EasyChair Preprint

№ 8518

Lightweight Searchable Encryption with Small Clients on Edge Cloud

Ruizhong Du, Haoyu Jiang and Mingyue Li

July 20, 2022

Lightweight Searchable Encryption with Small Clients on Edge Cloud

1st Ruizhong Du
Key Laboratory on High Trusted Information System in Hebei Province
Heibei University
Baoding, China
durz@hbu.edu.cn

2nd Haoyu Jiang
School of Cyberspace Security and Computing
Heibei University
Baoding, China
jianghaoyu133@gmail.com

3rd Mingyue Li
College of Cyber Science
NanKai University
TianJin, China
limingyue@mail.nankai.edu.cn

***Abstract*—In view of the limited storage and computing power of the client and the high delay of interaction with the cloud platform in public key searchable encryption, a new public key searchable encryption scheme SE-EPOMFC based on edge cloud network is proposed. The scheme adopts a multi cloud multi edge node architecture. By delegating the task of generating searchable ciphertext, trapdoor and general keyword set from the client to the edge node, the storage and computing overhead of the client is reduced. The edge network caches the frequently searched hot data, and the client can search on the edge network, so as to reduce the traffic load of the backbone network. At the same time, the response speed of the system is improved. A filtering algorithm based on partial homomorphic encryption is designed to filter completely mismatched tasks, which reduces the communication overhead between distributed systems and saves storage space for cloud services. The filtering algorithm can be calculated in the ciphertext state, which proves that it is safe under the collusion attack of semi trusted edge cloud nodes. In addition, the distributed two trapdoor public key cryptosystem is used to divide the keys for multiple nodes. Through the subset decisionmaking mechanism, the relationship between keywords is represented by binary strings to realize the search of multiple keywords. The simulation results show that the communication time of se-epomfc is saved by 25.46% in the case of task set matching degree II and 62.21% in the case of task set matching degree I.**

***Index Terms*—Distributed Search;Public Key Encryption with Keyword Search; Small Client; Edge Cloud**

I. Introduction

Since the emergence of cloud computing, cloud storage has become one of the effective ways for enterprises and individuals to store data with its unique storage advantages. Cloud computing faces serious privacy and security issues [1]. Therefore, data holders often encrypt sensitive data before outsourcing it to the cloud to protect privacy. When the data size is small, the user can download the entire encrypted data to the local computer and decrypt it, but in the increasingly popular big data applications, using this method incurs huge time and bandwidth costs in obtaining the required information [2]. The introduction of searchable encryption technology addresses the need to search data on ciphertexts. In 2004, Boneh et al. [3] proposed a public key-based searchable encryption scheme (Public Key Encryption with Keyword Search, PEKS), and first proposed the concept of multi-user search. However, PEKS has a large computational cost for ciphertext retrieval in a multi-user environment. That is, for the same data to be shared with different users, multiple searchable ciphertexts must be generated for the same data to match search trapdoors from different users, and the data holder hopes that for the same data, only one searchable ciphertext is required. It can meet the needs of different users. Similarly, for the requirement of multiple keywords, PEKS needs to run an algorithm to generate searchable ciphertext for each keyword. The data holder hopes that for a query composed of multiple keywords, the algorithm only needs to be run once. The literature [3,4,5,6,7,8,9,10,19,20,21,22,24,29] studies public key searchable encryption in multi-user, multi-data holder, multi-user The development of three functional requirements for keywords. However, the above PEKS scheme cannot resist Keyword Guessing Attack (Keyword Guess Attack, KGA). The main reason is that the keyword space is much smaller than the ciphertext space. In the actual search process, malicious users usually choose some frequently used hot words to search. Therefore, literature [14–19,24] studied KGA. In particular, the scheme [24] uses a common keyword set to convert multiple keywords into a fixed-length bit string while resisting KGA, and the size of the searchable ciphertext or trapdoor is constant, and supports multiple keywords Ciphertext search. However, due to the huge number of keywords in the general keyword set, the storage and computing overhead occupied by multiple users and multiple data holders is still not a small waste. Edge computing is a new distributed computing mode. With the help of edge nodes, users can compute and store resources closer to the physical location, which has the advantage of stronger real-time performance and is suitable for multi-user scenarios such as the Internet of Things [11]. Therefore, the introduction of the new technology of

edge computing can effectively solve the waste of complex computing and storage resources on the client side.

II. Contribution

In this paper, a SE-EPOMFC scheme is proposed, and a filtering algorithm is designed to reduce the system overhead.

A. Edge Compute

Edge nodes are introduced, and the client's computing tasks, storage tasks, and general keyword sets are delegated to edge nodes, which achieves weaker client-side storage and computing costs. Computing and storage on the side close to the user not only solves the problem of lack of client computing and storage resources, but also because the geographical distance between the edge node and the cloud is closer, the communication overhead caused by the cloud obtaining user search requests will also be significant. reduce.

B. Filtering

Using three techniques: Distributed Two-Trapdoor Public-Key Cryptosystem (DT-PKC), subset decision-making mechanism, and partially homomorphic encryption, we design a distributed pre-search filtering search task. The whole process is calculated under the ciphertext to ensure user privacy, and can reduce the overall communication overhead of distributed search when there are many meaningless tasks in the task set.

C. Evaluation

The simulation experiment evaluates the communication overhead of this scheme SE-EPOMFC and the scheme [24], as well as the storage and computing overhead of the client. The experimental results show that the scheme SE-EPOMFC saves 25.46% in communication time under the condition of task set matching degree II, and 62.21% in the case of task set matching degree I.

III. Related Works

In 2000, Song et al. [13] proposed the first symmetric searchable encryption scheme (Symmetric Searchable Encryption, SSE).

Although SSE has the advantages of small computation and high speed, both parties of communication use the same key, and the security is weaker than asymmetric encryption using key pairs.

Especially under the conditions of multiple users and multiple data holders, each pair of communicating parties needs to maintain a unique key, and there is a serious key management problem. The scheme of Wang et al.[29] supports multi-keyword queries, and the scheme of Hwang et al.[19] can prevent both internal and external KGA, but both only support single user, single data holder. In 2004, Boneh et al. [3] proposed the first public key searchable encryption scheme PEKS, which supports single user and multiple data holders. In the same year, Waters et al. [4] proposed a new scheme to construct encrypted audit logs. Golle et al. [5] proposed a searchable encryption scheme based on connected keywords, which can satisfy the search of multiple keywords at the same time, but the search trapdoor is too complicated.

For the needs of multi-user and single data holder, in 2006, Curtmola et al. [6] proposed a Naor-based broadcast encryption technology, but this scheme has a huge overhead of key revocation. In 2016, Sun et al. [20] utilized the combination of Ciphertext-Policy Attribute-Based Encryption (CP-ABE) and cross-tag to achieve multi-user access. In 2018, Wu et al. [22] proposed a verifiable PEKS scheme based on homomorphic encryption in a multi-user environment. With the development of scenarios such as the Internet of Things, it is becoming more and more important to design a searchable encryption mode that satisfies multiple users and multiple data holders. In 2008, Wang et al. [7] designed a trapdoor privacy-preserving keyword search, combined with the Shamir secret sharing algorithm and identity-based encryption to construct the first solution to meet the needs. In addition, Park et al. [8] constructed Searchable Keyword-Based Encryption (SKBE) using a proxy re-encryption technique. Both scenarios require a trusted key management server. Liu et al. [24] proposed the SE-EPOM scheme, which can satisfy multiple data holders, multi-user storage and search at the same time, and is suitable for multi-user scenarios of large companies. Huang et al. [24] constructed a PEKS scheme PE-MKS with multi-keyword search, but without support for constant-size search trapdoors, searchable ciphertexts, and their size is linear with the number of keywords.

For existing public key searchable encryption schemes, the construction mainly relies on bilinear pairings. However, the calculation of PEKS on the client side is more complicated [23]. Therefore, a common idea for designing lightweight PEKS schemes is to replace expensive bilinear pairings with certain operations. The literature [19,24] utilizes various cryptographic systems designed to ensure security.

We can also design an online and offline encryption like the scheme proposed by Liu et al. [25], dividing the complete process into two parts, and the offline part can be pre-computed, thus reducing the complex encryption overhead. With the rise of edge computing, edge nodes are often used to offload the pressure from clients or cloud centers. Wang et al.[12] proposed a lightweight PEKS scheme based on the industrial IoT environment, which delegates complex tasks to edge nodes and eliminates the heavy burden of most schemes due to bilinear pairing operations. computational cost.

PEKS faces the inherent problem of KGA. Chen et al. [14] proposed a new server-assisted PEKS scheme, SA-PEKS. Using an auxiliary server as an intermediary, data holders and users must authenticate their identity to the Keyword Server (KS) and run an interactive protocol before generating searchable ciphertexts and searching for trapdoors to return searchable secrets for secondary processing. Text, trapdoor. In this way, the searchable ciphertext, trapdoor generation process is turned into an online way, preventing KGA of malicious internal servers.

Chen et al. then proposed a dual-server test DS-PEKS scheme in [15]. Similar to the design idea of [14], the front-

end server is used to preprocess and send status information, ciphertext, and the back-end server Generate final trapdoor, searchable ciphertext and test.

The data holder in Sun et al.'s scheme [20] gives users an authorization key so that only authorized users can generate trapdoors and searchable ciphertexts.

The above solutions can only prevent KGA of internal malicious servers. Another threat faced by PEKS is that malicious external attackers may try to eavesdrop on public channels to obtain sensitive information.

The scheme proposed by Boneh et al. [3] uses a public safety channel transmission trapdoor, but the safety channel is expensive and inefficient. Baek et al. first proposed the concept of PEKS without a secure channel in the literature [16]. In this scheme, the private key of the storage server needs to be entered during the test process, which cannot be obtained by external attackers. In 2014, Huang et al. [17] proposed a secure channel-free conjunctive keyword ciphertext retrieval scheme SCF-PECKS. Immediately after Yang et al. [18] in 2018, they proved that the scheme SCF-PECKS cannot achieve resistance to KGA by enumerating three attacks.

In 2019, Hwang et al. [19] proposed a channelless secure PEKS scheme based on the ElGamal cryptosystem, while preventing internal and external KGA. In 2020, Liu et al. [24] proposed a multi-keyword PEKS scheme to perform search in a distributed system. An attacker needs to obtain the keys of multiple servers if he wants to execute KGA, so the scheme SE-EPOM successfully resists. Attacks from internal and external attackers

IV. Construction

A. System Model

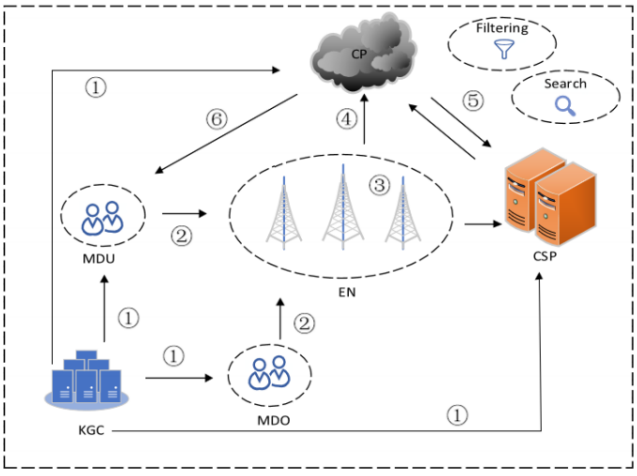

Fig. 1. System model

In the scheme SE-EPOMFC, the system consists of the following five real entity composition: a key generation center KGC, multiple data holders MDO, multi-user MDU, edge node EN, cloud platform CP and CSPs that provide auxiliary computing services. Figure 1 depicts our system system model.

1) The key generation center KGC is responsible for generating and distributing public parameters to multi-user MDU and multi-data holder MDO, and distributes some strong private keys to CP and CSP. In real life, it can be held by institutions with sufficient credibility such as the government.
2) Multi-user MDU and multi-data holder MDO rely on the public parameters generated from KGC to generate their own key pairs, and upload their respective public keys and keywords to the nearest edge node, multi-data holder MDO can Grants multi-user MDU search privileges.
3) After the edge node gets the public key and the keyword set, it collects and stores the general keywords from the nearby edge nodes, and integrates them together to form the general keyword set W.
4) In order to protect the privacy of multi-user MDU and multi-data holder MDO, edge nodes use subset decision-making mechanism and distributed two-trapdoor public key cryptosystem to generate corresponding searchable trapdoor SC and search trapdoor TD. , TD and general keyword set W are uploaded to cloud platform CP and CSP storage.
5) Cloud platform CP, with the help of computing service provider CSP, executes filtering algorithm and search algorithm to generate search results.
6) Finally, the generated search results are handed over to the multi-user MDU for decryption using the weak private key sk.

The edge node EN needs to directly observe the common keyword set and the keywords uploaded by users and data holders in order to generate searchable ciphertext SC and trapdoor TD. A man in the middle may try to masquerade as an edge node EN to obtain sensitive keyword information.

B. Threat Model

The key generation center is assumed to be completely trusted and honestly follow the protocol to generate public parameters and distribute them to various entities. Cloud platform CP and edge node EN are assumed to be semi-trusted, that is, they will honestly follow the protocol to complete their work, but will try to infer sensitive information from encrypted information. Multi-User MDU and Multi-Data Holder MDO are also semi-trusted entities that may try to eavesdrop on the channel to obtain sensitive information.

V. SE-EPOMFC

This section first defines the notation and terminology used in the paper in Table I, and introduces the overall workflow and designed algorithm of SE-EPOMFC.

A. Small Client Store

SE-EPOM implements searchable ciphertexts and trapdoors of constant size by introducing a common keyword set, expressing multiple keywords into binary strings and then encrypting them. But consider that when the number of keywords in the general keyword set is large enough, the

TABLE I
NOTATIONS

Notations	Meaning
W	The universal keyword set
W_{id}	All keywords contained in document with identifier id
W_t	The set of keywords of MDU
W_T	The set of keywords of MDO
D_{id}	Document with id
SC	Searchable ciphertext
TD	Trapdoor
T, t	The SC and TD's decimal form
T_i, t_i	The SC and TD's binary form
EN_i	The member of edge node group
$L(\cdot)$	Bit string length
$[\![\cdot]\!]_{pk_{\text{MDO}}}$	Keyword ciphertext encrypted with public key
$[\![T_i]\!]_{pk_{\text{MDO}}}$	A binary string encrypted with the MDO's public key

local computing and storage overhead of multi-user MDU and multi-data holder MDO will increase linearly. To solve this problem, the task of generating searchable ciphertext SC and searching trapdoor TD is delegated to edge nodes, and the execution process is given by Algorithm 1.

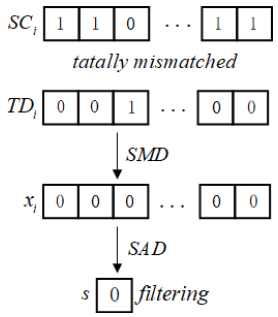

Fig. 2. filtering

B. Task Filtering

In the actual search process, a large number of interactive algorithms need to be used, and the communication is relatively complicated. In order to solve the problem of unnecessary communication overhead caused by a large number of mismatched tasks in the communication process, a Subset Decision Mechanism is designed based on the filtering algorithm is used to filter tasks that do not match at all. The scheme uses the Subset Decision Mechanism in SE-EPOM in the search process, and uses CP and CSP to perform search on the basis of ciphertext. However, when there are a large number of completely mismatched encrypted binary strings in the task set, the scheme SE-EPOM will generate redundant communication overhead. SE-EPOMFC adds a filtering algorithm in the search process, which is responsible for filtering out binary strings that do not match completely, as shown in Fig 2.

Only when the two binary strings do not match at all, does the filtering algorithm take effect. To calculate the ciphertext, only and can be used. In order to understand distinctly, the demonstrated example is executed in plaintext, but in practice, various complex communication protocols need to be run on encrypted binary strings. The filtering algorithm is used to determine whether two binary strings do not match at all. First, when each corresponding encrypted bit is different, the protocol will generate an encrypted binary string that all bits are 0. Then run the protocol to accumulate all bits and finally send the result to MDU for decryption. If the result is 0, indicating that the two binary strings must not match at all, remove it from the task set. Algorithm 1 shows the process of performing filtering in the plaintext state.

Algorithm 1: Filtering in plaintext
Input: $\mathcal{W}_t, \mathcal{W}_T, \mathcal{L}(\mathcal{W}) = \mathcal{L}(\mathcal{W}_{\mathcal{T}}) = \mathcal{L}(\mathcal{W}_{\sqcup})$
Output: Whether the task is filtered
1: Compute binary string: $(T_{n-1}, T_{n-2,\ldots,}T_0)$,$(t_{n-1}, t_{n-2,\ldots,}t_0)$
2: Set $i = 0$, $s = 0$
3: while $i < n$ do
4: $x_i = T_i - t_i$
5: $s = s + x_i$
6: end while
7: if $s = 0$ then
8: returen Filtering
9: else
10: returen No Filtering
11: end if

C. Concrete Scheme Construction

This section will introduce the workflow of SE-EPOMFC in detail, which is divided into five parts. KGC generates key pairs for each entity, EN generates SC and TD according to the requirements of MDU and MDO, and CP and CSP perform filtering and searching.

Algorithm 2: Ciphertext filtering
Input: $[\![t_i]\!]_{pk_{MDU}}$, $[\![T]\!]_{pk_{MDU}}$, pk_{MDO}, pk_{MDU}, SK_{CP}, SK_{CSP}
Output: 0 or 1
1: CP run SBD protocol with CSP
2: $SBD\left([\![T]\!]_{pk_{MDO}}\right) \rightarrow \left([\![T_{n-1}]\!]_{pk_{MDO}}, \ldots, [\![T_0]\!]_{pk_{MDO}}\right)$
3: CP run SMD and SAD protocol with CSP
4: Set $[\![f_{t_{i-1}}]\!]_{pk_{MDU}} = 0$
5: while $i < n$ do
4: $SMD\left([\![T_i]\!]_{pk_{MDO}}, [\![t_i]\!]_{pk_{MDU}}\right) \rightarrow [\![f_{t_{i-1}}]\!]_{pk_{MDU}}$
5: $SAD\left([\![f_{t_i}]\!]_{pk_{MDU}}, [\![f_{t_{i-1}}]\!]_{pk_{MDU}}\right) \rightarrow [\![f_{t_{i-1}}]\!]_{pk_{MDU}}$
6: end while
7: Let $[\![f_t]\!]_{pk_{MDU}} = [\![f_{t_{i-1}}]\!]_{pk_{MDU}}$
8: MDU decrypts $[\![f_t]\!]_{pk_{MDU}}$ with its weak private key sk_{MDU}
9: $D\left([\![f_t]\!]_{pk_{\text{MDU}}}\right) = f_t$
10: if $s = 0$ then
11: returen Filtering
12: else
13: returen No Filtering
14: end if

1. $(SK, SK_1, SK_2, PP) \leftarrow \text{Setup}(k)$. First, the KGC publishes the public parameter $PP = \{N, s\}$, sends it to MDU and MDO, sends partial strong private key $SK_1 = \lambda_1 SK_2 = \lambda_2$ to CP and CSP, and keeps $SK = \lambda$ secret.
   After the multi-user MDU and multi-data owner MDO get the public parameter $PP = \{N, s\}$, execute $KeyGen$ algorithm to generate their own key pair pk_i, sk_i, keyword set W and upload it to the edge node EN, weak private key ski secret save.

2. $[\![T]\!]_{pk_{MDO}} \leftarrow \text{Searchcp}\,(W, W_T, pk_{MDO})$.
Edge node EN runs Algorithm 1 to generate searchable ciphertext $[\![T]\!]_{pk_{MDO}}$. EN collects keywords from other edge nodes to form a general keyword set W, MDO extracts keywords from documents to form a sub-keyword set W_T, and calculates binary strings according to the index generation rules in Figure 2. Initialize a list whose length is equal to the length of the general keyword set W. If the keyword belongs to the general keyword set W, fill the corresponding bit with 1, otherwise fill with 0. Convert it into a decimal integer, use the Enc algorithm to encrypt the decimal integer to obtain a searchable ciphertext $[\![T]\!]_{pk_{MDO}}$, and upload the generated $[\![T]\!]_{pk_{MDO}}$ and general keyword set W to CP and CSP.
3. $[\![t]\!]_{pk_{MDU}} \leftarrow \text{Trapdoor}\,(W, W_t, pk_{MDU})$. The edge node EN runs Algorithm 1 to generate the search trapdoor $[\![t]\!]_{pk_{MDU}}$. Similar to the work of generating searchable ciphertexts, a general keyword set W, a user-interested keyword set W_t and an encryption algorithm Enc are required. The resulting $[\![t]\!]_{pk_{MDU}}$ is uploaded to the CP and CSP. CP and CSP are responsible for adding search trapdoor $[\![t]\!]_{pk_{MDU}}$ and searchable ciphertext $[\![T]\!]_{pk_{MDO}}$ to the task set.
4. $(0\backslash 1) \leftarrow \text{Filtering}([\![t]\!]_{pk_{MDU}}, [\![T]\!]_{pk_{MDO}}, pk_{MDU}, pk_{MDO}, SK_1, SK_2)$. First, CP and CSP perform Algorithm 3 in [24], input $[\![T]\!]_{pk_{MDO}}$, $[\![t]\!]_{pk_{MDU}}$, pk_{MDO}, pk_{MDU}, $SK_1 = \lambda_1$, $SK_2 = \lambda_2$, CP and CSP need to run the interactive SBD protocol, output $[\![\neg T_i]\!]_{pk_{MDO}}$, $[\![t_i]\!]_{pk_{MDU}}$. Secondly, as shown in the filtering algorithm 3 of this paper, input $[\![t_i]\!]_{pk_{MDU}}$, $[\![T]\!]_{pk_{MDO}}$, pk_{MDO}, pk_{MDU}, run the SBD protocol again to calculate $[\![T_i]\!]_{pk_{MDO}}$, and obtain two binary strings with all bits encrypted $[\![T_i]\!]_{pk_{MDO}}$, $[\![t_i]\!]_{pk_{MDU}}$. In these two substrings, the corresponding bits are multiplied by the SMD protocol to obtain $[\![f_{t_i}]\!]_{pk_{MDU}}$, and the SAD protocol is used to accumulate each bit in $[\![f_{t_i}]\!]_{pk_{MDU}}$, and the result is recorded as $[\![f_t]\!]_{pk_{MDU}}$. Finally, the user decrypts $[\![f_t]\!]_{pk_{MDU}}$ using the weak private key. If ft == 0, then the user returns the information that needs to be filtered to CP and CSP; otherwise, the user returns the information that does not need to be filtered to CP and CSP.
5. $(0\backslash 1) \leftarrow \text{Test}([\![t_i]\!]_{pk_{MDU}}, [\![T_i]\!]_{pk_{MDO}}, pk_{MDU}, pk_{MDO}, SK_1, SK_2)$. Performing a distributed search requires obtaining the public key pk_{MDU}, pk_{MDO}, CP and CSP provide partially strong private keys $SK_1 = \lambda_1$, $SK_2 = \lambda_2$, MDU and MDO provide filtered search trapdoors and searchable $[\![t_i]\!]_{pk_{MDU}}, [\![T_i$. CP and CSP continue to execute the following algorithm to complete the remaining distributed search.

1) CP and CSP use a filtering algorithm to choose $[\![\neg T_i]\!]_{pk_{MDO}}$, $[\![t_i]\!]_{pk_{MDU}}$, and perform a subset decision mechanism on the ciphertext to compute $[\![c_i]\!]_{pk_{MDU}}$, $[\![d_i]\!]_{pk_{MDU}}$.
2) CP and CSP extract each bit in $[\![c_i]\!]_{pk_{MDU}}$ and multiply to obtain $[\![R]\!]_{pk_{MDU}}$, and then add randomization seeds to $[\![c_i]\!]_{pk_{MDU}}$ to obtain $[\![f_i]\!]_{pk_{MDU}}$. The purpose of this step is to prevent users from inferring sensitive keyword information after decryption.
3)

After receiving the $[\![f_i]\!]_{pk_{MDU}}$, the user decrypts it with the weak private key sk_i. If the result outputs 0, it means that the trapdoor does not match the searchable ciphertext; if the result outputs 1, it means that the trapdoor matches the searchable ciphertext, and the user applies to the CP to obtain the relevant document.

D. Security

Definition 1. Let f be a deterministic functionality among parties $\mathcal{P} = (P_{MDO}, P_{MDU}, P_{CSP}, P_{CP})$ and $\prod$ be a protocol among $\mathcal{P} = (P_{MDO}, P_{MDU}, P_{CSP}, P_{CP})$. Futhermore, let $\mathcal{H} = \{\varnothing\}$, i.e., each party $P \in \mathcal{P}$ is semi-honest non-colluding parties. We say that $\prod$-securely computes f if there exists a set $\mathcal{S}\rangle\!\Updownarrow = (Sim_{MDO}, Sim_{MDU}, Sim_{CSP}, Sim_{CP})$ of PPT transformations such that for all semi-honest non-colluding adversaries $\mathcal{A} = (A_{MDO}, A_{MDU}, A_{CSP}, A_{CP})$, for all $x \in 0, 2^{\mu-1}$, $z \in 0, 2^{\mu-1}$ and for all parties $P \in \mathcal{P}$,

$$\left\{REAL^{P_i}_{f,\mathcal{P},\mathcal{S},z}(k,x)\right\}_{k\in N} \approx \left\{\text{IDEAL}^{P_i}_{\Pi,\mathcal{P},\mathcal{A},z}(k,x)\right\}_{k\in N}$$

Where $S = Sim$.

We will proves that according to Definition 1, the scheme SE-EPOMFC can be safely implemented.

Proof. Sim_{MDO} receives x as input and simulates A_{MDO} as follows: it computes $[\![T]\!]_{pk_{MDO}} \leftarrow \text{Enc}\,(T, pk_{MDO})$, returns $[\![T]\!]_{pk_{MDO}}$ to A_{MDO} and outputs A_{MDO}'s entire view. The view of A_{MDO} consists of $[\![T]\!]_{pk_{MDO}}$ to A_{MDO}. The view of A_{MDO} in both the real world and the semantic security of DT-PKC. (See Section Theorem 1 in [22] for details). Sim_{CP} simulates A_{CP} as follows: it generates encryptions of inputs $\left([\![\hat{t}]\!]_{pk_{MDO}}, [\![\hat{T}]\!]_{pk_{MDO}}\right) \leftarrow Enc(\hat{T}, pk_{MDO}, \hat{t}, pk_{MDU})$ on randomly chosen $\hat{T}, \hat{t} \in \left(0, \ldots, 2^{n-1}\right)$, computes $[\![\hat{t}]\!]_{pk_{MDU}}, [\![\hat{T}]\!]_{pk_{MDO}}, [\![\neg\hat{T}]\!]_{pk_{MDO}}$ with SMD protocol. The corresponding bits in the two encrypted binary strings are multiplied using the SMD protocol to obtain $[\![\hat{f}_{t_i}]\!]_{pk_{MDU}}$, use the SAD protocol to accumulate each bit in $[\![\hat{f}_{t_i}]\!]_{pk_{MDU}}$, and the result is recorded as $[\![\hat{f}_t]\!]_{pk_{MDU}}$. (See Section VI-C Theorem 2 in [22] for details).

After Sim_{MDO} and Sim_{CP} interact, determine the filtering task, it then generates the encryption of intermediate values $[\![\hat{c}_i]\!]_{\text{pk}_{\text{MDU}}}, [\![\hat{d}_i]\!]_{\text{pk}_{\text{MDU}}}, [\![\hat{R}_i]\!]_{\text{pk}_{\text{MDU}}}, [\![\hat{f}_i]\!]_{\text{pk}_{\text{MDU}}}$ in the same way of Proof of Theorem 3 Section VI-C in [22]. Finally, Sim_{CP} sends all the encryption of intermediate values to A_{CP}. If A_{CP} returns $\perp$, then Sim_{CP} returns $\perp$. The views of A_{CP} in both the real and the ideal world executions are indistinguishable, guaranteed by the fact that MDO is honest and the semantic security of DT-PKC. Sim_{CSP} simulates A_{CSP} as follows: it then generates the encryption of intermediate values by computing on and encrypting rangdomly chosen numbers in the same way of Proof of Theorem 3 Section VI-C in [22]. Sim_{CSP} sends these encryptions of intermediate values to A_{CSP}. If A_{CSP} returns $\perp$, then Sim_{CSP} returns $\perp$. The

views of A_{CSP} in both the real and the ideal world executions are indistinguishable, guaranteed by the fact that MDO is honest and the semantic security of DT-PKC.

VI. Performance Evaluation

A. Function Comparison

Table II shows the functional comparison of the implementation of each scheme from the two aspects of performance and security. MDU and MDO represent multiple users and data holders; SCS stands for small client store; MK stands for multiple keywords; and IKGA and OKGA stand for internal and external keyword guessing attacks. Reference [3] is the earliest PEKS scheme, which supports data upload by a single user and query by MDU. Its SC and TD must grow linearly with the number of keywords.References [12, 13] do not support the requirements of multiple users and multiple data holders, but in terms of security, reference [22] proposes a secure channel-free scheme, which satisfies internal keyword guessing attacks. Reference [12] is based on a cryptosystem, both internal and external keyword guessing attacks are satisfied. Reference [13] supports multi-user search of encrypted data uploaded by a single data holder. In the current index structure, each document is identified by a unique ID and contains several keywords. Its searchable ciphertext size is the same as $|W_{id}|$ related. In the search query, each keyword may appear in the document database DB, so the search trapdoor size is linearly related to $|DB||Q|$. Reference [17] satisfies most of the enumerated requirements, but the client needs to store the general keyword set W and undertake some computing tasks. SE-EPOMFC delegates the task of client computing searchable ciphertexts and trapdoors to edge nodes, achieving small client storage.

B. Experiment Environment

This section mainly compares and analyzes the differences between the scheme SE-EPOMFC and other schemes in client storage, computing, and search. The solution SE-EPOMFC uses python simulation, and the communication process uses socket programming to establish a reliable TCP connection. Two virtual machines were set up. KCG, CP, and MDO were deployed on a 2.90GHz, 4-core processor, with 6G memory, while CSP and MDU were deployed on a 2.90GHz, 2-core processor, with 4G memory. In order to compare with SE-EPOM, the 80 bit security level is set in the experiment and the parameter N is selected to be 1024 bit long.

The experiment mainly selects two kinds of parameters. One is the proportion of the task set that does not match the task completely (the search task does not match completely means that all the bits in the corresponding two binary strings do not match). We use the matching degree IV, III, II, and I to represent When the number of completely mismatched tasks in the task set is 0, the matching degree is IV, indicating that there are no completely mismatched tasks. When there are $\frac{1}{4}$ tasks in the task set do not match at all, the matching degree is III, and so on. Another parameter selects the number of keywords, which are 5, 10, 15, and 20 respectively.

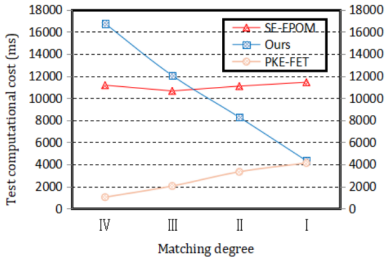

Fig. 3. search time overhead

C. search time overhead

Fig. 3 compares the time overhead of performing searches under different matching degree conditions for the three schemes. The PKE-FET scheme performs the search in a single server scenario, so the search time overhead is much smaller than the search scheme in a multi-server environment. SE-EPOM does not use the filtering algorithm and searches the task set directly. It can be seen that its time overhead fluctuates around 11s, which is not affected by the change of matching degree. While the scheme SE-EPOMFC uses the filtering algorithm, under the condition of matching degree IV and III, the time overhead generated by the filtering algorithm is more When the matching degree is below II, the filtering algorithm can significantly reduce the time overhead of SE-EPOM.

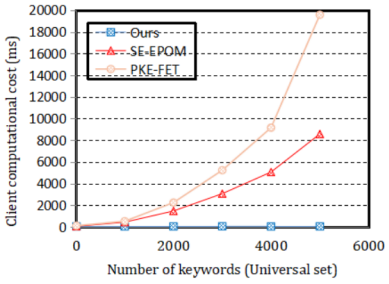

Fig. 4. generate index time overhead

D. computation and store cost

Figures 4 and 5 compare the computational time cost (generating indexes, trapdoors) and storage (storing a common set of keywords) on the client side for the two schemes. Since our scheme delegates client computing and storage tasks to edge nodes, the cost on the client side is nearly constant. The time and storage overhead of SE-EPOM will increase linearly with the increase of the number of general keywords. When the number of general keywords is 5000, the time spent is about 8.5s, and the storage will also occupy 40,000 bits. The time-saving filtering algorithm requires that more than half of the tasks in the task set are not completely matched, and it is not suitable for scenes with highly overlapping task matching numbers.

TABLE II
FUNCTION COMPARISON

	MDU	MDO	SCS	MK	OKGA	IKGA	Ciphertext size	Trapdoor size
[3]	×	✓	×	×	×	×	$O(\|W_{id}\|)$	$O(\|Q\|)$
[19]	×	×	×	×	✓	✓	$O(\|W_{id}\|)$	$O(\|W_{id}\|)$
[20]	✓	×	✓	✓	×	✓	$O(\|W_{id}\|)$	$O(\|W_{id}\|)$
[24]	✓	✓	×	✓	✓	✓	$O(1)$	$O(1)$
[29]	×	×	×	✓	×	✓	$O(\|W_{id}\|)$	$O(\|W_{id}\|)$
Ours	✓	✓	✓	✓	✓	✓	$O(1)$	$O(1)$

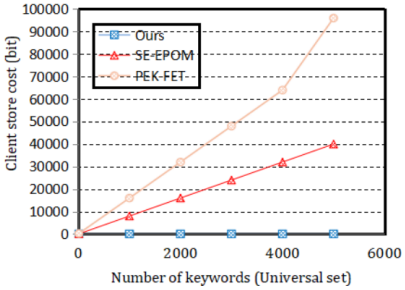

Fig. 5. verification store overhead

E. client store overhead

The filtering algorithm saves time and requires that more than half of the tasks in the task set be not completely matched. It is more suitable for special scenarios. For example, in a medical system, it is necessary to achieve a high degree of confidentiality for the privacy of patients. Doctors can only know their physical conditions; the rest of the information is unknown, and only a few keywords can be used to query patients to obtain information. In the current scenario, the patient, as the MDO, uploads encrypted information and can search the ciphertext on the CP, and the doctor, as the MDU, can use the edge device to sign and verify the correctness of the search results. The higher the degree of confidentiality of the patient's private information, the greater the possibility that the doctor will find a completely mismatched task, and a large number of mismatched tasks will eventually be generated. The VSE-EPOMFC saves doctors and patients time from having to perform complex verification.

VII. CONCLUTION AND FUTURE WORK

This paper designs a lightweight searchable encryption scheme with weak client-side storage on edge cloud, illustrates its definition of ideal reality model and keyword privacy model, and provides proofs. Compared with previous work, a weaker client-side storage is achieved, saving the client's complex computing and storage resources. By introducing edge nodes, the traffic load on the backbone network is reduced with the help of edge nodes' caching. A filtering algorithm is designed to reduce the communication overhead caused by distributed search. Experiments show that SE-EPOMFC is suitable for situations where there are many incomplete matching tasks in the task list. By comparing the differences of several schemes in offline keyword guessing attack, multi-keyword, multi-user, multi-data holder, etc., it shows that SE-EPOMFC has the advantages of overall functionality, security, computational storage cost, etc. certain advantages. Future work considers adding the function of dynamic update on the basis of the current scheme, and further researches the problem of keyword leakage in dynamic update.

REFERENCES

[1] Tabrizchi, H., Kuchaki Rafsanjani, M. A survey on security challenges in cloud computing: issues, threats, and solutions. J Supercomput 76, 9493–9532 (2020). https://doi.org/10.1007/s11227-020-03213-1

[2] H. Dai, Y. Ji, G. Yang, H. Huang and X. Yi, "A Privacy-Preserving Multi-Keyword Ranked Search Over Encrypted Data in Hybrid Clouds," in IEEE Access, vol. 8, pp. 4895-4907, 2020, doi: 10.1109/ACCESS.2019.2963096.

[3] D. Boneh, G. Di Crescenzo, R. Ostrovsky , and G. Persiano, "Public key encryption with keyword search," in International conference on the theory and applications of cryptographic techniques, 2004, pp. 506–522.

[4] Waters B R, Balfanz D, Durfee G, et al. Building an Encrypted and Searchable Audit Log[C]//NDSS. 2004, 4: 5-6.

[5] Golle P, Staddon J, Waters B. Secure conjunctive keyword search over encrypted data[C]//International conference on applied cryptography and network security. Springer, Berlin, Heidelberg, 2004: 31-45.

[6] Curtmola R, Garay J, Kamara S, et al. Searchable symmetric encryption: improved definitions and efficient constructions[J]. Journal of Computer Security, 2011, 19(5): 895-934.

[7] Wang P, Wang H, Pieprzyk J. Threshold privacy preserving keyword searches[C]//International Conference on Current Trends in Theory and Practice of Computer Science. Springer, Berlin, Heidelberg, 2008: 646-658.

[8] Park D J, Cha J, Lee P J. Searchable Keyword-Based Encryption[J]. IACR Cryptol. ePrint Arch., 2005, 2005: 367.

[9] Boneh D, Waters B. Conjunctive, subset, and range queries on encrypted data[C]//Theory of cryptography conference. Springer, Berlin, Heidelberg, 2007: 535-554.

[10] Waters B R, Balfanz D, Durfee G, et al. Building an Encrypted and Searchable Audit Log[C]//NDSS. 2004, 4: 5-6.

[11] L. Yang. Industry 4.0: A Survey on Technologies, Applications and Open Research Issues[J]. Journal of Industrial Information Integration, 2017, 6.

[12] W. Wang, P. Xu, D. Liu, L. T. Yang and Z. Yan, "Lightweighted Secure Searching Over Public-Key Ciphertexts for Edge-Cloud-Assisted Industrial IoT Devices," in IEEE Transactions on Industrial Informatics, vol. 16, no. 6, pp. 4221-4230, June 2020, doi: 10.1109/TII.2019.2950295.

[13] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in 2000 IEEE Symposium on Security and Privacy,Berkeley, California, USA, May 2000, pp. 44–55.

[14] R. Chen, Y . Mu, G. Yang, F. Guo, X. Huang, X. Wang, and Y . Wang,"Server-aided public key encryption with keyword search,"IEEE T rans. Information Forensics and Security, vol. 11, no. 12, pp. 2833–2842, 2016.

[15] R. Chen, Y . Mu, G. Yang, F. Guo, and X. Wang, "Dual-server public-key encryption with keyword search for secure cloud stor- age,"IEEE T rans. Information Forensics and Security, vol. 11, no. 4, pp. 789–798, 2016.

[16] Baek, J., Safavi-Naini, R. and Susilo, W., 2008, June. Public key encryption with keyword search revisited. In International conference on Computational Science and Its Applications (pp. 1249-1259). Springer, Berlin, Heidelberg.
[17] Hwang, M.S., Hsu, S.T. and Lee, C.C., 2014. A new public key encryption with conjunctive field keyword search scheme. Information technology and control, 43(3), pp.277-288.
[18] Lu, Y., Wang, G. and Li, J., 2018. On security of a secure channel free public key encryption with conjunctive field keyword search scheme. Information Technology and Control, 47(1), pp.56-62.
[19] Hwang, M.S., Lee, C.C. and Hsu, S.T., 2019. An ElGamal-like secure channel free public key encryption with keyword search scheme. International Journal of Foundations of Computer Science, 30(02), pp.255-273.
[20] Sun, S.F., Liu, J.K., Sakzad, A., Steinfeld, R. and Yuen, T.H., 2016, September. An efficient non-interactive multi-client searchable encryption with support for boolean queries. In European symposium on research in computer security (pp. 154-172). Springer, Cham.
[21] Huang, K., Tso, R. and Chen, Y.C., 2017. Somewhat semantic secure public key encryption with filtered-equality-test in the standard model and its extension to searchable encryption. Journal of Computer and System Sciences, 89, pp.400-409.
[22] Wu, D.N., Gan, Q.Q. and Wang, X.M., 2018. Verifiable public key encryption with keyword search based on homomorphic encryption in multi-user setting. IEEE Access, 6, pp.42445-42453.
[23] Shao, J. and Cao, Z., 2009, March. CCA-secure proxy re-encryption without pairings. In International Workshop on Public Key Cryptography (pp. 357-376). Springer, Berlin, Heidelberg.
[24] Liu, X., Yang, G., Susilo, W., Tonien, J., Liu, X. and Shen, J., 2020. Privacy-preserving multi-keyword searchable encryption for distributed systems. IEEE Transactions on Parallel and Distributed Systems, 32(3), pp.561-574.
[25] Liu, W., Liu, J., Wu, Q., Qin, B. and Liang, K., 2016, September. Online/offline public-index predicate encryption for fine-grained mobile access control. In European Symposium on Research in Computer Security (pp. 588-605). Springer, Cham.
[26] S. Kamara, P . Mohassel, and M. Raykova, "Outsourcing multi- party computation."IACR Cryptology ePrint Archive, vol. 2011, p. 272, 2011.
[27] X. Liu, R. H. Deng, K.-K. R. Choo, and J. Weng, "An efficient privacy-preserving outsourced calculation toolkit with multiple keys," IEEE T ransactions on Information Forensics and Security, vol. 11, no. 11, p. 2401, 2016.
[28] Samanthula, B.K., Chun, H. and Jiang, W., 2013, May. An efficient and probabilistic secure bit-decomposition. In Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security (pp. 541-546).
[29] Wang, P., Wang, H. and Pieprzyk, J., 2008, December. Keyword field-free conjunctive keyword searches on encrypted data and extension for dynamic groups. In International conference on cryptology and network security (pp. 178-195). Springer, Berlin, Heidelberg.