Leveraging Searchable Encryption through Homomorphic Encryption: A Comprehensive Analysis

Ivone Amorim * and Ivan Costa

Abstract: The widespread adoption of cloud infrastructures has revolutionized data storage and access. However, it has also raised concerns regarding the privacy of sensitive data. To address these concerns, encryption techniques have been widely used. However, traditional encryption schemes limit the efficient search and retrieval of encrypted data. To tackle this challenge, innovative approaches have emerged, such as the utilization of Homomorphic Encryption (HE) in Searchable Encryption (SE) schemes. This paper provides a comprehensive analysis of the advancements in HE-based privacy-preserving techniques, focusing on their application in SE. The main contributions of this work include the identification and classification of existing SE schemes that utilize HE, a comprehensive analysis of the types of HE used in SE, an examination of how HE shapes the search process structure and enables additional functionalities, and the identification of promising directions for future research in HE-based SE. The findings reveal the increasing usage of HE in SE schemes, particularly Partially Homomorphic Encryption. The popularity of this type of HE schemes, especially Paillier’s cryptosystem, can be attributed to its simplicity, proven security properties, and widespread availability in open-source libraries. The analysis also highlights the prevalence of index-based SE schemes using HE, the support for ranked search and multi-keyword queries, and the need for further exploration in functionalities such as verifiability and the ability to authorize and revoke users. Future research directions include exploring the usage of other encryption schemes alongside HE, addressing omissions in functionalities like fuzzy keyword search, and leveraging recent advancements in Fully Homomorphic Encryption schemes.

Keywords: searchable encryption; homomorphic encryption; secure search; data privacy; keyword search

MSC: 94A60; 68P25

1. Introduction

The rapid growth and widespread adoption of cloud infrastructures have revolutionized the way we store and access information. From personal data backup and file-sharing services such as Dropbox and Google Drive to enterprise-level data management and scalable infrastructure solutions, cloud storage has had a profound impact on our day-to-day lives, particularly in sectors such as healthcare [1,2] and education [3,4]. According to a report published by Netwrix [5], 80% of organizations store sensitive data in the cloud. The advantages it offers include high data availability, convenient access from anywhere, reduced infrastructure costs, unlimited storage space, and cost-effectiveness [6]. However, with cloud adoption come heightened security concerns as sensitive information entrusted to cloud servers faces potential vulnerabilities, such as unauthorized access, data breaches, and insider threats. Netwrix’s report [5] also found that 55% of healthcare organizations had experienced a data breach involving third-party entities in the last year, which was the second-highest percentage of all industry sectors, narrowly surpassed by the financial sector, where 58% of companies experienced similar breaches. These two sectors heavily
rely on third-party entities that will have access to their sensitive data, which is highly attractive to cybercriminals. Moreover, their results show that attacks have become more sophisticated and harder to spot.

To address these issues, ensuring the privacy of users’ sensitive data becomes crucial. The most common way to achieve privacy is through encryption, where data is encrypted before being stored in the cloud, providing end-to-end data privacy. However, traditional encryption schemes pose challenges when it comes to efficiently searching and retrieving data stored in encrypted form, limiting the usability of encrypted storage solutions. Naive approaches, such as downloading all ciphertexts, decrypting them, and searching on plaintexts, are impractical. Therefore, to tackle these challenges, innovative concepts such as secure search, private information retrieval (PIR), and searchable encryption (SE) have emerged. Secure search addresses the need to perform search operations while maintaining the privacy and confidentiality of the data [7]. PIR, closely related to secure search, enables users to retrieve specific data from a database without revealing which items they are accessing [8]. SE, on the other hand, is a cryptographic technique that allows for secure and efficient search operations over encrypted data [9]. It is important to note that the terms “secure search” and “searchable encryption” are sometimes used interchangeably, blurring the distinction between the two concepts. In practice, the boundaries between these terms can be fluid, with different researchers and practitioners using them to refer to related techniques and mechanisms. Additionally, PIR is often considered a component within secure search or SE approaches, as it enables private data retrieval from a database. In this work, we mainly use the term “searchable encryption” whereas “secure search” is used whenever we feel a distinction is needed.

SE can be achieved using several different cryptographic techniques as well as a combination of these techniques. Two of the most popular types of SE schemes are searchable symmetric encryption (SSE) and public key encryption with keyword search (PEKS), which, as the names indicate, rely on a secure symmetric encryption scheme or a secure asymmetric encryption scheme, respectively. Other techniques commonly used are attribute-based encryption (ABE), in order to perform fine-grained access control over a SE scheme, and order-preserving encryption (OPE), in order to efficiently allow for ranking of search results. There is, however, one notable approach within the realm of SE, which is the utilization of homomorphic encryption (HE), which is a cryptographic technique that allows computations to be performed directly on encrypted data [10]. This property makes it well-suited for privacy-preserving techniques, enabling search operations directly on encrypted data while maintaining confidentiality. Researchers have increasingly focused on this approach, as it offers an attractive framework for secure search without requiring costly setup procedures [11].

Numerous approaches have been proposed that leverage homomorphic properties to provide privacy-preserving technologies, particularly in the context of SE. However, there is a lack of dedicated studies analysing the advancements in this area. On the other hand, it is crucial for the scientific community to gain a comprehensive understanding of the state-of-the-art and identify promising directions for further exploration in this subject.

In this work, we aim to provide a thorough analysis of the application of HE to SE.

1.1. Related Work

Since the publication of the first work on SE, in 2020, due to Song et al. [12], several surveys and review papers have been proposed on this topic, which highlights the growing importance of secure data management in a world where the majority of data is stored in third-party cloud services.

Bösch et al. [13] presented the first survey on SE. In their work, a comprehensive discussion is given regarding the number of writers and readers supported by each scheme. The authors consider four different cases: single-writer/single-reader, multi-writer/single-reader, single-writer/multi-reader, and multi-writer/multi-reader. The schemes are organized based on their query expressiveness, and an analysis is conducted, comparing
efficiency and security aspects. Wang et al. \[14\] and Han et al. \[15\] published two new systematic surveys on SE. The former gives a broader perspective on the existing SE schemes, considering their application in both symmetric and asymmetric key environments, whereas the latter categorizes the systems according to three aspects: security requirements, search functionalities, and deployment model. Moreover, as stated by the authors, the latter also introduces a new deployment model that was not covered in the work of Bösch et al., called the Server–User model, in which the cloud server is the owner of the data and acts as both data owner and storage server.

In later years, other works have been published that provided reviews on SE methods \[16–19\].

More recently, in 2022, Andola et al. \[20\] published a comprehensive review paper that focuses on analysing the features and limitations of these techniques based on their performance and robustness against various types of attacks. One of the key aspects of this work is the in-depth analysis of each technique and the cryptographic basis that determines their efficiency. In the same year, Noorallahzade et al. \[21\] published a complete classification of SE schemes based on a comprehensive set of metrics, including search type, index type, results type, security models, type of implementation, the multiplicity of users, cryptographic primitives, and the technique used. For each category, the available schemes were compared and evaluated. Sharma \[9\] provided a comprehensive guide on SE for non-security experts. The main goal of this work was to help general practitioners to select the most suitable SE scheme for their specific needs by presenting a survey that details the existing schemes based on five key characteristics: key structure, search structure, search functionality, support for readers/writers, and reader capability. Both symmetric and asymmetric SE schemes are discussed. It also presents a comparative analysis, which may assist non-security experts in making informed decisions about their encryption needs.

Over the years, other works have been published that survey the application of SE in different contexts. For example, Zhang et al. \[22\] discussed the use of SE in healthcare systems, whereas Bader and Michala \[23\] focused on its application in the industrial Internet of Things (IoT). Other works have also analyzed how new technologies such as blockchain are being used to enhance the potential of SE mechanisms \[24,25\]. However, no study is devoted to the use of HE in secure search mechanisms. Nevertheless, HE is highly valuable in SE schemes due to its unique capabilities, such as allowing for computations to be performed directly on encrypted data, and enabling privacy-preserving search operations. With HE, users can encrypt their queries and evaluate them on encrypted data, ensuring the confidentiality of both the query and the data. Therefore, leveraging the recent advancements in HE can enhance SE schemes, making them more suitable for real-world applications. In this context, our main objective is to survey SE techniques that incorporate HE into their design, while also providing a comprehensive analysis of how HE is used, the benefits it brings, and identify potential research directions that should be explored within this domain.

1.2. Main Contributions

The main contributions of our work are:

1. To identify and classify existing SE utilizing HE. Our analysis covers several aspects of these systems, including the encryption techniques used, the structure of the search process (sequential scan or index-based), and the search capabilities offered (such as the ability to handle multiple keywords, regular expressions, wildcards, phrases, ranges, occurrences, and fuzzy keywords). Additionally, we study other functionalities like authorization and access revocation, static and dynamic approaches to SE, and correctness verification.

2. To conduct an extensive analysis of the most common types of HE used in SE schemes, focusing our analysis on the identification of whether the schemes employed fall under the categories of partially homomorphic encryption, somewhat homomorphic
encryption, or fully homomorphic encryption. Furthermore, we explicitly identify the HE schemes whenever possible.

3. To examine how HE is used to achieve the different properties and characteristics within SE, building upon the categorization mentioned earlier. Specifically, we investigate how HE shapes the search process structure, enhances search capabilities, and enables additional functionalities in SE.

4. To identify promising directions for future research and development in HE-based SE schemes, aiming to deliver more flexible and advanced solutions for SE.

1.3. Research Methodology

To perform a comprehensive analysis of the application of HE in SE, a systematic approach was adopted. This approach involved searching several academic databases, including ACM Digital Library, IEEE Xplore, Elsevier ScienceDirect, Scopus, and Web of Science.

To conduct the search in the previously mentioned databases, a list of keywords was identified after thoroughly reviewing the main literature on this area. The listed databases were searched for works that included at least one keyword related to SE and one related to HE in their title, abstract, or set of keywords. Keywords related to SE included “searchable”, “secure search”, and “keyword search”. Keywords related to HE included “homomorphic”.

As such, the main research query used was:

(“searchable” OR “secure search” OR “keyword search”) AND homomorphic.

The search was conducted in February 2023 and Table 1 presents the number of research publications obtained in each searched database. As it can be seen, a total of 645 results were returned.

Table 1. Number of search results obtained in each academic database.

<table>
<thead>
<tr>
<th>Database</th>
<th>N° of Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACM Digital Library</td>
<td>186</td>
</tr>
<tr>
<td>IEEE Xplore</td>
<td>68</td>
</tr>
<tr>
<td>Elsevier ScienceDirect</td>
<td>41</td>
</tr>
<tr>
<td>Scopus</td>
<td>217</td>
</tr>
<tr>
<td>Web of Science</td>
<td>133</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>645</strong></td>
</tr>
</tbody>
</table>

After conducting the search, we were able to identify duplicate results which were then eliminated. This process resulted in a total of 290 distinct research papers. To select the publications relevant for our study, it was necessary to perform another screening process. Therefore, a set of criteria for inclusion and exclusion were established and applied. These criteria are listed in Table 2.

Table 2. Inclusion and exclusion criteria.

<table>
<thead>
<tr>
<th>Type of Criterion</th>
<th>Criterion ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inclusion</td>
<td>IC1</td>
<td>It focuses on secure search methods or SE schemes that leverage the properties of HE in the search process.</td>
</tr>
<tr>
<td></td>
<td>IC2</td>
<td>It was published in peer-reviewed journals or conferences.</td>
</tr>
<tr>
<td></td>
<td>IC3</td>
<td>It was written in English.</td>
</tr>
<tr>
<td></td>
<td>IC4</td>
<td>It was published after 2016.</td>
</tr>
<tr>
<td>Exclusion</td>
<td>EC1</td>
<td>It does not provide sufficient information on how HE is applied.</td>
</tr>
<tr>
<td></td>
<td>EC2</td>
<td>It does not apply HE to improve the search method.</td>
</tr>
<tr>
<td></td>
<td>EC3</td>
<td>It was written in other languages than English.</td>
</tr>
<tr>
<td></td>
<td>EC4</td>
<td>It is not a peer-reviewed publication.</td>
</tr>
</tbody>
</table>
Finally, after applying the inclusion and exclusion criteria, a total of 23 papers, out of 290, were determined to comply with the inclusion and exclusion criteria. Therefore, these 23 papers were selected for our analysis.

1.4. Organization

This document is organized as follows: Section 2 introduces the main actors and processes involved in SE schemes, as well as how they can be characterized in terms of search structure, search functionalities, multiplicity of users, and other functionalities. Understanding the key concepts presented in this section is fundamental for comprehending the subsequent discussion on SE schemes that use HE. Section 3 focuses on HE, introducing its definition, main existing types, and approaches. HE plays a crucial role in this work, and understanding the basic related concepts and properties is essential to grasp its impact on enhancing secure search mechanisms. In Section 4, a comprehensive analysis of the selected works is presented, describing and categorizing each studied work. The discussion of the research trends is covered in Section 5, which is important to understand the latest advancements and emerging directions in this area. It provides valuable insights into ongoing research efforts, potential challenges, and future opportunities. Finally, in Section 6, we summarize the key findings, highlight the main contributions, and offer insights into the implications of the research. This section is important for drawing conclusions, summarizing the significance of the research, and inspiring further investigation.

2. Searchable Encryption

The term SE refers to a cryptographic technique that enables searching through encrypted data without first having to decrypt it. SE is particularly useful and important in scenarios where cloud services are used to store large amounts of sensitive data. In such cases, privacy-preserving techniques are necessary to search and retrieve stored sensitive data. Therefore, our focus in this work is on cloud-based SE systems.

SE techniques can be broadly categorized into two main types based on the underlying encryption process: symmetric and asymmetric. Similar to conventional symmetric encryption schemes, symmetric SE techniques utilize the same key for both encryption and decryption. On the other hand, asymmetric SE schemes employ two keys—one for encryption and another for decryption.

Figure 1 depicts a high-level architecture of a generic cloud-based SE system which typically consists of the following three main entities:

- **Data owner (DO)** The entity in charge of data encryption and its outsourcing to the cloud server is the data owner, also known as the “client” in relevant literature [9]. Generally, the data is produced by the DO who has legitimate control and ownership over it. However, in some scenarios, another entity, known as the data provider, may be responsible for generating the data. The SE methods that use indices to aid in the search process, also known as index-based SE schemes, typically require the data provider (whether it is the data owner or not) to encrypt the index and share it with the cloud.

- **Data user (DU)** An entity that wants to search through the encrypted data kept by the DO is referred to as a data user. Ideally, it can only conduct the search if the DO has already given permission. The DU is, therefore, responsible for sending a search request to the cloud server which processes the request and then retrieves the results. It is worth noting that, in some systems, the DO can also act as a DU.

- **Cloud server (CS)** The cloud server is the entity responsible for securely storing the encrypted data and providing a SE service to authorized DUs. It performs three main tasks after receiving the DO’s encrypted documents: storing the data, searching the data, and keeping the search data structures up to date. The way the CS performs the search depends on the SE scheme being used. For an index-based SE, the CS searches by comparing the search request with the encrypted index and then sends the findings to the authorized DU. However, for SE schemes that are not index-based, the process
of searching the encrypted data usually requires scanning the whole document, as will be discussed later in this section.

Figure 1. A conceptual overview of a cloud-based SE system.

The architecture of a SE system usually includes four processes involving the previously described entities. The specific algorithms used in each step may differ based on the specific design and requirements of the SE scheme.

The following description of each process has been made as broad as possible, taking into account various factors such as whether an index-based search mechanism is present or not, and whether it is a symmetric or asymmetric SE scheme.

1. Setup() The algorithm generates various system parameters \( P \) and the required keys \( K \) based on an input security parameter \( \lambda \). A public key and a private key must be generated for asymmetric SE, whereas only one key is required for symmetric SE. Typically, the system owner runs this algorithm.

2. Encryption() An encryption algorithm encrypts the data using the key(s) generated in the previous process \( K \), and it outputs a message \( M' \) obtained by encrypting the original data \( M \) using an encryption algorithm \( E \).

   If the SE scheme is index-based, the input set of keywords \( W \) will also be encrypted using the input key(s) \( K \), then it may be used to generate an index \( I \) of encrypted keywords. The DO then applies this algorithm to associate the encrypted index \( I \) with the encrypted message \( M' \) in order to create a searchable ciphertext \( (SC) \). The SC can then be uploaded to the CS. If the SE method is scan-based (meaning that the server will scan the encrypted data directly), the previous step can be skipped, and the message \( M' \) can be stored directly on the CS without generating an SC.

3. TokenGen() The creation of search queries is done by authorized users using this algorithm, also known as Trapdoor. It takes an input encryption key \( K \) and an input query \( Q = k_1, k_2, \ldots, k_m \) and generates a search token \( T_Q \). The specific implementation of the algorithm depends on whether the SE scheme is index-based or scan-based.

   For an index-based scheme, the algorithm encrypts each keyword in the query \( Q \) using the encryption key(s) \( K \) and generates an index \( I = w'_1, w'_2, \ldots, w'_m \) of encrypted keywords. The search token \( T_Q \) is then constructed from \( I \). For a scan-based scheme, the algorithm generates a scan token \( T_S \). Using the chosen search query \( Q \) and the scan token \( T_S \), the search token \( T_Q \) is then created. Once constructed, the search token \( T_Q \) is sent to the server, which uses it to search for relevant data in the encrypted database. The DU might be the only one who can perform the query, depending on the scenario.

4. Search() The search algorithm is used by the CS to search the encrypted data for matches to a search query. In an index-based scheme, the search algorithm applies the search token \( T_Q \) onto the searchable ciphertext \( SC \) to identify the set of encrypted keywords, and corresponding indices, that match the query. Then, it retrieves the corresponding encrypted data to the DU. In a scan-based scheme, the search algorithm applies the search token \( T_Q \) and the scan token \( T_S \) onto the encrypted data to identify the set of searchable data segments with keywords that match the search query’s keywords. The server sends to the DU the search results once it has found any matches.
2.1. Characterization of a SE Scheme

SE schemes can be categorized in various ways, and there are several categorizations of SE schemes in the literature. For example, Han et al. [15] categorized SE systems based on three aspects: security requirements, search functionalities, and deployment model. Noorallahzade et al. [21] presented a more comprehensive categorization by considering additional aspects such as search type, index type, results type, security models, type of implementation, multiplicity of users, and cryptographic primitives. Sharma [9] detailed existing SE schemes based on five key characteristics: key structure, search structure, search functionality, support for readers/writers, and reader capability.

In our work, we will consider a categorization of SE schemes that includes four categories: search structure, multiplicity of users, search functionalities, and other functionalities. By using this categorization, we then analyze each scheme and identify which category it belongs to, allowing us to make a proper analysis of the features that are most common in SE schemes that use HE.

The categorization we will use is represented in Figure 2, and each category is described in detail in the following sections for clarification.

Figure 2. Key characteristics of a cloud-based SE scheme.

2.1.1. Search Structure

An important aspect of a SE scheme is the search structure used to perform searches over the encrypted data, since it may significantly impact the scheme’s efficiency. Historically, the first scheme, due to Song et al. [12], used what is called a sequential scan search
structure. This search operation involves sequentially scanning through all the encrypted documents in the database to find those that match the search query. Although other works in the sequential scan category have been published, such as the one by Boneh et al. [26] on public encryption with keyword search, there are not many prominent research works that use this approach [18]. This is due to the fact that these techniques are very inefficient and are not well-suited for large databases or frequent search queries, as the search operation can become very time-consuming and computationally expensive. Moreover, this search method is prone to exposing sensitive information to the server, which compromises the privacy and security of the data [20]. A potential advantage of sequential scan-based SE schemes is that they can provide a simple and straightforward way to search over encrypted data without the need for complex indexing or search structures.

Another type of search structure commonly used in SE to improve search performance is the index-based search structure. In this approach, a special data structure, called Index, is associated with the encrypted documents. Instead of searching the contents of every document in the database, this new data structure enables comparison of the search query with the entries in the index. Additionally, using indexes enables working with files in various formats, including multimedia files, as well as compressed and encrypted files [19]. Nevertheless, with this method, the encrypted data and encrypted index must be sent by the DO to the CS. These two sets of information can be encrypted with the same encryption scheme, or we can use a SE scheme to encrypt the index and another reliable encryption scheme to encrypt the sensitive data. There are mainly three types of keywords’ indexes: simple index, inverted index, and tree index.

- **Simple index.** In systems that employ this strategy, each document is given an index before being encrypted and uploaded to the CS. The index is composed of words that are thought to be relevant to that document. This kind of index is appropriate for applications where it is necessary to upload a small number of documents to the CS [9].
- **Inverted index.** The term inverted-index comes from the process of building the index backwards. This is because, instead of associating each document with a set of keywords, the index is created by coupling each keyword with the set of documents where it appears. This approach significantly reduces the time required for searching, making it the most suitable search structure for applications that involve uploading a large number of documents to the CS [9].
- **Tree index.** A tree index is also a very efficient method to optimize the search process. Although there are various approaches to building a tree index, the basic idea is to create a tree-like structure containing the searchable keywords, by dividing the set of keywords into smaller sets. When a DU searches for a specific keyword, the CS will search the index-tree, starting at the root and traversing every relevant node until a match is found.

Compared to sequential scan approaches, the schemes that are index-based are highly efficient, reducing the number of comparisons made per file from $O(n)$ to $O(1)$. However, a drawback is the fact that the keywords that can be used in the query are limited to those that were extracted during index generation [19].

### 2.1.2. Multiplicity of Users

SE schemes allow DOs to securely outsource data storage and DUs to retrieve data based on specific search criteria. Based on the multiplicity of users involved, we further classify these schemes as **single-user**, if the DO assumes the role of DU, and **multi-user** if the DU is different from the DO.

### 2.1.3. Search Functionalities

The search functionalities of a SE scheme, as the name suggests, determine the kind of queries which can be performed, the filtering and sorting options and other tools that help data users to refine their search. In this work, and following a categorization similar
to Sharma [9], we distinguish between search functionalities directly associated with the number of keywords and others. The set of functionalities not directly linked with the number of keywords is referred to as “Miscellaneous” (Figure 2).

Regarding the number of keywords, a SE scheme can be classified as either single or multi-keyword. In the former, the DU is allowed to include just one search term in each query. Consequently, if he/she wants to search for multiple terms using this type of scheme, then he/she needs to perform multiple queries, one for each search term [19]. On the other hand, in a multi-keyword SE scheme, the user is allowed to perform a search with more than one search term in each query. However, it is important to mention that just because multiple keywords are supported, it does not necessarily imply that users can perform searches with an unlimited number of keywords. Some schemes may be restricted to a specific number of keywords.

Furthermore, in a multi-keyword SE scheme, a query can be represented as a simple list of search terms, as a conjunctive query (using the AND relation), or as a disjunctive query (using the OR relation). The multi-keyword SE may fall under the conjunctive keyword or disjunctive keyword categories, depending on how the query is expressed.

The existing SE schemes can be divided into groups based on various functionalities in addition to how many keywords they contain. In this work, we take into account the following: wildcards and regular expressions, fuzzy keyword search, phrase search, range search, occurrence search, and ranked search.

Regular Expressions and Wildcards

SE schemes that allow regular expression or wildcard based search queries are very useful when the DUs know the specific patterns of the keywords that they want to search for in the documents. In both cases, the query can be constructed using special characters to describe the keyword(s) pattern.

If a SE scheme allows wildcard queries, then the DUs can replace a single character in the search query with a symbol known as a wildcard. Generally, two types of wildcards are used: “?” and “*”, the latter being referred to as a multi-character wildcard and representing any number of characters, whereas the former only represents one character [27]. In case a regular expression-based SE scheme is used, a combination of special characters and operators can be employed to describe more complex patterns in search queries than those allowed by wildcards. Although regular expressions provide greater flexibility, wildcard-based SE schemes are a simpler alternative when searching for keywords with minor variations.

Fuzzy Keyword Search

A typical SE scheme, even when it allows wildcard or regular expression-based search queries, can only search for exact matches of keywords in ciphertexts. It excludes any typos or inconsistencies in the search term(s)’s format. But by including the fuzzy keyword search functionality in the SE scheme, this limitation can be overcome. With this feature, the scheme is able to search for results related to the correct spell word even if the DUs have misspelled it during their search [19].

Phrase Search

Often, instead of just looking for individual keywords, DUs prefer to find specific phrases. Even though this can be achieved using SE schemes that allow for conjunctive search queries, this requires the DUs to perform several additional steps. First, they need to convert the phrase into a conjunctive query and perform the search. Then, after obtaining the corresponding documents from the CS, they have to decrypt and screen them to find the ones that contain the phrase they are looking for [28]. As a consequence, this approach can be very inefficient when dealing with large databases. SE schemes with phrase search functionality, on the other hand, enable data users to directly search for phrases [9].
Range Search

SE schemes allowing range queries enable DUs to search for encrypted data within a specific range of values, meaning that they can perform searches based on an interval rather than just on exact matches. For instance, a data user may want to search for all documents in a database that were created between 2020 and 2022 or search for documents whose owners have more than 30 years old. In fact, the last example mentioned belongs to a specific subcategory of range queries called comparison queries [29]. This kind of search query can be very useful to avoid multiple single-keyword searches.

Occurrence Search

SE schemes with this property allow a data user to issue a query to the cloud server that includes both a search term and a value. This value specifies the minimum number of times that the chosen keyword must appear in a document for it to be retrieved in the search results. This feature can be very helpful to measure the importance of our keyword within each document in the database [19].

Ranked Search

This feature’s main purpose is to enhance the data user’s search experience by retrieving only relevant documents that match the query. To achieve this, various scoring functions have been adopted from the information retrieval research community [19]. The most frequently used scoring function is known as TF-IDF, where TF stands for term frequency, indicating the significance of that keyword in the document, and IDF stands for inverted document frequency, indicating the significance of the keyword over all documents in the database [18].

2.1.4. Other Functionalities

There are other functionalities that SE schemes may have and are important for some scenarios. For example, regarding access control functionalities, the ability to authorize and revoke DUs is crucial.

Authorize and Revoke Users

As the name indicates, this feature in a SE scheme means that the DO can grant new users search capability as well as revoke this ability. There are not many SE schemes with this functionality, and the ones that exist are inefficient and impractical. This is because enabling this functionality requires compromising security [20].

Static or Dynamic

Another important functionality in a SE scheme for real-world applications is the ability to allow dynamic updates. When no document is added, removed, or updated after creating the encrypted database, a SE scheme is considered static. Therefore, the entire set of data has to be available before data encryption. In contrast, a SE system is said to be dynamic if users can add, remove, or modify documents without endangering the encrypted data’s security or the system’s ability to search for and retrieve them. Dynamic SE schemes are more flexible and more suitable for practical uses. However, compared to dynamic SE schemes, static SE schemes are typically more effective and simpler to implement, making them a better option for simple scenarios where the data is relatively stable, and no frequent updates are needed. It is important to notice that, in a dynamic and index-based SE scheme, the index must be updated efficiently when a change is made to the document collection, and this has to be done without adding any leakage [19].

Verifiability

In most applications, the CS in a SE system is considered semi-trusted, meaning it may retrieve incomplete or incorrect results to the DU [19]. Therefore, it is important for a SE scheme to allow the DU to validate the search results that the CS has retrieved. Such
an SE scheme is designated as a verifiable SE scheme. As would be expected, this kind of scheme would suffer from higher computational overhead on all sites (at DO, at DU, and at the CS) when compared to a SE scheme not allowing verifiability [9]. Result verification may include correctness (if the results retrieved correspond to the query), completeness (if all pertinent documents are returned), and freshness (if the most recent version of the documents is retrieved to the DU) [19].

Delegate

This functionality allows an authorized user to delegate the search capability to another user. This functionality mainly appears in asymmetric SE schemes, also known as public key encryption with keyword search. These SE schemes include proxy re-encryption, that is, they have a semi-trusted third-party server (proxy server) to transform ciphertexts created with the DO’s public key into ciphertexts that the delegated user can decrypt [9]. Although this is not a very common type of SE, there are already several works addressing this feature [21].

3. Homomorphic Encryption

The main idea behind a HE scheme is to allow computations to be performed over encrypted data without the need to decrypt it first. This concept was introduced by Rivest et al. [30] and was originally called “privacy homomorphism”. A HE scheme is, therefore, any encryption scheme where the encryption function is a homomorphism. Formally speaking, let \( M \) denote the set of plaintexts and \( C \) the set of ciphertexts. Let \( \odot_M \) and \( \odot_C \) be operations in \( M \) and \( C \), respectively. An encryption scheme is said to be homomorphic if for any encryption key \( k \), the encryption function \( E \) satisfies the property below,

\[
\forall m_1, m_2 \in M, E(m_1 \odot_M m_2) \leftarrow E(m_1) \odot_C E(m_2),
\]

where \( \leftarrow \) means “can be directly obtained from” [31]. That is, in (1), \( E(m_1 \odot_M m_2) \) can be directly obtained from \( E(m_1) \odot_C E(m_2) \) without having to perform any decryption.

Depending on the type and number of operations that the system permits, HE schemes can be grouped into three categories: partially homomorphic encryption (PHE), somewhat homomorphic encryption (SWHE), and fully homomorphic encryption (FHE).

3.1. Partially Homomorphic Encryption

After the publication of Rivest et al.’s work, cryptography researchers began searching for HE schemes allowing to execute more than one operation directly on encrypted data. Nonetheless, over the next two decades, most attempts resulted in schemes that only allowed one type of operation, which are the ones known as PHE schemes. In these schemes, the homomorphic property is satisfied by just one operation, such as addition or multiplication, without imposing any limitations on the number of times that operation can be executed.

The well-known RSA public key cryptosystem [32] is a PHE scheme that only supports the usual product operation and is deterministic, meaning that when one uses the same key to encrypt the same plaintext, one will always get the same ciphertext.

Some years later, in 1982, Goldwasser and Micali [33] published the first probabilistic PHE scheme which inspired most of the PHE schemes published in the following decades, such as Benaloh [34], which was a generalization of Goldwasser et al.’s scheme, and Naccache and Stern [35] which was an improvement on Benaloh’s scheme.

Elgamal [36] introduced a PHE which also allows only the usual product, and Paillier [37] published a PHE which allows only the usual sum. This latter scheme is the most frequently used in SE schemes that use HE, as will be seen in Section 5.

3.2. Somewhat Homomorphic Encryption

In 2005, Boneh et al. [38] published the first HE scheme able to perform two operations: addition and multiplication. However, although additions can be executed an unlimited
number of times, it only allows one multiplication. This is the kind of HE schemes that are called SWHE, since the homomorphic property is satisfied by more than an operation but a limited number of times.

Shortly after the introduction of the initial HE scheme, several proposals for SWHE schemes emerged. Although they were not as appealing as PHE schemes due to their limitations, the extensive research focused on PHE led to the development of more complete schemes that served as stepping stones towards achieving an FHE scheme. In fact, many researchers consider the scheme presented by Boneh et al. [38] as a significant building block in the progression towards an FHE scheme.

3.3. Fully Homomorphic Encryption

In this type of HE schemes, the homomorphic property holds for different operations, and they can be performed an arbitrary number of times.

This category is considered by many as the “Holy Grail” of Cryptography, since it allows computations to be performed freely over encrypted data. It was in 2009, that Gentry published the first FHE scheme [39]. In his work, Gentry also proposed a method for constructing a general FHE scheme from one that is not fully homomorphic but has a certain capability for homomorphic evaluation [40]. Since then, HE has triggered considerable interest, leading to the proposal of novel FHE schemes based on Gentry’s idea. Among the most notable are FV [41], BGV [42], CKKS [43], and TFHE [44].

A common security concern in HE schemes is the accumulation of encryptions of 0. If a malicious actor is able to gather a significant number of encryptions of 0, it can potentially jeopardize the security of the scheme. To address this vulnerability, many schemes employ a technique referred to as “adding noise” to disguise encryptions of zero.

Nevertheless, as the number of computations increases, the noise in these schemes tends to grow, potentially leading to an incorrect decryption. To address this issue, Gentry introduced the concept of bootstrapping. However, this approach comes with a significant computational cost, and the complex mathematical foundations of Gentry’s scheme make it not practical for real-life applications.

According to Acar et al. [10], there are four main categories of FHE schemes that have been developed since Gentry’s work in 2009:

- **Ideal Lattices**—In this category there are essentially optimizations of Gentry’s scheme. Examples of this category include the work of Scholl and Smart [45] and the work of Gentry and Haveli [46].
- **Integers**—The first FHE based in integers appeared in 2010 [47]. The main motivation behind these schemes lies in their conceptual simplicity. However, their lack of practicality makes them the least preferred category among researchers.
- **(Rings) Learning With Error, (R)LWE**—Brakerski and Vaikuntanathan [48] were the first to propose a scheme in this category. These schemes are based on the LWE problem, which is one of the most challenging problems to solve in real-time even for post-quantum algorithms. LWE has an algebraic variant called RLWE, which is more useful in practical applications;
- **Nth degree-truncated polynomial ring unit**—These schemes allow computations between data that has been encrypted using various keys. López-Alt et al. [49] were the first to propose an HE scheme of this kind.

Due to its huge potential for cryptographic applications, HE schemes have already been proposed to solve a wide range of problems in several areas, including in big data and cloud computing, secure image processing, medical applications, electronic voting systems, private information retrieval, and biometric verifications, as mentioned by Challa [50] and Alloghani [51].

4. Analysis of SE Schemes That Utilize HE

In this section, we review and analyze the 23 research works which resulted from the selection process presented in Section 1. We only considered the proposed schemes in
which it is clear how HE is used. The goal of this analysis is to study their functionalities, taking in consideration the characteristics presented in Figure 2, and identify which of those characteristics leverage the utilization of HE and the type of HE which is used. Table 3 list the selected works and gives an overview of their functionalities. It also identifies the characteristics that are achieved using HE. This table serves as a complete resource for identifying and accessing the works that were analyzed in this study, ensuring transparency and facilitating future references.

Table 3. Categorization of selected SE schemes that utilize HE.

<table>
<thead>
<tr>
<th>Article</th>
<th>Search Structure</th>
<th>Search Functionalities</th>
<th>Other Functionalities</th>
<th>User</th>
</tr>
</thead>
<tbody>
<tr>
<td>[52]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[53]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[54]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[55]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[56]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[57]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[58]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[59]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[60]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[61]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[62]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[63]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[64]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[65]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[66]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[67]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[68]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[69]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[70]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[71]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[72]</td>
<td>x x</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

x: indicates that a scheme has a certain property; h: indicates that a scheme uses HE to achieve a certain property.

Our analysis is divided into the following categories: search structure, search functionalities, and other functionalities. We have not devoted a section to the category “Multiplicity of users” because this functionality is mentioned whenever we analyze one of the selected works. It is important to notice that it is out of the scope of this work to provide a detailed explanation of the cryptographic constructions, which can be found in the referenced works.

4.1. Search Structure

The search structure of an SE scheme can involve either a sequential scan of the entire database or an index-based approach, where an index is created to facilitate the search process, as mentioned in Section 2. In this analysis, we will focus on the sequential scan-based SE schemes, as all the others are index-based, and they will be discussed in the next sections. Additionally, it is important to recall that the concept of secure search is similar to that of SE, and consequently, both approaches will be included in this analysis. Notice that secure search methods typically involve verifying all the documents in the database to find the ones that match the query, which is similar to a sequential scan process.

The work of Akavia et al. [7] is the first of the selected papers to introduce a method to perform secure search on FHE encrypted data. The authors claim that this is achieved through the utilization of a polynomial with a degree that is logarithmic (as opposed to linear) in relation to the number of entries within the data array.

The core of their secure search protocol is the computation of a sketch of the first query match, which is done using an approach they call SPiRiT. This method returns the first piece of information that matches a lookup value in an encrypted array of data. Any standard semantically secure FHE can be used in their suggested scheme, provided that the plaintext modulus parameter is a prime number \( p \), in which case the homomorphic operations are
addition and multiplication modulo $p$. They claim to have a single communication round and that the communication overhead only grows with input and output sizes.

In 2020, Wen et al. [66] proposed a searching method to be used in a secure search scheme, meant for a single-user setting, named LEAF. This approach uses three new methods: localization, extraction, and reconstruction. The localization technique is used to divide the original database array into smaller intervals of equal length and to identify the first one that contains a matching item. The encrypted interval indexes containing the matched item are returned. Extraction is then used to find the interval that contains the first matched item for subsequent search operations on that interval, and Reconstruction is used to combine the information from these two techniques in order to properly locate our desired data, without needing to decrypt anything. It is worth mentioning that these authors claim that secure search consists of roughly two steps, namely matching, and searching. In the matching step, the server compares the client’s encrypted search query to all encrypted database items, returning a second encrypted array of 0s and 1s with a 1 denoting the item in the database that matches the query. The searching step returns all 1’s indexes and corresponding items to the client. Therefore, their work focuses on the searching step. FHE is used in this scheme to encrypt both sensitive data and search queries. In fact, one of the main goals of this scheme is to optimize the use of FHE in secure search by reducing the number of necessary computations, namely the number of multiplications. Moreover, the authors also propose a variant of this scheme, named LEAF+, which uses lazy bootstrapping. On the upside, the bootstrapping step can control the algorithm’s computational depth, and the larger the database, the greater the optimization effect. On the other hand, when the size of the database is small, this variant will be ineffective, since it brings many extra multiplication operations and computation depth.

Choi et al., in 2021 [11], suggested a secure search method that uses a standard CPA-secure (leveled) FHE, meant for single use, that also performs a sequential scan over the encrypted database to perform a search. In fact, in their approach, the computational task of secure search is divided in two steps called matching and fetching, which correspond to the matching and searching phases of Wen et al.’s approach. In the matching step, the cloud server compares the encrypted search query with all encrypted records in the database using the homomorphic properties of the underlying FHE scheme to find the ones that correspond to that query. Then, in the fetching step, those corresponding records are retrieved from the database, also using the homomorphic properties, and made available to the DU, which can then decrypt them. Regarding the fetching procedure, the authors propose two novel retrieving algorithms, namely the COIE scheme (based on power sums or bloom filters) and CODE scheme (based on bloom filter sets), and compare the performance of these algorithms with other well known retrieving algorithms like LEAF+ [66] and PIR.

In 2022, Iqbal et al. [53] proposed a mechanism to securely search encrypted audio data that has been outsourced to CS in a medical context, and which also performs a sequential scan. Their approach involves using the BGV FHE scheme to encrypt the data files, which are then sent to the cloud server for storage. When a search is requested, the CS performs a sequential scan on the stored encrypted files using homomorphic operations to find the documents that contain the searched keyword. Then, the retrieved documents are decrypted using the BGV scheme. In the experiments performed by Iqbal et al., the open-source programming library HElib version 2.1.0 was used. This library allows using BGV with bootstrapping and to apply enhancements such as the ciphertext packing technique by Smart–Vercauteren and the optimization technique by Gentry–Halevi–Smart [73].

In the recent work of Malik et al. [52], published in 2023, a single keyword SE scheme that uses PHE is presented, which uses the Paillier cryptosystem [37] to protect airport data that is outsourced to cloud servers. This scheme uses HE to perform search operations and provides a high level of security by hiding search patterns using trapdoors. Two approaches were proposed, one which exploits the deterministic properties of the Paillier cryptosystem, referred to as the “efficient SKSE”, and the other which takes advantage of its probabilistic properties, named the “secure SKSE”. The former is suitable for scenarios that prioritize a
lightweight approach over security. In contrast, the latter is more appropriate for scenarios in which the security has priority over performance. Both approaches use a sequential scan search structure and are designed for a single-user scenario where the airport acts as both the DO and DU of the encrypted data.

In this scheme, original data files are encrypted twice. The first encryption is done using the advanced encryption standard (AES) [74], after which the encrypted files are uploaded to the CS. The second encryption uses the Paillier cryptosystem to encrypt the original files, and the resulting encrypted data is also uploaded to the CS. These are the encrypted files that will be used to perform the sequential scan. As an output of this scan, the system retrieves an encrypted result which, after being decrypted by the DU using Paillier’s scheme, allows recovering the identifiers of the files containing the searched keyword. AES is then employed to decrypt these files and recover the original data.

4.2. Search Functionalities

In this section, we analyze the selected works that address search functionalities. We have divided this section based on the following characteristics: regular expressions and wildcards, conjunctive search, range search, phrase search, and ranked search. We do not provide separate sections devoted to the ability to allow a multi-keyword search or disjunctive search, because all the works that possess those functionalities also possess other search functionalities and are analyzed in their corresponding sections. Moreover, the papers that have a specified characteristic can be easily identified in Table 3.

4.2.1. Regular Expressions and Wildcards

SE schemes allowing wildcard queries are sparse, and only two were found during our research. The first scheme in question was proposed by Yang et al. in 2020 [65]. In that work, they presented a novel index-based SE scheme designed for a multi-user environment. The proposed scheme allows wildcard queries as well as user authorization and revocation. Additionally, the user can issue “AND” or “OR” queries on search keywords and can also obtain the top-\(k\) documents that have the highest relevance scores.

This scheme uses a simple index approach. More specifically, a document index contains three pieces of information: the document ID, its corresponding keywords, and the key utilized in the symmetric encryption scheme used to protect the documents. The index is encrypted using a PHE scheme, namely Paillier’s cryptosystem, before being outsourced to the cloud server. On the other hand, the documents themselves are encrypted with any secure symmetric encryption scheme.

The authors cover a wide range of different algorithms to look for a match within the search protocols, regarding different types of wildcard queries. These algorithms can be split into three categories: zero wildcards (1 algorithm), one wildcard (3 algorithms), and two wildcards (4 algorithms). More specifically, for one wildcard we have the following possibilities for a query: \( * + Y_1, Y_1 + * \) and \( Y_1 + * + Y_2 \), \((Y_1, Y_2)\) are strings of any size). When two wildcards are present in the query, we have the following possibilities: \( * + Y_1 + *, Y_1 + \) \( * + Y_2 + *, Y_1 + * + Y_2 \) and \( Y_1 + * + Y_2 + * + Y_3 \), \((Y_1, Y_2, Y_3)\) are strings of any size). These algorithms take advantage of the homomorphic properties of the Paillier’s cryptosystem to encrypt the keywords in a way that comparisons are possible to identify matches. Finally, with respect to the authorization and revocation functionality, PHE is not utilized. Despite this, the system allows a DO to grant research privileges to other users for a specified period of time and to automatically revoke these privileges once the authorization period expires.

In 2021, Yin et al. [58], suggested a new SE scheme which uses FHE and allow some types of wildcard queries. The scheme was designed to achieve compound substring query on multiple attributes. A substring query, as the name suggests, is a query that allows to search for a contiguous sequence of characters within a string. For example, the substring query “cat” would return all the results which contain the substring cat, e.g., caterpillar, cats, concatenate, ducat. The proposed scheme can, in fact, support two types of substring patterns: \( * + s + * \) and \( s1 + * + s2 \), where \( s, s1 \), and \( s2 \) represent queried substrings and \( * \).
represents any string of any length. This is achieved by constructing a tree index structure using a modified version of the well-known position heap technique.

In this scheme, the sensitive data is encrypted using a symmetric key encryption scheme indistinguishable under a chosen-plaintext attack, such as AES, and the tree index is encrypted using an FHE scheme, such as BGV, and a pseudorandom function. The FHE scheme is used to encrypt the ID of each node in the tree, and the pseudorandom function is used to encrypt the concatenation of all edges labels along the path from the root to that node.

Based on the properties of the FHE scheme, the authors also designed an algorithm to calculate the intersection of search results for different searched keywords and therefore achieve the compound substring query on multiple attributes. The compound formula, in this case, consists of conjunctive and disjunctive expressions of the substrings queried and, consequently, this scheme allows conjunctive and disjunctive queries as defined in Section 2.

We have not found any literature proposing a SE scheme that enables the use of regular expressions and utilizes HE.

4.2.2. Conjunctive Search

The ability to perform a conjunctive search is very attractive, and among the selected works, six of them allow this functionality. However, only 4 of those papers make use of HE to facilitate the conjunctive search. Most of those papers possess other search functionalities and are analyzed in the corresponding sections. Interestingly, only one work provides conjunctive search and no other search functionality. This work is due to Wang et al., which, in 2022 [56], proposed an index-based SE scheme, for a multi-user setting, that supports conjunctive keyword search and uses a special PHE scheme to hide the search pattern. Their work specifically takes advantage of an auxiliary server and the additive homomorphic encryption scheme PBC [75] to effectively achieve the conjunctive keyword search property while ensuring that the DU can only learn the desired search result. In fact, the auxiliary server is introduced to allow the system to achieve the desired properties by adopting what the authors refer to as the double trapdoor decryption mechanism, which is available in this scheme.

To achieve the conjunctive keyword search property, the authors use the polynomial representation of a multiset and encrypt the polynomials using the BCP cryptosystem to maintain their confidentiality. The search is then carried out jointly by the cloud server and the auxiliary server using both additive and multiplicative homomorphic properties.

This work does not detail what kind of encryption scheme is used to encrypt the documents before uploading them to the database.

4.2.3. Phrase Search

The use of HE to enable phrase search in SE schemes is not very common. In fact, only 2 out of the 23 analyzed SE schemes are capable of performing phrase search and use HE to allow that functionality. The first one was proposed in 2019, by Shen et al., and is named P3 [70]. This scheme is meant for a multi-user setting and supports multi-keyword search, specifically phrase search, and conjunctive keyword search. The main idea of the scheme is to build an inverted index containing not only the document identifiers where each keyword appears, but also the location of each keyword in those documents.

To protect the location information, the scheme encrypts it using a PHE scheme from Boneh et al. [38], whose homomorphic properties allow the server to analyze if two encrypted keywords are adjacent. Moreover, this enables the DU to obtain precise search results from a single interaction with the cloud server. Note that phrase search is a specific type of conjunctive query where the position of queried keywords matters.

The document identifiers and keywords are protected using other techniques, specifically a pseudorandom permutation primitive and a secure kNN technique, respectively.
Subsequently, in 2022, Hou et al. [61], proposed a SE scheme that also uses the PHE scheme from Boneh et al. to protect the location of keywords and enable phrase search. However, they propose a different structure for the index, which they call a virtual binary tree. This index tree is only a logical structure used to store the keywords and related information. Its elements are stored in a hash table, if they are a leaf node, or are mapped to a bloom filter otherwise. Homomorphic properties are used similarly to the work of Shen et al. [70], allowing the scheme to check if the keywords are adjacent. Additionally, their approach allows for dynamic updates, which is an advantage when compared to the previous scheme.

4.2.4. Range Search

The ability to perform range search is another functionality which is not usual to have in SE schemes that use HE. In our research, only one work, due to Guo et al. [69], was found. This work was published in 2019, and it presents a probabilistic threshold range search scheme meant for the multi-user setting in the IoT context. This scheme addresses the problem of performing range searches over multidimensional uncertain data while allowing false positives by returning all the encrypted data that have a probability of being within the range of interest, higher than a given threshold value.

The concept of the scheme is that DOs obtain uncertain data from the IoT devices, which are modelled as multidimensional objects. As such, each IoT object is represented by an uncertain region and its probabilistic density function. A piece of data collected from such an object is called instance and comprises three components: the identification of the object, the coordinates of the instance, and its probability.

Their approach utilizes a KD-tree, which is a data structure for indexing \textit{d}-dimensional point data distributed in a \textit{d}-dimensional space. Each node in the KD-tree consists of one instance and its corresponding range, calculated based on the instance’s coordinates. The nodes are encrypted using an order-preserving encryption (OPE) scheme, whereas the instance probabilities are encrypted using Paillier’s cryptosystem.

It is noteworthy that the scheme relies on two cloud servers. Specifically, the first cloud server is responsible for: storing both the encrypted data and the encrypted KD-tree, performing KD-tree searches, and sending the results to the second server. Then, the second server is entitled to filter the results based on the threshold provided in the search query, and sending the obtained encrypted documents to the DU.

When the first cloud server receives a query, it conducts a search over the KD-tree by comparing the values of the instances’ coordinates and using the homomorphic addition of the Paillier cryptosystem to calculate the upper and lower appearance probability of each IoT object with respect to the search, so the results fall within the range queried. The results are then sent to the second server for filtering, and finally, the filtered results are sent to the DU.

4.2.5. Ranked Search

The ability to perform a ranked search is the most frequently observed feature in SE that use HE, with 10 out of the 23 analyzed works having this property. Furthermore, all of these works utilize HE to achieve that functionality.

In 2018, Elizabeth et al. [71] proposed a multi-user top-\textit{k} ranked SE (TSED) scheme that allows dynamic updates. The scheme is based on an inverted index, which uses a binary vector for each keyword to indicate whether each file contains that keyword (similar to the Z-index proposed by Wu et al. [72]).

In this scheme, sensitive data is encrypted using a secure symmetric encryption scheme, such as AES. Two PHE schemes are then used to encrypt the two components of the inverted index. Specifically, Paillier’s cryptosystem is used to encrypt each keyword, whereas Goldwasser–Micali (GM) [33] is used to encrypt each binary index vector.

TSED scheme allows for top-\textit{k} ranked searches using the TF-IDF rule. This process is done by a secure coprocessor which is responsible for computing the scores for the query
keyword using the encrypted score index, for ranking the results based on their relevance to the query and for returning the top-\textit{k} document identifiers to the CS which then retrieves the corresponding documents do the DU.

A variation of this scheme was proposed in 2020 [63], also by Elizabeth et al., which was designated verifiable top-\textit{k} ranked SE over encrypted cloud data with dynamic updates (VSED). This scheme, besides its dynamic and ranked search capabilities similar to the VSED scheme, it also has the verifiability capability.

In VSED scheme, a similar encrypted inverted index is constructed, but now the authors use a secret orthogonal vector and the Paillier cryptosystem [37]. Both the index and a trapdoor which is used when performing searches are encrypted using the PHE scheme, as well as the ranking score which is computed beforehand and, similarly to what is done in TSDE, a secure coprocessor is used to find the top-\textit{k} ranked searches based in the TF-IDF rule.

The Paillier cryptosystem is also used in the update process, which is done using two different algorithms: one to add new keywords and the other one to delete existing ones. To verify the query results received by the CS, this scheme uses the well-known mechanism for message authentication named HMAC [76]. Regarding sensitive data, this system allows to encrypt it using a secure symmetric encryption scheme such as AES, as in TSED.

In 2019, Boucenna et al. [68] proposed a SE scheme, named secure inverted index-based SE (SIIS), meant for a multi-user setting, which supports ranked search using HE. In fact, this scheme also allows user access rights management by using a CP-ABE to encrypt the data collection. As the name of the scheme indicates, it involves the construction of inverted indices. More specifically, two separate inverted indices are used. One is used to store similarity scores computed using the double score weighting formula [77]. By using the dummy documents technique [78] to allow keyword privacy, the second inverted index is created to manage the users’ access rights and lower the number of false positives. The entries in this second index are the users’ IDs, which are used to identify the collection of documents to which they have access. Both indexes are then encrypted using the FHE scheme BGV whose homomorphic properties are used to perform the search.

In 2020, Zhang et al. [62] proposed a secure ranked search scheme over encrypted data in hybrid cloud computing, meant for a single-user setting. The term “hybrid cloud computing” refers to the usage of both a public and private cloud, where the latter is mainly used to perform costly computations that would have to be performed on the client’s side otherwise. In this scheme, the private cloud has the ability to encrypt and decrypt data and most communication rounds are done between the private and public clouds.

In this proposal, the authors use the Ocapi BM25 ranking model [79], and a not specified FHE scheme is used to encrypt TF and IDF separately, on the private cloud. This information is then used to build an inverted index, which is done in the public cloud.

Regarding the encryption of sensitive data, it can be performed with any secure symmetric encryption scheme, such as AES. Finally, the proposed scheme also relies on a retrieval technique to be used by the private cloud to download the encrypted documents, although none is specified by the authors.

In the same year, Li et al. [64], proposed a two-server ranked dynamic SE scheme (TS-RDSE), meant for a single-user setting, which supports multi-keyword search. This scheme uses two cloud servers to perform the searching and sorting, with one of them being responsible for storing the encrypted data and the other one for storing the secret key. This scheme uses orthogonal vectors and two PHE schemes, namely Paillier’s and GM, in the encryption of an inverted-index, which contains TF-IDF scores that are used to perform ranked search. More specifically, the index is divided into a search index, which uses both Paillier and GM for encryption, and a weight index, which uses only Paillier.

Since the encryption of the index is based on both PHE schemes, TS-RDSE protocols for adding or deleting documents also take advantage of the homomorphic properties in order to efficiently update the index.
Regarding the encryption of sensitive data, it can be encrypted with any secure symmetric encryption scheme.

In 2020, Yang et al. [67] proposed an index-based SE scheme meant for the multi-user setting which support multiple data owners. This scheme also supports multi-keyword queries, top-k ranked results and user authorization/revocation.

The authors use a PHE scheme, more specifically the Paillier cryptosystem with threshold decryption (PCTD) to encrypt the index, which is constituted by the keywords along with their weight, documents IDs and documents keys (the weight of a keyword can be computed using TF-IDF rule, for example). Given a certain query, the authors propose a novel algorithm to compute the relevance scores regarding such query, named secure multiple keyword search protocol across domains (MKS).

The encryption of the files themselves is performed by any secure symmetric encryption scheme.

In 2021, Tosun et al. [60] proposed a multi-user SE scheme, named fully secure document similarity (FSDS) which combines secure K nearest neighbor (SK-NN), a secure algorithm that operates on data sets in Euclidean space and measures similarity using Euclidean distance, with SWHE. In this scheme, the sensitive data and the search queries are represented as TF—IDF vectors which are encrypted with a variant of SK-NN that the authors named mSk-NN. The searchable index, which is generated from the documents using TF-IDF representation, is encrypted first with the mSk-NN, and then by a SWHE scheme named FV [41]. The overall concept of the scheme is to compute the k nearest neighbors to a given query using the cosine similarity comparison metric. The authors claim that using a combination of mSK-NN and SWHE results in a more efficient and secure system, compared to an approach that only uses SWHE to encrypt the queries. This is because the amount of computations required is minimised.

Liu et al. proposed, in 2022 [54], an index-based multi-keyword SE scheme which uses FHE to support ranked search, namely, to retrieve the top-k documents most pertinent to the search query. The documents themselves are encrypted using a symmetric encryption scheme (not specified), whereas an FHE scheme is used to encrypt the indexes and the search queries. The FHE used was developed by the same authors and is named fully homomorphic order-preserving encryption (FHOPE) [80]. Over encrypted data, this encryption approach provides homomorphic addition, homomorphic multiplication, and order comparison. As a result, it is used to compute the relevance scores over the encrypted data and to support the search operation. The DO is in charge of extracting the keywords, FHOPE-encrypting the searchable index, and uploading it to a CS in this system. The top-k most pertinent documents are returned to the DU by the CS after it has completed the search operation and ranking score operation on behalf of the DU.

In the same year, Andola et al. [57] presented a multi-user SE system that enables multi-keyword searches and uses HE properties to construct and search over an index. This scheme also allows ranked search, which is achieved using TF-IDF. The authors encrypt the index with elliptic curve-based ElGamal [81], and the encryption of sensitive data can be done with any secure symmetric encryption algorithm.

4.3. Other Functionalities

In this section, we analyze the selected works that offer other functionalities, although they are not common. Specifically, we did not find any papers that use HE to provide the ability to “Authorize or Revoke” access, nor did we find any works addressing the “Delegate” functionality. As for the remaining functionalities, we have identified six approaches that are “Dynamic” and just one which uses HE to provide the “Verifiability” characteristic.

4.3.1. Dynamic

Even though dynamic SE schemes are more likely to be employed in real-world applications, not many works have been proposed to address this problem. In our research, we identified only five published studies that exploit the HE properties to develop dynamic
SE schemes. There are, however, other schemes, which are dynamic, but they do not utilize the HE properties to achieve this functionality, as can be seen in Table 3.

Before 2021, three dynamic SE schemes that use HE to provide index updates were proposed, namely the TSED scheme proposed by Elizabeth et al. [71], a variant of this scheme, named VSED [63], and the TS-RDSE scheme proposed by Li et al. [64]. These schemes are designed for a single-user setting.

Prakash et al. proposed, in 2021 [59], a dynamic and index-based SE scheme, named PINDEX, which is intended for a multi-user setting. Their approach suggests a dynamic index construction method that is multi-linked, and which uses the PHE scheme proposed by Paillier, to encrypt the index, along with secret orthogonal vectors as building blocks. In this scheme, a DO can add and delete keywords or documents without having to reconstruct the encrypted index stored in the CS and the homomorphic properties are used on both the search and update processes.

Furthermore, the proposed scheme achieves forward privacy because of the probabilistic nature of the Paillier cryptosystem and the usage of secret orthogonal vectors. Notice that, in a dynamic SE scheme, forward privacy is a critical requirement since it ensures that newly added data does not reveal any information about previously searched queries. This is especially important in a multi-client setting where different users search different queries, and new data should not reveal anything about previous searches.

Gan et al. [55], in 2022, proposed a new dynamic searchable symmetric encryption scheme for multi-user settings, which uses XOR homomorphic function to ensure forward privacy.

The authors introduced two novel data structures to achieve efficient multi-user search and forward privacy: private links and a public search tree. Each client has three options, namely to search their own private link, the public search tree, or both. The proposed scheme also employs a state-based approach to manage database updates. This involves maintaining a state variable that tracks the current state of the database and allows for efficient updates without requiring its complete re-encryption. The XOR-homomorphic function is used in both the update and search processes. The documents themselves are encrypted using a symmetric cryptosystem.

4.3.2. Verifiability

There is only one paper which uses HE to provide the verifiability characteristic. In fact, only one more paper was found in our research which has this property, but does not use HE to achieve it, namely the work of Elizabeth and Prakash [63]. The former paper is due to Wu et al. and was published in 2018 [72]. Their work presented a verifiable public key encryption scheme with keyword search meant for multi-user settings.

In this scheme, a standard proxy re-encryption public key algorithm is used to encrypt sensitive data. The authors also suggest a novel index construction, named Z-index, which uses a binary vector for each keyword to indicate whether each file contains that keyword. The index is constructed using an inverted data structure and is encrypted using an FHE scheme, namely DGHV [47], and a homomorphic hash function. The main objective of this scheme is to avoid using query trapdoors and therefore improve search efficiency. The verifiability feature is achieved using the homomorphic hash function.

5. Research Trends and Discussion

The research on SE schemes enhanced by HE properties has significantly increased in recent years. This trend aligns with the broader adoption of HE in various domains. In this section, we aim to provide an overview of the current trends in this research area as well as highlight their significant implications on the design of the studied approaches. Specifically, we will discuss the types of HE schemes used in SE, the application of HE in search structures, in search functionalities, and in other types of functionalities. Finally, we will also discuss the ability to allow multi-users, even though achieving this functionality does not directly rely on HE.
5.1. Types of HE Schemes Used in SE

The use of HE in SE schemes has increased in recent years. In fact, 74% of the 23 selected works were published within the last three years. This is consistent with the overall trend observed in the increased application of HE in diverse areas.

Our analysis revealed that PHE is the most frequently type of HE used in SE, as shown in Figure 3. This is not surprising since PHE is the simplest type of HE among the three. On the other hand, FHE schemes have the most potential, but they are still inefficient. However, despite this, a considerable percentage of SE schemes utilize FHE due to its potential benefits (38% use FHE for 57% of systems using PHE).

![Figure 3. Types of HE schemes used in SE.](image)

Table 4. HE schemes used in selected SE approaches.

<table>
<thead>
<tr>
<th>Article</th>
<th>PHE</th>
<th>SWHE</th>
<th>FHE</th>
</tr>
</thead>
<tbody>
<tr>
<td>[52]</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[53]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[54]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[55]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[56]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[57]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[11]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[58]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[39]</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[60]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[61]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[62]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[63]</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[64]</td>
<td>x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 4. Cont.

<table>
<thead>
<tr>
<th>Article</th>
<th>Paillier</th>
<th>EC El Gamal</th>
<th>XOR</th>
<th>Boneh</th>
<th>PCTD</th>
<th>GM</th>
<th>BCP</th>
<th>SWHE</th>
<th>FHE</th>
</tr>
</thead>
<tbody>
<tr>
<td>[65]</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[66]</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>[67]</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[68]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[69]</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[70]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[71]</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>[72]</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

x: indicates that a scheme has a certain property.

5.2. HE Usage in Search Structures

The utilization of HE in SE schemes has been well explored to provide more efficient search structures. This applies to both sequential scan-based approaches and index-based ones. In all the papers that have been studied, HE is used to design the search structure of the respective scheme. It is worth mentioning that index-based SE schemes are predominant (see Figure 4), which is expected since most current applications require efficient search processes on large encrypted databases.

5.3. HE Usage in Search Functionalities

The usage of HE to improve or allow for search functionalities is very prevalent in the selected works. There is, however, a functionality which is not present in none of the works, namely the ability to perform a fuzzy keyword search. Figure 5 illustrates the number of works that support each search functionality, considering the functionalities that appeared at least once.

From our analysis, we observed that ranked search is the most prevailing search functionality, being present in 10 out of the 23 analyzed works. Moreover, in all of those works, this functionality is achieved using the homomorphic properties of the underlying HE scheme. The most commonly used protocol involves ranking documents according to TF-IDF, where the information needed to compute relevance scores is encrypted using HE. Therefore, relevance scores can be computed in the public cloud. One of these schemes, FASE [54], which allows for multi-keyword queries, firstly ranks documents by keyword matching degree, i.e., how many queried keywords are present in the documents, then the secondary criteria are the respective relevance scores. Among all studied schemes, there is only one that uses a technique other than TF-IDF to rank their search results, namely the double score weighting formula (firstly proposed by Boucenna et al. [77] and used in the work of Boucenna et al. [68]).
Another important observation, now regarding schemes that allow for top-$k$ results, is that the majority of these schemes do not allow it in a direct and single round of communication, i.e., most schemes either rely on PIR [60,67], or they include another entity in the framework, such as a coprocessor [63,71], a collaborate server [64] or a private cloud [62], which have access to the private key and are responsible for ranking search results.

Regarding the multiplicity of keywords, we observed that the majority of the works supports multi-keyword queries. There are, however, some articles that support only single-keyword queries [7,52,53,55,62]. Moreover, the ones that support multi-keyword queries can be further split into two categories: multi-keyword for ranked search and multi-keyword meant for conjunctive or disjunctive queries.

Some of the articles allow only for conjunctive queries [56,61,70,72], where the latter two also allow for phrase search, which basically is a conjunctive query where the order of keywords matters. Disjunctive queries are allowed in two of the studied works, namely the one published by Yin et al. [58] and the one by Yang et al. [65]. These approaches also allow conjunctive queries. Additionally, they allow for wildcards in each of the queried keywords, where the latter has a protocol to return the top-$k$ results when the query performed is disjunctive. Besides the above-mentioned work [65], some other schemes allow for multi-keyword queries specifically for ranked search [54,57,60,63,64,67,68]. From all these papers, only the VSED scheme [63] has a slightly different approach on the ranking protocol. More specifically, before ranking the documents, it performs a disjunctive keyword search, to guarantee that every returned document has at least one queried keyword.

5.4. HE Usage in Other Functionalities

HE has also been explored to provide functionalities such as dynamic updates and verifiability. However, during our study, we did not find any work that mentioned the ability to delegate the search. Moreover, only two works allow the revocation of users, and they do not use HE for this purpose.

Among the properties within this category, “Dynamic” is the most frequent, and is always achieved using the properties of HE. Implementing dynamic updates is challenging, especially on index-based schemes, since they required some sort of index updates. Nonetheless, schemes that utilize HE to encrypt the index offer the advantage of performing computations directly on ciphertexts, thereby facilitating efficient index updates.

From the studied articles that allow for dynamic updates, half of them rely on an inverted index [63,64,71]. Hou et al. [61] uses a VBTree, which works as a tree index where the search is performed over keywords, similar to an inverted index. The work of Prakash et al. [59] relies on a novel index type called the multi-linked index.
Regarding the ability to update, most of the approaches allow only for updates of keywords over files. However, Prakash et al. [59] and Li et al. [64] also allow for dynamic updates of files, besides allowing for regular updates over keywords.

The capability to authorize or revoke users is crucial for ensuring data privacy, yet it is not commonly addressed in the selected articles. Only two of them incorporate this property [65,67], and neither of them exploits HE schemes to achieve this functionality. Both articles follow a similar protocol, involving the generation of a search token with an expiration date. Additionally, they allow users to query multiple DOs simultaneously by requesting an authorization token from all DOs at the same time.

5.5. Multiplicity of Users

The multiplicity of users is another property that we have analyzed in the selected works. Even though this property is not directly achieved using HE, we observed that schemes meant for the multi-user setting are the most common, with a total of 63% allowing this functionality (Figure 6). Nonetheless, there are a couple of papers that show different scenarios where single-user SE schemes are suitable. Malik et al. [52], presented an application where the single entity is an airport, and Iqbal et al. [53] presented a use case where both the DO and DU are the same hospital.

![Figure 6. Multiplicity of users.](image)

Regarding the schemes that are meant for a multi-user setting, the majority rely on user authorization [54,57,60,65,67,68,70,72]. Note also that Yang et al. [67] proposed a multi-user SE scheme that allows DUs to perform search queries over multiple DOs’ data at the same time. From the remaining schemes, it is worth highlighting the work of Gan et al. [55], which achieves the multi-user functionality by allowing users to search through a private link, a public search tree, or both.

6. Conclusions and Future Work

In this work, we have provided a comprehensive analysis of the research trends on searchable encryption (SE) schemes that utilize homomorphic encryption (HE). This analysis serves as a valuable tool for researchers, enabling them to gain insights into this specific area of research and make well-informed decisions regarding future research directions.

We focused our study on the types of HE schemes used, their application to enhance search structures, to allow several functionalities such as ranked, conjunctive, disjunctive, range, and phrase search, as well as verifiability, dynamic updates, and the ability to add and revoke users. This analysis was conducted on a carefully selected set of 23 works, following a well-defined research methodology.

Our findings revealed that HE usage in SE schemes has increased in recent years, with 75% of the selected works published within the last three years. The most commonly used type of HE schemes in SE is PHE, which accounted for the majority of the analyzed schemes. The popularity of PHE schemes, particularly the widespread adoption of Paillier’s cryptosystem, can be attributed to its simplicity, proven security properties, and availability in several open-source libraries, making it easily accessible for researchers.
When considering the usage of HE in search structures, we found that building the index using HE is the most prevalent approach. This is not surprising, since SE schemes that rely on sequential scans are not suitable for large databases. Additionally, we analyzed the types of indexes used in these schemes and observed that normal indexes are the most common choice, followed by inverted indexes and tree indexes.

Regarding the usage of HE in search functionalities, we found that ranked search is the most prevalent functionality. Relevance scores are computed using homomorphic properties and are often based on TF-IDF ranking. Multi-keyword queries are widely supported, either for ranked search or for conjunctive/disjunctive queries. However, fuzzy keyword search functionality was not found in any of the analyzed works.

In future research, there is a need to explore and improve upon various functionalities when searching over encrypted data using HE schemes. Although dynamic updates consistently take advantage of homomorphic properties, other functionalities such as authorizing and revoking users, verifiability, and delegation of search capability require further development. Among the analyzed works, only one study [72] utilized a HE scheme to achieve verifiability, while none of the articles addressed the ability to authorize and revoke users. However, we found two works that, although not employing HE schemes directly, utilized a cryptographic primitive called homomorphic message authentication to facilitate verifiability, namely the work of Zhang et al. [62], and the work of Wan et al. [82]. Furthermore, Zhang et al.’s work utilized this primitive to enable the authorization and revocation of users’ access. Therefore, considering the potential benefits and the limited exploration in this area, it is worth further exploring and investigating the utilization of homomorphic message authentication and similar cryptographic primitives for achieving verifiability, and the ability to add and revoke users.

Although FHE schemes were present in nearly half of the studied articles, we believe that the percentage could be significantly higher given the recent advancements in FHE schemes. For example, the TFHE scheme, introduced in 2020 [44], is an FHE scheme worth exploring. Leveraging other encryption schemes alongside HE, such as attribute-based encryption used for user access control (e.g., it is used in the work of Boucenna et al. [68]), or OPE for facilitating ciphertext comparison (e.g., it is used in FASE [54]), could lead to the development of more sophisticated schemes.

In our study, we also concluded that the studied articles cover several important search functionalities. However, there were some notable omissions, particularly in the areas of fuzzy keyword search and occurrence queries. Fuzzy keyword search is one of the most desirable functionalities in real-life applications, making it crucial to explore HE schemes that can support this functionality. One approach, as suggested by Dong et al. [83], involves using Hamming distances. Additionally, occurrence queries were not addressed in any of the studied schemes, but we observed that most articles allowing ranked searches could easily accommodate occurrence queries since they already require information such as TF to rank documents.

Finally, it is important for future research to focus on reducing the reliance on retrieval protocols like PIR. The emphasis should be on developing SE schemes that operate effectively with a minimum number of communication rounds, aiming to minimize the overall overhead.

Overall, our work provides a comprehensive summary of the latest research trends in SE enhanced by HE. We emphasize critical components related to the HE schemes most frequently employed, possible search structures, and search functionalities. This contribution significantly advances the scientific understanding in this domain, enabling researchers to explore state-of-the-art methodologies, leverage existing knowledge, and explore new research directions.
Author Contributions: Conceptualization, I.A. and I.C.; Data curation, I.A. and I.C.; Formal analysis, I.C.; Investigation, I.A. and I.C.; Methodology, I.A.; Visualization, I.C.; Project administration, I.A.; Supervision, I.A.; Validation, I.A.; Writing—original draft, I.A and I.C.; Writing—review and editing, I.A. and I.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the Norte Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, through the European Regional Development Fund (ERDF), within project “Cybers SeC IP” (NORTE-01-0145-FEDER-000044).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All the data are already provided in the manuscript, with quoted reference.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

- AES: Advanced encryption standard
- CP-ABE: Ciphertext-policy attribute-based encryption
- CPA: Chosen plaintext attack
- CS: Cloud server
- DO: Data owner
- DU: Data user
- FHE: Fully homomorphic encryption
- FHOPE: Fully homomorphic order-preserving encryption
- FSIDS: Fully secure document similarity
- GM: Goldwasser–Micali
- HE: Homomorphic encryption
- IoT: Internet of Things
- OPE: Order-preserving encryption
- PCTD: Paillier cryptosystem with threshold decryption
- PHE: Partially homomorphic encryption
- PIR: Private information retrieval
- RE: Regular expression
- SC: Searchable ciphertext
- SE: Searchable encryption
- SIIS: Secure inverted index-based searchable encryption
- SK-NN: Secure K nearest neighbor
- SWHE: Somewhat homomorphic encryption
- TF-IDF: Term frequency inverse document frequency

References


64. Li, Y.; Zhou, F.; Xu, Z.; Ge, Y. An Efficient Two-Server Ranked Dynamic Searchable Encryption Scheme. IEEE Access 2020, 8, 86328–86344. [CrossRef]


Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.