ISSN 2277-2685

IJESR/April-May. 2025/ Vol-15/Issue-2s/1644-1651

Ibrahim Farwah Affan *et. al.*, / International Journal of Engineering & Science Research

Privacy-Preserving Search On Encrypted Health Records In The Cloud

Ibrahim Farwah Affan[1], Gulam Jeelani[2], Abdullah Ameer Ahmed[3], Mr. Syed Juber[4]

[1,2,3]B.E. Student, Department of IT, Lords Institute of Engineering and Technology, Hyderabad

[4]Assistant Professor, Department of IT, Lords Institute of Engineering and Technology, Hyderabad

juber@lords.ac.in

Abstract—***Cloud-based Personal Health Record systems (CB-PHR) have great potential in facilitating the management of individual health records. Security and privacy concerns are among the main obstacles for the wide adoption of CB-PHR systems. In this paper, we consider a multi-source CB-PHR system in which multiple data providers such as hospitals and physicians are authorized by individual data owners to upload their personal health data to an untrusted public cloud. The health data are submitted in an encrypted form to ensure data security, and each data provider also submits encrypted data indexes to enable queries over the encrypted data. We propose a novel Multi-Source Order-Preserving Symmetric Encryption (MOPSE) scheme whereby the cloud can merge the encrypted data indexes from multiple data providers without knowing the index content. MOPSE enables efficient and privacy-preserving query processing in that a data user can submit a single data query the cloud can process over the encrypted data from all related data providers without knowing the query content. We also propose an enhanced scheme, MOPSE+, to more efficiently support the data queries by hierarchical data providers. Extensive analysis and experiments over real datasets demonstrate the efficacy and efficiency of MOPSE and MOPSE+***

Keywords—Cloud-based Personal Health Record (CB-PHR), Security and privacy, Multi-source system, Encrypted data, Order-Preserving Symmetric Encryption (MOPSE), Privacy-preserving query processing, MOPSE+, Efficient query support

I. INTRODUCTION

C LOUD-BASED Personal Health Record systems (CBPHR) such as Microsoft HealthVault[1] and ZebraHealth[2] are rising. A typical CB-PHR system consists of three entities: data owners, data providers and a cloud server. In CBPHR system, data owners and data providers are defined as patients themselves and hospitals, respectively. Data owners can directly authorize data providers to upload their PHRs to the cloud. The CB-PHR system allows data owners to access their PHRs anytime and anywhere, be better prepared for medical appointments and unexpected emergencies, maintain a more complete picture about personal health, and even achieve fitness goals. Data providers can explore the PHR system to provide better medical services by sharing, collaborating, and engaging with the patients in new ways.

Privacy concerns are among the main obstacles for the wide adoption of CB-PHR systems. In particular, many people have deep concerns that there can be unauthorized access to their sensitive PHRs. For example, the cloud may have business interest in analysing the PHRs, and it may also have malicious employees or even be hacked. A natural way to alleviate the privacy concerns is to let data owners and providers upload encrypted PHRs to the cloud which does not possess the decryption keys [1]–[5], [8]. Since PHRs can be in huge volume, it is very inefficient for data owners or providers to retrieve all the encrypted PHRs from the cloud when only a small portion of them are needed. To enable efficient queries over encrypted PHRs, the B+-tree technique [2]–[5], [9] is proposed to build an index for each patient's PHRs. The data index allows the cloud server to quickly find all the PHRs matching a particular data query. To further resolve the privacy concerns about data indexes and queries, searchable encryption schemes [10]–[20] are proposed to encrypt data indexes and queries as well. These schemes allow the cloud server to perform efficient queries over encrypted PHRs directly based on the encrypted indexes and queries while blind to the index and query content.

Traditional searchable encryption schemes [10]–[20] are designed for generic cloud platforms and not optimized for CB-PHR systems. In particular, the PHRs of different data providers for the same data owner may be highly correlated and associated with the same attributes (e.g., symptoms). If a traditional search encryption scheme is used, each data provider needs to independently generate the encrypted data index for submission to the cloud server. Therefore, the data owner needs to manage the keys with different data providers and also submit a dedicated data query for each data provider even if query conditions are exactly the same. A plausible solution to this issue is to let all the data providers use a common key assigned by the data owner to encrypt the data indexes associated with him. This method, however, is vulnerable to the compromise of a single data provider.

In this paper, we propose a very efficient PHR system with strong privacy guarantees. In our system, each data owner authorizes multiple data providers to submit encrypted health records and data indexes to the cloud server. Our system differs from prior work in two desirable features. First, each data provider of

the same data owner uses a unique symmetric key for encrypting data indexes, thus resisting single point of compromise. Second, each data owner needs not manage the keys with individual health providers and can submit a single encrypted query to the cloud server for searching over the encrypted health data from all his data providers. The second feature enables very efficient query processing.

Our system is built upon Multi-source Encrypted Indexes Merge (MEIM), a novel technique we propose in this paper. MEIM allows the cloud server to merge multiple encrypted data indexes from different health providers of the same patient without violating the patient's privacy. It also permits the patient to generate a single encrypted query over all his health providers' encrypted data stored at the cloud server. The fundamental building block of MEIM is a novel Multisource Order-Preserving Symmetric Encryption (MOPSE) primitive we develop. MOPSE preserves the order of multiple data indexes encrypted by different symmetric keys.

We also propose an MOPSE+ primitive to support hierarchical authorization queries whereby the health providers with higher privileges can query the cloud server for the encrypted data from those with lower privileges. Such hierarchical access patterns are quite common in practice. We confirm the security and efficiency of our system by comprehensive theoretical analysis and extensive exper3No gender implication. iments with a real dataset [21]. Our results show that (1). the query performance for data users in MOPSE and MOPSE+ is faster $n \times^4$ than that in tradition OPSE; (2). the query performance for data providers in MOPSE and OPSE are almost the same and less than that in MOPSE+.

II. RELATED WORK

Recently, privacy preserving in PHRs have drawn re-searchers' attention [2]–[7]. In this section, we review three categories of work: searchable encryption, order-preserving symmetric encryption and other related work. Search en- cryption schemes guarantee that the untrusted entity gains nothing about what data owners are searching for, and order- preserving symmetric encryption can guarantee the order of the ciphertexts following with that of the plaintexts. There has been a lot of works for searchable encryption [3], [12]– [19], [22], order-preserving symmetric encryption [20], and other related work [8-15]

Data access control is another vital issue in cloud storage system. Many researchers [11]–[16] focused on multi-authority access control scenario. In [21], the authors aimed to address multi-authority problem, and proposed a data access scheme DAC-MACS. However, Hong et al. [12] found that Yang's work may be attacked due to utilizing a bidirectional reEncryption method, and described their attack method. To address the data privacy problem and the user identity privacy both, Jung et al. [13] proposed AnonyControl scheme. In [14], the authors stated that the prior work cannot overwhelm the common shortcoming, like low efficiency and single point bottleneck. So they proposed RAAC, a robust access control scheme. To reduce the overhead of decryption, Chase and Chow [15] proposed a scheme to remove the trusted central authority. In [16], the authors proposed a threshold multi-authority ciphertext-policy attribute-based encryption access control solution (named TMACS). However, none of them can be directly utilized to address our problem due to ignoring hierarchical authentication query problem.

Existing System

In the domain of cloud-based Personal Health Record (CB-PHR) systems, existing approaches primarily rely on traditional searchable encryption schemes to manage and query encrypted health data. These systems, as highlighted in prior works such as [3], [10]-[20] are designed for generic cloud platforms and focus on securing data while enabling search functionality over encrypted records. Typically, a CB-PHR system involves three entities: data owners (patients), data providers (e.g., hospitals or physicians), and an untrusted cloud server. In these systems, personal health records (PHRs) are encrypted before being outsourced to the cloud to ensure data privacy, addressing concerns about unauthorized access by the cloud or external entities.

However, existing searchable encryption schemes exhibit several limitations when applied to multi-source CB-PHR environments. Firstly, they are not optimized for scenarios where multiple data providers contribute PHRs for a single data owner. Each data provider independently generates encrypted data indexes using distinct keys, requiring the data owner to manage multiple keys and submit separate queries for each provider, even when query conditions are identical. This results in inefficient query processing and increased computational overhead for the data owner. For instance, schemes like those in [14] (symmetric key cryptography) and [15] (public key cryptography) support equality queries but lack efficient support for range queries across multiple sources. Additionally, approaches such as [17] encrypt indexes page-by-page, reducing communication overhead but burdening users with key management complexity. Other methods, such as SafeQ [5] and QuerySec [14], aim to support range queries but either return imprecise results or fail to address multi-source and hierarchical access scenarios effectively. Moreover, traditional Order-Preserving Symmetric Encryption (OPSE) schemes [20], [14] preserve ciphertext order for single-source data but cannot maintain order across multiple sources with different keys, limiting their applicability in CB-PHR systems.

These Shortcomings*—inefficient multi-source query handling, lack of hierarchical access support, and poor scalability—hinder the adoption of CB-PHR systems, particularly in terms of privacy-preserving search efficiency and practicality.*

Proposed System

To address the limitations of existing systems, the proposed system introduces a novel framework, termed Multi-source Encrypted Indexes Merge (MEIM), tailored for privacy-preserving search over encrypted PHRs in a multi-source cloud environment. This system, detailed in the referenced study, enhances security and efficiency by integrating two innovative schemes: Multi-Source Order-Preserving Symmetric Encryption (MOPSE) and its enhanced variant, MOPSE$^+$. The proposed CB-PHR system retains the three-entity model (data owners, data providers, and cloud server) but introduces significant advancements in data management and query processing.

In the proposed system, multiple data providers (e.g., hospitals, physicians, or personal health devices) are authorized by a data owner to upload encrypted PHRs and corresponding encrypted data indexes to an untrusted public cloud. Unlike existing systems, each data provider uses a unique symmetric key to encrypt its data indexes, mitigating the risk of a single point of compromise. The MEIM technique enables the cloud server to merge these encrypted indexes from multiple sources without decrypting them, preserving index privacy. This is achieved through MOPSE, which extends traditional OPSE by preserving the order of ciphertexts across different keys. Consequently, a data owner can submit a single encrypted query to search over all providers' encrypted PHRs, significantly reducing query overhead compared to submitting separate queries per provider in existing systems.

The system employs a Multi-Dimensional B-Tree (MDBT) structure to index PHRs, supporting multi-attribute queries (equality, subset, and range). MOPSE ensures that the cloud can process these queries efficiently over the merged indexes without accessing the plaintext content of the data, indexes, or queries, thus guaranteeing privacy. To further enhance practicality, MOPSE$^+$ introduces hierarchical authorization, allowing higher-privileged data providers (e.g., research hospitals) to query data from lower-privileged providers (e.g., personal devices) securely. This hierarchical model, built upon the HPBPE scheme [20], addresses real-world access patterns absent in prior works.

Experimental evaluation using the Nursery Dataset from the UCI Machine Learning Repository demonstrates that MOPSE and MOPSE$^+$ outperform traditional OPSE in query efficiency. For data owners, query performance is improved by a factor of nnn (where nnn is the number of data providers), while maintaining comparable performance to OPSE for data providers. MOPSE$^+$, despite higher encryption overhead (e.g., 42.5 ms per attribute value versus 1289.894 ms for MOPSE with n=3n=3n=3), supports hierarchical queries efficiently. Security analysis confirms that the system protects index and query privacy against honest-but-curious cloud servers and untrusted providers, with rigorous proofs establishing resistance to chosen-plaintext attacks (CPA) under RDSP and IDSP assumptions.

The Proposed System*---- thus offers a secure, efficient, and scalable solution for CB-PHR systems, overcoming the multi-source and hierarchical limitations of existing approaches while maintaining strong privacy guarantees.*

III. SYSTEM AND THREAT MODELS

A. SYSTEM MODEL AND WORKFLOW

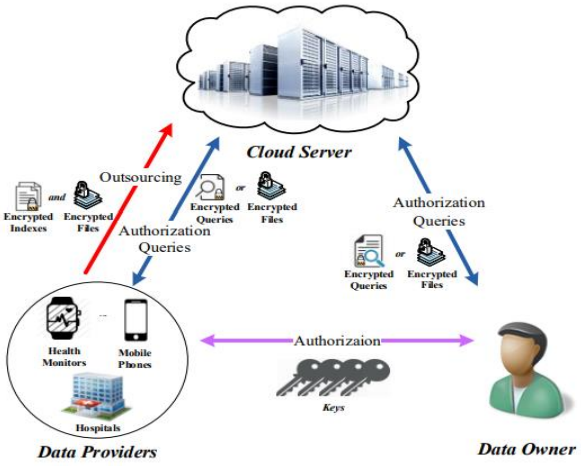

FIGURE1: Architecture of the Proposed System

We consider a generic CB-PHR system shown in Fig. 1. There are three kinds of entities: the cloud server, data owners, and data providers. A data owner refers to a patient who own the PHRs. In contrast, a data provider can refer to a patient himself, any of his health providers such as a physician or hospital, and even his personal health monitoring device. The cloud server stores and provides anytime, anywhere access to the PHRs submitted by the data providers of each data owner.

Each data owner has strong privacy concerns for his PHRs. His data providers thus must encrypt the PHRs before outsourcing them to the cloud server. To ensure efficient search for the encrypted PHRs, each data provider additionally uploads a data index to the cloud server. The data owner or any of his authorized data providers can submit data queries to the cloud server. Both data indexes and queries should be encrypted as well to prevent information disclosure. The cloud server explores the data indexes to locate the PHRs satisfying each query without the capability or need to decrypt the PHRs, data indexes, or data queries. Finally, the cloud server returns the corresponding

encrypted PHRs to the requesting data user who can decrypt them with the right decryption key.

TABLE 1 : Notation

Symbol	Definition
R and r_d	Integer range set and integer range (l_d,u_d)
n	MDBTs numbers
ψ	Attribute numbers in original index
γ	Health records numbers
φ	A large prime
$\mathbb{G}_j$, j={1,2,$\mathcal{T}$}	Three cyclic groups of order φ
g_j, j={1,2,$\mathcal{T}$}	The group element of $\mathbb{G}_j$, j={1,2,$\mathcal{T}$}
$\mathbb{V}$ and $\mathbb{V}^*$	Two vector spaces
x and **y**	Two vectors $(g_1^{x_1}, \cdots, g_1^{x_n})$ and $(g_2^{y_1}, \cdots, g_2^{y_n})$
$\mathbb{A}$ and $\mathbb{A}^*$	Two canonical bases of $\mathbb{V}$ and $\mathbb{V}^*$
$\vec{x}/\vec{v}$	Plaintext/Predicate vector

B. THREAT MODEL

We assume a conventional threat model as follows. The cloud server is honest-but-curious by faithfully running the system but having strong interest in the content of the PHRs, data indexes, and queries. We also assume that data providers are untrusted and may try to acquire the PHRs generated by other providers. Besides, we assume that the communications within our system are secured using traditional mechanisms such as TLS (Transport Layer Security) [22]. Finally, we assume that the cloud server does not collude with data providers to compromise data owners' privacy.

The privacy of data owners can be classified into PHR privacy, index privacy, and query privacy. Since our system stores encrypted PHRs at the cloud server which has no decryption keys, PHR privacy is easily achieved as long as the underlying encryption primitive is unbreakable. We thus focus on index privacy and query privacy hereafter. Index privacy is considered compromised if the content of any encrypted index is known to the cloud server or any data provider other than the source data provider. In contrast, query privacy is said to be breached in either scenario below. First, the content of any encrypted query is disclosed to anyone other than the data user (i.e., the data owner or any of his authorized data providers). Second, a data provider generates a valid query without obtaining the authorization of the data owner.

IV. MULTI-SOURCE ENCRYPTED INDEX MERGING

Our system features a novel Multi-source Encrypted Index Merging (MEIM) technique whereby the cloud server can merge the encrypted data indexes from different data providers of the same data owner without decrypting individual indexes. MEIM allows a data user to submit a single data query for the PHRs from all the data providers of the data owner. In what follows, we introduce the basics of indexing and querying PHRs, followed by the MEIM design.

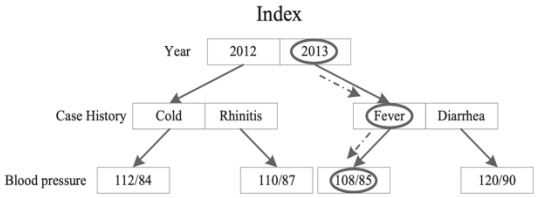

FIGURE 2: An MDBT index example with three attributes, year, case history and blood pressure. The nodes satisfying the query are marked with black circles, and the arrows show the matching processing.

A. *INDEXING AND QUERYING PHRS*

We explore the classic multi-dimensional B-tree (MDBT) [9] to index PHRs. In particular, we assume that each PHR has the same set of queryable attributes such as name, year, disease, and blood pressure. The values of each attribute can be naturally numerical (e.g., year) or non-numerical (e.g., disease), and we convert non-numerical attribute values into numerical ones with the coding technique in [13]. Each attribute corresponds to one layer on the MDBT, and each node (the root, non-leaf nodes, and leaf nodes) has one or multiple attribute values depending on the particular PHRs. In addition, each leaf node has a pointer to the corresponding encrypted PHR. Fig. 2 shows an exemplary MDBT index for PHRs with three attributes: year, disease, and blood pressure.

The MDBT index supports multi-attribute equality, subset, and range queries. For example, a query related to Fig. 2 can be (2012 $\leq$ year $\leq$ 2013) $\wedge$ (disease $\in$ {Cold, F ever}) $\wedge$ (blood pressure = 108/85), where $\wedge$ denotes the conjunction operator. The sub-queries for the year, disease, and blood pressure attributes correspond to range, subset, and equality queries, respectively. The nodes matching this query in each layer are marked with black circles in Fig. 2. The leaf node with value 108/85 points to the PHR that should be returned to the data user. Equality queries are specific range queries, and subset queries can be easily transformed into range or equality queries. So we focus on range queries hereafter.

B. *OVERVIEW OF MEIM*

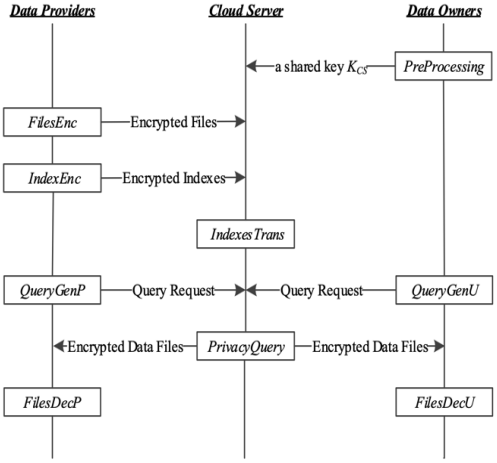

FIGURE 3: This figure shows the MEIM workflow. The details are represented in Section III-B.

Fig. 3 depicts the work flow of MEIM with the following function modules for each involved entity.

Data owner: (1) Pre-processing generates a group key KDP, a secret key K kept to itself, and a secret key KCS shared with the cloud server; (2) QueryGenU generates his data queries; and (3) FilesDecU decrypts the encrypted PHRs returned by the cloud server.

Data provider: (1) FilesEnc encrypts the PHRs with a symmetric-key algorithm and then encrypts the symmetric key with a public-key algorithm; (2) IndexEnc creates the MDBT index and then encrypts it with our Multi-source Order-Preserving symmetric encryption (MOPSE) scheme; (3) QueryGenP generates its data queries; and (4) FilesDec decrypts the encrypted PHRs returned by the cloud server.

Cloud server: (1) IndexTrans merges multiple encrypted MDBT indexes and then segments the merged index into two indexes: a S-index and an H-index; (2) PrivacyQuery explores the S-index and H-index to process the encrypted queries and then returns the corresponding PHRs to data users or data providers.

C. ***MULTI-SOURCE ORDER-PRESERVING SYMMETRIC ENCRYPTION***

We propose a novel MOPSE scheme for each data provider to encrypt the MDBT index to enable later merging at the cloud server. MOPSE is adapted from Order-Preserving Symmetric Encryption (OPSE) [14]. To introduce OPSE and MOPSE, we first introduce the following definition.

Definition 1. For D_1, $D_2 \subset \mathbb{N}$ with $|D_1| \leq |D_2|$, a function f: $D_1 \rightarrow D_2$ is order-preserving (aka. strictly increasing) if $\forall i, j \in D_1$, $f(i) > f(j)$ iff $i > j$.

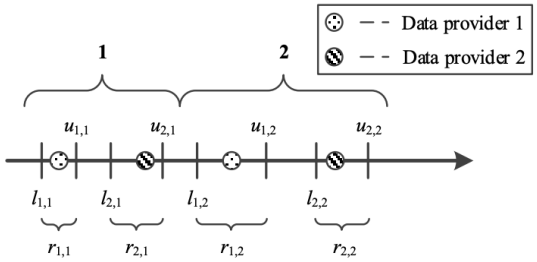

FIGURE 4: An example of data range $R=\{R_1, \cdots, R_n\}$. All ciphertexts for 1 locate at the left of that for 2.

Consider a deterministic encryption scheme {K, Enc(K, •), Dec(K, •)} with the key space K, the plaintext space D, the ciphertext space R, the encryption function Enc(K, •), and the decryption function Dec(K, •). We call this scheme an OPSE scheme iff Enc(K, •) is an order preserving function from D to R for $\forall K \in K$. OPSE assumes a single data source with the same key K, so it is not applicable to our multi-source scenario in which different data providers use different keys to encrypt the MDBT indexes for the same data owner. More specifically, assume that two data providers use keys K_1 and K_2, respectively. We then have $Enc(K_1, d_1) < Enc(K_1, d_2)$ and $Enc(K_2, d_1) < Enc(K_2, d_2)$ for $\forall d_1 < d_2$. But $Enc(K_2, d_2) - Enc(K_1, d_1)$ can be zero, positive or negative. We thus propose MOPSE to preserve the order of ciphertexts encrypted with various symmetric keys. For the above example, $Enc(K_2, d_2)$ is larger than $Enc(K_1, d_1)$ in MOPSE.

D. *GENERATING ENCRYPTED INDEX*

In the following subsections, we depict how the MEIM mechanism runs by an example in Fig. 5. First, data providers generate plaintext indexes with a specific structure MDBT, e.g., indexes A and B in Fig. 5. Subsequently, data providers utilize MOPSE to generate encrypted indexes $Enc(K_1, A)$ and $Enc(K_2, B)$. Here, each label $I_{i,d}$ is represented by $\{M_{i,d} \| P_{i,d}\}$. For instance, the label $I^B_{,1}$ of the value 1 for data provider B is $\{M^B_{,1} \| 30\}$, where $M^B_{,1}$ is a prefix family (each prefix is a 128-bit hash key) and $P^B_{,1}$ is 30. Finally, all these encrypted indexes are outsourced to the cloud server.

E. *TRANSFORMING ENCRYPTED INDEXES*

Upon receiving encrypted indexes, the cloud server recursively merges them from the root to the leaf nodes. In this process, MEIM compares the values of $P_{i,d}$ to determine the relative order of nodes within the merged index. For example, the merged index Z in Fig. 5 is formed from Enc(K_1, A) and Enc(K_2, B). Since $P_{A,1} = 15$ in Enc(K_1, A) is less than $P_{B,3} = 70$ in Enc(K_2, B), the entry $I_{A,1}$ appears before $I_{B,3}$ in index Z. The cloud server then splits the merged index Z into two separate indexes: the S index and the H index. In the S index, each $M_{i,d}$ retains the same value as in Z, whereas in the H index, each $h_{i,d}$ is generated by encoding the corresponding $P_{i,d}$ value from Z using the OPMPH function H_s. This function satisfies Definition 7 and is constructed using the shared key K_{CS}.As previously illustrated, the labels $I_{A,3}$ and $I_{B,3}$ in index Z are represented as $\{M_{A,3} \parallel 75\}$ and $\{M_{B,3} \parallel 70\}$, respectively. Therefore, the ordering of $M_{A,3}$ and $M_{B,3}$ in the S index follows the same order as in Z. If $H_s(70) = 78$, then $h_{B,3}$ in the H index becomes 78.

F. *PRIVACY-PRESERVING QUERY*

After receiving the trapdoor Q_e issued by data owners, the cloud server iterative processes the query on H index from the root to the leaf nodes as shown in Alg. 1. If the query fails, the algorithm returns Null; Otherwise, the encrypted files. Likewise, for the trapdoor T_e from data provider, the matching is also an iterative as depicted in Alg. 1. Only the matching condition in Line 4 of Alg. 1 follows the Theorem 1, which is proof in [19].

Algorithm 1: PrivacyQuery

Input: (H, Q_e) or (S, T_e)
Output: Encrypted files

1 Extract 1st query attribute value in Q_e or T_e as or $\tilde{t}$;
2 Remove 1st query attribute value in Q_e or T_e;
3 **for** Traverse each element χ in H or S index from left to right **do**
4 **if** $\chi.P \in \tilde{q}$ or $\chi.M \cap \tilde{t} \neq \varphi$ **then**
5 **if** χ belongs to a Leaf node **then**
6 return Encrypted file;
7 **else**
8 PrivacyQuery(H.childnode, Q_e) or PrivacyQuery(S.childnode, T_e);

V. CONCLUSION AND FUTURE SCOPE

In this paper, we explore the problem of privacy-preserving query for multi-source in the cloud-based PHR environment. Different from prior works, our proposed MEIM mechanism enables authenticated data owner to achieve secure, convenient, and efficient query over multiple data providers' data. To implement the efficient query, we introduce MDBT as the data structure. To reduce the overhead of query generation of data owner, and allow the cloud server to securely query, we propose a novel multiple order-preserving symmetric encryption (MOPSE) scheme. To make our model more practical, we propose an enhanced multiple order-preserving symmetric encryption (MOPSE+) scheme to satisfy the hierarchical authenticated query. Moreover, we leverage rigorous security proof to prove that our schemes are security.

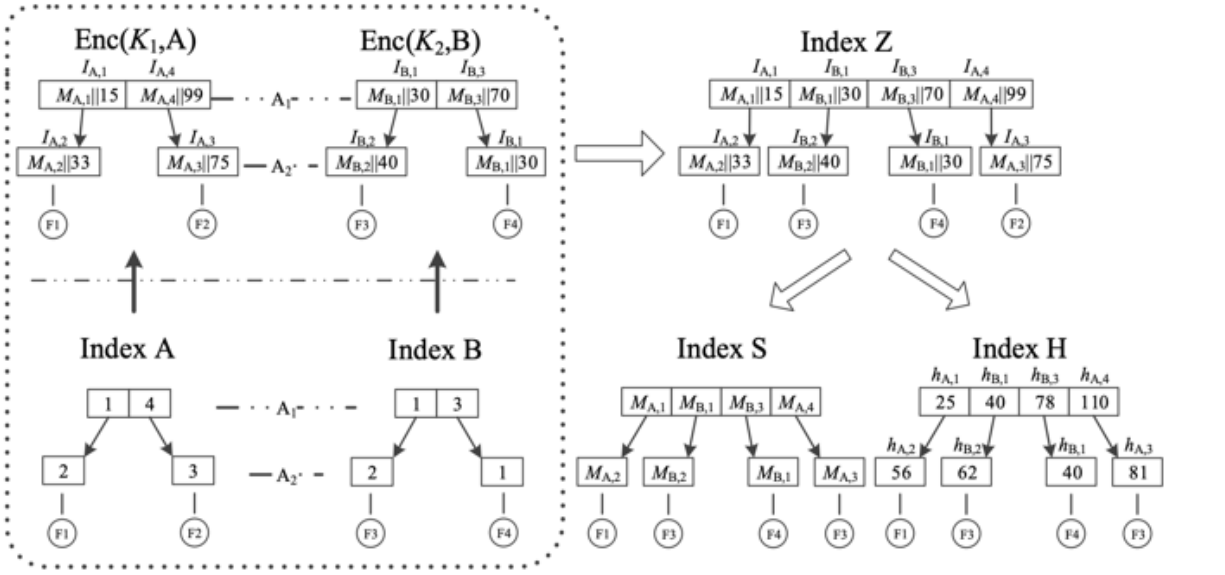

FIGURE 5: An example of MEIM. The dashed part is processed by two different data providers, who generate indexes A and B, and encrypt indexes to Enc(K_1,A) and Enc(K_2,B). The cloud server runs the other part, e.g., merging indexes Enc(K_1,A) and Enc(K_2,B) to Z and segmenting Z into two indexes S and H. Here $I_{i,d} = \{M_{i,d}||P_{i,d}\}$ denotes the ciphertext of the value d for index i.

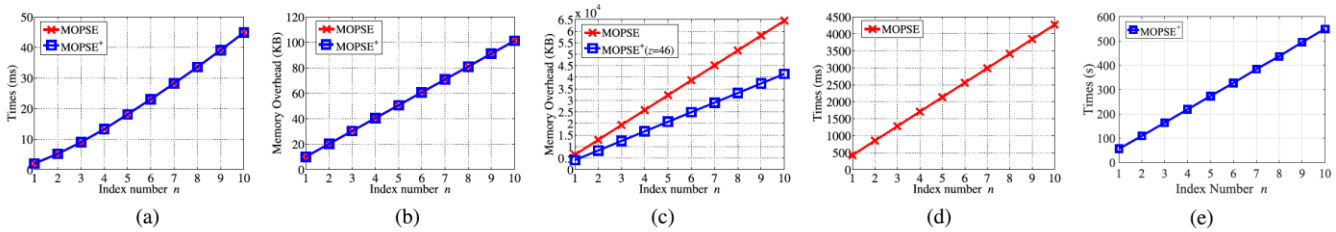

FIGURE 6. Fig. 6a and 6b show the index generation time and memory overhead, respectively. The memory overhead of encrypted indexes is represented in Fig. 6c. Fig. 6d and 6e depict the encrypting overhead under MOPSE and MOPSE$^+$. (a) Index generation time. (b) Index memory overhead. (c) En-index memory overhead. (d) Encryption under MOPSE. (e) Encryption under MOPSE$^+$.

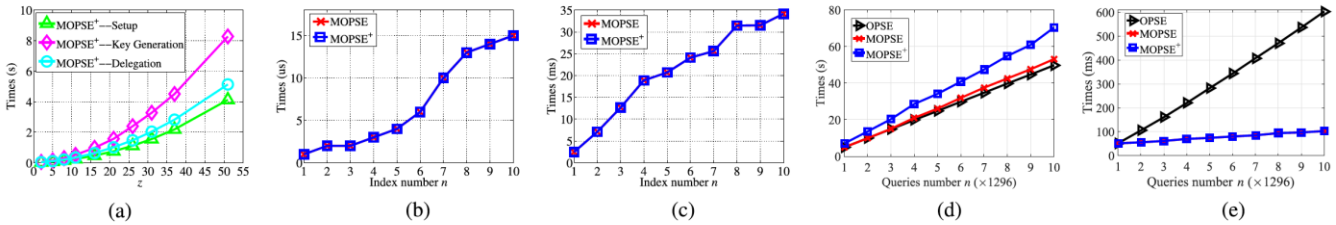

FIGURE 7. Fig. 7a shows the time of setup, key generation and delegation generating in MOPSE$^+$. The transforming overhead includes indexes merging time (Fig. 7b) and index segmenting time (Fig. 7c). Figure 7d and 7e depict the query consumption for data providers/owners under OPSE, MOPSE and MOPSE$^+$, respectively. (a) Time overhead in MOPSE$^+$. (b) Indexes merging time. (c) Index segmenting time. (d) Query of data providers. (e) Query of data owner.

Finally, we demonstrate that the MEIM mechanism is computationally efficient by implementing our schemes and running in a real dataset.

Al thought our work only focuses on CB-PHR system, it can be theoretically extended to various scenarios, mobile data collection, recommendation system, and so forth. However, the devices, like mobile devices, have limited computation and memory resource. For all this, we will discuss lightweight schemes in our future work.

VI. REFERENCES

[1] C. Wang, B. Zhang, K. Ren, J. Roveda, C. Chen, Z. Xu, "A privacy-aware cloud-assisted healthcare monitoring system via compressive sensing," in INFOCOM'14, Toronto, Canada, 2014.

[2] J. Sun, X. Zhu, C. Zhang, Y. Fang, "HCPP: Cryptography based secure ehr system for patient privacy and emergency healthcare," in ICDCS'11, Minneapolis, Minnesota, 2011.

[3] M. Li, S. Yu, N. Cao, W. Lou, "Authorized private keyword search over encrypted data in cloud computing," in ICDCS'11, Minneapolis, Minnesota, 2011. [4]

[4] J. Benaloh, M. Chase, E. Horvitz, K. Lauter, "Patient controlled encryption: ensuring privacy of electronic medical records," in: ACM workshop on CCS'09, New York, NY, 2009.

[5] M. Li, S. Yu, Y. Zheng, K. Ren, W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," IEEE T Parall Distr., vol. 24, no. 1, pp. 131 – 143, 2013.

[6] M. Li, S. Yu, K. Ren, W. Lou, "Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multiowner settings," in SecureComm'10, Singapore, 2010.

[7] X. Ma, Y. Zhu, X. Li, "An efficient and secure ridge regression outsourcing scheme in wearable devices," Computers & Electrical Engineering, 2017, DOI: 10.1016/j.compeleceng.2017.07.019.

[8] J. Liu, X. Huang, J. Liu, "Secure sharing of personal health records in cloud computing: ciphertext-policy attribute-based signcryption," Future Gener Comp Sy., vol. 52, pp. 67 – 76, 2015.

[9] P. Scheuermann, M. Ouksel, "Multidimensional B-trees for associative searching in database systems," Inform Syst., vol. 7, no. 2, pp. 123 – 137, 1982.

[10] K. Xue, J. Hong, Y. Xue, D. Wei, N. Yu, P. Hong, "CABE: A New Comparable Attribute-based Encryption Construction with 0-Encoding and 1-Encoding," IEEE Trans Comput., vol. 66, no. 9, pp. 1491 – 1503, 2017.

[11] K. Xue, S. Li, J. Hong, Y. Xue, N. Yu, P. Hong, "Two-Cloud Secure Database for Numeric-Related SQL Range Queries with Privacy Preserving," IEEE Trans Inf Forensics Secur., vol. 12, no. 7, pp. 1596 – 1608, 2017.

[12] R. Curtmola, J. Garay, S. Kamara, R. Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient constructions," in CCS'06, Alexandria, VA, 2006.

[13] Y. Zhu, Z. Huang, T. Takagi, "Secure and Controllable k-NN Query over Encrypted Cloud

Data with Key Confidentiality," J Parallel Distr Com, vol. 89, no. C, pp. 1 – 12, 2016.

[14] D. Song, D. Wagner, A. Perrig, "Practical techniques for searches on encrypted data," in IEEE S&P'00, Berkeley, CA, 2000.

[15] D. Boneh, G. Crescenzo, R. Ostrovsky, G. Persiano, "Public key encryption with keyword search," in EUROCRYP'04, Interlaken, Switzerland, 2004.

[16] Y. Zhu, Z. Wang, Y. Zhang, "Secure k-NN Query on Encrypted Cloud Data with Limited Key-disclosure and Offline Data Owner," in PAKDD'16, Auckland, New Zealand, 2016.

[17] B. Iyer, S. Mehrotra, E. Mykletun, G. Tsudik, Y. Wu, "A framework for efficient storage security in RDBMS," in EDBT'04, Heraklion, Crete, Greece, 2004.

[18] Q. Liu, C. C. Tan, J. Wu, G. Wang, "Efficient information retrieval for ranked queries in cost-effective cloud environments," in INFOCOM'12, Orlando, FL, 2012.

[19] Y. Zhu, Z. Wang, J. Wang, "Collusion-Resisting Secure Nearest Neighbor Query over Encrypted Data in Cloud," in IWQoS'16, Beijing, China, 2016.

[20] R. Agrawal, J. Kiernan, R. Srikant, Y. Xu, "Order preserving encryption for numeric data," in SIGMOD'04, New York, NY, 2004.

[21] A. Asuncion, D. Newman, "UCI machine learning repository," 2010.

[22] Q. Liu, C. C. Tan, J. Wu, G. Wang, "Cooperative private searching in clouds," J Parallel Distr Com, vol. 72, no. 8, pp. 1019 – 1031, 2012.