CS498 - Mini Project II

Study on privacy preserving techniques of cloud computing

Name of the author:

Anjuru Lokesh(18BCS006)

Ch N V Avinash(18BCS021)

Diddi Geetha Krishna(18BCS025)

Instructor:

Dr. Malay Kumar

Department of Computer Science and Engineering

Indian Institute of Information Technology Dharwad

Nov 30, 2021

Contents

List of Tables

List of Figures

1 Introduction

In reality, patients always encrypt their EHRs(Electronic Health Records) before outsourcing, making the range query impossible. In cloud-based electronic healthcare (eHealth)[1] systems, range query is an important data search technique. It enables authorised doctors to retrieve target electronic health records (EHRs) that are outsourced by patients from the cloud server. Here, I will explain some of the threats and algorithms to overcome those threats in cloud security to enable range queries over encrypted EHRs from multiple patients. The MSOPE scheme is IND-MSOCPA secure, according to security analysis. (Ordered plaintext attack from multiple sources)

However the possibility of a risk of patients Electronic Health Records (EHRs) weakens the interest in using the Cloud-based eHealth system. Patients will be concerned about illegal usage of their EHRs stored in the public cloud because a cloud server is always considered a semi-trusted party. Doctors will also be concerned about privacy breaches while examining EHRs on a semi-trusted cloud server, because searching activities may expose EHR content. As a result, while adopting a cloud-based eHealth system, the content of EHRs should be protected.

Order-preserving encryption (OPE) allows for efficient range queries on encrypted data, which strikes a balance between speed and security. Order-preserving encryption has been frequently utilised in encrypted databases due to the fact that the ciphertext and plaintext are in the same order. For a single data source situation involving a data provider and a cloud server, many OPE techniques have recently been presented. The majority of the publications are the-

oretical attempts to push security concepts to their limits, such as indistinguishability under an ordered selected plaintext attack (IND-OCPA). Single data source approaches, on the other hand, are incompatible with cloud-based eHealth systems because they cannot allow privacy-preserving range searches across EHRs from numerous patients, which is a critical feature of cloud-based eHealth systems.

We start by identifying three real-world challenges to cloud-based eHealth systems: privacy leakage, frequency analysis, and same data inference. We introduce a security concept named indistinguishability under multi-source ordered chosen plaintext attacks to capture the security that thwarts these threats (IND-MSOCPA). Then, for cloud-based eHealth systems, we offer a multi-source order-preserving encryption (MSOPE) technique that allows doctors to perform privacy-preserving range queries across encrypted EHRs from multiple patients. This work builds on our prior research by improving encryption efficiency and adding capability for privacy-preserving range queries.

The following is a summary of our contributions:

- We provide the IND-MSOCPA security concept, which includes the protection against threats in real cloud-based eHealth systems, such as privacy leakage, frequency analysis, and identical data inference.
- For cloud-based eHealth systems, we introduce the MSOPE technique, which allows doctors to conduct range queries across outsourced ciphertexts (i.e., encrypted EHRs). The MSOPE technique is based on a safe comparison protocol that uses a small number of homomorphic encryption operations, which enhances its computing efficiency greatly.

- The MSOPE system is IND-MSOCPA secure, as shown by a formal security proof. Extensive testing has shown that the MSOPE approach is more efficient in terms of computational overhead than the original version of this work.

2 Literature Survey

Nowadays , everything everywhere is changing to the cloud. So while it comes to the Medical department, when we use cloud for medical as present technology has many flaws there is a privacy risk. Means the reports that is Electronic Health Records (EHRs) were stored in cloud and it was authorised by a user that is patient and it was given access to the doctor by the permission of the user. Meanwhile, authorized doctors can perform queries on EHRs[2] provided by hundreds of thousands of patients, and later collect qualified biomedical data to build a diagnosis model.As an important technique for collecting qualified biomedical data, range query returns a set of interesting EHRs between an upper bound and a lower bound, helping doctors to investigate specific disease. A cloud server is always considered as a semi-trusted party and therefore patients will worry about the unauthorized use for their EHR in the public cloud.Here,I will focus on some of the threats in the cloud and how to overcome those threats. So, in this article, I'm going to talk about the vulnerabilities

that patients face when it comes to their Ehrs, as well as the remedies to those concerns, because normal cloud has some risk of data loss. Patients can encrypt and upload data to the cloud, and the key to decrypt it will be stored with them. They can share the key with their doctor and authorise them as users, but after giving access to anyone else, they can re-encrypt their data using our proposed algo-

rithms and upload it to the cloud again.This way, you can prevent EHRs from being decrypted or guessed by others because we use different randomised encrypted ciphertext for the same text and also because we use 32 and 64 bit encryption technologies, which makes it difficult to guess, and after decrypting according to our proposed algorithm, it decrypts and returns the file in excel format.

The Cloud will be unable to access any patient or user information since it is stored in encrypted ciphertext, which it cannot decrypt without the user's permission because it requires keys to decipher the encrypted ciphertext. So, using our encryption, we can try to fix some problems, and users may also grant doctors access to their EHRs, but doctors will not be able to decrypt it in the cloud unless they have the key, or they will be able to decrypt and examine their EHRs using patients' devices. Patients can download and change the key in any way they like by uploading it to the cloud again. This will make it more secure.

We will use index searching because it is completely encrypted to get the correct information from the cloud. This index can be changed whenever the user reuploads the files, and this key will be stored alongside the user encrypted keys, so the user will not have to worry about the key separately because it will be present in the key files given to them. This ensures cloud security, privacy, and many other benefits. I briefly covered the process of using our suggested algorithm in this article, and by reading it, you will gain an understanding of the complete challenges, the algorithms we are using, and the answers, as well as the implementation of our end algorithm.

Multi-source order-preserving encryption techniques have been employed in cloud-based eHealth systems to enable numerous patient

cases. Multi-source order-preserving encryption techniques secure EHR contents while also allowing for quick range queries across several encrypted EHRs. Most existing multi-source techniques are feasible, but they leak both plaintext value and distance between any two plaintexts. Meanwhile, these systems are vulnerable to frequency analysis.

Let's take a quick look at the algorithms we're employing in our proposed approach. After decryption, we use Privacy-Preserving Range Query Schemes to conduct queries, and for encryption, we use MOSPE+ algorithms to preserve the data. And we are using some of the standard encryption algorithms, and by reading our paper, you will have a clear understanding of the problem, the users, and who is included, as well as what data is stored, as well as the traditional algorithms' advantages, definitions, and disadvantages, as well as our proposed algorithms and the implementation results we obtained. As a result, after reading this text, you will have a thorough understanding of everything.

3 Privacy-Preserving Range Query Schemes

There are two privacy-preserving range query schemes in cloud-based eHealth Systems:

(1) privacy preservation, i.e., unauthorized users cannot get the content of EHRs; (2) practicality and potency, i.e., approved users will perform economical vary question over the protected EHRfunctionality and efficiencies. Order-preserving encryption (OPE) permits economical vary question on encrypted knowledge, that balances the potency and security. For one data-source scenario as well as an information supplier and a cloud server, OPE techniques are

bestowed. Single knowledge supply schemes, on the opposite hand, privacy preserving incompatible with cloud-based eHealth systems as a result of they're unable to support privacy-preserving vary queries across EHRs from multiple patients, that could be a essential feature of cloud-based eHealth systems.

3.1 DisAdvanatges

Multi-source (or abbreviated as multi-provider or multi-user) order-preserving encryption schemes are employed in cloud-based eHealth systems. These schemes not solely defend the privacy of EHR contents however conjointly change economical question over them. The majority of today's multi-source systems focus on utility, however they leak both plaintext value and distance between any two plaintexts.

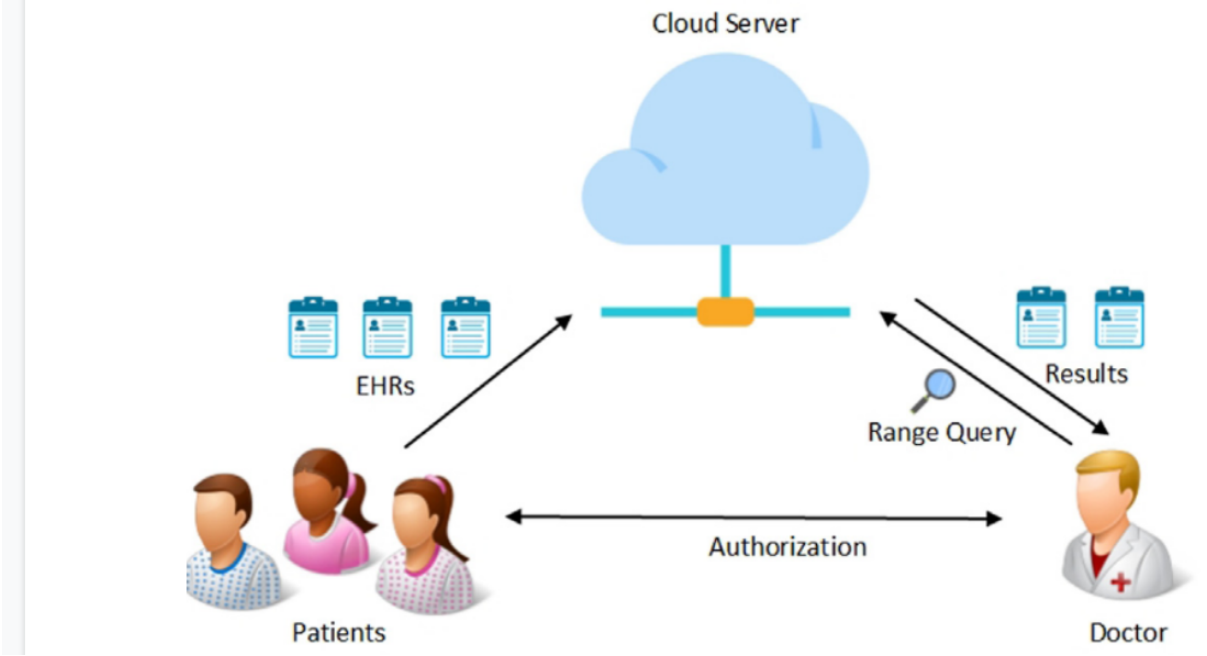


Figure 1: Architecture

3.2 Algorithms used in Privacy Preserving Range Query Schemes

3.2.1 Symmetric Key Encryption:

A Symmetric key encryption contains 3 polynomial time algorithm. They are:

- SKE Generation
- SKE Encryption
- SKE Decryption

3.2.2 Homomorphic Encryption(HOM):

Homomorphic encryption is a type of encryption that allows users to operate on encrypted data without having to decrypt it first. These computations are then encrypted, resulting in an output that is identical to what would have been produced if the operations had been done on unencrypted data. For privacy-preserving outsourced storage and computing, homomorphic encryption can be applied. This makes it easy to encrypt data before sending it to commercial cloud environments for processing.

A Homomorphic Encryption contains 3 polynomial time algorithm. They are:

- HOM Generation
- HOM Encryption
- HOM Decryption

We use additive homomorphic encryption in MSOPE scheme.

3.2.3 Secure comparing Protocol:

The secure comparing protocol is a two-party secure computation process that involves two patients, Pi and Pj. To generate it, we use the Paillier cryptosystem [3] (an additive homomorphic encryption). The secure comparing protocol takes two EHR ciphertexts encrypted by SKE as input, wi,x and wj,y, and outputs the comparison result R, which is one bit size and indicates if wi,x is less than wj,y. Algorithm 1 specifies the secure comparison procedure.

3.2.4 Secret State Generation:

The secret state generation algorithm generates the secret state T and the ciphertext domain M. This algorithm generates a empty AVL Tree T as Secret State.

3.2.5 Encryption:

For EHR wi,x provided by Pi, the MSOPE encryption algorithm gives an MSOPE ciphertext ci,x. Tree Node Construction, Secret State Update, and Ciphertext Generation are the three algorithms that make up the MSOPE encryption algorithm. This algorithm invokes Tree node construction algorithm, Secret state update algorithm, Cipher text generation algorithm.

3.2.6 Tree node construction algorithm:

In the tree node construction algorithm, the patient uses his SKE key to encrypt wi,x, and later generates the tree node message ti,x. This algorithm generates the message as an AVL Tree node.

3.2.7 Secret state update algorithm:

Secret State Update algorithm adds a node to the AVL Tree if the node is not already present in the tree.

3.2.8 Cipher text generation algorithm:

In the ciphertext generation algorithm, the cloud server generates order-preserving ciphertexts for EHRs stored in T . The ciphertexts are generated from T and the ciphertext domain M. We use recursion to generate the orderpreserving ciphertexts for EHRs in T . We use Min and Max to denote the lower bound and the upper bound of recursion respectively. The ciphertext generation algorithm returns the corresponding order-preserving ciphertext of wi,x for Pi.

3.2.9 Decryption:

The MSOPE decryption algorithm is used for Pi to decrypt his encrypted EHR ci,x.

3.2.10 Range Query:

Using secure comparing protocol and SKE Encryption and Decryption we can peform a Range Query over EHRs. This algorithm returns a set of Encrypted nodes which can be decrypted using SKE algorithms by the doctor.

4 Attacks

Considering the Honest-but-Curious (HbC) model, which has been widely adopted in privacy-preserving search work in cloud computing.The HbC adversary will follow the protocol honestly but be curious about the data content.

Patient: We assume that each patient in our scheme is HbC adversary, Someone is only allowed to see his or her own EHRs but is inquisitive about what other patients' EHRs contain.

Cloud Server: Meanwhile, the cloud server is viewed as a HbC adversary, as it is interested in the content of patients' EHRs and the doctor's inquiry.

Doctor: Because he or she is authorised to run range queries on patients' EHRs, the doctor is deemed an honest user.

4.1 Privacy Leakage

Privacy leakage means that a cloud server may recover the encrypted EHRs to the plaintext form if the encrypted EHRs leak additional information other than the order Information. Since AES encryption is a pseudo-random function .Assume that c 1 and c 2 are equal. Then the fixed interval [l i , d , u i , d] is partially leaked, because [c1,c2] is subset of [l i , d , u i , d] . Note that the fixed interval would be fully leaked when c 1 = l i , d and c 2 = u i , d . Because l I d and u I d would be reused multiple times, the adversary might recover the ciphertext c * to the plaintexts form d if c [c 1, c 2].

4.2 Frequency analysis

Frequency analysis implies that a cloud server could analyze the distribution of encrypted EHRs and additional recover the encrypted EHRs to the plaintext type as a result of some order-preserving cryptography schemes reveal the distribution info of EHRs and therefore the content of EHRs aren't distributed uniformly. The encrypted

field “gender” of EHRs could also be recovered by the cloud server as a result of the encrypted EHRs reveals the distribution of “gender” and therefore the cloud server learns that the keyword “female” seems additional oft than “male” in an exceedingly medical specialty hospital.

4.3 Identical Data Inference

Unauthorized patients may attempt to extract the information of EHRs by searching identical encrypted EHRs. Assume that Alice and Bob are both 38 years old, and that the OPE encryption for the number ”38” is ”357”. As a result, both Alice and Bob’s encrypted ”age” fields are ”357”.

5 The MSOPE Scheme

5.1 Overview Of the MSOPE Scheme

The MSOPE theme is meant for privacy-preserving vary question on encrypted outsourced EHRs. The work-flow of privacy-preserving vary question involves 2 phases,

EHRs outsourcing : during this section, Cs invokes the key State Generation formula to get Associate in Nursing empty AVL tree, and fix the ciphertext domain. Then, every patient P i belongs to P invokes the secret writing formula to inscribe his/her EHRs, and later uploads the encrypted EHRs to Cs .

EHRs querying : during this section, there square measure 2 completely different knowledge users. On the one hand, D invokes the vary question to inscribe the higher limit and lower limit of EHRs so obtains the desired encrypted EHRs. Later D invokes the decoding formula to decode the desired EHRs. On the opposite hand, PI

might question his/her EHRs and invoke the decoding formula to decode the outsourced encrypted EHRs.

5.2 Construction Of the MSOPE Scheme

The MSOPE[3] system is based on a multi-source randomised order that randomly permutes the order of identical data from various patients. We create multi-source randomised order by covertly building T and then generating order-preserving ciphertexts for EHRs gathered from different patients based on T. Secret State Generation, Encryption, Decryption, and Range Query are all polynomial-time algorithms in the MSOPE scheme.

6 Multi-source Encrypted Indexes Merge

MEIM is a new method that allows a cloud server to combine several encrypted data indexes from different health providers for the same patient without compromising on the patient's privacy. It also allows the user to create a single encrypted query that searches all of his health providers' encrypted data on the cloud server. MEIM is built around a novel Multi-source Order-Preserving Symmetric Encryption (MOPSE) primitive. The order of various data indices encrypted by separate symmetric keys is preserved by MOPSE.

7 Proposed model with Advanced Security

7.1 The problem of privacy-preserving query for multi-source

The 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, we propose a novel multiple order-preserving symmetric encryption (MOPSE) scheme. To make the model more practical, we propose an enhanced multiple orderpreserving symmetric encryption (MOPSE +) scheme to satisfy the hierarchical authenticated query.

8 MOPSE+

We can construct a hierarchical structure for data providers with the basic solution MOPSE, which involves issuing all keys of data providers with lower privileges to those with higher privileges. The technique, however, is inefficient, because the overhead of key management grows linearly with the number of data providers with lesser privileges, and data providers must generate a large number of queries (the number is proportional to the number of providers). We presented an upgraded system (called MOPSE +) based on our earlier scheme to combat these inefficient difficulties

Our system uses a novel Multi-source Encrypted Index Merging (MEIM) technique, which allows the cloud server to combine encrypted data indexes from different data providers for the same data owner without having to decrypt the data. Indices that are unique to each person. MEIM allows a data user to submit a single data query for all of the data owner's PHRs from all of the data providers. In what follows, we introduce the basics of indexing and querying PHRs, followed by the MEIM design.

9 Algorithms

Algorithm is divided into :

- Indexing and Querying Ehr

- Overview of MEIM
- Continuous investment in cybersecurity and security technology.
- Multi-Source Order-Preserving Symmetric Encryption.(MSOPE)
- Generating Encrypted Index

PrivacyQuery: If and only if the following formula is valid, given a query range [a,b] and an attribute value d, d[a,b]

10 Implementation and Results

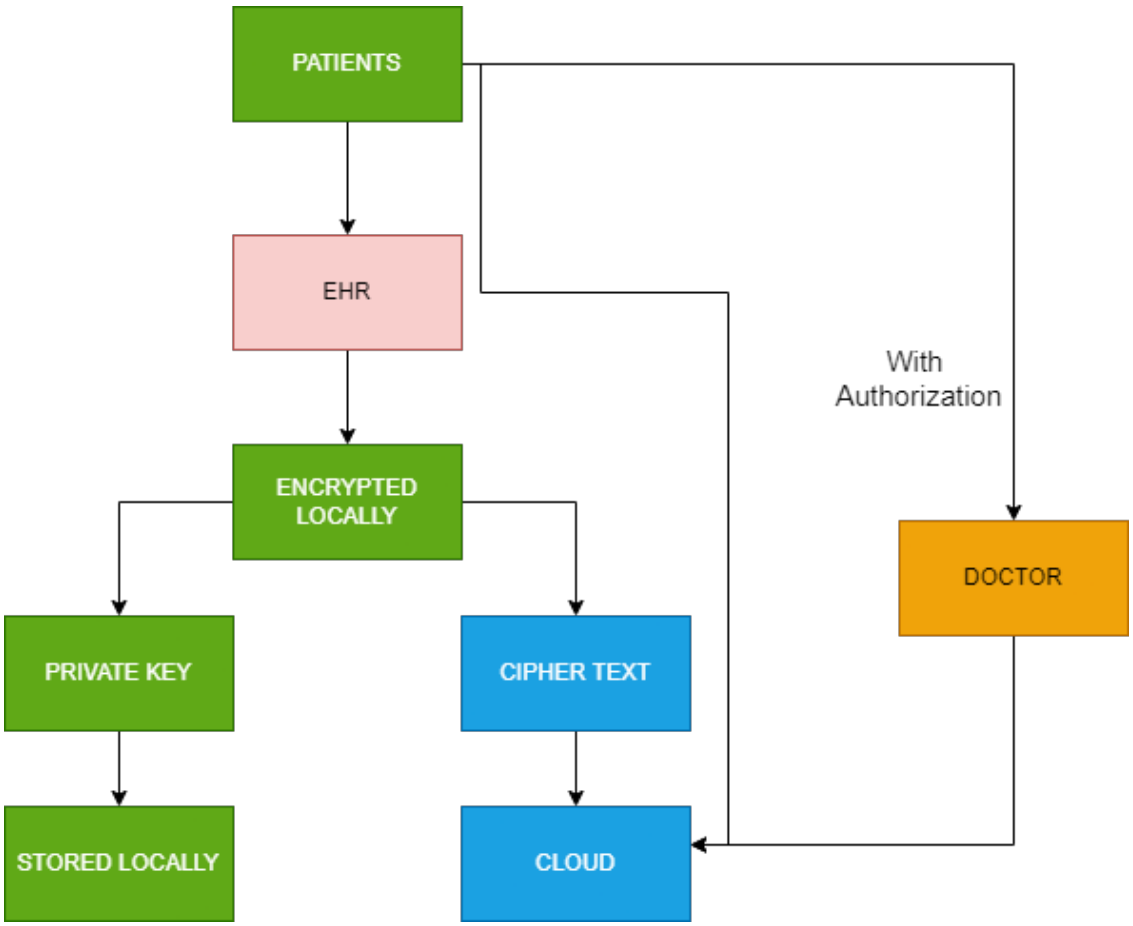


Figure 2: Implementation Flowchart

We have tested and implemented our proposed algorithm, we have used fernet encryption algorithm to implement randomized and secured encrypted cipher and we are storing the encrypted cipher text in string format but the resulted encrypted text will be in byte format so after downloading the encrypted text to decrypt we will first encode downloaded string into bytes again and then we will decrypt the stored values and we will convert it to an excel file.

Here we are taking an EHR in xlsx format and we are encrypting each row value by taking the header as column value and firstly we are creating a dictionary and then we are creating two more dictionaries one is encrypted text and the other is key to decrypt and the key will be given out to user as json format and the other encrypted cipher text will be stored as json but in mongo db database as mongo db database uses dictionaries or json format it will be easy to download and decrypt so in this way we have implemented our proposed algorithm and we are successful in integration with uploading the encrypted files to db and then download again to decrypt it. Here the keys must be carefully stored by users so that in the keys when they want to download it we have also embedded the ids so this ids we use to search and get the perfect match to download uploaded file. And the people who are having the keys can download and decrypt and check and when they again re-upload the files their ids and encrypted text and keys will be changed in this way it will be helpful in giving more security to users. Below are some snapshots to see the results of our implementation.

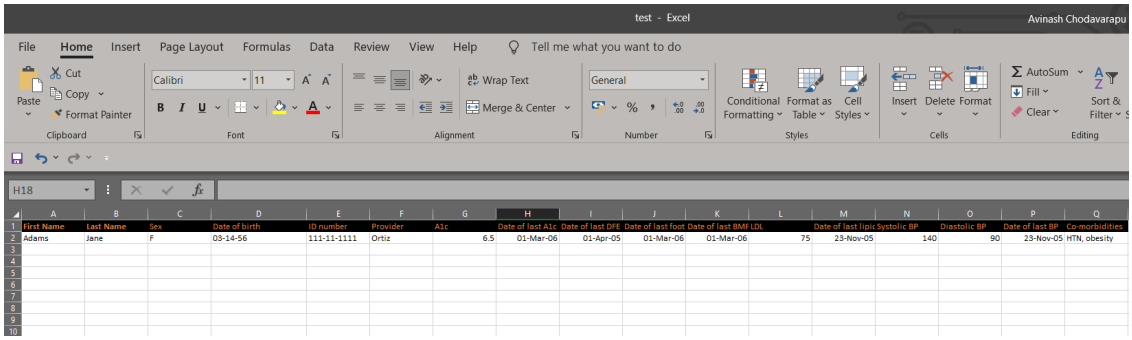


Figure 3: Input file Image

This is the image input excel file which we are taking to encrypt the file and after taking the file we are making it as input python dictionary by using header of excel as key in dictionary and value as the value of the column in excel file. After Converting it into dictionary we are encrypting it and store the encrypted details as two json files one is of encrypted cipher text and the other is of key to decrypt the encrypted cipher text.

Figure 4: Results after encrypting and decrypting data

After Encrypting the data we are also generating a randomised id which is unique and adding it to two generated files. As we use this id to download file from database and it helps to identify the exact encrypted file of user. So After adding id to the files we save the key locally in users computer and now we upload the file which contains encrypted cipher text to database or cloud. In this way no one can find out the file details even who is the owner of the file as there will be no identity revealed to cloud or any other user as it contains complete encrypted text and only ids.

Now if the user want the file we just ask for key and using key file as we already store id in it we take id and search the file using the id in key and we fetch it and download it to local cloud and decrypt it and we even after decrypting we make it as excel file which we have taken as input. So in this we losslessly transfer files and also save the privacy of the user.

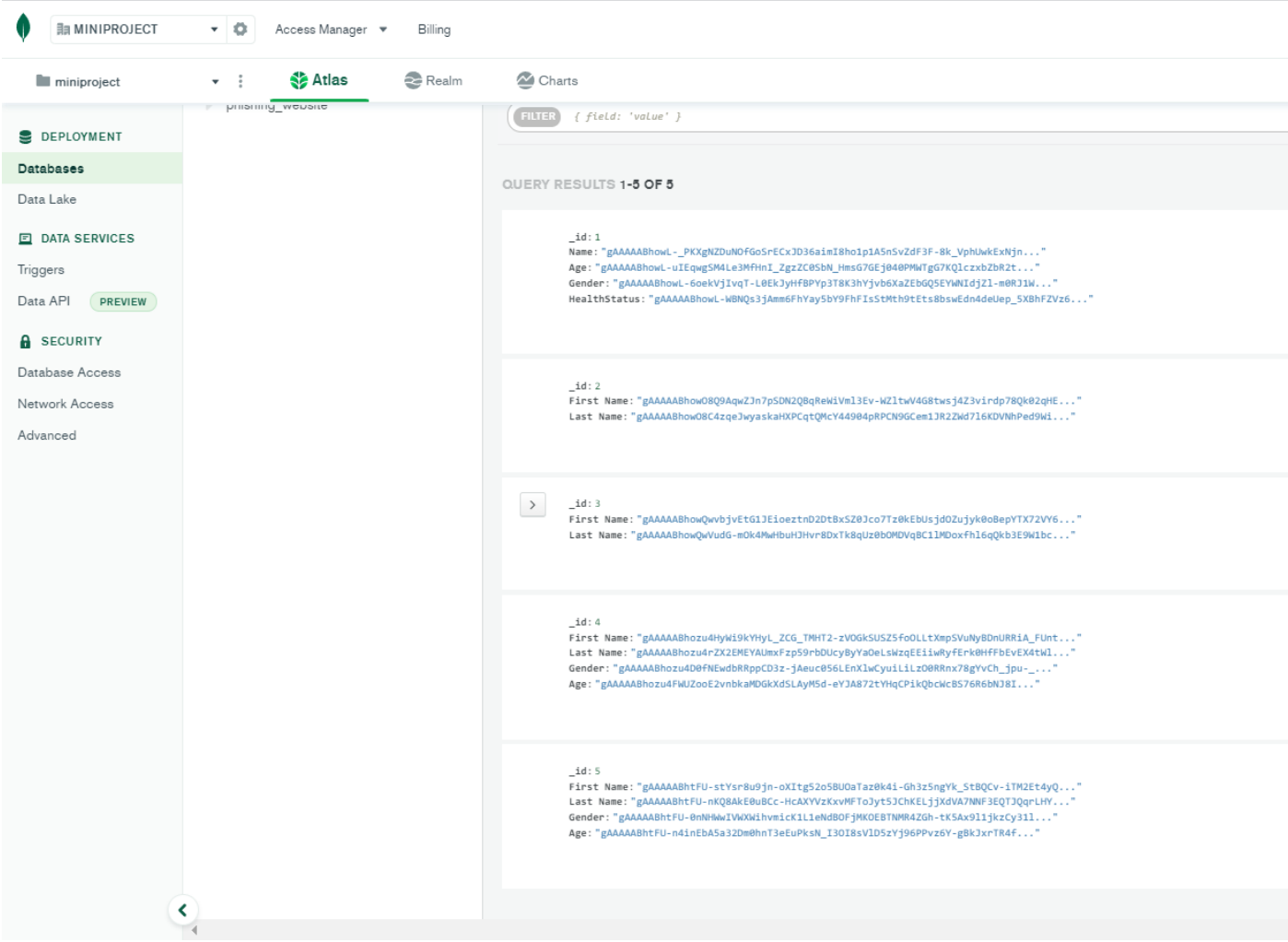
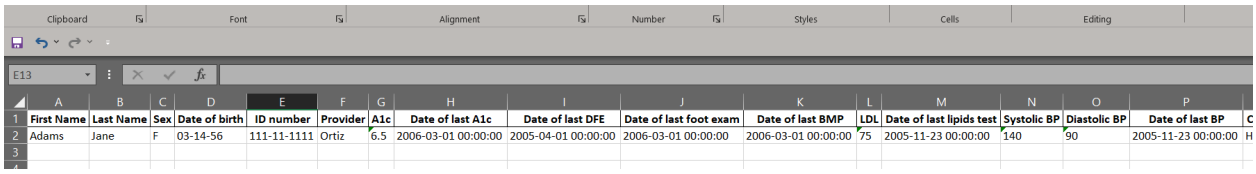


Figure 5: Files in Database(Cloud)

Here is the image of the data base where we store the encrypted details of the user and here we can see that we cant identify anything untill and unless we know the id and even after knowing id without having keys we cant download and decrypt it. As we are using AES 128Bit Encryption algorthims.



First Name	Last Name	Sex	Date of birth	ID number	Provider	A1c	Date of last A1c	Date of last DFE	Date of last foot exam	Date of last BMP	LDL	Date of last lipids test	Systolic BP	Diastolic BP	Date of last BP
Adams	Jane	F	03-14-56	111-11-1111	Ortiz	6.5	2006-03-01 00:00:00	2005-04-01 00:00:00	2006-03-01 00:00:00	2006-03-01 00:00:00	75	2005-11-23 00:00:00	140	90	2005-11-23 00:00:00

Figure 6: Output file Image

This is the output image and we can see that the image is an excel file and it has all the details of the user and it has decrypted data. So in this way our algorithm protects users privacy and solves the problems faced by user or patients for their privacy concerns about EHR.

11 Discussion and Conclusions

In this work, I identify three vulnerabilities in cloud-based eHealth systems: privacy leakage, frequency analysis, and identical data inference. I've also established the indistinguishability under multi-source ordered chosen plaintext attack (IND-MSOCPA)[4] security concept for multi-source order-preserving encryption to capture the security characteristics that protect against attacks.

I also proposed the MSOPE technique, which enables doctors in cloud-based eHealth systems to perform privacy-preserving range searches across outsourced EHRs from a large number of patients. I created a formal security proof to show that the MSOPE scheme is IND-MSOCPA[4] secure. Extensive performance testing has shown that the MSOPE method is more efficient than the original version.

In the implementation part,we have used Advanced Encryption Standard algorithm in combination of randomisation. AES generates 64 bit url safe encrypted cipher text and 32-64bit key to decrypt the cipher text.For each and every input randomized chipher text will be generated and overall AES is an 128 bit encryption algorithm .It uses 64 bit for the key encryption and 64 bit decryption .Using this combination of AES and randomized algorithms we can overcome the attacks like privacy leakage,frequent analysis and identical data integrity etc and also we can perform range queries over the data set easily and can analysis the results for creating diagnosis model.Overall it is a Ind-Msope safe algorithm to use in privacy preserving range query schemes in cloud security.

References

[1] . Huang, R. Lu, H. Zhu, J. Shao, and X. Lin, *FSSR: Fine-grained EHRs sharing via similarity-based recommendation in cloud-assisted ehealthcare system.* in: Proceedings of ACM ASIACCS, ACM, 2016.

[2] . Grubbs, K. Sekniqi, V. Bindschaedler, M. Naveed, and T. Ristenpart, *Leakage- abuse attacks against order-revealing encryption.* in: Proceedings of IEEE Security and Privacy (SP), IEEE, 2017.

[3] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, *Order preserving encryption for numeric data.* in: Proceedings of ACM SIGMOD, ACM, 2004.

[4] A. Boldyreva, N. Chenette, and A. O'Neill, *Order-preserving encryption revisited: Improved security analysis and alternative solutions.* in: Proceedings of CRYPTO, Springer, 2011.