# An analysis of Structured Encryption compared to other secure computation technologies

## A review of Structured Encryption schemes compared to Oblivious RAM, Multi-party Computation, Homomorphic Encryption and Trusted Execution Environments in the context of computing on encrypted data

**Dorian Herbiet**[1]

**Supervisor: Lilika Markatou**[1]

[1]**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 22, 2025

Name of the student: Dorian Herbiet
Final project course: CSE3000 Research Project
Thesis committee: Lilika Markatou, Tim Coopmans

## Abstract

With the ever-growing cloud capacity in our society, being able to harness that immense computational power while keeping our data private has become a strong point of interest. This can be achieved with Structured Encryption, which allows a data structure to be encrypted and stored remotely, but still queryable by the client owning the encryption key. This report is a literature review of the field of Structured Encryption (StE). We analyze the state-of-the-art technologies and their characteristics on the following aspects: security, efficiency, functionality and usability, and discuss their capabilities and limitations. We then compare StE schemes with other promising technologies in the area of computation on encrypted data: Fully Homomorphic Encryption, Oblivious RAM, Secure Multiparty Computation and Trusted Execution Environments.


# 1 Introduction

Outsourcing data and computation to cloud environments is very common nowadays [1]. However, as a client, we do not have any guarantees about the intentions of cloud providers. For some situations, such as sensitive data with high privacy requirements, it cannot simply be assumed that the server is honest. Such sensitive data includes medical records, for which the GDPR forbids the outsourcing in plaintext [2]. Instead of requiring the server to prove that it is safe, we can use technologies to ensure that the server cannot use or tamper our sensitive data.

This problem is addressed by the field of encryption, that allows data to be stored on a server without leaking any information. Classical encryption techniques, such as AES, store the entire data as a big block of ciphertext, which does not give the server any possibility to perform operations on it, essentially making the computation power of the cloud useless.

Structured Encryption (StE) is a technology that allows the server to perform operations on encrypted data [3]. It is a set of techniques that encrypts and stores sensitive data on an untrusted server, while still keeping the structure of the data to allow the client to outsource operations securely. The structure is present, but the server is not able to see it.

In this survey, our goal is to give a clear overview of state-of-the-art Structured Encryption technologies and their characteristics. We will also compare StE with other methods for computing on encrypted data to critically assess its performance for this task in theoretical and practical scenarios. For this purpose, this report will answer the following two questions in depth:

1. What are the capabilities and limitations of modern Structured Encryption technologies ?
2. How does Structured Encryption perform compared to Oblivious RAM (ORAM), Fully Homomorphic Encryption (FHE), Trusted Execution Environments (TEEs) and Secure Multiparty Computation (MPC) ?

This paper is organized in the following sections: Section 2 outlines the historical background of Structured Encryption and details the method used to perform this literature review. Sections 3 present the state-of-the-art StE technologies and Section 4 discusses their characteristics. Section 5 compares StE against the other secure computation technologies mentioned above. Section 6 covers the ethical implications of the research and StE technologies in general. Finally, Section 7 summarizes the main findings of this research, provides suggestions for future work, and concludes the paper.

# 2 Motivation

## 2.1 Background

The ancestor of Structured Encryption was proposed by Song et al. in 2000 [4]. They proposed a technique to perform secure searches on symmetrically encrypted data, which they called Symmetric Searchable Encryption (SSE). It allowed keyword searches on encrypted documents using symmetric-key cryptography. This scheme was further enhanced by Goh in 2003 who improved the efficiency of the scheme by introducing secure indices [5]. This technique remained static: once uploaded, the encrypted data could not be modified. Kamara et al. introduced the first provably secure dynamic SSE scheme in 2012, which supported the addition and deletion of documents [6].

In its current state, SSE only allowed the search for keywords in an unordered collection of documents. In 2010, Chase and Kamara generalized this concept to any type of structured data. They defined a new, broader encryption framework to perform queries on structured data: Structured Encryption [3]. They explained how to implement search on encrypted matrices and labeled data (unordered list of documents labeled with keywords), as well as neighbor and adjacency queries on graphs. The schemes initially introduced by Chase and Kamara were static. SSE is considered a subpart of STE, focused on labeled data.

The field of Structured Encryption is governed by three fundamental aspects: functionality (also referred to as expressiveness), security and efficiency. These three components form an implicit trade-off for all StE schemes. It was formally expressed as a trilemma by Kamara and Moataz [7].

The first aspect is functionality: new protocols were created for different data structures, such as encrypted multimaps [8], to provide StE with a wider range of real-world applications. Kamara and Moataz introduced in 2018 an encryption scheme supporting a large class of SQL queries on encrypted databases [9].

The second aspect is security. The first precise definition of the security of StE/SSE schemes was given by Curtmola et al. in 2006 [10], based on the leakage to the server. Leakage is defined as the information unintentionally revealed to an attacker. Many studies then further explored the realm of attacks on encryption schemes exploiting various kinds of leakage (information leaked to the server) [11] [12] [13] [14].

The last aspect of the Structured Encryption trilemma is efficiency. Sublinear query efficiency was achieved in [10] and [3] for static encryption schemes, and later, similar efficiency was also achieved for dynamic schemes, as presented

by Kamara et al. in 2012 [6].

All improvements in the field of Structured Encryption try to balance this three-sided trade-off: improve a scheme on one aspect without compromising on the other two. In practice, Structured Encryption scheme usually do not provide perfect theoretical security in order to achieve real-world efficiency, or functionality [8].

### 2.2 Methodology

To answer the research questions stated in the introduction, a literature review was performed. We used the snowball sampling method with backward reference search to collect all the scientific papers analyzed in this report, starting with a small batch of papers on the topic of StE, then recursively iterating over the reference of each paper and add the relevant ones to the pool of papers under review [15] [16].

To assemble the initial batch of papers, a search query was performed on Scopus to gather the most relevant studies on the topic of Structured Encryption. We chose Scopus as search engine for its extensive coverage of peer-reviewed scientific publications and complex search queries functionality. We used the following search criteria:

- The search was performed using the keywords *"structured encryption"*, *"encrypted multimap"* and NOT *"homomorphic"* as they cover the general field without broadening the search unnecessarily.
- Only papers written in English were taken into consideration for this review.
- Conference reviews were excluded to only take conference papers and articles into account.

This resulted in a list of 45 papers matching the search criteria. From then, we discarded papers on encrypted graphs and networks to focus on multimap encryption schemes, because of the time and space constraint of this research. We still present a high overview of graph encryption scheme functionalities in Section 4.1. The exact Scopus query can found in Appendix A.

To perform the comparison between Structured Encryption and the other computation technologies, I benefited from the expertise of my fellow teammates: Eugen Bulboacă for FHE, Pedro Gomes Moreira for MPC, Vlad Popescu for TEEs and Sergiu Nicolae Stancu for ORAM. For this research, we each focused on those four aspects of our technology: functionality, efficiency, security and usability. (see Section 4)

## 3 Technologies

### 3.1 Definitions

A private-key associative Structured Encryption scheme is defined formally by Chase and Kamara as a tuple of five polynomial-time algorithms $\Pi = (Gen, Enc, Token, Query_e, Dec)$ [3]. The algorithms are defined as follows:

- $K \leftarrow Gen(1^k)$ is a probabilistic algorithm that takes as input a security parameter $k$ and outputs a private key $K$.
- $(\gamma, c) \leftarrow Enc_K(\delta, M)$ is a probabilistic algorithm that takes as input a private key $K$, a data structure $\delta$, and a sequences of private and semi-private data $M$. It outputs an encrypted data structure $\gamma$ and a sequence of ciphertexts $c$.
- $\tau \leftarrow Token_K(q)$ is a (possibly probabilistic) algorithm that takes as input a private key $K$ and a query $q$ and outputs a token $\tau$.
- $(J, v) := Query_e(\gamma, \tau)$ is a deterministic algorithm that takes as input an encrypted data structure $\gamma$ and a token $\tau$. It outputs a set of pointers $J$ and a sequence of semi-private data $v_I$.
- $m_j := Dec_k(c_j)$ is a deterministic algorithm that takes as input a secret key $K$ and a ciphertext $c_i$ and outputs a message $m_j$.

In this context, an associative scheme is a scheme where one can associate an item $v_i$ with a data item $m_i$ in such a way that a query returns the associated items in addition to the pointers. These items are referred to as semi-private data as they can be retrieved with the right token, unlike data items. Associativity is a useful property as it allows to "chain" multiple encryption scheme to create more complex schemes.

Schemes can be classified according to different properties:

- **Response-hiding / Response-revealing:** A response-revealing scheme, as its name suggests, reveals the response to a query to the server. This means that the server cannot decrypt the data at rest, but once some data element is queried, the server will learn its content. A response-hiding scheme keeps the response hidden.
- **Static / Dynamic:** A static scheme only supports search operations on the encrypted data. A dynamic scheme supports insertions and deletions as well.
- **Interactive / Non-interactive:** An interactive scheme requires multiple rounds of communications to perform an operation on the encrypted data, while a non-interactive scheme only requires a single round of communication.

Dynamism of StE schemes comes in different flavors. George et al. introduce the notion of *full dynamism* which means that there is the scheme does not enforce a limit to the numbers of documents and the number of keywords present in the multimap. It also allows any length for the tuples associated to the keywords [17]. While this resizability is an interesting feature, it has been shown that not fixing a maximum length for the tuple makes the scheme vulnerable to simple file-injection attacks (a similar attack is presented in Section 4.3) [18]. Therefore, all the dynamic schemes considered in this review enforce a limit to the size of tuples in the multimap.

### 3.2 Security Notions

Curtmola et al. [10] formally defined security for SSE/StE schemes. They proposed two definitions:

- **Non-Adaptative Security**: also referred to as **CKA1** (Chosen Keyword Attack 1). A cryptographic scheme is non-adaptatively secure if it is only secure when the

adversary must choose all their queries in advance, without seeing any output or behavior of the scheme. This is a weak adversarial model, and is not considered to be secure in the general case. It can however form a basis model when designing an encryption scheme as it is an easy property to prove. The scheme can then be further improved to achieve adaptative security.

- **Adaptative Security**: also referred to as **CKA2** (Chosen Keyword Attack 2). A cryptographic scheme is adaptatively secure if it is secure even when the adversary adapts its strategy and queries based on previous outcomes and behavior. Since proving that a scheme is adaptatively-secure can turn out quite complex, a simplification of this model was proposed, relying on the Random Oracle Model [19]. All cryptographic primitives used to construct the encryption scheme (hash functions, etc) are modeled as random oracles: idealized blackbox returning random outputs for each new input. Using the ROM simplifies the proof for adaptative security and allows constructing efficient schemes based on idealized assumptions about the primitives used. This model is the de facto standard for StE schemes.

The strongest security paradigm is **Universally Composable Security**. It was defined by Canetti in 2001 [20] for cryptographic protocols, and imported to SSE and StE in 2012 by Kurosawa and Ohtaki [21]. A universally composable protocol remains secure even when composed with other arbitrary protocols, it behaves like a perfect functionality. Achieving this level of security is ideal, but often comes at a high cost in terms of efficiency.

With the rise of dynamic StE schemes, new security aspects must be taken into consideration. Insertions and deletions of data can leak more information than simple searches. Two main aspects have been clearly defined: Forward Privacy (also called Forward Security) and Backward Privacy (also called Backward Security).

**Forward Privacy**: A scheme is forward-private if insertions do not reveal any information about previous searches and updates. For example, the server should not be able to learn whether a newly inserted element matches previous search queries. It was first introduced by Stefanov et al. [22], a few years after the first dynamic StE scheme [6]. It was then refined by Bost in its Sophos scheme [23].

**Backward Privacy**: A scheme is backward-private if deletions ensure that subsequent searches do not reveal any information about the deleted elements. It was defined by Bost et al. [24] in 2017 and comes in three different flavors of strength:

I. (Strong): When performing a search, the scheme leaks the documents currently matching the searched keyword, when they were inserted and the total number of updates performed on that keyword.

II. (Medium): When performing a search, the scheme leaks the documents currently matching the searched keyword, when they were inserted, and when all the updates on that keyword happened (but not their content).

III. (Weak): When performing a search, the scheme leaks the documents currently matching the searched keyword, when they were inserted, when all the updates on that keyword happened, and which deletion update canceled which insertion update.

### 3.3 Encryption Primitives

The strength of Structured Encryption comes from its modularity. StE schemes for simple data structures can be used as building blocks to construct a new scheme for a more complex data structure. The most simple scheme introduced in the original study by Chase and Kamara [3] is an encrypted multimap. A multimap is a structure that maps keys to a (variable-length) list of values. Because of its simplicity and versatility, it has been the main focus for most of the schemes created afterwards, and the foundation stone of many complex StE schemes (see Section 3.4). A dictionary is a special case of a multimap, where each key is mapped to a single value. Hence, a multimap encryption scheme can also be used to encrypt a dictionary.

Some of the most frequent uses of multimaps are *labelings* and graphs. A labeling is essentially the process of assigning keywords to each element of a set of documents. A multimap can be used to encode a labeling: each keyword forms a key, mapped to the list of documents matching it. The entire field of Symmetric Searchable Encryption therefore relies on that structure to perform efficient keyword search queries. A graph can be encoded in a multimap as a an adjacency list: each node in the graph is mapped to a list containing its neighbors.

These encodings have been given by Chase and Kamara [3] as motivations for the use of encrypted multimaps. Their original scheme encrypts the multimap using a combination of hash tables, pseudo-random functions and permutations along with classical symmetric encryption protocols such as AES.

More recent schemes are based on the same principle, but with enhanced features and security. The current state-of-the-art schemes are **dprfMM** and **XorMM** [25] [26].

### 3.4 Complex Structures

**Boolean Queries**

In the field of SSE/StE, boolean queries are search queries where multiple keywords can be used with the following boolean operators: AND, OR and NOT. Cash et al. proposed **OXT**: the first StE scheme supporting boolean queries with a worst-case linear search complexity, but sublinear query complexity for most queries [27]. This framework relies on a two data structures:

- **TSet:** this structure is used to retrieve the documents matching the least frequent keyword in the conjunctive query.
- **XSet:** this structure is used to check that the documents retrieved from TSet match the conjunctive query.

OXT remains the state-of-the-art boolean queries scheme. It is practically efficient on huge datasets, as Cash et al. proved using the entire English Wikipedia as a dataset. Kamara and Moataz proposed the IEX scheme in 2017, which achieves similar efficiency, but can be extended to be dynamic

and forward-secure [7]. Li et al. improved OXT in 2022 by creating a new version that hides the volume leakage, i.e. the amount of documents matching each keyword in the conjunctive query, which is a fundamental security threat (see Section 4.3) [28]. They achieved volume-hiding at the cost of a linear overhead in the total number of key-value pairs.

**Range Queries**

Range queries are a specific type of queries performed on numerical attributes of some datasets to retrieve all the data elements where the attribute value is inside the queried range. In the SSE/StE context, we want to be able to query the structurally encrypted data using range queries such that the server does not learn the range endpoints, nor the order between the retrieved attributes and the endpoints. The first scheme supporting such queries was introduced by Faber et al. in 2015 [29] [30]. They extended OXT by converting range queries into a disjunction of exact match queries. In their work, they also extended OXT to support substring and wildcard queries, allowing for a larger set of queries, which they called *rich queries*. While efficient, their scheme leaks to the server which documents match each value in the queried range.

Modern schemes supporting range queries transform the range checking problem into a set intersection problem, hiding the order relation and the endpoints [31] [32]. They created a scheme with no storage overhead that performs range queries in $O(N \log N)$, where $N$ is the amount of data elements.

**Encrypted Databases**

Thanks to their modular nature, multimap encryption schemes can be used as building blocks for more complex data structures. They can be seen as a blackbox, making the construction of advanced schemes more simple and the security guarantees easier to prove. This construction also allows complex schemes to benefit from improvements made on the underlying schemes. The most notable structure that has been constructed from multimaps is an StE Relational Database framework, called **SPX**, allowing (a subset of) SQL queries, introduced by Kamara and Moataz in 2018 [9]. Recall that a Relational Database consists of multiple tables, each having their own columns (attributes) and rows (entries).

Kamara and Moataz were not the first to propose an encrypted database system, previous database encryption schemes, such as CryptDB, were based on Property-Preserving Encryption (PPE) and Order-Preserving Encryption (OPE) [33] [34] [35] [36]. These systems are efficient and are legacy-friendly, however they have been shown to have weak security due to the characteristics of PPE and OPE. PPE is a type of deterministic encryption, and therefore leaks equality: two equal plaintexts will have the same ciphertext. OPE, by its nature, leaks the order of encrypted numerical data. The leakage can be used to perform inference attack on the encrypted database, as it has been shown by Naveed et al., who showed that a lot of information could be discovered, with medical records for example [37]. Grubbs et al. have also shown that theoretical threat models on which encrypted databases are built do not take into account leakage from the Database Management System itself, such as logs and diagnostic tables [38].

The structurally encrypted database framework SPX is built using multimaps and dictionaries. Different representations of the database are built, each designed to perform a particular query operation, which combined, allow for efficient query processing.

- Row-wise representation: a multimap $MM_R$ that maps the coordinate of every row (row rank / table identifier pair) in the database to the contents of that row.
- Column-wise representation: a multimap $MM_C$ with the same structure as $MM_R$ for columns.
- Equality-relation representation: a multimap $MM_V$ that maps each value in every column to all the rows that contain the same value for that column.
- Column-pair representation: a set of multimaps, one for each column $c$ in the database. Each multimap $MM_c$ maps a pair of column coordinates to all the rows that have the same value in both those columns. These are then stored in a dictionary $EDX$.

This construction provides optimal query complexity for the class of conjunctive SQL queries, which corresponds to queries of the form:

$$\text{Select } attributes \text{ From } tables$$
$$\text{Where } (att_1 = X_1 \wedge ... \wedge att_l = X_l)$$

It achieves optimal complexity under the condition that $(s_1 + ... + s_t)/h = O(1)$ where $t$ is the number of tables in the query, $s_i$ is the number of columns present in table $i$ and $h$ is the number of attributes in the *Select* term of the query. It is important to note that conjunctive queries are a completely distinct concept of conjunctive keyword searches and boolean queries described above.

This framework is a pure StE framework, in the sense that it doesn't rely on cryptographic primitives such as ORAM and FHE (their definition can be found in Section 5). It was built on top of classical StE schemes [3; 39; 27; 10; 6]. ORAM is however used on one of the data structure in order to achieve forward-privacy at the cost of a polylogarithmic blowup for the update operations. An implementation of this scheme is also proposed with Zero-Leakage (ZL) building blocks, like TWORAM [1] which is built on ORAM (See Section 5) or FZL [40] [41]. This implementation incurs an additional polylogarithmic overhead for queries.

Thanks to the modularity of the SPX framework, newer StE schemes can be used as building blocks, further improving its security. It can use the most modern pure StE scheme, such as **XorMM** and **d-DSE**, that also hide the volume-leakage, or mixed StE-ORAM technique also hiding the access pattern, such as $EMM_{avl}$ [42] [26] [18].

SPX has been further improved by Cash et al. [43] who introduced hybrid indexing. Previously, SPX performed fully precomputed joins, which required the client to download and decrypt a quadratic amount of rows. In this study, Cash et al.

[1] Note: this protocol is not ZL by default, but its block size can be carefully parameterized to achieve ZL.

have shown that using partially computed joins in some specific cases performed better, up to 231 times faster on simple queries on the City of Chicago's Data Portal [44] and Sakila's [45] datasets. In their scheme, the client chooses at query time which type of join will be used for their query based on a simple heuristic, which chooses the optimal plan in 68% of the cases.

# 4 Characteristics and Discussion

In this section, we will present in details the characteristics of state-of-the-art StE technologies. For this purpose, we will discuss in detail the three aspects of the trilemma introduced by Kamara et Moataz: functionality, efficiency and security [7]. We added a fourth aspect, usability, which outlines how StE can practically be used for real-world applications.

## 4.1 Functionality

Structured Encryption now supports a wide range of queries for multiple types of data: labelings, SQL databases (Kamara et al.), graphs (Guo et al.), etc [9] [46]. We will now cover the specific features provided by StE for different data types.

For encrypted documents indexed by keywords (multimaps), StE schemes support dynamic addition and deletion of documents and keywords, single-keyword searches, boolean queries (with the logical operators AND, OR and NOT), proposed by Kamara et al. and improved by Zhu et al., and *rich* queries, proposed by Faber et al., which includes search by substring and wildcards [7] [47] [30].

For numerical data, StE additionally supports range queries, consisting of querying all the data for which a numerical attribute is comprised in the queried range. The most recent range query scheme was proposed by Basudan et al. in 2024 [32].

For encrypted graphs, StE schemes can be used to perform adjacency and focused subgraph queries, as shown by Chase and Kamara, which are essential for web search engines [3] [48]. Guo et al. have also successfully performed secure shortest path queries on encrypted graphs [46].

It is now even used to store encrypted versions of concept graphs and knowledge graphs, which are key structures in the area of artificial intelligence and natural language processing, as shown by Poh et al. and Xue et al. [49] [50].

## 4.2 Efficiency

All modern StE schemes built after the work by Cash et al. achieve sublinear query efficiency in terms of the size of the dataset [27].

As StE aims towards practical efficiency, a lot of work was put early on in the field to allow queries to be executed efficiently using parallel processing. The most notable studies have been performed by Kamara et al., Cash et al., and Lai et al. [51] [39] [52]. With a large amount of processors, queries are ran in $O(\log N)$, where $N$ is the amount of documents in the dataset, which means that the query execution time is independent from the result size.

While reducing the asymptotic complexity is an essential process in the creation of an efficient SSE/StE scheme, the I/O efficiency is a decisive criterion to determine if a scheme can practically be used [53]. Cash et al. define an efficiency trade-off for I/O operations between the server storage size and the spatial locality of memory accesses. They prove that a lower bound is inherently caused by the security requirements of SSE/StE schemes: a secure scheme must either use $\omega(N)$ storage space or perform searches in with $\omega(1)$ memory accesses, where $N$ is the amount of keyword-document pairs in the multimap.

Modern schemes achieve $O(N)$ (optimal) storage space, while maintaining the amount of memory access efficient [54] [55]. Because of the trade-off mentioned above, they cannot provide constant search time and apply the security requirements simultaneously, but they achieve sub-logarithmic search complexity.

Very recently, concurrency in the context of StE scheme has been studied by Agarwal et al. [56]. This is an important feature for an environment where multiple clients should access the multimap stored on a server. They built an encrypted multimap scheme that allows lock-free concurrent query and append operations.

The performance of StE schemes is sufficient for real-world scenarios, as it only causes a sublinear overhead compared to plaintext storage. Real-world applications will be shown in Section 4.4. However, efficiency alone is not enough to determine if Structured Encryption should be used or not, the key is aspect is security, which we will develop in the next section.

## 4.3 Security

Different types of attacks have been tried on StE schemes, and their success rate has been thoroughly analyzed. We will first discuss passive attacks, where the adversary does not modify the client's data, then We will go over active attacks, where the adversary has the opportunity of interacting with and changing the client's data.

The first kind of attack performed on StE/SSE are keyword inference attacks on schemes that leak the access pattern (which data is accessed and when) such as the original scheme proposed by Chase and Kamara. They aim to discover which keyword is queried in the multi-map, and what documents it is linked to. The very first one was performed by Islam et al. in 2012, with the goal of decrypting queries sent to the server, and Cash et al. proposed a new, slightly more effective inference attack in 2015. [11] [12]. These attacks, respectively known as the IKK (in reference to its authors' initials) and Count attacks, rely on strong assumptions about the data, mainly the attacker must have almost full data knowledge: both attacks require the attacker to know at least 70% of the dataset to become slightly successful. (see Appendix B)

The main takeaway from those two passive attacks is to avoid using StE schemes on public datasets, as an adversary could easily infer queries, but using StE on a completely private dataset reduces the risk of such attacks.

Further research has been done in the topic of inference attacks. Volume leakage has been identified as a major threat for the security of cloud-based StE schemes [14]. This claim about volume leakage has proven especially true in the context of schemes supporting range queries. In their studies,

Kellaris et al. show that a quadratic number of queries (in the amount of distinct plaintext values) is sufficient to reconstruct the data structure.

Two main studies by Lacharité et al. and Grubbs et al. have performed even more efficient attacks on such schemes with a high success rate [57] [58]. They showed that $O(N \log N)$ queries (where N is the amount of distinct plaintext values) are enough to fully recover the data structure.

The next attack we will present is an active attack, called file-injection attack. It was proposed by Zhang et al. in 2016 [13], aimed for encrypted multimaps with keywords. They considered a scenario where the server can inject files to the client, who then encrypts and store them. Except for this file injection, the server behaves semi-honestly. These attacks are quite easy to perform in practice: if a client is using an StE/SSE scheme to store and search over encrypted emails (e.g. Pmail), the server can simply send an email to the client [59]. This attacks performs significantly better than the IKK inference attack and the Count attack, as it is already 40% effective with only 1% of the dataset known by the attacker. (see Appendix B) This attack can be counteracted with a simple measure: forward-privacy, which ensures that insertions of new (possibly maliciously crafted) files do not leak any information about future queries. This attack highlights the importance of forward-privacy for all dynamic Ste schemes.

The final attack we will discuss is again a leakage-abuse attack from a passive adversarial model, designed by Hoover et al. against encrypted SQL databases [60]. It is the first attack that exploits the cross-column equality leakage, which is only present in the specific context of StE databases. They show that StE databases are often not significantly more secure than databases built on Property-preserving Encryption like CryptDB, which was introduced by Popa et al. in 2011, if they leak the response volume. The mitigation techniques proposed include volume-hiding schemes or strong encryption primitives like ORAM and FHE (See Section 5) [33] [25].

These attacks show that, while StE schemes propose an attracting efficiency, their vulnerabilities should not be overlooked. They provide more privacy than plaintext multimap storing, but it gives a false sense of security. In order to achieve a satisfactory security, StE schemes must balance by loosing some of their efficiency.

The attacks mentioned above highlights the importance of volume-hiding, which is only achieved in recent schemes such as the one proposed by Patel et al. and Wang et al. [25] [61] [26].

Dynamic schemes must also implement forward and backward-privacy to mitigate file-injection attacks [24]. Old StE schemes that do not provide those features should not be considered secure and should not be used for sensitive data because of their vulnerability to attacks [14].

There exists a type of StE/SSE schemes called Verifiable StE/SSE that provides security guarantees about a malicious server tampering with the data [62]. Because most schemes focus on the semi-honest model, this type of scheme has not received much attention from researchers, therefore we will not focus on it.

### 4.4 Usability

From the beginning, achieving practical real-world efficiency has been one of the main goal of StE and SSE, with researchers testing their encryption schemes against real-world use cases and datasets [27].

In 2022, MongoDB, a document-oriented database, built their Queryable Encryption framework based on the work of Kamara and Moataz [63] [9] [64]. Starting from version 6.0, MongoDB allows user to manage an end-to-end encrypted database built on top of StE schemes with practical efficiency. However, no studies have been performed to critically assess the performance overhead. It offers the following features: keyword queries, range queries and substring queries. It was built to maximize compatibility, and can therefore be used with most of the existing databases.

Other encrypted database projects exist, such as Cipherstash and Crypteron [65] [66].

## 5 Comparison

In this section, we will perform a comparison of StE with the four computation techniques on encrypted data: Secure Multiparty Computation, Fully Homomorphic Encryption and Trusted Execution Environments.

We will compare and contrast StE with each technique individually, and present a table containing the main differences between them (this table can be found in Appendix C). We will then analyze a very specific use case, Private Set Intersection, to assess what technology performs best and gain insight on their performance in practical real-world scenarios.

### 5.1 General Comparison

**Fully Homomorphic Encryption (FHE)**

FHE is a type of encryption that enables computation on encrypted data without decrypting it, preserving confidentiality throughout the entire computation. The first feasible construction, using a lattice-based cryptosystem, was introduced by Craig Gentry in 2009 [67].

FHE is a general purpose cryptographic primitive. It allows for arbitrary computations that can be expressed as circuits on encrypted data. Its use cases are therefore larger than the encrypted search provided by Structured Encryption. It can perform computation on any kind of data, even on real numbers, thanks to its approximate arithmetic feature [68]. Any feature provided by StE can be performed by FHE.

While StE is designed with practical efficiency in mind, FHE is built for maximum security, and does not achieve fast enough operations for real world applications yet.

FHE achieves IND-CPA (Indistinguishability under Chosen Plaintext Attack) security, and even IND-CCA1 (Indistinguishability under Chosen Ciphertext Attack) in some cases. Attempts to increase the security level to IND-CCA2 have proven to be unsuccessful [69]. It is built for a malicious adversarial model, but StE is built on the semi-honest model, where the server does not perform any malicious action except listening, which makes these two techniques hard to compare security-wise. It is however fair to say that FHE is built to stand against stronger adversarial models.

**Oblivious RAM (ORAM)**

ORAM is a technique that hides the access patterns to memory to prevent leakage of sensitive information. Access sequences do not reveal any information about the data being accessed in the ORAM. The concept was introduced by Goldreich and Ostrovsky who combined their previous work into one paper, published in 1996 [70].

ORAM is used for a similar purpose as StE, that is to store encrypted data on a server, and access it without that the server learns anything about the data. ORAM simply stores data without a structure, it can be used for structured data, but the client must keep track of the structure themselves. While they share a common objective, these techniques differ in the procedure to achieve it.

First of all, ORAM hides the access pattern, the server does not know which data block is accessed [70]. StE does not hide the access pattern, it only hides the structure of the data. This means that ORAM is a general purpose solution, that can be used for any kind of data on which the client desires to perform encrypted queries, while StE is used for specific data structures.

In the efficiency aspect, we will consider two metrics: storage space and query time. The most efficient ORAM technique, Ring ORAM, requires $O(\log N)$ storage on the client, and $O(N)$ on the server, where $N$ is the amount of data blocks [71]. StE on the other hand does not require any storage on the client (except for the encryption keys obviously) for the same storage complexity server-side.

The block access time of Ring ORAM is $O(\log N)$. For the purpose of searchable encryption, this is not practically efficient, and produces a massive communication overhead. ORAM in its current state is therefore not a viable solution compared to efficient StE schemes.

ORAM can however be used within an StE scheme to combine its access pattern hiding property with the practical efficiency of StE. For example, the TWORAM scheme makes use of ORAM to achieve zero-leakage [40]. The most recent scheme proposed by Boldyreva et al. also uses ORAM to hide the query, access and volume pattern [18], at the cost of a small overhead.

**Secure Multiparty Computation (MPC)**

MPC is a type of cryptographic protocols allowing multiple parties to jointly compute a function over their private inputs. The function is evaluated securely, ensuring correctness and privacy of the inputs, even in the presence of a malicious party. This concept was introduced by Yao in 1982 [72].

Similarly to FHE, MPC can be used for the computation of an arbitrary functions between two (or more) parties while keeping the inputs private [73]. It can therefore technically be used for encrypted search like StE, but the field of private search should focus on non-interactive techniques with a sublinear complexity in the amount of data elements, instead of general-purpose tool like MPC or FHE [3]. Also, StE is designed for a client-server architecture, e.g. a cloud environment, while MPC is built on computation between multiple peers, in a broader setting than just a client and a server.

Another similarity of MPC shared with FHE is the focus on security: MPC protocols aim for a high security standard, often at the cost of efficiency, while StE tolerates information leakage to achieve practical efficiency.

StE and MPC share however the same threat model. Both are generally built for a semi-honest adversary, but some MPC protocols also provide security guarantees against malicious adversaries. One method to achieve this is to embed ZK (Zero Knowledge) Proofs at each step of the computation that ensure that each party is computing their part honestly without revealing that part [74]. Most MPC protocols provide computational security, meaning that it can only be broken by an adversary with unlimited computational power, but some, like Shamir's Secret Sharing provide information-theoretic security, which means that it cannot be broken, even with unlimited computational power [75].

**Trusted Execution Environments (TEEs)**

TEEs are isolated areas within a processor that provide secure execution of code as well as data confidentiality and integrity, even in the presence of a compromised operating system. TEEs can be categorized into two main types: hardware-based and VM-based TEEs. A notable implementation is Intel SGX, introduced by Costan and Devadas [76].

The key difference between TEEs and the four other technologies is the use of secure hardware. While StE, ORAM, FHE and MPC are implemented by relying solely on software cryptography, TEEs are built at a low level [76]. They come in two flavors: Enclave-based and VM-based TEEs (also called Confidential Virtual Machines). Enclave-based TEEs consist of an isolated enclave located in user-space, thus relying on the host OS to perform system calls. VM-based TEEs are, as their name suggests, based on secure VMs communicating with the hypervisor. They require a long time to boot, but achieve near-native efficiency [77]. Compared to StE, which achieves a sublinear overhead compared to plaintext storage, TEEs achieve a constant overhead compared to a native machine.

TEEs are aimed to be a general-purpose solution for computation on an untrusted server (where only the hardware is trusted), which allows for arbitrary computation, much like FHE and MPC. They are designed for the strong threat model where the server software is compromised, where StE just assumes a semi-honest server.

Because of their hardware nature, they are vulnerable to a completely different category of attacks than StE: side-channels attacks, exploiting indirect signals such as power and timing, and architectural attacks, exploiting design flaws in hardware components. These types of attack are very hard to mitigate, as shown by Muñoz et al. [78].

## 5.2 Private Set Intersection

ñ In this section, we will compare StE against the other secure computation techniques on a very specific use case: Private Set Intersection (PSI). In the context of secure computation, it is one of the most useful operation. It enables two (or more) parties to compute the intersection of their individual private datasets without leaking any other information to the other parties.

This cryptographic technique has a variety of use cases. Private messenger applications such as Signal needs to dis-

cover which of your contacts are also users of the app [79]. However, you don't want to reveal your full contact list, and Signal doesn't want to disclose their entire user base. Signal therefore uses PSI to allow both you and their server to discover shared contacts. PSI is also used to measure the ad conversion without revealing all user data, which would violate the GDPR and in collaborative financial fraud detection and collaborative medical research with sensitive health data sharing [80].

Very recently, Agarwal et al. proposed an efficient updatable PSI scheme based on StE [81]. Updatable means that both parties can arbitrarily insert/delete elements without having to re-encrypt their entire set. Similarly to classical StE schemes, they built their PSI protocol for a semi-honest server, based on a new encrypted set structure: **ESX**. Its structure is inspired by modern Path ORAM techniques: the set is stored in a tree of dynamically resizable buckets [82]. It is the first scheme to support arbitrary insertions and deletions, which it achieves in a time linear in the amount of updates already performed.

Before the StE protocol explained above, the best updatable PSI scheme was based on an MPC technique called Decisional Diffie-Hellman [83] and performs its operation in a time linear in the amount of updates. However, it only supported a weaker form of deletions: elements were deleted after a specific amount of epochs.

Trusted Execution Environments and Fully Homomorphic Encryption were also explored to implement PSI. Kerschbaum et al. proposed a 2-part protocol, consisting of a pre-processing phase (called offline phase) where heavy cryptographic functions are computed, and a very efficient online phase were the two parties interact [84]. In this scheme, a TEE is used during the offline phase. Chen et al. proposed a PSI scheme based on leveled FHE in 2017 [85]. These two schemes are very efficient, with a computation overhead linear in the size of the smaller set (or even sublinear for TEEs), however they don't support dynamic updates.

It is interesting to note that static schemes have a computation overhead for queries linear in the size of the larger set, while dynamic schemes have an overhead linear in the amount of updates performed. For a scenario where the sets to intersect are fixed, methods such as static MPC, FHE and TEEs provide the best execution time. In the case of dynamic sets, StE is currently the most efficient solution that provides both insertions and deletions in a reliable fashion.

# 6 Responsible Research

## 6.1 Reproducibility and Transparency

The methodology used to gather the knowledge presented in this report has been transparently detailed in Section 2.2, for reproducibility reasons. The exact Scopus query used is shown in Appendix A. All the papers on the topic of StE can be found in the result of this query, or by following references.

The last query was performed on the 6th of June 2025, therefore reproducing this query at a later date will yield different results as new papers could have been published, or papers might have been deleted from Scopus.

## 6.2 Ethical Considerations

In a literature review, we are subject to bias when choosing which papers to include or exclude, and what search query we perform. In this survey, because of the time and space constraints, we chose to focus on encrypted multimaps, which are a widely used instance of StE, without diving into encrypted graphs or other data structures.

Being aware of this bias, we covered use cases and functionality of encrypted graphs without exploring the details of the underlying StE schemes.

Generative AI was only used to provide grammatical and syntactic improvements to the text, not to create content.

The impact of this paper extends outside of research on Structured Encryption or related technologies. Indeed, readers might use this paper to choose an encryption technology to handle their own data. As we do not know the type of data and its security requirements, we must ensure that all the information present in this report is correct to allow the reader to make an informed decision. The consequences of a reader using an encryption scheme thinking it is secure while it is in reality vulnerable could be dire. Think of a hospital using an insecure SSE scheme to outsource their private medical records for example which could leak the private information of the patients.

# 7 Conclusion and Future Work

In this report, we analyzed the state-of-the-art Structured Encryption technologies, which allow a client to store a data structure encrypted on a remote server while still being able to efficiently perform queries on the data. It was first introduced by Chase and Kamara in 2010, but the field now includes many different schemes for different purposes [3].

StE allows for a variety of queries on structured data. For multimaps storing keyword-annotated documents, modern StE schemes allow for single-keyword search, boolean queries, substring and wildcard searches [7] [30], as well as the addition and deletion of new keywords and documents [86]. For numerical data, range queries can be performed [32]. The combination of the preceding schemes can be used to create a structurally-encrypted database supporting a large subset of SQL [9] [43]. More complex schemes exist to encrypt graphs, and perform focused subgraph and shortest path queries [3] [46].

These features of StE are achieved with sublinear query complexity in the amount of data elements. To achieve such efficiency, StE must accept to leak some information to the server. Many attacks have been performed on schemes leaking the access and volume information (what data is accessed and how large the query result is) [14] [57]. Dynamic schemes have also been proved to be vulnerable to injection attacks [13]. This shows that very fast queries come at a security cost, and slower schemes providing better security and less leakage should be preferred over insecure faster schemes.

Compared to FHE and MPC, StE only provide functionality for queries over encrypted structured data, while FHE and MPC provide a universal ways of performing secure computations between untrusted parties. FHE can be used a secure cryptographic primitive to perform very secure search as well,

but its efficiency makes it unpractical for real-world applications. StE performs very well for dynamic PSI (Private Set Intersection), which is a problem usually handled by MPC [81].

TEEs are not built for the same threat model as StE, but can also be used for encrypted search, and achieve near-native performance. They are however sensitive to attacks that are very hard to mitigate: architectural attacks and side-channel attacks, which do not apply to StE [78].

ORAM can be used a secure cryptographic primitive combined with StE to provide schemes hiding the access pattern, hence with improved security, at the cost of a small overhead [40]. On its own, ORAM hides only the access pattern of data, and doesn't keep track of the data structure like StE.

There have not been much comparison between the five techniques: FHE, StE, MPC, TEEs and ORAM. Future research could focus on creating a precise benchmark between those techniques on specific problems and use cases, with a clear table comparing the difference in performance of each technique.

On the specific topic of StE, there has not been much research about server-side querying, a scenario where the server should be able to perform some type of queries on the structurally encrypted data. Agarwal et al. explored this concept for their PSI implementation, but they have been the only one [81]. The field could also benefit from new schemes designed for active adversarial models, such as Verifiable SSE/StE, and studies on the weaknesses and strengths of such schemes against the state-of-the-art schemes.

In conclusion, StE schemes are an efficient way to query over encrypted data, that can and are used in real-life situations, but secure schemes come at an efficiency cost, which is preferable to the leakage of sensitive information by very fast but insecure schemes.

## A Scopus Query

The exact Scopus query used for this research is the following:

*( TITLE-ABS-KEY ( "structured encryption" ) OR TITLE-ABS-KEY ( "encrypted multimap" ) AND NOT TITLE-ABS-KEY ( "homomorphic" ) ) AND ( EXCLUDE ( LANGUAGE , "Chinese" ) ) AND ( EXCLUDE ( DOCTYPE , "cr" ) )*

## B Attacks against StE

The success rate of the IKK attack by Islam et al., the Count attack by Cash et al., and the file-injection attack by Zhang et al. are plotted in Figures 1 and 2.

## C General Comparison Table

The general comparison between FHE, ORAM, MPC, StE and TEE is summarized in Tables 1 and 2. It covers the aspects of functionality, efficiency, security and usability.

## D PSI Protocols

Table 3 compares the PSI protocols mentioned above, based on StE, MPC, FHE and TEEs. It outlines the main difference in terms of dynamic update capabilities, overhead and information leakage.

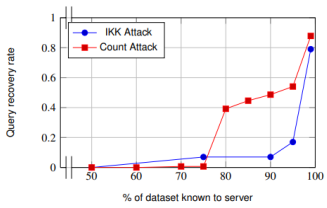


Figure 1: Comparison of the query recovery rates when server has partial knowledge of true document set for the IKK and Count attacks. Enron dataset, 500 keywords, 150 queried uniformly. [12]

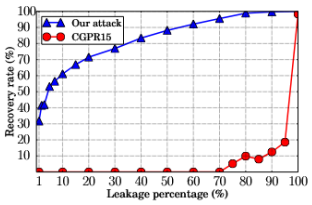


Figure 2: Comparison of Count and File-injection attacks for query reconstruction corresponding to a single token. [13]

## References


[1] Flexera. Flexera 2024 state of the cloud report. Technical report, Flexera Software LLC, 2024. 13th Annual Report, published March 12, 2024.

[2] European Union. General Data Protection Regulation (GDPR) - Article 32: Security of processing. https://gdpr-info.eu/art-32-gdpr/, 2016. Regulation (EU) 2016/679 of the European Parliament and of the Council.

[3] Melissa Chase and Seny Kamara. Structured Encryption and Controlled Disclosure. In Masayuki Abe, editor, *Advances in Cryptology - ASIACRYPT 2010*, pages 577–594, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[4] Dawn Xiaoding Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In *Proceeding 2000 IEEE Symposium on Security and Privacy. S&P 2000*, pages 44–55, May 2000. ISSN: 1081-6011.

[5] Eu-Jin Goh. Secure Indexes, 2003. Published: Cryptology ePrint Archive, Paper 2003/216.

[6] Seny Kamara, Charalampos Papamanthou, and Tom Roeder. Dynamic searchable symmetric encryption. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, CCS '12, pages 965–976, New York, NY, USA, 2012. Association for Computing Machinery. event-place: Raleigh, North Carolina, USA.

[7] Seny Kamara and Tarik Moataz. Boolean searchable symmetric encryption with worst-case sub-linear complexity. In *Advances in Cryptology–EUROCRYPT 2017: 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques,*

Table 1: Functionality and Usability Comparison of Privacy-Enhancing Techniques

| Technique | Computation Type | Parties Communication | Applicability | Use Cases |
|---|---|---|---|---|
| FHE | Any computation | Non-interactive Client–server | Available in open-source libraries | Medical data analysis Recommender systems Confidential ML |
| MPC | General computation (excluding specialized protocols) | Multiple clients or distributed parties | Used in practice but with limitations | Secure auctions DNA comparison Collaborative research |
| ORAM | Data access | Non-interactive Client(s)–server(s) | Used in secure processors and oblivious DBs | SGX integration ObliDB, Signal protocol |
| StE | Specific data access on encrypted structures | Non-interactive Client–server | Practical protocols for specific structures | Encrypted DBMS (e.g. MongoDB) |
| TEE | Any computation | Interactive Client–server with attestation service | Optional in real world cloud deployment | Data analytics Trusted AI workloads Medical Federated Learning |

Table 2: Security and Performance Comparison of Privacy-Enhancing Techniques

| Technique | Threat Model | Information Leakage | Performance Overhead |
|---|---|---|---|
| FHE | IND-CCA2 Adaptive attack | None by itself | High: Key Generation & Polynomial Operations |
| MPC | Semi-honest or malicious | Nothing beyond function output | Constant or Linear |
| ORAM | Semi-honest or malicious | Leakage through side-channel attacks | Logarithmic |
| StE | Semi-honest | Access pattern sometimes response volume | Sublinear |
| TEE | Malicious actor controlling server | Access patterns, plaintext in CPU | Generally near-native, bottleneck in I/O heavy |

Table 3: Comparison of PSI protocols implemented using different techniques. $N_x$ = size of the smaller set, $N_y$ = size of the larger set, $N_{tot}$ = size of the accumulated sets, and $u$ = number of updates performed.

| | **Pre-processing (Offline Phase)** | **Insertions** | **Deletions** | **Computation Overhead** | **Leakage** |
|---|---|---|---|---|---|
| **StE** [81] | No | Yes | Yes | $O(u \times \text{polylog}(N_{tot}))$ | Size of the update set in each epoch |
| **MPC** [83] | No | Yes | Partially | $O(u)$ (amortized) | Size of the update set in each epoch |
| **FHE** [85] | No | No | No | $O(N_x \log N_y)$ | No leakage |
| **TEEs** [84] | Yes | No | No | $O\left(\frac{N_x \log N_y}{\log \log N_y}\right)$ (online phase) | No leakage |


*Paris, France, April 30–May 4, 2017, Proceedings, Part III 36*, pages 94–124. Springer, 2017.

[8] Seny Kamara and Tarik Moataz. Encrypted Multi-Maps with Computationally-Secure Leakage, 2018. Published: Cryptology ePrint Archive, Paper 2018/978.

[9] Seny Kamara and Tarik Moataz. SQL on structurally-encrypted databases. In *Advances in Cryptology–ASIACRYPT 2018: 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2–6, 2018, Proceedings, Part I 24*, pages 149–180. Springer, 2018.

[10] Reza Curtmola, Juan Garay, Seny Kamara, and Rafail Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*, CCS '06, pages 79–88, New York, NY, USA, 2006. Association for Computing Machinery. event-place: Alexandria, Virginia, USA.

[11] Mohammad Saiful Islam, Mehmet Kuzu, and Murat Kantarcioglu. Access Pattern Disclosure on Searchable Encryption: Ramification, Attack and Mitigation. In *Proceedings of the 19th Annual Network and Distributed System Security Symposium (NDSS)*, 2012.

[12] David Cash, Paul Grubbs, Jason Perry, and Thomas Ristenpart. Leakage-Abuse Attacks Against Searchable Encryption. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 668–679, New York, NY, USA, 2015. Association for Computing Machinery. event-place: Denver, Colorado, USA.

[13] Yupeng Zhang, Jonathan Katz, and Charalampos Papamanthou. All your queries are belong to us: the power of {File-Injection} attacks on searchable encryption. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 707–720, 2016.

[14] Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. Generic Attacks on Secure Outsourced Databases. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 1329–1340, New York, NY, USA, 2016. Association for Computing Machinery. event-place: Vienna, Austria.

[15] Julian P. T. Higgins, James Thomas, Jacqueline Chandler, Miranda Cumpston, Tianjing Li, Matthew J. Page, and Vivian A. Welch. *Cochrane Handbook for Systematic Reviews of Interventions*. Wiley-Blackwell, 2 edition, 2019.

[16] Charlie Parker, Sam Scott, and Alistair Geddes. Snowball sampling. *SAGE research methods foundations*, 2019.

[17] Marilyn George, Seny Kamara, and Tarik Moataz. Structured Encryption and Dynamic Leakage Suppression. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021*, pages 370–396, Cham, 2021. Springer International Publishing.

[18] Alexandra Boldyreva and Tianxin Tang. Encrypted multi-map that hides query, access, and volume patterns. In *International Conference on Security and Cryptography for Networks*, pages 230–251. Springer, 2024.

[19] Mihir Bellare and Phillip Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, CCS '93, pages 62–73, New York, NY, USA, 1993. Association for Computing Machinery. event-place: Fairfax, Virginia, USA.

[20] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*, pages 136–145. IEEE, 2001.

[21] Kaoru Kurosawa and Yasuhiro Ohtaki. UC-secure searchable symmetric encryption. In *Financial Cryptography and Data Security: 16th International Conference, FC 2012, Kralendijk, Bonaire, Februray 27-March 2, 2012, Revised Selected Papers 16*, pages 285–298. Springer, 2012.

[22] Emil Stefanov, Charalampos Papamanthou, and Elaine Shi. Practical dynamic searchable encryption with small leakage. *Cryptology ePrint Archive*, 2013.

[23] Raphael Bost. Sophos: Forward secure searchable encryption. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1143–1154, 2016.

[24] Raphaël Bost, Brice Minaud, and Olga Ohrimenko. Forward and backward private searchable encryption from constrained cryptographic primitives. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1465–1482, 2017.

[25] 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 *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pages 79–93, 2019.

[26] Jianfeng Wang, Shi-Feng Sun, Tianci Li, Saiyu Qi, and Xiaofeng Chen. Practical volume-hiding encrypted multi-maps with optimal overhead and beyond. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, CCS '22, page 2825–2839, New York, NY, USA, 2022. Association for Computing Machinery.

[27] David Cash, Stanislaw Jarecki, Charanjit Jutla, Hugo Krawczyk, Marcel-Cătălin Roşu, and Michael Steiner. Highly-scalable searchable symmetric encryption with support for boolean queries. In *Advances in Cryptology–CRYPTO 2013: 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, pages 353–373. Springer, 2013.

[28] Tianci Li, Jiaojiao Wu, and Jianfeng Wang. Efficient volume-hiding encrypted multi-maps with support for conjunctive queries. In *IFIP International Conference on ICT Systems Security and Privacy Protection*, pages 467–485. Springer, 2022.

[29] Stanislaw Jarecki, Charanjit Jutla, Hugo Krawczyk, Marcel Rosu, and Michael Steiner. Outsourced symmetric private information retrieval. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, pages 875–888, New York, NY, USA, 2013. Association for Computing Machinery. event-place: Berlin, Germany.

[30] Sky Faber, Stanislaw Jarecki, Hugo Krawczyk, Quan Nguyen, Marcel Rosu, and Michael Steiner. Rich queries on encrypted data: Beyond exact matches. In *Computer Security–ESORICS 2015: 20th European Symposium on Research in Computer Security, Vienna, Austria, September 21-25, 2015, Proceedings, Part II 20*, pages 123–145. Springer, 2015.

[31] Juntao Gao, Xinxiang Liu, Xuelian Li, and Baocang Wang. Leakage-Suppressed Range Query Scheme for Structured Data in IoT. *IEEE Systems Journal*, 16(3):3531–3542, September 2022.

[32] Sultan Basudan and Abdulrahman Alamer. Efficient Privacy-Preserving Range Query With Leakage Suppressed for Encrypted Data in Cloud-Based Internet of Things. *IEEE Access*, 12:187652–187664, 2024.

[33] Raluca Ada Popa, Catherine MS Redfield, Nickolai Zeldovich, and Hari Balakrishnan. Cryptdb: Protecting confidentiality with encrypted query processing. In *Proceedings of the twenty-third ACM symposium on operating systems principles*, pages 85–100, 2011.

[34] Mihir Bellare, Alexandra Boldyreva, and Adam O'Neill. Deterministic and efficiently searchable encryption. In *Advances in Cryptology-CRYPTO 2007: 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007. Proceedings 27*, pages 535–552. Springer, 2007.

[35] Rakesh Agrawal, Jerry Kiernan, Ramakrishnan Srikant, and Yirong Xu. Order preserving encryption for numeric data. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 563–574, 2004.

[36] Alexandra Boldyreva, Nathan Chenette, Younho Lee, and Adam O'neill. Order-preserving symmetric encryption. In *Advances in Cryptology-EUROCRYPT 2009: 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings 28*, pages 224–241. Springer, 2009.

[37] Muhammad Naveed, Seny Kamara, and Charles V Wright. Inference attacks on property-preserving encrypted databases. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 644–655, 2015.

[38] Paul Grubbs, Thomas Ristenpart, and Vitaly Shmatikov. Why your encrypted database is not secure. In *Proceedings of the 16th workshop on hot topics in operating systems*, pages 162–168, 2017.

[39] David Cash, Joseph Jaeger, Stanislaw Jarecki, Charanjit Jutla, Hugo Krawczyk, Marcel-Cătălin Roşu, and Michael Steiner. Dynamic Searchable Encryption in Very-Large Databases: Data Structures and Implementation, 2014. Published: Cryptology ePrint Archive, Paper 2014/853.

[40] Sanjam Garg, Payman Mohassel, and Charalampos Papamanthou. TWORAM: Efficient oblivious RAM in two rounds with applications to searchable encryption. In *Annual International Cryptology Conference*, pages 563–592. Springer, 2016.

[41] Seny Kamara and Tarik Moataz. Structured Encryption. In Sushil Jajodia, Pierangela Samarati, and Moti Yung, editors, *Encyclopedia of Cryptography, Security and Privacy*, pages 1–3. Springer Berlin Heidelberg, Berlin, Heidelberg, 2019.

[42] Dongli Liu, Wei Wang, Peng Xu, Laurence T Yang, Bo Luo, and Kaitai Liang. {d-DSE}: Distinct Dynamic Searchable Encryption Resisting Volume Leakage in Encrypted Databases. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 2563–2580, 2024.

[43] David Cash, Ruth Ng, and Adam Rivkin. Improved Structured Encryption for SQL Databases via Hybrid Indexing. In Kazue Sako and Nils Ole Tippenhauer, editors, *Applied Cryptography and Network Security*, pages 480–510, Cham, 2021. Springer International Publishing.

[44] City of Chicago. City of chicago data portal. https://data.cityofchicago.org/, 2021. Accessed: 2021.

[45] MySQL. Sakila sample database. https://dev.mysql.com/doc/sakila/en/, 2021. Accessed: 2021.

[46] Jingjing Guo and Jiacong Sun. Secure shortest distance queries over encrypted graph in cloud computing. *Wireless Networks*, 30(4):2633–2646, 2024.

[47] Xueling Zhu, Huaping Hu, Shaojing Fu, Qing Wu, and Bo Liu. Efficient boolean sse: A novel encrypted database (edb). In *International Conference on Frontiers in Cyber Security*, pages 373–387. Springer, 2020.

[48] Ronny Lempel and Shlomo Moran. SALSA: the stochastic approach for link-structure analysis. *ACM Transactions on Information Systems (TOIS)*, 19(2):131–160, 2001. Publisher: ACM New York, NY, USA.

[49] Geong Sen Poh, Moesfa Soeheila Mohamad, and Muhammad Reza Z'aba. Structured encryption for conceptual graphs. In *International Workshop on Security*, pages 105–122. Springer, 2012.

[50] Yujie Xue, Lanxiang Chen, Yi Mu, Lingfang Zeng, Fatemeh Rezaeibagha, and Robert H Deng. Structured encryption for knowledge graphs. *Information Sciences*, 605:43–70, 2022. Publisher: Elsevier.

[51] Seny Kamara and Charalampos Papamanthou. Parallel and dynamic searchable symmetric encryption. In *Financial Cryptography and Data Security: 17th International Conference, FC 2013, Okinawa, Japan, April 1-5, 2013, Revised Selected Papers 17*, pages 258–274. Springer, 2013.

[52] Russell WF Lai and Sherman SM Chow. Parallel and dynamic structured encryption. In *International Conference on Security and Privacy in Communication Systems*, pages 219–238. Springer, 2016.

[53] David Cash and Stefano Tessaro. The locality of searchable symmetric encryption. In *Advances in Cryptology–EUROCRYPT 2014: 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings 33*, pages 351–368. Springer, 2014.

[54] Gilad Asharov, Moni Naor, Gil Segev, and Ido Shahaf. Searchable symmetric encryption: optimal locality in linear space via two-dimensional balanced allocations. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 1101–1114, New York, NY, USA, 2016. Association for Computing Machinery. event-place: Cambridge, MA, USA.

[55] Ioannis Demertzis, Dimitrios Papadopoulos, and Charalampos Papamanthou. Searchable encryption with optimal locality: Achieving sublogarithmic read efficiency. In *Advances in Cryptology–CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part I 38*, pages 371–406. Springer, 2018.

[56] Archita Agarwal, Seny Kamara, and Tarik Moataz. Concurrent encrypted multimaps. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 169–201. Springer, 2024.

[57] 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 (SP)*, pages 297–314, May 2018. ISSN: 2375-1207.

[58] 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 (SP)*, pages 655–672. IEEE, 2017.

[59] Tony Pr. Pmail: Encrypted Email via SMTP and IMAP. https://github.com/tonypr/Pmail, 2025. GitHub repository, accessed June 7, 2025.

[60] Alexander Hoover, Ruth Ng, Daren Khu, Yao'An Li, Joelle Lim, Derrick Ng, Jed Lim, and Yiyang Song. Leakage-Abuse Attacks Against Structured Encryption for SQL. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 7411–7428, Philadelphia, PA, August 2024. USENIX Association.

[61] Jiafan Wang and Sherman SM Chow. Simple storage-saving structure for volume-hiding encrypted multimaps: (a slot in need is a slot indeed). In *IFIP Annual Conference on Data and Applications Security and Privacy*, pages 63–83. Springer, 2021.

[62] Moesfa Soeheila Mohamad and Geong Sen Poh. Verifiable structured encryption. In *International Conference on Information Security and Cryptology*, pages 137–156. Springer, 2012.

[63] Lily Hay Newman. A long-awaited defense against data leaks may have just arrived. https://www.wired.com/story/mongodb-queryable-encryption-databases/, August 2022. Accessed: 2025-06-09.

[64] MongoDB, Inc. Mongodb official website. https://www.mongodb.com. Accessed: 2025-06-13.

[65] Cipherstash, Inc. Cipherstash official website. https://www.cipherstash.com. Accessed: 2025-06-13.

[66] Crypteron, Inc. Crypteron official website. https://www.crypteron.com. Accessed: 2025-06-13.

[67] Craig Gentry. *A fully homomorphic encryption scheme*. Stanford university, 2009.

[68] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. In *Advances in cryptology–ASIACRYPT 2017: 23rd international conference on the theory and applications of cryptology and information security, Hong kong, China, December 3-7, 2017, proceedings, part i 23*, pages 409–437. Springer, 2017.

[69] Hyung Tae Lee, San Ling, and Huaxiong Wang. Analysis of gong et al.'s cca2-secure homomorphic encryption. *Theoretical Computer Science*, 640:104–114, 2016.

[70] Oded Goldreich and Rafail Ostrovsky. Software protection and simulation on oblivious rams. *Journal of the ACM (JACM)*, 43(3):431–473, 1996.

[71] Ling Ren, Xiangyao Yu, Christopher W Fletcher, Marten Van Dijk, and Srinivas Devadas. Design space exploration and optimization of path oblivious ram in secure processors. In *Proceedings of the 40th Annual International Symposium on Computer Architecture*, pages 571–582, 2013.

[72] Andrew C Yao. Protocols for secure computations. In *23rd annual symposium on foundations of computer science (sfcs 1982)*, pages 160–164. IEEE, 1982.

[73] David Evans, Vladimir Kolesnikov, Mike Rosulek, et al. A pragmatic introduction to secure multi-party computation. *Foundations and Trends® in Privacy and Security*, 2(2-3):70–246, 2018.

[74] Yehuda Lindell and Benny Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 52–78. Springer, 2007.

[75] David W Archer, Dan Bogdanov, Benny Pinkas, and Pille Pullonen. Maturity and performance of programmable secure computation. *IEEE security & privacy*, 14(5):48–56, 2016.

[76] Victor Costan and Srinivas Devadas. Intel sgx explained. *Cryptology ePrint Archive*, 2016.

[77] Masanori Misono, Dimitrios Stavrakakis, Nuno Santos, and Pramod Bhatotia. Confidential vms explained: An empirical analysis of amd sev-snp and intel tdx. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 8(3):1–42, 2024.

[78] Antonio Muñoz, Ruben Ríos, Rodrigo Román, and Javier López. A survey on the (in) security of trusted execution environments. *Computers & Security*, 129:103180, 2023.

[79] Signal Technology Foundation. Signal official website. https://www.signal.org. Accessed: 2025-06-13.

[80] Mihaela Ion, Ben Kreuter, Erhan Nergiz, Sarvar Patel, Shobhit Saxena, Karn Seth, David Shanahan, and Moti Yung. Private intersection-sum protocol with applications to attributing aggregate ad conversions. *Cryptology ePrint Archive*, 2017.

[81] Archita Agarwal, David Cash, Marilyn George, Seny Kamara, Tarik Moataz, and Jaspal Singh. Updatable Private Set Intersection from Structured Encryption, 2024. Published: Cryptology ePrint Archive, Paper 2024/1183.

[82] Emil Stefanov, Marten van Dijk, Elaine Shi, T-H Hubert Chan, Christopher Fletcher, Ling Ren, Xiangyao Yu, and Srinivas Devadas. Path oram: an extremely simple oblivious ram protocol. *Journal of the ACM (JACM)*, 65(4):1–26, 2018.

[83] Saikrishna Badrinarayanan, Peihan Miao, and Tiancheng Xie. Updatable private set intersection. *Proceedings on Privacy Enhancing Technologies*, 2022(2), 2022.

[84] Florian Kerschbaum, Erik-Oliver Blass, and Rasoul Akhavan Mahdavi. Faster secure comparisons with offline phase for efficient private set intersection. *arXiv preprint arXiv:2209.13913*, 2022.

[85] Hao Chen, Kim Laine, and Peter Rindal. Fast private set intersection from homomorphic encryption. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1243–1255, 2017.

[86] Haochen Dou, Zhenwu Dan, Peng Xu, Wei Wang, Shuning Xu, Tianyang Chen, and Hai Jin. Dynamic Searchable Symmetric Encryption With Strong Security and Robustness. *IEEE Transactions on Information Forensics and Security*, 19:2370–2384, 2024.