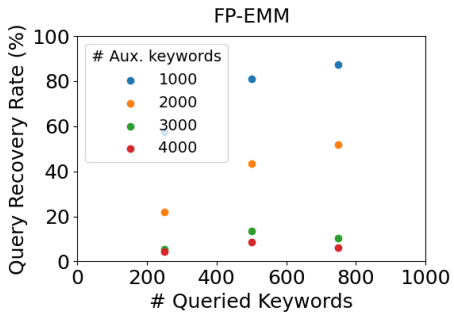
DP-EMM. The experimental results are shown in 1d. The attack does not seem to be affected by the number of observed queries and the choice of ϵ much.



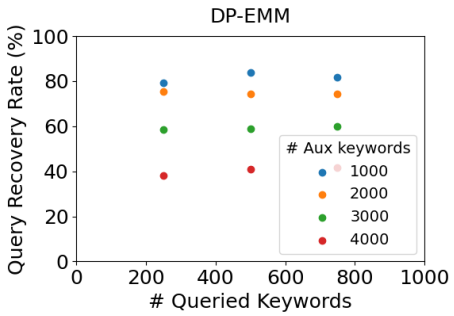
(a) **DPAP-SE** [14]. The security parameters used are $m = 6, k = 2, p = 0.89$ and $q = 0.045$.



(b) **PRT-EMM** [33]. λ is set to $0.25n_{max}$.



(c) **FP-EMM** [45].



(d) **DP-EMM** [45]. $\epsilon = 0.2$.

Figure 2: Experimental results with varying number of keywords in auxiliary information.

5.3 Varying the Number of Keywords in Auxiliary Information

Our experimental results on varying the number of keywords in the auxiliary information are shown in Figure 2. The security parameters we used can be found in the captions. Interestingly, the constructions behave very differently with respect to the number of keywords in the auxiliary information. In particular, our attack on **DPAP-SE** [14] is able to recover more than 70% of the keywords even with 4000 queried keywords. Our attack on **PRT-EMM** [33] works reasonably well with large numbers of keywords in the auxiliary information, managing over 50% query recovery rate except for the case with 250 queried keywords and 4000 keywords in the auxiliary information.

Our attacks are less successful on **FP-EMM** and **DP-EMM** [45]. For **FP-EMM**, the query recovery rate falls rapidly as soon as the number of keywords in the auxiliary information is greater than 2000, whereas our attack on **DP-EMM** has lower query reconstruction rates with more than 3000 keywords in the auxiliary information. This suggests that **FP-EMM** and **DP-EMM** (with our choice of parameters) introduces more uncertainty than the other schemes.

5.4 Varying the Level of Noise in Auxiliary Information

Given that there is no widely accepted way of modelling noise in auxiliary information, we opt to use different numbers of documents in auxiliary information as a way to simulate different levels of noise – fewer documents means more noise. We use absolute distance and modified probability score to measure the level of noise introduced in each set of experiments we run. We treat the co-occurrence matrix used as the target database as a sample from a distribution of co-occurrence matrices specified by the co-occurrence matrix used as auxiliary information. There is no known test statistics for this scenario so we propose our own measurements for the level of noise.

Absolute Distance. Inspired by the Kolmogorov–Smirnov test [51], we define absolute distance to be the maximum absolute difference between the target co-occurrence matrix and auxiliary co-occurrence matrix:

$$D = \max_{i,j} \left| \frac{\bar{M}_{P(i),P(j)}}{N} - M_{i,j} \right|,$$

where $\bar{M}$ is the co-occurrence matrix generated from the target database (without using any construction on top), M is the co-occurrence matrix generated for auxiliary information, P is the true keyword assignments between the queries and keywords, and N is the number of documents in the target database. Intuitively, more noisy auxiliary information means a larger absolute distance.

Modified Probability Score. The second measurement of the level of noise we propose is the probability score. As the name suggests, the measurement is simply:

$$\mathbf{Pr}\left[\bar{M} \mid M\right].$$

It is clear that less noisy auxiliary information produces a larger probability score.

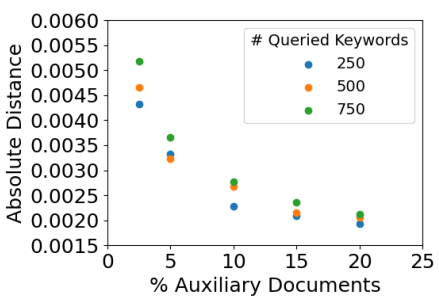
The probability score is very small for our datasets, so we use

$$D = \log(-\log(\mathbf{Pr}\left[\bar{M} \mid M\right]))$$

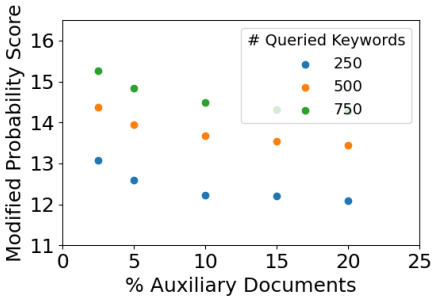
as a modified probability score instead. Less noisy auxiliary information produces a larger modified probability score just as before.

Measurements on the Level of Noise. The measurements on the level of noise for the auxiliary datasets used in our attacks can be found in Figure 3. It can be seen clearly that the absolute distance and modified probability score increase as less documents are used as auxiliary information.

Our experimental results on varying auxiliary information are shown in Figure 4. The security parameters we used can be found in the captions. We observed that the attacks do not perform well when only 2.5% of the documents are used to construct the auxiliary information. This is likely due to the fact that the keywords are not identically distributed within the documents, as indicated by Figure 3. On the other hand, our attacks have comparable query reconstruction rates with 10% and 50% of the documents in the auxiliary

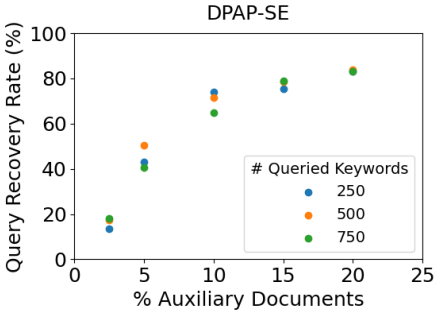


(a) Absolute distance.

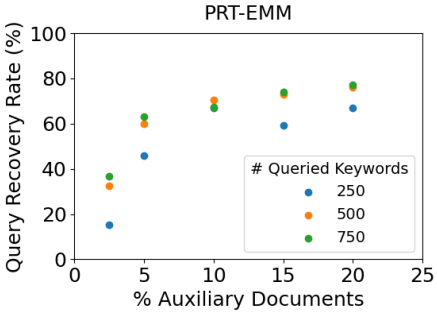


(b) Modified probability score.

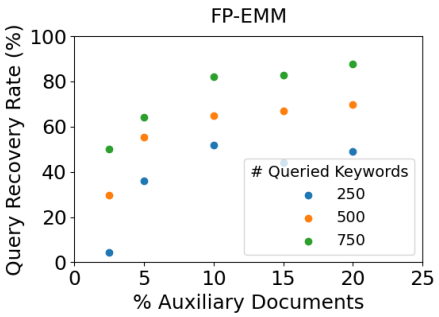
Figure 3: Measurements of the level of noise of the auxiliary data in our experiments.



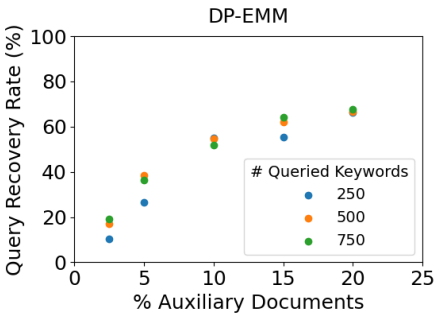
(a) **DPAP-SE** [14]. The security parameters used are $m = 6, k = 2, p = 0.89$ and $q = 0.045$.



(b) **PRT-EMM** [33]. λ is set to $0.25n_{max}$.



(c) **FP-EMM** [45].



(d) **DP-EMM** [45]. ϵ is set to 0.2.

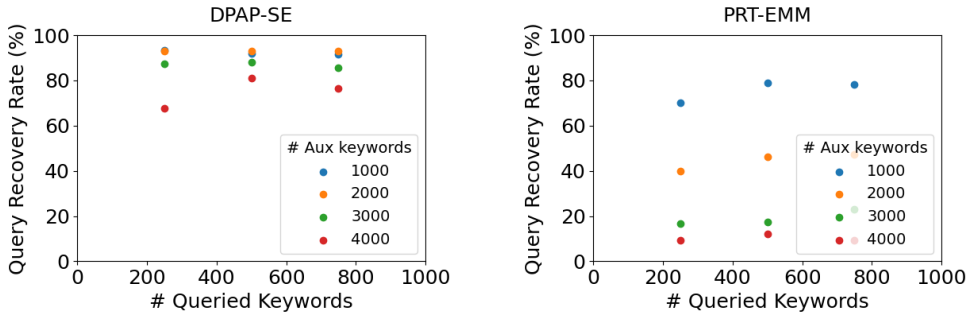
Figure 4: Experimental results with varying auxiliary information.

information, suggesting that 10% of the documents is sufficient as auxiliary information and that our attacks are robust in a noisy setting.

5.5 Experimental Results with Stemming

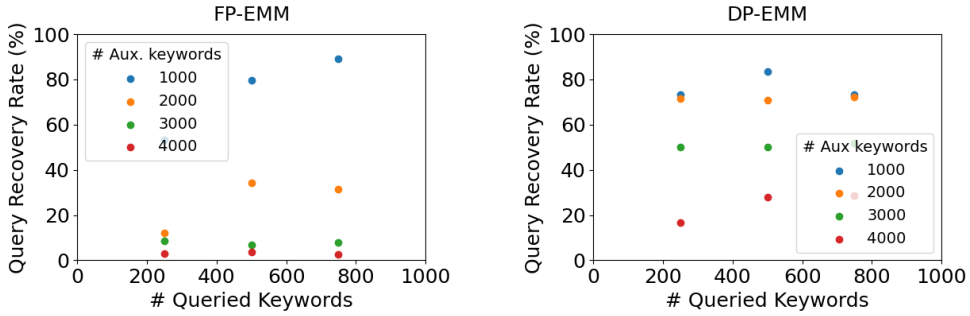
This section provides additional experimental results on the Enron dataset after stemming. We used the Porter Stemming Algorithm [47] implemented in the Natural language toolkit [49] as our stemming algorithm. The security parameters used for the constructions can be found in the captions.

Varying the Number of Keywords in Auxiliary Information. Our experimental results with varying number of keywords in auxiliary information is shown in Figure 5. The security parameters for the constructions used in our experiments can be found in the captions. The experimental results with stemming agree with those without stemming (Section 5.3), except for **PRT-EMM**, which performs worse with stemming as the number of keywords in auxiliary information increases. This is possibly because significantly more noise is introduced with stemming for **PRT-EMM**, as indicated by the frequency distribution plot in Figure 8 (Appendix D).



(a) **DPAP-SE** [14]. The security parameters used are $m = 6, k = 2, p = 0.89$ and $q = 0.045$.

(b) **PRT-EMM** [33]. λ is set to $0.25n_{max}$.



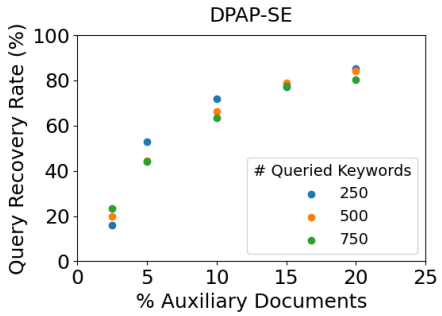
(c) **FP-EMM** [45].

(d) **DP-EMM** [45]. $\epsilon = 0.2$.

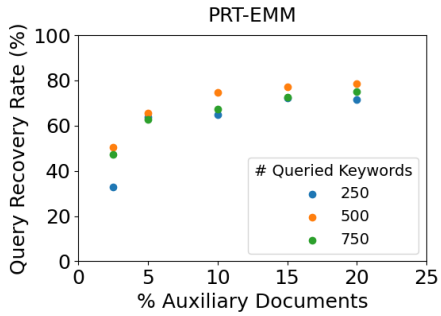
Figure 5: Experimental results with varying number of keywords in auxiliary information.

Varying the Level of Noise in Auxiliary Information. Our experimental results with varying level of noise in auxiliary information is shown in Figure 6. Just like before, the most frequent 1000 keywords are used to build auxiliary information. The security parameters for the constructions used in our experiments can be found in the captions.

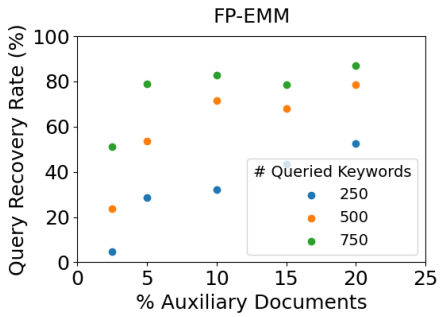
The performance of our attacks on auxiliary information with stemmed keywords is almost indistinguishable from those on auxiliary information with unmodified keywords.



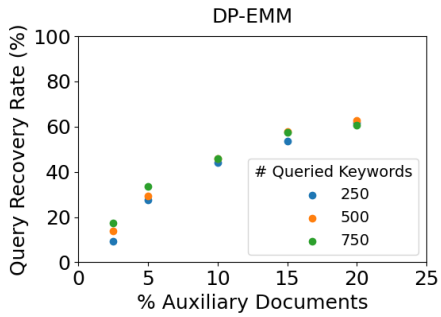
(a) **DPAP-SE** [14]. The security parameters used are $m = 6, k = 2, p = 0.89$ and $q = 0.045$.



(b) **PRT-EMM** [33]. λ is set to $0.25n_{max}$.



(c) **FP-EMM** [45].



(d) **DP-EMM** [45]. ϵ is set to 0.2.

Figure 6: Experimental results with varying auxiliary information.

6 Discussion

In this section, we discuss the implications of our attacks, the choices made in our experiments, and the practicality of our attacks.

On Differentially Private Access Patterns. Our attack on **DPAP-SE** [14] serves as a warning about the potential pitfalls of applying techniques from the differential privacy literature to STE without appropriately modeling and analyzing the resulting leakage. As pointed out by Chen *et al.* in [14], differentially private access patterns provide provable guarantees of the form: an adversary cannot distinguish between queries over keywords such that their access pattern leakage is within a *small* statistical distance of each other. As demonstrated by our experiments, the provable guarantees provided by differential privacy do not necessarily translate into security guarantees against leakage-abuse attacks *in general*.

We note here that the authors of [14] did establish the security of **DPAP-SE** (for certain

sets of parameters) against existing attacks [30, 9], which necessarily rely on *exact* co-occurrence leakage. However, it is perhaps unwise to assume that security against a small set of known attacks translates to security against all possible attacks. Rather, one should assume that attacks can always get stronger. This is precisely what we demonstrate by showcasing stronger leakage-abuse attacks that work even in the presence of "noisy" co-occurrence pattern leakage.

On System-Wide leakage. Our attacks on state-of-the-art STE schemes built from volume-hiding EMMs highlight the threats posed by system-wide leakage and highlight the need to revisit existing security definitions that ignore such leakage. In particular, suppose that we ask the following question:

How do we transition from volume-hiding EMMs to efficient leakage-hiding STE schemes for document collections?

Unfortunately, the authors of [33, 45] do not offer a concrete answer to this question. As demonstrated by our attacks, the naturally efficient (and widely prevalent) *structure-only* [13] method for achieving this transition (that of applying volume-hiding EMMs to only the encrypted index) inevitably incurs system-wide leakage and, in fact, result in insecure systems. The alternative approach (that of applying volume-hiding EMMs to the entire database) is secure but renders all state-of-the-art STE schemes impractical.

Note that many of the STE constructions built from EMMs were designed prior to the proposal of volume-hiding EMMs; in their original form, these constructions used EMMs that themselves leak the exact access-pattern. In this case, the leakage from the encrypted search index subsumes the leakage from encrypted document retrieval (see [15, 12] for relevant discussions). In that context, security definitions that focussed purely on the encrypted search index proved simple and useful in analyzing the leakage profiles for these constructions.

However, this analysis approach is no longer valid when such access pattern-revealing EMMs are replaced by volume-hiding EMMs. In this case, the additional system-wide leakage during encrypted document retrieval is no longer covered by security definitions that focus purely on the encrypted search index. At the same time, this leakage is observable by any adversary that takes a system-wide view of STE.

We believe that our attacks on system-wide leakage fundamentally motivate the need to revisit: (a) how the leakage of STE schemes is currently modeled in the literature, and (b) how countermeasures should be designed to minimize leakage.

On Keyword Frequencies for Our Experiments. As pointed out by Blackstone et al. [5], the choice of client queries is an important consideration to make when evaluating the performance of any leakage-abuse attack. In our experiments, we used keywords with *high-frequency* of occurrence in the document collection, for both the auxiliary data available to the adversary, as well as the queries that the adversary eventually observes during the attack phase. We note here that all of the existing leakage-abuse attacks on STE for document collections in [30, 9, 5] only work for high-frequency keywords. In fact, as reported by

Blackstone et al. [5], the Count attacks in [30] and [9] have zero query reconstruction rate on keywords in the Enron corpus with frequency less than 20 (even in the known-data setting where the entirety of the target database for evaluation is known to the adversary as auxiliary information). Blackstone *et al.* [5] also pointed out that, for their own attacks, query reconstruction rates drop significantly if keywords with low-frequency are used to build auxiliary datasets and to generate queries. Our attacks exhibit a similar trend.

On the Practicality of Our Attacks. As already mentioned, the leakage-abuse attacks proposed in this paper are inference attacks and work when the adversary has access to auxiliary data that is independent of but statistically "close" to the target database. We believe that this a weaker (and more realistic) assumption on adversarial capabilities compared to the assumptions made in previous attacks [30, 9, 5] which are all of the known-data variety. Our attacks also achieve high keyword recovery rates even when the target schemes use aggressive security parameters, or when the auxiliary data available to the adversary is relatively noisy (as arises when we sample the auxiliary data from a small portion of the database). These observations further reinforce the practicality of our proposed attacks. We could potentially extend/strengthen our attacks based on other related attacks on PPE/STE. For example, we could use Bayesian inference as in [4] if a good model for the prior distribution could be established.

A potential drawback of our attacks is that they assume auxiliary information involving high-frequency keywords. This is a relatively strong assumption in practice (although one made by all previous leakage-abuse attacks); in a real-world scenario, it is likely that the user queries keywords with a mixture of frequencies. One can of course filter out leakage from low frequency keywords based on response volume before running our attacks. It is also possible that our attacks could be improved to work for low-frequency keywords as well, though we leave this as an open problem.

In summary, our results demonstrate the security limitations of state-of-the-art STE schemes with perturbed leakage profiles, and serve as motivation for designing STE schemes with reduced leakage profiles that satisfy more comprehensive, system-wide security definitions.

References

[1] Hime Aguiar e Oliveira Junior, Lester Ingber, Antonio Petraglia, Mariane Rembold Petraglia, and Maria Augusta Soares Machado. *Adaptive Simulated Annealing*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[2] Ghous Amjad, Seny Kamara, and Tarik Moataz. Breach-resistant structured encryption. *Proc. Priv. Enhancing Technol.*, 2019(1):245–265, 2019.

[3] Mihir Bellare, Alexandra Boldyreva, and Adam O'Neill. Deterministic and efficiently searchable encryption. In Alfred Menezes, editor, *Advances in Cryptology – CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 535–552, Santa Barbara, CA, USA, August 19–23, 2007. Springer, Heidelberg, Germany.

[4] Vincent Bindschaedler, Paul Grubbs, David Cash, Thomas Ristenpart, and Vitaly Shmatikov. The tao of inference in privacy-protected databases. *Proc. VLDB Endow.*, 11(11):1715–1728, July 2018.

[5] Laura Blackstone, Seny Kamara, and Tarik Moataz. Revisiting leakage abuse attacks. In *ISOC Network and Distributed System Security Symposium – NDSS 2020*, San Diego, CA, USA, February 23-26, 2020. The Internet Society.

[6] Alexandra Boldyreva, Nathan Chenette, Younho Lee, and Adam O'Neill. Order-preserving symmetric encryption. In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 224–241, Cologne, Germany, April 26–30, 2009. Springer, Heidelberg, Germany.

[7] Alexandra Boldyreva, Nathan Chenette, and Adam O'Neill. Order-preserving encryption revisited: Improved security analysis and alternative solutions. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 578–595, Santa Barbara, CA, USA, August 14–18, 2011. Springer, Heidelberg, Germany.

[8] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 253–273, Providence, RI, USA, March 28–30, 2011. Springer, Heidelberg, Germany.

[9] David Cash, Paul Grubbs, Jason Perry, and Thomas Ristenpart. Leakage-abuse attacks against searchable encryption. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *ACM CCS 2015: 22nd Conference on Computer and Communications Security*, pages 668–679, Denver, CO, USA, October 12–16, 2015. ACM Press.

[10] David Cash, Paul Grubbs, Jason Perry, and Thomas Ristenpart. Leakage-abuse attacks against searchable encryption. Cryptology ePrint Archive, Report 2016/718, 2016. `http://eprint.iacr.org/2016/718`.

[11] David Cash, Joseph Jaeger, Stanislaw Jarecki, Charanjit S. Jutla, Hugo Krawczyk, Marcel-Catalin Rosu, and Michael Steiner. Dynamic searchable encryption in very-large databases: Data structures and implementation. In *ISOC Network and Distributed System Security Symposium – NDSS 2014*, San Diego, CA, USA, February 23–26, 2014. The Internet Society.

[12] David Cash, Stanislaw Jarecki, Charanjit S. Jutla, Hugo Krawczyk, Marcel-Catalin Rosu, and Michael Steiner. Highly-scalable searchable symmetric encryption with support for Boolean queries. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 353–373, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.

[13] Melissa Chase and Seny Kamara. Structured encryption and controlled disclosure. In Masayuki Abe, editor, *Advances in Cryptology – ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 577–594, Singapore, December 5–9, 2010. Springer, Heidelberg, Germany.

[14] G. Chen, T. Lai, M. K. Reiter, and Y. Zhang. Differentially private access patterns for searchable symmetric encryption. In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, pages 810–818, 2018.

[15] Reza Curtmola, Juan A. Garay, Seny Kamara, and Rafail Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006: 13th Conference on Computer and Communications Security*, pages 79–88, Alexandria, Virginia, USA, October 30 – November 3, 2006. ACM Press.

[16] Ioannis Demertzis, Dimitrios Papadopoulos, Charalampos Papamanthou, and Saurabh Shintre. SEAL: Attack mitigation for encrypted databases via adjustable leakage. In Srdjan Capkun and Franziska Roesner, editors, *USENIX Security 2020: 29th USENIX Security Symposium*, pages 2433–2450. USENIX Association, August 12–14, 2020.

[17] Alexandros G. Dimakis, Brighten Godfrey, Martin J. Wainwright, and Kannan Ramchandran. Network coding for distributed storage systems. *CoRR*, abs/cs/0702015, 2007.

[18] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, *TCC 2006: 3rd Theory of Cryptography Conference*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284, New York, NY, USA, March 4–7, 2006. Springer, Heidelberg, Germany.

[19] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st Annual ACM Symposium on Theory of Computing*, pages 169–178, Bethesda, MD, USA, May 31 – June 2, 2009. ACM Press.

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

[21] Oded Goldreich. Towards a theory of software protection and simulation by oblivious RAMs. In Alfred Aho, editor, *19th Annual ACM Symposium on Theory of Computing*, pages 182–194, New York City, NY, USA, May 25–27, 1987. ACM Press.

[22] Brian Gough. *GNU Scientific Library Reference Manual - Third Edition*. Network Theory Ltd., 3rd edition, 2009.

[23] Paul Grubbs, Marie-Sarah Lacharité, Brice Minaud, and Kenneth G. Paterson. Pump up the volume: Practical database reconstruction from volume leakage on range queries. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 315–331, Toronto, ON, Canada, October 15–19, 2018. ACM Press.

[24] Paul Grubbs, Marie-Sarah Lacharité, Brice Minaud, and Kenneth G. Paterson. Learning to reconstruct: Statistical learning theory and encrypted database attacks. In *2019 IEEE Symposium on Security and Privacy*, pages 1067–1083, San Francisco, CA, USA, May 19–23, 2019. IEEE Computer Society Press.

[25] Paul Grubbs, Thomas Ristenpart, and Vitaly Shmatikov. Why your encrypted database is not secure. In Alexandra Fedorova, Andrew Warfield, Ivan Beschastnikh, and Rachit

Agarwal, editors, *Proceedings of the 16th Workshop on Hot Topics in Operating Systems, HotOS 2017, Whistler, BC, Canada, May 8-10, 2017*, pages 162–168. ACM, 2017.

[26] Paul Grubbs, Kevin Sekniqi, Vincent Bindschaedler, Muhammad Naveed, and Thomas Ristenpart. Leakage-abuse attacks against order-revealing encryption. In *2017 IEEE Symposium on Security and Privacy*, pages 655–672, San Jose, CA, USA, May 22–26, 2017. IEEE Computer Society Press.

[27] Zichen Gui, Oliver Johnson, and Bogdan Warinschi. Encrypted databases: New volume attacks against range queries. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019: 26th Conference on Computer and Communications Security*, pages 361–378. ACM Press, November 11–15, 2019.

[28] Zichen Gui, Kenneth G. Paterson, Sikhar Patranabis, and Bogdan Warinschi. Swissse: System-wide security for searchable symmetric encryption. *IACR Cryptol. ePrint Arch.*, 2020:1328, 2020.

[29] Warren He, Devdatta Akhawe, Sumeet Jain, Elaine Shi, and Dawn Xiaodong Song. ShadowCrypt: Encrypted web applications for everyone. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *ACM CCS 2014: 21st Conference on Computer and Communications Security*, pages 1028–1039, Scottsdale, AZ, USA, November 3–7, 2014. ACM Press.

[30] Mohammad Saiful Islam, Mehmet Kuzu, and Murat Kantarcioglu. Access pattern disclosure on searchable encryption: Ramification, attack and mitigation. In *ISOC Network and Distributed System Security Symposium – NDSS 2012*, San Diego, CA, USA, February 5–8, 2012. The Internet Society.

[31] Seny Kamara and Tarik Moataz. Boolean searchable symmetric encryption with worst-case sub-linear complexity. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part III*, volume 10212 of *Lecture Notes in Computer Science*, pages 94–124, Paris, France, April 30 – May 4, 2017. Springer, Heidelberg, Germany.

[32] Seny Kamara and Tarik Moataz. SQL on structurally-encrypted databases. In Thomas Peyrin and Steven Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018, Part I*, volume 11272 of *Lecture Notes in Computer Science*, pages 149–180, Brisbane, Queensland, Australia, December 2–6, 2018. Springer, Heidelberg, Germany.

[33] Seny Kamara and Tarik Moataz. Computationally volume-hiding structured encryption. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019, Part II*, volume 11477 of *Lecture Notes in Computer Science*, pages 183–213, Darmstadt, Germany, May 19–23, 2019. Springer, Heidelberg, Germany.

[34] Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. Generic attacks on secure outsourced databases. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016: 23rd Conference on Computer and Communications Security*, pages 1329–1340, Vienna, Austria, October 24–28, 2016. ACM Press.

[35] Marie-Sarah Lacharité, Brice Minaud, and Kenneth G. Paterson. Improved reconstruction attacks on encrypted data using range query leakage. In *2018 IEEE Symposium on Security and Privacy*, pages 297–314, San Francisco, CA, USA, May 21–23, 2018. IEEE Computer Society Press.

[36] Marie-Sarah Lacharité and Kenneth G. Paterson. Frequency-smoothing encryption: preventing snapshot attacks on deterministically encrypted data. *IACR Transactions on Symmetric Cryptology*, 2018(1):277–313, 2018.

[37] Shangqi Lai, Sikhar Patranabis, Amin Sakzad, Joseph K. Liu, Debdeep Mukhopadhyay, Ron Steinfeld, Shifeng Sun, Dongxi Liu, and Cong Zuo. Result pattern hiding searchable encryption for conjunctive queries. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 745–762, Toronto, ON, Canada, October 15–19, 2018. ACM Press.

[38] Billy Lau, Simon P. Chung, Chengyu Song, Yeongjin Jang, Wenke Lee, and Alexandra Boldyreva. Mimesis aegis: A mimicry privacy shield-A system's approach to data privacy on public cloud. In Kevin Fu and Jaeyeon Jung, editors, *USENIX Security 2014: 23rd USENIX Security Symposium*, pages 33–48, San Diego, CA, USA, August 20–22, 2014. USENIX Association.

[39] Sahar Mazloom and S. Dov Gordon. Secure computation with differentially private access patterns. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 490–507, Toronto, ON, Canada, October 15–19, 2018. ACM Press.

[40] Muhammad Naveed, Seny Kamara, and Charles V. Wright. Inference attacks on property-preserving encrypted databases. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *ACM CCS 2015: 22nd Conference on Computer and Communications Security*, pages 644–655, Denver, CO, USA, October 12–16, 2015. ACM Press.

[41] OpenMP Architecture Review Board. OpenMP application program interface version 5.0, 2018.

[42] Simon Oya and Florian Kerschbaum. Hiding the access pattern is not enough: Exploiting search pattern leakage in searchable encryption, 2021.

[43] Rasmus Pagh and Flemming Friche Rodler. Cuckoo hashing. In Friedhelm Meyer auf der Heide, editor, *Algorithms — ESA 2001*, pages 121–133, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[44] Omkant Pandey and Yannis Rouselakis. Property preserving symmetric encryption. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 375–391, Cambridge, UK, April 15–19, 2012. Springer, Heidelberg, Germany.

[45] Sarvar Patel, Giuseppe Persiano, Kevin Yeo, and Moti Yung. Mitigating leakage in secure cloud-hosted data structures: Volume-hiding for multi-maps via hashing. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019: 26th Conference on Computer and Communications Security*, pages 79–93. ACM Press, November 11–15, 2019.

[46] Raluca A. Popa, Catherine M. S. Redfield, Nickolai Zeldovich, and Hari Balakrishnan. Cryptdb: protecting confidentiality with encrypted query processing. In *ACM SOSP 2011*, pages 85–100, 2011.

[47] M.f. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[48] David Pouliot and Charles V. Wright. The shadow nemesis: Inference attacks on efficiently deployable, efficiently searchable encryption. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016: 23rd Conference on Computer and Communications Security*, pages 1341–1352, Vienna, Austria, October 24–28, 2016. ACM Press.

[49] NLTK Project. *Natural Language Toolkit*. `https://www.nltk.org/`.

[50] Zhiwei Shang, Simon Oya, Andreas Peter, and Florian Kerschbaum. Obfuscated access and search patterns in searchable encryption, 2021.

[51] N. Smirnov. Table for Estimating the Goodness of Fit of Empirical Distributions. *The Annals of Mathematical Statistics*, 19(2):279 – 281, 1948.

[52] Dawn Xiaodong Song, David Wagner, and Adrian Perrig. Practical techniques for searches on encrypted data. In *2000 IEEE Symposium on Security and Privacy*, pages 44–55, Oakland, CA, USA, May 2000. IEEE Computer Society Press.

[53] CMU William W. Cohen, MLD. Enron email dataset.

[54] Yupeng Zhang, Jonathan Katz, and Charalampos Papamanthou. All your queries are belong to us: The power of file-injection attacks on searchable encryption. In Thorsten Holz and Stefan Savage, editors, *USENIX Security 2016: 25th USENIX Security Symposium*, pages 707–720, Austin, TX, USA, August 10–12, 2016. USENIX Association.

A Derivations of the Co-occurrence Matrices

Derivation for **DPAP-SE** [14]. Let **DB** be a database and $\mathsf{q}_1, \ldots, \mathsf{q}_l$ be non-repeating search queries with associated keywords $\mathrm{kw}_1, \ldots, \mathrm{kw}_l$ on **DB** encrypted with the searchable encryption scheme above [14]. The diagonal entries of the co-occurrence matrix $\bar{M}(\mathsf{q}_1, \ldots, \mathsf{q}_l; \mathbf{DB})$, i.e. the query response volumes, represent the numbers of shards retrieved by the client. For a particular query q_i, the number of shards retrieved is determined by:

- The number of shards which contain keyword kw_i before the pre-processing step, and the keyword is not removed from them.
- The number of shards which do not contain keyword kw_i before the pre-processing step, but the keyword is added to them.

Formally, the diagonal entries of the co-occurrence matrix can be expressed in terms of the true query response lengths as:

$$\begin{aligned}\bar{M}(\mathsf{q}_1, \ldots, \mathsf{q}_l; \mathbf{DB})_{i,i} \sim\ & \mathbf{Bin}(m \cdot |\mathbf{DB}(\mathrm{kw}_i)|, p) \\ & + \mathbf{Bin}(m \cdot |\mathbf{DB}| - m \cdot |\mathbf{DB}(\mathrm{kw}_i)|, q),\end{aligned}$$

where m comes from splitting the documents into shards, $\mathbf{DB}(\mathrm{kw}_i)$ denotes the set of documents containing keyword kw_i associated to query q_i, $|\mathbf{DB}|$ denotes the number of documents in database $\mathbf{DB}$, and $\mathbf{Bin}(\cdot)$ denotes a binomial distribution.

For the off-diagonal entries of the co-occurrence matrix, assume without loss of generality that the keywords in concern are kw_i and kw_j. The co-occurrence count for keywords kw_i and kw_j can increase if:

- A shard contains one of the keywords, say kw_i, and the keyword is not removed by the scheme. At the same time, the other keyword, kw_j in this case, is added to the shard.
- A shard contains none of the keywords, and both of the keywords are added to the shard.

On the other hand, the co-occurrence count for keywords kw_i and kw_j can decrease if a shard contains both of the keywords and at least one of the keywords is removed.

The actual distribution of the off-diagonal entries of the co-occurrence matrix is complicated due to dependencies. However, if we ignoring the fact that we already know the query response lengths for keywords kw_i and kw_j, the off-diagonal entries of the co-occurrence matrix can be approximated as:

$$
\begin{aligned}
&\bar{M}(\mathsf{q}_1, \ldots, \mathsf{q}_l; \mathbf{DB})_{i,j} \\
\sim&\mathbf{Bin}(m \cdot |\mathbf{DB}(\mathrm{kw}_i, \mathrm{kw}_j)|, p^2) \\
+&\mathbf{Bin}\left(m \cdot (|\mathbf{DB}| - |\mathbf{DB}(\mathrm{kw}_i)| - |\mathbf{DB}(\mathrm{kw}_j)| + |\mathbf{DB}(\mathrm{kw}_i, \mathrm{kw}_j)|), q^2\right) \\
+&\mathbf{Bin}(m \cdot |\mathbf{DB}(\mathrm{kw}_i)| - |\mathbf{DB}\mathrm{kw}_i, \mathrm{kw}_j|, pq) \\
+&\mathbf{Bin}(m \cdot |\mathbf{DB}(\mathrm{kw}_j)| - |\mathbf{DB}\mathrm{kw}_i, \mathrm{kw}_j|, pq).
\end{aligned}
$$

Derivation for **PRT-EMM** [33]. Recall that in **PRT-EMM**, the query response lengths are padded or truncated according to:

$$n'_{\mathsf{key}} = \lambda + F_{sk}(\mathsf{key}||n_{\mathsf{key}}).$$

Let $\mathbf{DB}$ be a multi-map and $\mathsf{q}_1, \ldots, \mathsf{q}_l$ be non-repeating search queries with associated keys $\mathsf{key}_1, \ldots, \mathsf{key}_l$ on $\mathbf{DB}$ encrypted with **PRT-EMM**. We abuse the notation $\mathsf{key}(\mathsf{q}_i)$ to mean the key associated to q_i. By denoting the maximum value of the PRF F as $|F|$, the diagonal entries of the co-occurrence matrix can be expressed as:

$$\bar{M}(\mathsf{q}_1, \ldots, \mathsf{q}_l; \mathbf{DB})_{i,i} \sim \lambda + \mathbf{Uniform}(0, |F|),$$

where $\mathbf{Uniform}(\cdot)$ is a uniform distribution.

There are three cases to be considered for the off-diagonal entries of the co-occurrence matrix. Without loss of generality, let the keys in concern be keys key_i and key_j. In the first case, both of the query response lengths associated to the keys are larger than the true query response lengths. This corresponds to $n'_{\mathsf{key}_i} - |\mathbf{DB}(\mathsf{key}_i)|$ random document retrievals

for queries on key key_i and $n'_{\mathsf{key}_j} - \left|\mathbf{DB}(\mathsf{key}_j)\right|$ random document retrievals for queries on key key_j. These random document retrievals can create additional co-occurrence counts among themselves or with the real document retrievals. The co-occurrence counts in this case can be approximated by:

$$\begin{aligned}
&\bar{M}(\mathsf{q}_1, \ldots, \mathsf{q}_l; \mathbf{DB})_{i,j} \\
\sim &\left|\mathbf{DB}(\mathsf{key}_i, \mathsf{key}_j)\right| \\
+&\mathbf{Hypergeometric}\left(n'_{\mathsf{key}_i} - \left|\mathbf{DB}(\mathsf{key}_i)\right|, \left|\mathbf{DB}\right|, n'_{\mathsf{key}_j}\right) \\
+&\mathbf{Hypergeometric}\left(n'_{\mathsf{key}_j} - \left|\mathbf{DB}(\mathsf{key}_j)\right|, \left|\mathbf{DB}\right|, n'_{\mathsf{key}_i}\right),
\end{aligned}$$

where $\mathbf{Hypergeometric}(n, N, K)$ denotes a hypergeometric distribution which makes n draws without replacement, from a population of size N that contains exactly K objects with the desired feature.

In the second case, one of the query response lengths is truncated and the other one is padded. Without loss of generality, let key key_i be the truncated key and key key_j be the padded key. Then, the co-occurrence count associated to keys key_i and key_j can be modelled as a process where the co-occurrence count is first reduced by the truncation and then increased by the padding. Its distribution is given below:

$$x \sim \mathbf{Hypergeometric}\Big(n'_{\mathsf{key}_i}, \left|\mathbf{DB}(\mathsf{key}_i)\right|, \left|\mathbf{DB}(\mathsf{key}_i, \mathsf{key}_j)\right|\Big),$$

$$\begin{aligned}
\bar{M}(\mathsf{q}_1, \ldots, \mathsf{q}_l; \mathbf{DB})_{i,j} \sim &x + \mathbf{Hypergeometric}\Big(n'_{\mathsf{key}_j} - \left|\mathbf{DB}(\mathsf{key}_j)\right|, \\
&\left|\mathbf{DB}\right|, n'_{\mathsf{key}_i} - x\Big).
\end{aligned}$$

Finally, in the last case, both of the query response lengths are truncated. Similar to above, the distribution of the co-occurrence count associated to keys key_i and key_j can be expressed as:

$$x \sim \mathbf{Hypergeometric}\Big(n'_{\mathsf{key}_i}, \left|\mathbf{DB}(\mathsf{key}_i)\right|, \left|\mathbf{DB}(\mathsf{key}_i, \mathsf{key}_j)\right|\Big),$$

$$\begin{aligned}
\bar{M}(\mathsf{q}_1, \ldots, \mathsf{q}_l; \mathbf{DB})_{i,j} \sim &\mathbf{Hypergeometric}\Big(n'_{\mathsf{key}_j}, \\
&\left|\mathbf{DB}(\mathsf{key}_j)\right|, x\Big).
\end{aligned}$$

Derivation for new volume-hiding multi-maps in [45]. The volume-hiding multi-maps in [45] are special cases of **PRT-EMM** [33], where the query response lengths are either padded to the maximum query response length or ones that are larger than the true query response lengths. Specifically, for the full padding version (**PRT-EMM**),

$$\bar{M}(\mathsf{q}_1, \ldots, \mathsf{q}_l; \mathbf{DB})_{i,i} \sim 2 \max_{\mathsf{key}} \left|\mathbf{DB}(\mathsf{key})\right|.$$

And for the differentially-private version (**DP-EMM**),

$$\bar{M}(\mathsf{q}_1, \ldots, \mathsf{q}_l; \mathbf{DB})_{i,i} \sim 2\left|\mathbf{DB}(\mathsf{key})\right| + n^* + \mathbf{Lap}(2/\epsilon),$$

where n^* is a fixed constant to offset the query response length in case the latter random variable is negative.

For the co-occurrence counts, we get:

$$\begin{aligned}
&\bar{M}(\mathsf{q}_1, \ldots, \mathsf{q}_l; \mathbf{DB})_{i,j} \\
\sim &\left|\mathbf{DB}(\mathsf{key}_i, \mathsf{key}_j)\right| \\
+&\mathbf{Hypergeometric}\left(n'_{\mathsf{key}_i} - \left|\mathbf{DB}(\mathsf{key}_i)\right|, \left|\mathbf{DB}\right|, n'_{\mathsf{key}_j}\right) \\
+&\mathbf{Hypergeometric}\left(n'_{\mathsf{key}_j} - \left|\mathbf{DB}(\mathsf{key}_j)\right|, \left|\mathbf{DB}\right|, n'_{\mathsf{key}_i}\right),
\end{aligned}$$

where n'_{key_i} and n'_{key_j} are the padded query response lengths for keyword kw_i and kw_j respectively.

B Derivations of the Likelihood Functions

Likelihood Function and its Decomposition. The likelihood function $\mathbf{L}\left[P \mid \bar{M}, M\right]$ can be written as follows:

$$\begin{aligned}
&\mathbf{L}\left[P \mid \bar{M}, M\right] \\
=&\mathbf{Pr}\left[\bar{M}, M \mid P\right] \\
=&\sum_{M' \in \mathcal{N}^{N \times N}} \mathbf{Pr}\left[\bar{M}, M, M' \mid P\right] \\
=&\sum_{M' \in \mathcal{N}^{N \times N}} \mathbf{Pr}\left[\bar{M} \mid M, M', P\right] \mathbf{Pr}\left[M' \mid M, P\right] \\
=&\sum_{M' \in \mathcal{N}^{N \times N}} \mathbf{Pr}\left[\bar{M} \mid M', P\right] \mathbf{Pr}\left[M' \mid M\right],
\end{aligned}$$

where N is the number of documents and $\mathcal{N}^{N \times N}$ is all N by N natural number valued matrices. In the third line of the equation, we used the law of total probability to turn the likelihood into a summation over all possible real co-occurrence matrices. The lines after break the probability into a sum of products of two probabilities. The first probability $\mathbf{Pr}\left[\bar{M}, M' \mid P\right]$ is the probability that $\bar{M}$ is the observed co-occurrence matrix and M' is the real co-occurrence matrix given P is the permutation. The second probability is the probability of getting M' as the real observed co-occurrence matrix knowing that M is the auxiliary co-occurrence matrix.

We assume the same structure of the auxiliary co-occurrence matrix M for all of our leakage functions so its derivation is shared by all three leakage functions. We note that only some of the real co-occurrence matrices generate a non-zero likelihood, as the sum of off-diagonal

entries of a row must be less or equal to the diagonal entry for correctness. By writing a row of a matrix M without the i-th entry as $M_{i,\cdot}$, for those real co-occurrence matrices, we can derive the probability as:

$$\begin{aligned}&\mathbf{Pr}\left[M' \mid M\right]\\=&\sum_i \mathbf{Pr}\left[M'_{i,i} \mid M_{i,i}\right]\mathbf{Pr}\left[M'_{i,\cdot} \mid M'_{i,i}, M_{i,\cdot}\right].\end{aligned}$$

In the second line, the first term is the probability of getting $M'_{i,i}$ documents containing keyword kw_i, and the second term is the probability of observing the off-diagonal co-occurrence counts.

Derivation for **DPAP-SE**. Recall that in **DPAP-SE** [14], the documents are split into shards and the keywords for the shards are randomized. This means that each diagonal entry of the observed co-occurrence matrix contain the counts from the real shards which have kept the keyword, and the counts from the other shards which have gained the keyword from the randomization process. Similarly, each off-diagonal entry of the observed co-occurrence matrix contain the counts from the real shards which have kept both of the keywords, and the other counts from the other shards which have gained one of the keywords or both of them from the randomization process. Let p be the probability that a shard keeps its keywords, q be the probability that a fake keyword is introduced to a shard, and m to be the number of shards, we can express the first term in the likelihood decomposition as:

$$\begin{aligned}&\mathbf{Pr}\left[\bar{M}, M' \mid P\right]\\=&\prod_{i=j}\mathbf{Pr}\left[\bar{M}, M' \mid P\right]\times\prod_{i<j}\mathbf{Pr}\left[\bar{M}, M' \mid P\right]\\=&\prod_{i}\mathbf{Pr}\left[\bar{M}_{i,i}, M' \mid P\right]\times\prod_{i<j}\mathbf{Pr}\left[\bar{M}_{i,j}, M'_{P(i),P(j)} \mid P\right]\\=&\prod_{i}\mathbf{Pr}\left[\mathbf{Bin}\left(mM'_{P(i),P(i)}, p\right)+\mathbf{Bin}\left(mM'_{P(j),P(j)}, q\right)=\bar{M}_{i,i}\right]\\&\times\prod_{i<j}\mathbf{Pr}\Bigg[\mathbf{Bin}\left(mM'_{P(i),P(j)}, p^2\right)\\&+\mathbf{Bin}\left(m\left(M'_{P(i),P(i)}-\sum_{k>1}M'_{P(i),P(k)}\right), pq\right)\\&+\mathbf{Bin}\left(m\left(M'_{P(j),P(j)}-\sum_{k>1}M'_{P(j),P(k)}\right), pq\right)\\&+\mathbf{Bin}\left(m\left(N-M'_{P(i),P(i)}-M'_{P(j),P(j)}+M'_{P(i),P(j)}\right), q^2\right)=\bar{M}_{i,j}\Bigg].\end{aligned}$$

Derivation for **PRT-EMMs**. For **PRT-EMMs** [33], query response lengths may be truncated by a random amount. This means that based on the query response length in the auxiliary co-occurrence matrix M' and that in the observed co-occurrence matrix, an attacker can estimate how many documents in the off-diagonal entries are expected to be

removed. For observed co-occurrence count between keywords kw_j and kw_j where $i \neq j$, the real process can be modelled as a sequential application of two hypergeometric distributions on the real co-occurrence count.

$$\begin{aligned}
&\mathbf{Pr}\left[\bar{M}, M' \mid P\right] \\
=&\prod_{i<j} \mathbf{Pr}\left[\bar{M}_{i,j}, M' \mid P\right] \\
=&\prod_{i<j} \sum_{k} \mathbf{Pr}\left[\textbf{Hypergeometric}\left(M'_{P(i),P(i)}, M'_{P(i),P(j)}, \bar{M}_{i,i}\right) = k\right] \\
&\mathbf{Pr}\left[\textbf{Hypergeometric}\left(M'_{P(j),P(j)}, k, \bar{M}_{j,j}\right) = \bar{M}_{i,j}\right].
\end{aligned}$$

Derivation for **FP-EMMs** [45]. To simplify the first term of the likelihood decomposition, we assume independence of the entries in the observed co-occurrence matrix. Without loss of generality, we assume that all query response lengths are padded to m. This means we can express the probability as:

$$\begin{aligned}
&\mathbf{Pr}\left[\bar{M}, M' \mid P\right] \\
=&\prod_{i<j} \mathbf{Pr}\left[\bar{M}_{i,j}, M'_{P(i),P(j)} \mid P\right] \\
=&\prod_{i<j} \mathbf{Pr}\Big[\textbf{Hypergeometric}\big(2N, 2m - 2M'_{P(i),P(i)}, \\
&2m - 2M'_{P(j),P(j)}\big) = \bar{M}_{i,j} - M'_{P(i),P(j)}\Big].
\end{aligned}$$

Derivation for **DP-EMMs** [45]. The first term of the likelihood decomposition for differentially private volume-hiding EMMs [45] is similar to that of the full padding version, except that the query response lengths are padded according to a Laplacian distribution as opposed to padding to the maximum query response length. Let n^* be the constant to offset the Laplacian random variable $\mathbf{Lap}(2/\epsilon)$, the first term of the likelihood decomposition can be expressed as:

$$\begin{aligned}
&\mathbf{Pr}\left[\bar{M}, M' \mid P\right] \\
=&\sum_{i=j} \mathbf{Pr}\left[\bar{M}, M'\right] + \sum_{i<j} \mathbf{Pr}\left[P \mid \bar{M}, M' \mid P\right] \\
=&\sum_{i} \mathbf{Pr}\left[\bar{M}_{i,i}, M'_{P(i),P(i)} \mid P\right] + \sum_{i<j} \mathbf{Pr}\left[\bar{M}_{i,j}, M'_{P(i),P(j)} \mid P\right] \\
=&\sum_{i} \mathbf{Pr}\left[2M'_{P(i),P(i)} + n^* + \mathbf{Lap}(2/\epsilon) = \bar{M}_{i,i}\right] \\
&+ \sum_{i<j} \mathbf{Pr}\Big[\textbf{Hypergeometric}(2N, 2\bar{M}_{i,i} - 2M'_{P(i),P(i)}, \\
&2\bar{M}_{j,j} - 2M'_{P(j),P(j)}) = \bar{M}_{i,j} - 2M'_{P(i),P(j)}\Big].
\end{aligned}$$

Approximation Techniques. As it can be seen, it is computationally infeasible to sum over all possible real co-occurrence matrices. We propose to sum over all possible real co-occurrence matrices such that $\mathbf{Pr}\left[M' \mid \bar{M}\right]$ is significant. In our experiment, we used symmetric endpoints on every entry of M' such that the resultant interval covers at least 95% of the probability density function. We use Normal approximation in the first term of the likelihood decomposition for **PRT-EMM** [33] to remove the need of a convolution. To further improve the computational efficiency, we used simple rectangle rule to approximate large summations, such as the convolutions in the first term of the likelihood decomposition for **DPAP-SE** [14].

Speeding up the `Score` function. The `Score` functions are by far the most computationally demanding functions of our attacks. If we just implement them naïvely, the amount of computation required in an iteration is proportional to l^2, where l is the number of non-repeating queries observed (it is also the dimension of the observed co-occurrence matrix $\bar{M}$). However, we note that the score functions in our attacks are essentially likelihood functions of the shape

$$\prod_{i \leq j} \mathbf{Pr}\left[\bar{M}_{P(i),P(j)}, M\right],$$

and the neighbourhood function `Neighbour` only changes the assignment P for one or two values. Without loss of generality, let $P(a)$ be the changed assignment. It means only the probabilities with $P(a)$ involved are changed, that is, the new likelihood function can be written as

$$\prod_{\substack{i \leq j \\ i,j \neq a}} \mathbf{Pr}\left[\bar{M}_{P(i),P(j)}, M\right] \times \prod_{i \leq a} \mathbf{Pr}\left[\bar{M}_{P(i),P(a)}, M\right] \times \prod_{a < j} \mathbf{Pr}\left[\bar{M}_{P(a),P(j)}, M\right].$$

The terms in the first product were already computed in the previous iteration so they can be used directly. The only terms that need re-computation are in the second and third products. This reduces the amount of computation required for the `Score` function (from the second iteration onwards) to something that is proportional to l.

In our implementation, we maintain an l-by-l matrix where the i, j-th entry of the matrix records $\mathbf{Pr}\left[\bar{M}_{P(i),P(j)}, M\right]$. Only l (or $2l$ if the assignment is changed on two queries) of these entries are updated according to the likelihood function, and the score function simply outputs the sum of the entries of this matrix.

C Detailed Pseudocodes

Algorithm 2 Neighbourhood Generation Algorithm for **DPAP-SE** [14]

procedure NEIGHBOUR($P, \bar{M}, M$)
  $i \xleftarrow{\$} \{1, \ldots, |\mathrm{kw}(\mathbf{DB})|\}$
  $j \xleftarrow{\$} \{1, \ldots, |\mathrm{kw}(\mathbf{DB})|\}$
  $b_0 \leftarrow kpNM_{j,j} + kqN(1 - M_{j,j}) - 1.96kNM_{j,j}(1 - M_{j,j}) - 1.96kN(p + q)$
  $b_1 \leftarrow kpNM_{j,j} + kqN(1 - M_{j,j}) + 1.96kNM_{j,j}(1 - M_{j,j}) + 1.96kN(p + q)$
  if there exists k such that $P(k) = j$ **then**
    $b_2 \leftarrow kpNM_{P(i),P(i)} + kqN(1 - M_{j,j}) - 1.96kNM_{P(i),P(i)}(1 - M_{P(i),P(i)}) - 1.96kN(p + q)$
    $b_3 \leftarrow kpNM_{P(i),P(i)} + kqN(1 - M_{j,j}) - 1.96kNM_{P(i),P(i)}(1 - M_{P(i),P(i)}) - 1.96kN(p + q)$
  /* *Check the condition with k only if it exits* */
  while $\neg(b_0 < \bar{M}i, i < b_1) \vee \neg(b_2 < \bar{M}_{k,k} < b_3)$ **do**
    Resample i, j, k
  $P' \leftarrow P$
  $P'(i) \leftarrow j$
  if there exists k such that $P(k) = j$ **then**
    $P'(k) \leftarrow P(i)$
  return P'

Algorithm 3 Neighbourhood Generation Algorithm for **DP-EMM** [45]

procedure NEIGHBOUR($P, \bar{M}, M$)
  $i \xleftarrow{\$} \{1, \ldots, |\mathrm{kw}(\mathbf{DB})|\}$
  $j \xleftarrow{\$} \{1, \ldots, |\mathrm{kw}(\mathbf{DB})|\}$
  $b_0 \leftarrow NM_{j,j} - 1.96NM_{j,j}(1 - M_{j,j}) - 1.96/\epsilon$
  $b_1 \leftarrow NM_{j,j} + 1.96NM_{j,j}(1 - M_{j,j}) + 1.96/\epsilon$
  if there exists k such that $P(k) = j$ **then**
    $b_2 \leftarrow NM_{P(i),P(i)} - 1.96NM_{P(i),P(i)}(1 - M_{P(i),P(i)}) - 1.96/\epsilon$
    $b_3 \leftarrow NM_{P(i),P(i)} + 1.96NM_{P(i),P(i)}(1 - M_{P(i),P(i)}) + 1.96/\epsilon$
  /* *Check the condition with k only if it exits* */
  while $\neg(b_0 < \bar{M}i, i < b_1) \vee \neg(b_2 < \bar{M}_{k,k} < b_3)$ **do**
    Resample i, j, k
  $P' \leftarrow P$
  $P'(i) \leftarrow j$
  if there exists k such that $P(k) = j$ **then**
    $P'(k) \leftarrow P(i)$
  return P'

	Without Stemming	With Stemming
# documents	480000	480000
# keywords	33366	24947
# keyword-document pairs	17415721	16881119
Max. keyword frequency	23989	40714
Min. keyword frequency	1	1
Mean keyword frequency	522.0	676.7
Max. # keywords per document	3483	2939
Min. # keywords per document	1	1
Mean # keywords per document	36.8	35.7

Figure 7: General statistics of the Enron email corpus after pre-processing.

Algorithm 4 Neighbourhood Generation Algorithm for **PRT-EMM** [33] and **FP-EMM** [45]

procedure $\textsc{Neighbour}(P, \bar{M}, M)$
  $i \xleftarrow{\$} \{1, \ldots, |\mathrm{kw}(\mathbf{DB})|\}$
  $j \xleftarrow{\$} \{1, \ldots, |\mathrm{kw}(\mathbf{DB})|\}$
  $P' \leftarrow P$
  $P'(i) \leftarrow j$
  if there exists k such that $P(k) = j$ **then**
    $P'(k) \leftarrow P(i)$
  return P'

D Experimental Data

D.1 General Information about Enron Email Corpus

The Enron email corpus [53] is a collection of over 600 thousand emails generated by 158 employees of the Enron Corporation and acquired by the Federal Energy Regulatory Commission (FERC) during its investigation of the Enron scandal. At the conclusion of the investigation, and upon the issuance of the FERC staff report, the email corpus is released to the public for historical research and academic purposes. The Enron dataset is widely used as a target for cryptanalysis on structured encryption [30, 9, 5] as it is one of the only public real-world datasets.

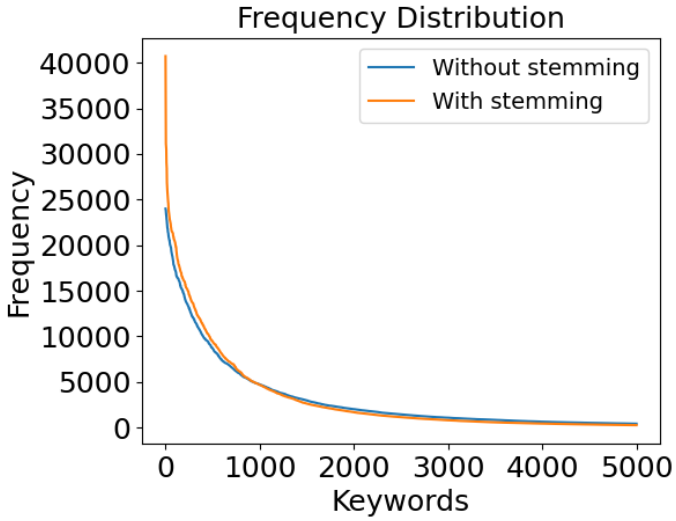


Figure 8: Frequency distribution of the 5000 most frequent keywords.

D.2 Pre-processing

We implemented our email processing and keyword extraction script in python using the Natural Language Toolkit [49] module as the tokeniser. The English stop words and other keywords with frequency higher than 5% are removed. The experiments in Section 5 are run with the keywords as they are, whereas the experiments in Section 5.5 are run with stemmed keywords. Further details of how stemming is used can be found in Section 5.5.

D.3 General Statistics of Enron Email Corpus

Figure 7 gives some general statistics of the Enron email corpus after pre-processing. We note that if the index-only EMMs are used on the documents directly to build a searchable database for the Enron email corpus, the storage overhead is over 36× (the ratio between the number of keyword-document pairs and number of keywords), assuming that the documents are padded to the maximum size by the underlying EMMs.

Figure 8 shows the frequency distribution of the 5000 most frequent keywords after pre-processing.