Forward Privacy Preservation in IoT-Enabled Healthcare Systems

Ke Wang, Chien-Ming Chen, *Member, IEEE*, Zhuoyu Tie, Mohammad Shojafar, *Senior Member, IEEE*, Sachin Kumar and Saru Kumari

Abstract—In recent years, Internet of things (IoT)-enabled health monitoring wearable devices have become a trend in healthcare systems, regularly collecting vital sign data from patients and uploading them to the cloud. Through on-demand search queries, data are shared with third-party healthcare service providers (HSPs) to monitor patients' health status and provide timely diagnoses. To ensure privacy and security, patient health data should be encrypted before being uploaded to the cloud. The cloud can give search encryption services. However, current searchable encryption technologies still have problems with forward privacy security and verifiability. This paper proposes an IoT-cloud-enabled healthcare data system incorporating a searchable encryption method with forward privacy and verifiability. By designing a trapdoor permutation function, we render the resulting output indistinguishable from meaningless random data to the adversary. Thus, the adversary cannot judge the relationship between a newly inserted record and a past search token, and therefore, the system realizes forward privacy or forward secrecy. We propose a multi-keyword search verification mechanism based on a pseudo-random function (PRF). Our approach solves verifying the correctness of search results in the top-k search scenario with partial search results. A formal security analysis proves that our scheme achieves forward privacy preservation, which can help guarantee healthcare data privacy. Additionally, a performance evaluation shows that our method is efficient and effective, providing an information security system to preserve patient privacy in IoT-enabled healthcare systems.

Index Terms—Internet of things (IoT), healthcare systems, privacy preservation, forward privacy

I. INTRODUCTION

THE combination of the Internet of things (IoT) and cloud computing has become a technological trend in healthcare

systems. When collected data from IoT devices are stored in the cloud and integrated into a coherent system, continuous remote monitoring and intelligent treatment of personal health problems become possible, improving patient healthcare outcomes. Such systems enable people with limited access to hospitals or limited mobility to remotely access excellent healthcare services. It allows parents to monitor their children's health and caregivers to ensure that elderly patients receive treatment as needed. For instance, in this application, a set of wearable devices periodically collects critical vital signs from a data owner (i.e., a patient). The system aggregates this information into personal health information (PHI) files and stores it on cloud servers. These PHI files are generally shared with third-party health service providers (HSPs), including doctors, through on-demand queries to monitor patients' health status and provide timely diagnoses. The combination of cloud computing and IoT wearable devices in the healthcare industry is beneficial for saving data storage space, reducing information technology (IT) costs, and improving patient treatment efficiency. However, issues of data security and personal privacy remain critical concerns in medical information systems.

Before PHI files are uploaded to cloud storage, encryption can support privacy protection functions in an e-healthcare system, but this also touches on other challenges. When third-party HSPs send on-demand queries to the cloud storage, the cloud is expected to return logically related query results rather than irrelevant results. Concurrently, encryption renders regular use of remote search in PHI files particularly challenging. Searchable encryption (SE) technology provides a promising solution to the problem of encrypted file search by adding an encrypted search index. The data owner first constructs an encrypted document index and uploads it to the cloud with the encrypted document. Any legitimate user can generate a search token, sometimes called a trapdoor. According to the received search token, the server searches the encrypted data and finally returns the search results to the user. Throughout the process, the document, search index, and search token all remain encrypted. The server can complete the search without obtaining the unencrypted plaintext information, and data privacy is effectively protected.

However, in a real environment, the data is dynamic and subject to users' change over time. Therefore, searchable encryption should support dynamic updates to protect the stored data and search for privacy. Also, the system is allowed

Ke Wang is with College of Information Science and Technology, Jinan University, Guangzhou, 518055, China (e-mail: wangke@gmail.com).

Chien-Ming Chen is with Shandong University of Science and Technology, Shandong 266510, China (e-mail: chienming.taiwan@gmail.com).

Zhuoyu Tie is with Department of Computer Science, Harbin Institute of Technology, Shenzhen 518055, China (e-mail: tiezhuoyu@outlook.com).

Mohammad Shojafar is with Network Security and Privacy LAB, 5G & 6G Innovation Centre, University of Surrey, Guildford, United Kingdom (e-mail: m.shojafar@surrey.ac.uk, m.shojafar@ieee.org).

Sachin Kumar is with Department of Computer Science and Engineering, Ajay Kumar Garg Engineering College, Ghaziabad, India (e-mail: sachink2k21@gmail.com, imsachingupta@rediffmail.com).

Saru Kumari is with the Department of Mathematics, Chaudhary Charan Singh University, Meerut, Uttar Pradesh, India (e-mail: saryusiiohi@gmail.com).

Corresponding author: Saru Kumari

to modify the security index and encrypted documents themselves dynamically. Nevertheless, this modification of encrypted data causes a forward security problem; that is, the insert operation reveals the inserted data's content. The server can match the index corresponding to the newly inserted data using a legitimate search token generated by past users. Using this access, a user can distinguish data containing previous keyword searches, which still can result in a healthcare data privacy violation through unauthorized information disclosure. Moreover, considering that the third-party storage server is usually regarded as a semi-trusted entity, it may deliberately return wrong search results to mislead users. To save computing resources, the server may also submit empty sets as search results to users. Therefore, searchable encryption schemes should be port users' verification of the correctness of the results.

Forward privacy and verifiability, or forward secrecy, has received recent attention in the field of searchable encryption (SE). The purpose of forward security is to prevent the server from judging whether the updated content contains keywords from previous user search requests. It also helps verifiability consider the malicious modification of search data on the server, which requires an additional authentication mechanism to ensure that users can judge whether search results are correct. However, few studies have been conducted on SE schemes that satisfy both forward security and verifiability. It remains necessary to design appropriate verification mechanisms to provide a defense for SE schemes' forward security properties against potential attack.

Searchable symmetric encryption (SSE), first introduced by Song et al. [1], is designed to protect remote data privacy, with a search time linear to documents' length. Considering the need to update data on remote servers, Kamara et al. [2] proposed a dynamic, searchable symmetric encryption (DSSE) with an optimal search time to allow the data owner to dynamically modify the encrypted data, i.e., to perform an insert or delete operation.

Key-value (KV) store systems [3,4] with high performance and scalability have become popular recently. Yuan et al. [5] proposed a distributed DSSE scheme based on Redis KV, and the authors in [6] subsequently proposed an encrypted and distributed KV store with an EncKV scheme. These methods provide a range-match search model hiding order relations and partial information with ciphertexts and order-revealing encryption (ORE). The search time for an exact-match model is optimal, while the search time for a range-match model is linear to the number of records.

Although considering adaptive adversaries for exact-match models and non-adaptive adversaries for range-match models, EncKV does not preserve forward privacy. Forward privacy is a method for limiting the leakage of inserted text in dynamic SE schemes [7], with a search time logarithmic to the number of keyword-document pairs. Forward privacy requires that the adversary be unable to determine whether a newly added data contains the keywords searched in the past. Forward privacy is

increasingly being considered in DSSE schemes [8-9]. Chen et al. [10] used similar ideas to construct a forward secure public key SE scheme, contrasting with other such schemes by using bi-linear pairing. In the following year, Chen et al. [13] further proposed a public key SE scheme satisfying both forward security and backward security. Backward security requires that an attacker cannot determine whether an inserted or deleted entry contains keywords that have been searched in the past.

Inspired by EncKV and existing forward privacy research, we propose FEncKV, a distributed SSE method on a key-value store, with forward privacy. There are some existing schemes with forward privacy [11,12,14,15]; however, they are only designed for exact-match search, while FEncKV constructs a range-match search model with an ORE scheme [6]. We summarize our contribution in this paper as follows:

- We propose a scheme called FEncKV, based on a trapdoor permutation and a status count.
- We prove that FEncKV has the feature of forward privacy, meaning that an adversary is unable to determine the relationship between a previous search query and a newly added record.

The remaining sections are organized as follows. The proposed IoT-cloud-enabled healthcare data system and their related components are described in Section II. The security analysis is reported Section III. Section IV explains the experimental evaluation. Finally, Section V concludes the paper and gives an outlook.

II. PROPOSED METHOD

In this section, first, we describe our IoT-enabled healthcare system, specify the threat assumptions, and then explain the privacy components designed in the proposed scheme.

A. System model

Fig. 1 presents the general model of searchable encryption for IoT-enabled healthcare systems. In this figure, three types of entities are defined in the system used by the FEncKV scheme: data owner (DO), healthcare service providers (HSPs) and healthcare cloud server provider (HCSP).

Fig. 1. General model for SE for IoT-enabled healthcare systems

The data owner DO (usually with wearable devices) is responsible for encrypting the document to be uploaded, establishing an index, and then sending the encrypted data and index to the HCSP. For verifiable schemes, the data owner also

needs to generate summary information for the search results in advance and submit it to the server. The healthcare service providers HSPs send the corresponding search request to the server according to their own needs and submits the encrypted search token. The cloud then returns the search results to HSPs. In the verifiable scheme, the proof information will be returned together with the search results. The HSPs need to verify the search results and finally decide whether to receive the search results.

HCSP is responsible for receiving and storing encrypted documents and security indexes submitted by data owners. After the HSPs initiate the search request, they search the security index according to the received search token, get the corresponding search results, and finally return the search results to the data user. For the verifiable class scheme, the cloud server usually needs to calculate a proof according to the summary information to provide the HSPs with the verification of the search results. The server is usually regarded as a semi-trusted entity in the system. The server will collect data privacy as much as possible on the premise of complying with the encryption protocol.

B. Threats

Threats are mainly from semi-honest adversaries, who faithfully follow the protocol but try to collect privacy information on patient data and search queries. The adversary cannot know the private keys of the client but is able to access the data transmitted over public channels, for example, encrypted key-value pairs, the secure index, and search tokens. In addition, the search and access patterns are exposed to the adversary. The search pattern exposes the search history for a given search token, and the access pattern exposes the set of user entry data for each search query.

Currently, EncKV does not consider that attackers can learn background information on datasets and queries, e.g., the partial or entire content of queries or their distribution [16-18]. Actively malicious attackers (as mentioned in [19-21]) are not considered in the range-match model, which may be addressed by an orthogonal study.

C. FEncKV Design

We improve on EncKV to satisfy the condition of forward privacy. This approach ensures that an adversary cannot learn the relationship between an inserted record and a previous search query. Fig. 2 presents our system flow. The records stored in the relational database of the cloud transform into label value. The label is generated by ID and attribute name, and the value of the corresponding attribute is symmetrically encrypted, as t_1 and t_2 . The secondary index α and β is then established, and the label is generated by using the condition attribute and the corresponding value. The record's ID is symmetrically encrypted to obtain the value, which is then inserted into the table in the form of the label value key-value pair. In the search process, a label in the secondary index is

obtained by searching the token. The corresponding value is taken out. The ID of the record is obtained by decryption. Then, the encrypted value is obtained using the ID, and the result is attributed to the label. The value of the result attribute in the corresponding record is obtained by decryption.

TABLE I
SYMBOL TABLE

Symbol	Description
R	the set of record IDs $\{r_1, \dots, r_n\}$
R_v	the set of all record IDs with conditional attributes equal to v
ER_v	encrypted R_v
er	encrypted r
C_v	conditional attribute
C_r	consequence attribute
V	all the possible values of conditional attributes $\{v_1, \dots, v_m\}$
V_r	the value of C_r in the result attribute of search results
v_r	the value of result attribute C_r for record r
ev	encrypted v
T_v	search token
I^{rng}	range index $\{I_1^{rng}, \dots, I_N^{rng}\}$ distributed over N nodes

Fig. 2. System flow

The main contribution of FEncKV is that we used a trapdoor permutation function to transform the state counter instead of the traditional method, thus we achieved forward privacy and verifiability for searchable encryption schemes. Because the data owner and data user have the public key and private key, they can transform the state counter forward or backward; while the server node only has the public key, it can only transform the state counter unidirectionally. Thus, it can prevent the server from using the fields in the past search token to predict the fields in the insert update token, and then prevent the server from judging whether the new insert record contains the attributes and values searched in the past, so as to achieve forward security.

D. Range-match index and query protocol

1) Secure index

The range-matched index building protocol is as shown in Algorithm 3, with which the client builds a range-match index $\{I_1^{rng}, \dots, I_n^{rng}\}$ for n server nodes. For attribute C_v , the

corresponding status counter c_i is initialized to a random number, and each time $c_i \leftarrow \pi_{sk}^{-1}(c_i)$ to the next status with private key sk of client. After generating the index label α , the client calls the function $OREnc$ to get ct_R . Both ct_R and $Enc(k_R, R)$ are hidden with a mask $H_2(t_2, c_i)$ to generate β . Finally, key-value pair (α, β) is inserted into the range-match index I_i^{rng} for server node i .

Algorithm 1. Build_{mg} : Build Range-Match Index

Input: Private key k_r, k_o, sk , values $\{v_1, \dots, v_m\}$ on attribute C_v , PRFs $\{H_1, H_2, G_1, G_2\}$, trapdoor permutation π .

Output: Encrypted indices $\{I_1^{rng}, \dots, I_n^{rng}\}$.

```

1: Initialize a hash table  $S$  to maintain counters;
2: for  $v_j \in \{v_1, \dots, v_m\}$  do
3:    $i \leftarrow route(R)$ ;
4:    $t_1 \leftarrow G_1(k_r, C_v \parallel i)$ ;
5:    $t_2 \leftarrow G_2(k_r, C_v \parallel i)$ ;
6:   if  $S.find(i) = \perp$  then
7:     Pick a random number  $r$ ;
8:      $c_i^j \leftarrow r$ ;
9:   else
10:     $c_i \leftarrow S.find(i)$ ;
11:   end if
12:    $\alpha \leftarrow H_1(t_1, c_i)$ ;
13:    $ct_R \leftarrow OREnc(k_o, v_j)$ ;
14:    $\beta \leftarrow H_2(t_2, c_i) \oplus (ct_R \parallel Enc(k_R, R))$ ;
15:    $c_i \leftarrow \pi_{pk}(c_i)$ ;
16:    $S.put(i, c_i)$ ;
17:    $I_i^{rng}.put(\alpha, \beta)$ ;
18: end for
```

2) Secure query

Algorithm 2 describes a range-match query protocol. For a value v on attribute C_v , the client generates tokens (t_1, t_2, ct_L, c_i) and sends them to each server node. Given a token, node i calculates label α using PRF . Next, the node obtains β from range-match index I_i^{rng} and reveals the encrypted IDs r and ct_R . If $OREcmp(ct_L, ct_R) == true$, the record with ID r is matched. The server node next transforms the status counter $c_i \leftarrow \pi_{pk}(c_i)$ with the client's public key pk and repeats the above operation. Finally, the client receives encrypted IDs and fetches v_r via L .

Algorithm 2. Query_{mg} : Range-Match Query

Input: Private key k_r, k_o , result value attribute C_r , query condition value v on attribute C_v , order condition cmp , PRFs $\{H_1, H_2, G_1, G_2\}$,

trapdoor permutation π .

Output: Encrypted matched result $\{v_r\}$

Client.Token

```

1: for  $i \in \{1, \dots, n\}$  do
2:    $t_1 \leftarrow G_1(k_e, C_v \parallel i)$ ;
3:    $t_2 \leftarrow G_2(k_e, C_v \parallel i)$ ;
4:    $ct_L \leftarrow OREtoken(k_o, v, cmp)$ ;
5:    $c_i \leftarrow S.find(i)$ ;
6:   Send  $(t_1, t_2, ct_L, c_i)$  to node  $i$ ;
7: end for
Nodei.ExtQuery
1:  $\alpha \leftarrow H_1(t_1, c_i)$ ;
2: while  $find(\alpha) \neq \perp$  do
3:    $\beta \leftarrow find(\alpha)$ ;
4:    $ct_R \parallel r \leftarrow I_i^{rng}.get(H_2(t_2, c_i) \oplus \beta)$ ;
5:    $c_i \leftarrow \pi_{pk}(c_i)$ ;
6:   if  $OREcmp(ct_L, ct_R) == true$  then
7:     Return to  $r$  client;
8:   end if
9:   Fetch  $v_r$  via  $r$ ;
10: end while
```

III. SECURITY ANALYSIS

In this part, we analyze the security of the proposed scheme.

A. Forward Privacy

FEncKV scheme is with forward privacy because a server node fails to determine the relationship between a newly added record and a previous search query. In contrast to EncKV, the transform of the status counter in FEncKV is limited: with private key sk and public key pk , the client is able to calculate $\pi_{pk}(c_i)$ and $\pi_{sk}^{-1}(c_i)$, while the server node is only able to compute $\pi_{pk}(c_i)$ with the client's public key pk . A previous search token is invalid for a new inserted record with the same value, because the two have different status counters.

For the range-match search model, a value v on attribute C_v , $t_1 = G_1(k_e, C_v \parallel i)$, $t_2 = G_2(k_e, C_v \parallel i)$, and the corresponding updated index entry $\langle \alpha, \beta \rangle$, $\alpha = H_1(t_1, c_i)$ and $\beta = H_2(t_2, c_i) \oplus (ct_R \parallel Enc(k_R, R))$. Similarly, knowing public key pk of the client, with search token (t_1, t_2, ct_L, c_i) , an adversary $\mathcal{A}$ is unable to distinguish a new added index label $\alpha' = H_1(t_1, \pi_{sk}^{-1}(c_i))$ from a random number. Thus, in Step 4, given previous search tokens $\{t_1, \dots, t_m\}$, $\mathcal{A}$ learns nothing about the next inserted label-value pair. In Step 6, given an inserted index entry $\langle \alpha', \beta' \rangle$, $\mathcal{A}$ fails to determine whether the added record matches the previous value.

B. Security on Range-Match Queries

The leakage function of the range-match model FEncKV scheme is as follows:

$$\mathcal{L}_1^{ng}(\mathbf{C}) = (\{m_i\}_n, \langle \alpha, \beta \rangle, \langle l, v^* \rangle) \quad (1)$$

where $\mathcal{L}_1^{ng}$ describes the leakage of building range-match index, and

$$\mathcal{L}_2^{ng}(v_c, cmp, C_v, C_r) = (\{t_1^i, t_2^i, c_i\}_n, \{\{\alpha, \beta, r, l, v^*\}_{cnt_i}\}_n, ct_L, \{\{b_{dif}\}_{cnt_i}\}_n, M_{q \times q}, T_q) \quad (2)$$

where b_{dif} is the ORE comparison block, and ct_L is the ORE token. Unlike $\mathcal{L}_2^{ext}$, $M_{q \times q}$ only traces the search query with the same attribute. The value information is protected by ORE encryption. Furthermore, we know that

$$\mathcal{L}_3^{ng} = (R, v, C, \mathbf{C}) = (\alpha, \beta, r, l, v^*). \quad (3)$$

where $\mathcal{L}_3^{ng}$ represents the leakage of the insert operation. Because FEncKV is forward private, the relationship between a record and previous search queries is not exposed. Next, we provide a formal definition of such security.

Definition 1. Let $\mathcal{A}$ be a probabilistic polynomial time (PPT) adversary, and $\mathcal{S}$ be a PPT simulator. There are two probabilistic games **Real** $_{\mathcal{A}}(k)$ and **Ideal** $_{\mathcal{A}, \mathcal{S}}(k)$ as follows:

In the **Real** $_{\mathcal{A}}(k)$ game, given k , the client first generates private keys. $\mathcal{A}$ selects a dataset $\mathbf{D}$ and sends it to the client, receiving a range-match index. Then, $\mathcal{A}$ makes some search queries to the client and, for each query, receives the corresponding search token and search result. $\mathcal{A}$ asks the client for insert operation access on some records and receives the corresponding index entry. These queries are nonadaptive. Finally, $\mathcal{A}$ returns a bit b as output.

In the **Ideal** $_{\mathcal{A}, \mathcal{S}}(k)$ game, given k , $\mathcal{A}$ selects a dataset $\mathbf{D}$ and sends it to $\mathcal{S}$. $\mathcal{S}$ simulates the range-match index with $\mathcal{L}_1^{ng}$ and returns it to $\mathcal{A}$. Then, $\mathcal{A}$ makes some search queries and sends $\mathcal{L}_2^{ng}$ to $\mathcal{S}$. For each query, $\mathcal{A}$ receives the corresponding search token and search result from $\mathcal{S}$. Also, $\mathcal{A}$ ask $\mathcal{S}$ for insert operation access for some records. $\mathcal{S}$ gets $\mathcal{L}_3^{ng}$ and returns the corresponding index entry. These queries are nonadaptive. Finally, $\mathcal{A}$ returns a bit b as output.

We say that the FEncKV range-match protocol is adaptively secure with $(\mathcal{L}_1^{ng}, \mathcal{L}_2^{ng}, \mathcal{L}_3^{ng})$ for PPT adversaries $\mathcal{A}$, if there exists a PPT simulator $\mathcal{S}$ such that

$$|\Pr[\mathbf{Real}_{\mathcal{A}}(k) = 1] - \Pr[\mathbf{Ideal}_{\mathcal{A}, \mathcal{S}}(k)]| \leq \text{negl}(k). \quad (4)$$

Theorem 1. The FEncKV range-match protocol is non-adaptively secure with $(\mathcal{L}_1^{ng}, \mathcal{L}_2^{ng}, \mathcal{L}_3^{ng})$ under the random oracle model if $H_1, H_2, G_1, G_2, F_1, F_2, F_3, P, Q$ are secure PRF, and π is a secure trapdoor permutation.

Proof. Here, we construct a simulator $\mathcal{S}$ to let adversary $\mathcal{A}$ fail to distinguish **Real** $_{\mathcal{A}}(k)$ from **Ideal** $_{\mathcal{A}, \mathcal{S}}(k)$. Let $\mathcal{H}_{H1}, \mathcal{H}_{H2}$,

and $\mathcal{H}_r$ be random oracles. Given leakage $\mathcal{L}_1^{ng}$, $\mathcal{S}$ simulates a secure index $\{\tilde{l}_1^{ng}, \dots, \tilde{l}_n^{ng}\}$, where $\tilde{l}_i^{ng}$ contains m_i pairs $\langle \tilde{\alpha}, \tilde{\beta} \rangle$. Each $\tilde{\alpha}$ is a $|\alpha|$ -bit random string and $\tilde{\beta}$ is a $|\beta|$ -bit random string.

For an insert operation, $\mathcal{S}$ receives leakage $\mathcal{L}_3^{ng}$. Due to the forward privacy feature, the relationship between the previous search query and an inserted record is not leaked. $\mathcal{S}$ generates $\langle \tilde{\alpha}, \tilde{\beta} \rangle$ and $\langle \tilde{l}, \tilde{v}^* \rangle$ as random strings to simulate $\langle \alpha, \beta \rangle$ and $\langle l, v^* \rangle$.

When it comes to search query q , $\mathcal{S}$ is given the leakage $\mathcal{L}_2^{ng}$. It constructs a search token $(\tilde{l}_1, \tilde{l}_2, \tilde{ct}_L, \tilde{c})$ for node i , determining first whether the attribute has been searched by checking $M_{q \times q}$.

- If there is no such a previous search query q' , $\mathcal{S}$ picks three random string to simulate $\tilde{l}_1$, $\tilde{l}_2$, and $\tilde{ct}_L$. To simulate the status counter $\tilde{c}$, $\mathcal{S}$ determines a polynomial number N and we assume that the number of matched entries is not more than N . He picks $\tilde{c}_N$ a random string, and repeatedly calculates $\tilde{c}_{j-1} \leftarrow \pi_{pk}(\tilde{c}_j)$, $j \in [1, N]$. Table CNT stores those status counters $(\tilde{c}_1, \dots, \tilde{c}_N)$. For convenience, we note $\text{CNT}[\tilde{l}_1][j] = \tilde{c}_j$. $\mathcal{S}$ sends $(\tilde{l}_1, \tilde{l}_2, \tilde{c}_1)$ as simulated search token to $\mathcal{A}$.
- If there is such a previous search query q' , and their matched results (check T_q) are equal, it means that no record with same value was inserted between two search queries, and the search token $(\tilde{l}_1, \tilde{l}_2, \tilde{ct}_L, \tilde{c})$ of q' is still valid; If their matched results are different, it means some records have been inserted and the status counter should be updated. $\mathcal{S}$ gets the current status counter $\tilde{c} \leftarrow \text{CNT}[\tilde{l}_1][cnt_i - (|T_q| - |T_{q'}|)]$, where $|T_q| - |T_{q'}|$ is the number of inserted records between two search queries. $\mathcal{S}$ sends $(\tilde{l}_1, \tilde{l}_2, \tilde{ct}_L, \tilde{c})$ as simulated search token to $\mathcal{A}$.

Next, $\mathcal{S}$ needs to validate the simulated token $(\tilde{l}_1, \tilde{l}_2, \tilde{ct}_L, \tilde{c})$ for the simulated index $\tilde{l}_i^{ng}$. For each matched $\langle \alpha, \beta \rangle$ and $\langle l, v^* \rangle$, $\mathcal{S}$ selects an unused entry $\langle \tilde{\alpha}, \tilde{\beta} \rangle$ and $\langle \tilde{l}, \tilde{v}^* \rangle$ from, $\tilde{l}_i^{ng}$, and lets $\mathcal{H}_{H1}(\tilde{l}_1, \tilde{c}) \leftarrow \tilde{\alpha}$. Given the encrypted ID r , $\mathcal{S}$ simulates $\tilde{r} \leftarrow \mathcal{H}_r(r)$. Knowing the different blocks b_{dif} the ORE comparison $\mathcal{S}$ needs to simulate $\tilde{ct}_R$ satisfying

$\text{OREcmp}(\tilde{c}_{t_L}, \tilde{c}_{t_R}) = \text{true}$, regarding $\tilde{c}_{t_L}$ as $(\tilde{c}_{t_{L1}}, \dots, \tilde{c}_{t_{Lb}})$, and $x_i, q_i \leftarrow \tilde{c}_{t_{Li}}$. For the block b_{diff} , which is the first permuted block differing between $\tilde{c}_{t_L}$ and $\tilde{c}_{t_R}$, $\mathcal{S}$ generates a random string $\tilde{r}$ and simulates $\tilde{z} \leftarrow \tilde{q}_{b_{\text{diff}}} + Q(\tilde{x}_{b_{\text{diff}}}, \tilde{r})$ as $\tilde{c}_{t_R}$ the b_{diff} -th block. Then, it generates a random string for remaining $\tilde{c}_{t_R}$ blocks. Next, it lets $\mathcal{H}_{H2}(\tilde{t}_2, \tilde{c}) \leftarrow \tilde{\beta} \oplus (\tilde{c}_{t_R} \parallel \tilde{r})$. Due to the semantic security of secure PRF, $\mathcal{A}$ cannot differentiate the simulated tokens and results from the real tokens and results.

IV. PERFORMANCE EVALUATION

In this section, we assess the proposed privacy-aware scheme compared to the literature. We implemented the FEncKV method in Java and performed experiments on multiple entities: six hosts acted as cloud server nodes, two devices (wearable devices) acted as data owners, and one desktop acted as a healthcare service provider. Each host had a 4-core CPU (AMD EPYC™ Rome 2.6 GHz), 16 GB RAM, and used the Ubuntu Server 16.04.1 LTS 64 operating system. We adopt Redis (v3.2.0) to provide a key-value pair store, and Thrift (v0.9.2), a remote procedure call (RPC) framework, to implement communication between server nodes and the HSP.

TABLE II
TIME COMPLEXITY

	EncKV.Ext	EncKV.Rng	FEncKV.Ext	FEncKV.Rng
Index	$O(R)$	$O(2^d b^2 R)$	$O(E R)$	$O((2^d b^2 + E) R)$
Search	$O(R_v)$	$O(2^d b^2 R)$	$O(E R_v)$	$O((2^d b^2 + E) R)$

Table II presents the time complexity analytic terms for each scheme component of the proposed scheme. In this table, the symbol "E" represents the high consumption exponential operation of trapdoor replacement.

We compare the index's time complexity (data owner side) and search operations (cloud side) between FEncKV and EncKV. In the range search mode, the index building operation of FEncKV and EncKV requires the ORE encryption scheme. The time complexity is related to the bit length index of each data block in ORE encryption. In this table, by observing the data, we find that the index building task's time complexity is linearly related to the total amount of data. This conforms to the above time complexity analysis, that is, the time complexity of the precise search mode of FEncKV and EncKV is the same when the number of bits is determined. The time complexity of the range search is also $O(E|R|)$ and $O(|R|)$, where $|R|$ represents the total number of records. In FEncKV, the time consumption of index establishment is related to the bit length d and block number b of ORE encryption: the value of each record is encrypted into two ciphertexts with lengths of $2^d b^2$ bits.

A. Latency and energy consumption in wearable devices side

In our scheme, we implement PRF with HMAC-256 and build PRP on a three-round Feistel block cipher. A symmetric encryption scheme was implemented with AES-128. The secure hash function is SHA256. We use RSA for trapdoor permutation. We implemented both EncKV and FEncKV in Java. We mainly evaluated the index-building stage of FEncKV and also implemented EncKV for comparison.

TABLE III
TIME CONSUMPTION OF BUILDING INDEX WITH DIFFERENT TOTAL RECORDS

	0.1K (s)	1K (s)	2K (s)	4K (s)	8K (s)	10K (s)
Plaintext	0.03	0.31	0.64	1.25	2.45	3.14
EncKV	0.05	0.38	0.73	1.45	2.84	3.55
FEncKV	0.25	1.41	2.60	5.21	10.04	12.65
EncKV 2-bit	0.29	1.71	3.26	6.26	12.09	15.17
EncKV 4-bit	0.45	2.95	5.95	11.60	23.18	29.79
FEncKV 2-bit	0.45	2.39	4.59	8.78	16.93	21.35
FEncKV 4-bit	0.58	3.73	7.17	14.15	27.77	34.90

Table III shows the specific data from the experiment. With the total number of data increasing from 100 to 10,000, the building index's time consumption between the FEncKV and EncKV increases from 0.25s and 0.05s, respectively, to 12.65s and 3.55s. For the range index of 2-bit ORE encryption, the time consumption increases from 0.45s and 0.29s to 21.35s and 15.17s, respectively. We find that the time consumption is significantly less than that of 4-bit ORE encryption. This is because despite having the same time complexity, the 4-bit ORE still occupies more memory space, increasing the experimental program's burden to request memory from the operating system. Table IV shows the energy consumption with different total records. Commonly wearable devices build an index and upload the captured data to the cloud once per day, and 10K is almost the most significant record number of the captured data per day. When the record number is 10K, FEncKV 4-bit costs the energy of 6.98J. It is acceptable for wearable devices.

TABLE IV
ENERGY CONSUMPTION OF BUILDING INDEX WITH DIFFERENT TOTAL RECORDS

	0.1K (J)	1K (J)	2K (J)	4K (J)	8K (J)	10K (J)
Plaintext	0.006	0.062	0.128	0.250	0.490	0.628
EncKV	0.010	0.076	0.146	0.290	0.568	0.710
FEncKV	0.050	0.282	0.520	1.042	2.008	2.530
EncKV 2-bit	0.058	0.342	0.652	1.252	2.418	3.034
EncKV 4-bit	0.090	0.590	1.190	2.320	4.636	5.958
FEncKV 2-bit	0.090	0.478	0.918	1.756	3.386	4.270
FEncKV 4-bit	0.116	0.746	1.434	2.830	5.554	6.980

In the verification aspect of this study, we compared our scheme with VMKSE [22] and VDERS [23]. FEncKV uses multiple-set hashes as proof information for verification of search results, VDERS uses an RSA accumulator to generate verification information, and VMKSE uses an obfuscated bloom filter. For the record type data, the complexity of the verification mechanism is the same, the time consumption of building the verification table is $O(|R|)$ and is linearly related to

the total number of records. The time required to verify the search results is $O(|R_v|)$, which linearly correlates to the size of the search results.

TABLE V
TIME CONSUMPTION OF BUILDING VERIFICATION TABLE WITH DIFFERENT TOTAL RECORDS

	0.1K (s)	1K (s)	2K (s)	4K (s)	8K (s)	10K (s)
VDERS	0.42	1.96	3.58	6.82	13.32	16.30
VMKSE	0.11	0.69	1.32	2.58	5.19	6.58
FEncKV	0.13	0.38	0.58	0.88	1.55	2.16

To enable HSPs to verify the search results, data owners (wearable devices) need to build a secure validation table in advance to upload the proof information to the cloud for search results corresponding to each keyword. As the total number of records increases, the search results' size corresponding to each keyword increases, so the time required to build the validation table increases.

The way to build the verification table is to obtain the corresponding search result for each different search value, and calculate the RSA accumulator, obfuscation bloom filter GBF, or multiset hash for the search result as the proof of the corresponding search value, and then store the "search value proof" in the verification table. Therefore, the time needed to establish the verification table is the sum of the time required to calculate the search results' proofs corresponding to all values and insert them into a table. A comparison experiment between the establishment of the validation table and the verification phase is presented in Table V. Compared with VMKSE and VDERS, FEncKV consumed less time. When the total number of records was 1K to 10K, the time consumption of FEncKV building verification table was lower than that of VDERS and VMKSE. From the observation, the time consumption of building the verification table is linearly related to the total number of records. Table V presents the specific time consumption of three schemes. Also, Table VI illustrates the energy consumption from 0.026J to 0.432J under the total records from 0.1K to 10K.

TABLE VI
ENERGY CONSUMPTION OF BUILDING VERIFICATION TABLE WITH DIFFERENT TOTAL RECORDS

	0.1K (J)	1K (J)	2K (J)	4K (J)	8K (J)	10K (J)
VDERS	0.084	0.392	0.716	1.364	2.664	3.260
VMKSE	0.022	0.138	0.264	0.516	1.038	1.316
FEncKV	0.026	0.076	0.116	0.176	0.310	0.432

B. Latency for search algorithms in the cloud side

TABLE VII
RESPONSE LATENCY WITH DIFFERENT SEARCH RESULT SIZE(MS)

	1PCS	10PCS	20PCS	50PCS	100PCS
Plaintext	2.69	3.33	3.30	3.44	3.81
EncKV	2.82	3.48	3.73	4.34	5.39
FEncKV	3.00	3.44	3.71	4.45	5.75
EncKV 2-bit	366	345	355	350	346
EncKV 4-bit	346	352	348	352	360
FEncKV 2-bit	388	387	382	391	391

FEncKV 4-bit	385	382	398	387	403
--------------	------------	------------	------------	------------	------------

In the search phase, as shown in Table VII, by changing the size of the search results from 1 to 100, it can be observed that the time consumption of precise search for FEncKV scheme and EncKV increases from 3ms and 2.82ms to 5.75ms and 5.39ms, respectively approximately linearly correlated to the search results, which is in accordance with the time complexity. For the range search mode, the time consumption of FEncKV and EncKV fluctuates from 382ms to 403ms and 345ms to 366ms, respectively. This is because when the total number of data is determined, the time complexity is independent of the size of the search results.

TABLE VIII
SEARCH TIME LATENCY WITH INCREASING SERVER NODES

	1(s)	2(s)	3(s)	4(s)	5(s)	6(s)
Plaintext	0.255	0.155	0.128	0.109	0.092	0.085
EncKV	0.524	0.34	0.29	0.259	0.238	0.229
FEncKV	0.655	0.423	0.331	0.305	0.27	0.245
EncKV 2-bit	4.615	2.581	1.964	1.549	1.202	1.115
EncKV 4-bit	4.824	2.614	1.903	1.519	1.305	1.125
FEncKV 2-bit	5.344	2.846	2.18	1.645	1.37	1.204
FEncKV 4-bit	5.629	3.408	2.254	1.803	1.507	1.337

As shown in Table VIII, with the increasing number of server nodes, the search phase's time consumption is significantly reduced. With only one server node, the time consumption of FEncKV and EncKV are 0.655s and 0.524s, respectively. While with six server nodes, the corresponding time consumption decreases to 0.245s and 0.229s. For range search with 2-bit ORE encryption, the related time consumption decreases from 5.344s and 4.615s to 1.204s and 1.115s, respectively. The time consumption of 4-bit ORE range search in FEncKV is slightly higher than that of 2-bit ORE range search, and the 4-bit ORE range search in EncKV is close to 2-bit ORE range search.

TABLE IX
VERIFICATION MESSAGE SIZE

Verification message size	
VMKSE	31 · R · 128-bit
VDERS	1024-bit
FEncKV	256-bit

Focusing on Table IX, in this experiment, the obfuscation bloom filter of VMKSE is set to be 128-bit data blocks with 31 times the number of inserted elements, and the length has a linear correlation with total records number $|R|$. When the total amount of data is 10000, the size of verification information in VMKSE will reach 4.75 MB, which will affect the search efficiency in real application scenarios. The sizes of proof information of FEncKV and VDERS are only 256 bits and 1024 bits.

C. Security performance

To verify the forward security performance of our proposed scheme, we used a hierarchical search attack (HAS) [24] to

attack EncKV and FEncKV and recorded the number of records required for injection.

HSA is an advanced injection attack method. Considering that users may refuse to encrypt abnormal documents (where the number of keywords is greater than the threshold T , t is far less than $|K|$), the keyword space K can be divided into several smaller subspaces before attacking. The active adaptive HSA attack first injects some information. After observing the user query results, this method confirms the subspace to which the query belongs and then injects documents into the target subspace. This attack can further reduce the number of injected documents. The number of records of injection attacks is shown in Table X.

TABLE X
NUMBER OF RECORDS OF INJECTION ATTACKS

Adaptive-HSA	
EncKV.Ext	$\prod V_C $
FEncKV.Ext	$\prod V_C $
EncKV.Rng	$ C /T + \lceil \log T \rceil - 1$
FEncKV.Rng	$(C /2T) \cdot (\lceil \log 2T \rceil + 2)$

Let all possible attribute sets be $\{C_1, C_2, \dots\}$, and all possible values of attribute $C \in \{C_1, C_2, \dots\}$ be $V_C = \{v_1, v_2, \dots\}$. Threshold T represents the limit of the number of attributes contained in each record.

Table XI presents the number of injected files required for adaptive HSA attacks under different threshold values T . The results indicate that more records need to be injected to attack the FEncKV scheme with forward security.

TABLE XI
NUMBER OF INJECTED RECORDS UNDER DIFFERENT THRESHOLDS T IN
ADAPTIVE HSA ATTACK

	10	20	40	80	160	320
EncKV.Rng	503	254	130	69	39	24
FEncKV.Rng	1750	1000	567	320	176	96

From Table XI, we understand that the higher the number of injected records, the more alert the user, and the easier it is to detect the attack. When the threshold is set at $T=10$, the EncKV scheme needs to inject 503 records, while our proposed improvement FEncKV needs to inject 1750 records. With a threshold set at $T=320$, the EncKV only needs to inject 24 records, while FEncKV needs to inject 96 records. Therefore, in general, the test results can reflect that our FEncKV scheme has superior resistance to privacy attacks, and its forward security performance is significantly better.

V. CONCLUSION AND FUTURE DIRECTIONS

In this study, we designed a trapdoor permutation method and proposed a verifiable forward searchable encryption scheme. After each insertion update, the corresponding state counter is transformed by the trapdoor replacement function

and a private key to replace the incrementing method each time in the original scheme. Because the trapdoor permutation function's output is indistinguishable from random numbers to the adversary, the adversary cannot judge the relationship between a newly inserted record and a past search token FEncKV has forward private security or a forward private secrecy cryptographic feature. Besides, we proposed a multi-keyword search verification mechanism based on a pseudo-random function. Our approach solved the problem of verifying the correctness of search results in the top-K search scenario with only partial search results. The experimental results show that the proposed FEncKV scheme is suitable for IoT-enabled healthcare systems.

We will continue to design a new searchable encryption algorithm for the cloud-edge-terminal integrated applications in future research. Also, we plan to enhance our scheme by considering the recent automatic privacy verification mechanisms, such as [25] to provide a robust and reliable IoT-enabled healthcare, especially for large-scale multi-purpose IoT devices connecting as infotainment wearable devices and gather information and transferring them through the hop-to-hop software networks.

REFERENCES

- [1] D. X. Song, D. Wagner, and A. Perrig, "Practical Techniques for Searches on Encrypted Data," in IEEE Symposium on Security & Privacy, 2002.
- [2] S. Kamara, C. Papamanthou and T. Roeder, Dynamic searchable symmetric encryption, in computer and communications security, 2012, pp. 965-976.
- [3] Ma W, Zhu Y, Li C, et al. BiloKey: A Scalable Bi-Index Locality-Aware In-Memory Key-Value Store, in IEEE Transactions on Parallel and Distributed Systems, 30(7), 2019:1528 - 1540.
- [4] Anwar A, Cheng Y, Huang H, et al. Customizable Scale-Out Key-Value Stores, in IEEE Transactions on Parallel and Distributed Systems, 2020, 31(9):2081-2096.
- [5] X. Yuan, X. Wang, C. Wang, C. Qian, and J. Lin, Building an Encrypted, Distributed, and Searchable Key-value Store, in computer and communications security, 2016.
- [6] Guo Y, Yuan X, Wang X, et al. Enabling Encrypted Rich Queries in Distributed Key-value Stores, in IEEE Transactions on Parallel and Distributed Systems, 2018, 30(6): 1283-1297.
- [7] Li H, Yang Y, Dai Y, et al. Achieving Secure and Efficient Dynamic Searchable Symmetric Encryption over Medical Cloud Data, in IEEE Transactions on Cloud Computing, 2017:1-1.
- [8] Wang Q, He M, Du M, et al. Searchable Encryption over Feature-Rich Data, in IEEE Transactions on Dependable & Secure Computing, 2018:1-1.
- [9] Li H, Yang Y, Dai Y, et al. Achieving Secure and Efficient Dynamic Searchable Symmetric Encryption over Medical Cloud Data, in IEEE Transactions on Cloud Computing, 2017:1-1.
- [10] Chen B, Wu L, Kumar N, et al. Lightweight Searchable Public-key Encryption with Forward Privacy over IoT Outsourced Data, in IEEE Transactions on Emerging Topics in Computing, 2019, PP (99):1-1.
- [11] Li H, Yang Y, Dai Y, et al. Achieving Secure and Efficient Dynamic Searchable Symmetric Encryption over Medical Cloud Data, in IEEE Transactions on Cloud Computing, 2020, 8(2):484-494.
- [12] Song X, Dong C, Yuan D, et al. Forward Private Searchable Symmetric Encryption with Optimized I/O Efficiency, in IEEE Transactions on Dependable & Secure Computing, 2017, 17(5): 912-927.
- [13] Chen B, Wu L, Wang H, et al. A Blockchain-Based Searchable Public-Key Encryption with Forward and Backward Privacy for Cloud-Assisted Vehicular Social Networks, in IEEE Transactions on Vehicular Technology, 2020, 69(6):5813-5825.
- [14] Xiong H, Mei Q, Zhao Y, et al. Scalable and Forward Secure Network Attestation with Privacy-Preserving in Cloud-Assisted Internet of Things, in IEEE Sensors Journal, 2019, 19(18): 8317-8331.
- [15] Li H, Liu L, Lan C, et al. Lattice-Based Privacy-Preserving and Forward-Secure Cloud Storage Public Auditing Scheme, in IEEE Access, 2020, 8: 86797-86809.

- [16] Yao T, Tan Z, Wan J, et al. SEALDB: An Efficient LSM-tree based KV Store on SMR Drives with Sets and Dynamic Bands, in IEEE Transactions on Parallel and Distributed Systems, 2019, 30(11): 2595-2607.
- [17] Yuan, Xingliang & Guo, Yu & Wang, Xinyu & Wang, Cong & Li, Baochun & Jia, Xiaohua. (2017). EncKV: An Encrypted Key-value Store with Rich Queries. 423-435. 10.1145/3052973.3052977.
- [18] Yue Y, He B, Li Y, et al. Building an Efficient Put-Intensive Key-Value Store with Skip-Tree, in IEEE Transactions on Parallel & Distributed Systems, 2017, 28(4):961-973.
- [19] Zhou R, Zhang X, Du X, et al. File-Centric Multi-Key Aggregate Keyword Searchable Encryption for Industrial Internet of Things, in IEEE Transactions on Industrial Informatics, 2018, 14(8): 3648 - 3658.
- [20] Lu Y, Li J, Zhang Y. Secure Channel Free Certificate-Based Searchable Encryption Withstanding Outside and Inside Keyword Guessing Attacks, in IEEE Transactions on Services Computing, 2020, PP (99):1-1.
- [21] Miao Y, Tong Q, Deng R, et al. Verifiable Searchable Encryption Framework against Insider Keyword-Guessing Attack in Cloud Storage, in IEEE Transactions on Cloud Computing, 2020, PP (99):1-1.
- [22] Y. Su, J. Wang, Y. Wang and M. Miao, Efficient Verifiable Multi-Key Searchable Encryption in Cloud Computing, in IEEE Access, 2019, 7:141352-141362.
- [23] Liu, Qin & Tian, Yue & Wu, Jie & Peng, Tao & Wang, Guojun, Enabling Verifiable and Dynamic Ranked Search Over Outsourced Data, in IEEE Transactions on Services Computing, 2019, PP. 1-1.
- [24] Zhang Y, Katz J, Papamanthou C, All Your Queries Are Belong to Us: The Power of File-Injection Attacks on Searchable Encryption, in Proceedings of the Security Symposium, 2016: 707-720.
- [25] Taheri R, Shojafar M, Alazab M, Tafazolli, R, FED-IIoT: A Robust Federated Malware Detection Architecture in Industrial IoT, in IEEE Transactions on Industrial Informatics, 2020, PP (99):1-1.

Ke Wang is a Professor with Jinan University, Guangzhou, China. He has published more than 60 research papers in IoT and AI. Currently, he works as a senior researcher at the College of Information Science and Technology, Jinan University. Many of his ideas had been realized in real applications. His main research interests include AI and IoT.

Chien-Ming Chen (Senior Member, IEEE) received the Ph.D. degree from National Tsing Hua University, Taiwan. He is currently an Associate Professor with the Shandong University of Science and Technology, China. His current research interests include network security, the mobile Internet, IoT, and cryptography. He also serves as an Associate Editor for IEEE ACCESS and an Executive Editor for the International Journal of Information Computer Security.

Zhuoyu Tie received the B. D. degree from Northeastern University, China, and received the M. D. degree from Harbin Institute of Technology (HIT), China. His main research interest is information security.

Mohammad Shojafar (M'17-SM'19) is a Senior Lecturer (Associate Professor) in the network security and an Intel Innovator, and a Marie Curie Alumni, working in the 5G & 6G Innovation Centre (5GIC/6GIC) at the University of Surrey (UoS), Guildford, UK. Before joining UoS, he was a Senior Researcher and a Marie Curie Fellow at the University of Padua (UNIPD), Italy. In 2019, he was a Senior Researcher working on a network security project with Ryerson University (RU) and Telus Communications Inc. (TELUS) in Toronto, Canada. Also, he was CNIT Senior Researcher at the University of Rome Tor Vergata contributed to 5G PPP European H2020 "SUPERFLUIDITY" project. Dr. Mohammad was a PI of PRISENODE project, a 275k euro Horizon 2020 Marie Curie global fellowship project in the areas of Fog/Cloud security collaborating at UNIPD. He was contributed to some Italian projects like GAUCHO, SAMMClouds, and SC2. He received his Ph.D. degree from Sapienza University of Rome, Rome, Italy, in 2016 with an "Excellent" degree. He is an Associate Editor in IEEE Transactions on Consumer Electronics, IEEE Systems Journal, and Computer Networks. For additional information: <http://mshojafar.com>

Sachin Kumar is working as a Professor of Computer Science and Engineering at Ajay Kumar Garg Engineering College, Ghaziabad, India. He received a Ph.D. degree in Computer Science from Chaudhary Charan Singh University, Meerut, India. He has published more than 30 papers in the SCI/ Scopus Index Journals. He delivered a talk and chaired sessions in various International/National conferences of repute. He has authored five books on Computer Science. He also guided 04 Ph.D. and 10 M.Tech. Scholars. His area of interest includes Computer Networks and Applied Cryptography.

Saru Kumari is currently an Assistant Professor with the Department of Mathematics, Chaudhary Charan Singh University, Meerut, Uttar Pradesh, India. She received her Ph.D. degree in Mathematics in 2012 from Chaudhary Charan Singh University, Meerut, UP, India. She has published more than 215 research papers in reputed International journals and conferences, including more than 195 research papers in various SCI-Indexed Journals. She is on the editorial board of more than a dozen of International Journals, of high repute, under IEEE, Elsevier, Springer, Wiley and others including IEEE Systems Journal. She has served as the Guest Editor of many special issues in many SCI Journals under IEEE, Elsevier, Springer and Wiley. She has been involved in the research community as Technical Program Committee (TPC) member or PC chair for more than a dozen of International conferences of high repute. She is also serving as a reviewer of dozens of reputed Journals including SCI-Indexed of IEEE, Elsevier, Springer, Wiley, Taylor & Francis, etc. Her current research interests include information security and applied cryptography.