Proceedings of the Estonian Academy of Sciences,
2020, **69**, 3, 266–276
https://doi.org/10.3176/proc.2020.3.09
Available online at www.eap.ee/proceedings

E-HEALTH

Secure e-health cloud framework for patients' EHR storage and sharing for Indian Government healthcare model

Indra Priyadharshini S[a*] and Vigilson Prem M[b]

[a] Department of Computer Science and Engineering, R.M.K. College of Engineering and Technology, R.S.M. Nagar, Puduvoyal, 601206 Tiruvallur District, India
[b] Department of Computer Science and Engineering, R.M.D. Engineering College, R.S.M. Nagar, Kavaraipettai, 601206 Tiruvallur District, India

Received 3 February 2020, accepted 28 March 2020, available online 16 July 2020

Abstract. Medical field is experiencing a huge paradigm shift from traditional healthcare model to electronic healthcare model. Cloud computing facilitates healthcare industry to provide continuous, on-demand services anytime, anywhere. Cloud computing facilitates management but it is also challenging to provide privacy and security in cloud computing. In this paper we propose a secure, privacy-preserving health cloud which allows data exchange between hospitals, healthcare centres, doctors and patients. To accomplish security and privacy, we implement homomorphic encryption (HE), which allows computations to be carried out on encrypted data without even decrypting them. To achieve secure sharing of data among authorized entities, proxy re-encryption (PRE) scheme is combined with homomorphic encryption. Our secure e-health cloud framework achieves performance improvement compared to the Paillier cryptosystem.

Key words: health cloud, homomorphic encryption, primary healthcare centre, health data analytics, health record exchange, cloud security.

1. INTRODUCTION

The healthcare industry is one of the largest service-oriented sectors in India in terms of revenue and employment. Health expenditure spent by the public sector is also on the rise. Deloitte Touche Tohmatsu India has forecasted that the Indian healthcare industry, worth about US$ 100 billion, is expected to rise at a CAGR of 23 per cent to US$ 280 billion by 2020 with increased digital adoption. The healthcare market can increase threefold to US$ 372 billion by 2022 [1]. So, government and healthcare institutions are working towards the innovative ways of delivering health care services efficiently with easing the payment methods. Cloud computing offers best solutions to address these challenges through unlimited, on-demand services anytime, anywhere. Cloud-based healthcare solutions also help hospitals and health centres to decrease their investments in infrastructure and also decrease maintenance costs. Health cloud environment can be extended (scaled up) by adding resources (like servers, storage) if the demand is increased or the resources can be maintained at the same level or revoked if the demand is lower (scaled-down), to fit a dynamic workload. Healthcare solutions built over cloud can also have inbuilt mechanisms like disaster recovery and redundancy to mitigate the failures and reduce the impact of business downtime. The health cloud acts as a central data repository through which efficient access and information sharing is achieved.

There are challenges that come along with the benefits of cloud computing. As the patient's data are sensitive when moved to the cloud, the security and privacy of the patient are in question. Privacy, in this paper, refers to the

* Corresponding author, indra.priyadharshini@gmail.com

right of the individual [26] to know what kind of information is disclosed about her/him in public, right to know what kind of information about them is stored, with whom the information is shared and how it is going to be used. For example, the patient's electronic health records (EHRs) are moved to the cloud by the doctor, by the hospital, or even by the patient itself. When these EHRs are accessed by the data owner (the one who pushes the EHRs to the cloud) or by other users for any computation, the EHRs have to be decrypted. So, the data is in danger while in transit. In case the computations are carried out by the cloud service provider (CSP) itself, the data owner has to trust the honest-but-curious CSP. Assuring secure and private transfer of EHRs to the cloud under public access is always a conundrum. The identity of the patients must also be preserved in the public domain [27]. According to the Health Insurance Portability and Accountability Act (HIPAA) [2], the Privacy Rule is to assure that individuals' health information is properly protected while allowing the flow of health information needed to provide and promote high quality healthcare and to protect the public's health and well being. The India's version of HIPAA act is proposed. It is called Health Data Privacy and Security Act (HDPSA). HDPSA is being worked out by the Health and Family Welfare Department of the Indian Government and is likely to take HIPAA of the USA as a model. Currently, ISO27799:2016 standard is the advisory standard for information security management in healthcare solutions for the Information Technology ACT 2000 (ITA-2000) and its amendments. Implementation of ISO 27799:2016 covers General Data Protection Regulation (GDPR). ITA-2000 has adopted several principles also from GDPR [25]. These laws and legal regulations emphasize the fact that the technological healthcare solutions must guarantee data privacy and anonymity of the patients.

In this paper we intend to build a framework for the e-health cloud which would securely store sensitive EHRs in the cloud by encrypting them. We implement homomorphic encryption (HE) to encrypt EHRs, which allows to carry out computations with the encrypted data. The computations made on the ciphertext produces an encrypted output, which in turn can be decrypted by the user or the data owner who holds the decryption keys. Homomorphic encryption (HE) is a type of encryption that allows computation on ciphertext, generating an encrypted result, which, when decrypted, matches the outcome of the operations as if they had been performed on the plaintext. The purpose of homomorphic encryption is to allow computation on encrypted data [3]. For secure sharing of EHRs among trusted entities, proxy re-encryption is used. Proxy re-encryption allows proxies to re-encrypt the ciphertext in such a way that it can be decrypted by another user without using the private key of the data owner [28].

2. RELATED WORKS

Cloud computing has gained much popularity and importance in recent years. It has become inevitable these days in any business model. Healthcare is one of the largest revenue generating industries in India. As the entire country is on the verge of digitization, providing healthcare services online is developing fast. This is easily achievable with the advantages of cloud computing but it comes at the cost of security and privacy as sensitive patient's data need to be shared on the public platform like the Internet.

There are plenty of schemes available to enforce security of the cloud. One of those schemes is homomorphic encryption (HE). Homomorphic encryption has been in the field of cryptography since 1978, when Rivest et al. first investigated the RSA algorithm. The concepts of homomorphism have been investigated for a long time to explore the homomorphic properties in various cryptographic algorithms and to find ways of designing any algebraically homomorphic encryption schemes. The first theoretical implementation of fully homomorphic encryption (FHE) [4] was built by Craig Gentry in 2009 in his PhD thesis. Gentry's homomorphic encryption scheme allows user to perform any operations on the ciphertext and output is also in the encrypted form. The encrypted result is decrypted to plaintext output with relevant key.

Acar et al. [5] discussed various homomorphic algorithms available in the literature. They also explained somewhat homomorphic encryption (SHE) algorithms and partial homomorphic encryption (PHE) which are building blocks for achieving practical implementation of HE.

Abbas et al. [6] described various privacy preserving approaches for e-health clouds. The paper discussed different cryptographic and non-cryptographic approaches (access policies) to enforce privacy and security of the e-health clouds.

IBM has been involved in constant research for several years in the field of homomorphic encryption. It has brought excellent solutions in the process of coupling healthcare and machine learning. Bocu and Costache [7] have invented a system for managing health record metrics from local monitoring devices in IBM cloudlet and used Apache Spark for processing the data and HE for securing the data.

Li et al. [8] have introduced multi-hop identity-based proxy re-encryption using HE, which can be utilized for data forwarding, email forwarding or can be applied for access control policies. The authors claim that HE is proven secure under learning with errors (LWE) assumption.

A new parallel implementation of HE for securely storing data in the cloud is done by Sethi et al. [9]. This paper discusses the implementation of DGHV homomorphic

encryption scheme [9] and authors have also executed a new procedure to reduce the noise generated by ciphertexts when the data gets homomorphically encrypted.

Chen et al. [10] have utilized NTRU algorithm to enforce privacy and avoid intrusion while sharing medical data in cloudlets. They split and encrypt the data in many ways to strengthen the security.

Zhang et al. [11] have invented an efficient privacy-preserving disease prediction (PPDP) scheme in health clouds. They have constructed the prediction models using a single-layer perceptron algorithm. They have evaluated breast cancer and heart disease dataset and tested error rate of classifier.

Zhang et al. [12] stated that HE schemes are the perfect solutions for securing outsourced data in the cloud. But fully homomorphic encryption (FHE) algorithms are prone to statistical analysis with some probability if an intruder or malicious cloud deploys attacks by observing user's reactions to results.

Hassan [13] state that implementation of FHE needs enormous computational power to process the encrypted data.

Based on the above findings, we have decided to implement SHE schemes which may not require bootstrapping technique. SHE is an optimized and less functional encryption scheme which can be used for securing outsourced computations in the cloud, which is more efficient compared to the FHE.

3. OUR CONTRIBUTIONS

First and foremost amongst the public healthcare services provided in India are primary health centres (PHC) and community health centres (CHCs). The Declaration of Alma-Ata [15] is the initiative for the Government of India to enhance the services provided by the primary health centres (PHCs). In this work we analyse the working pattern of PHCs where health workers and health assistants work on the fields to collect medical data from the population, and they specially focus on the programs like infant immunization, anti-epidemic programs, birth control, parental care, emergency medical care in addition to the regular medical treatment. All the data collected from the patients are merely kept in ledgers or stand-alone computers.

The EHRs or other health information exchange between PHCs and government hospitals is a time-consuming process through insecure medium. Sometimes they just exchange documents using email. They store data in simple excel sheets and share them without any security procedures. Government healthcare institutions are also using fax machines to share information as it is less likely to be attacked compared to the Internet, but communication through faxes is a slow process. Instead, modern technologies like clouds [10] can be effectively utilized to share the health information among hospitals, medical researches, doctors and insurance companies. Figure 1 depicts secure health cloud model, where the cloud security is provided through homomorphic encryption.

Our idea is to regularize, format, collect and connect the data collected from all primary health centres (PHCs) via a secure, privacy-preserving health cloud. This secure health cloud gives authorized access to the doctors and researchers in the government hospitals (GHs). Moving one step further, the GHs can analyse the EHRs to make meaningful inferences and predictions. These results can be used by the Department of Health and Family Welfare (DoHFW) for accurate decision-making.

The way to accomplish our goal of constructing a secure and privacy-preserving health cloud is the following – primary health workers and health assistants (immediate supervisors) are given a mobile application through which they collect data from the patients. The data are collected from patients during: i) periodic censuses; ii) visits to primary health centres (unusual visits); iii) periodic visits (e.g. infant immunization, parental care, etc.); iv) vaccination in Polio Medical Camps or other medical camps. Each health worker/health assistant is provided with encryption keys after an authorization and validation process by the key server. The collected data are encrypted and moved to the health cloud. The doctors and researchers of the government hospitals also hold the keys to access the health cloud. In addition, they can perform predictive analysis with the data stored in the health cloud. The outcomes of the analysis are sent to the Department of Health and Family Welfare (DoHFW) for further actions

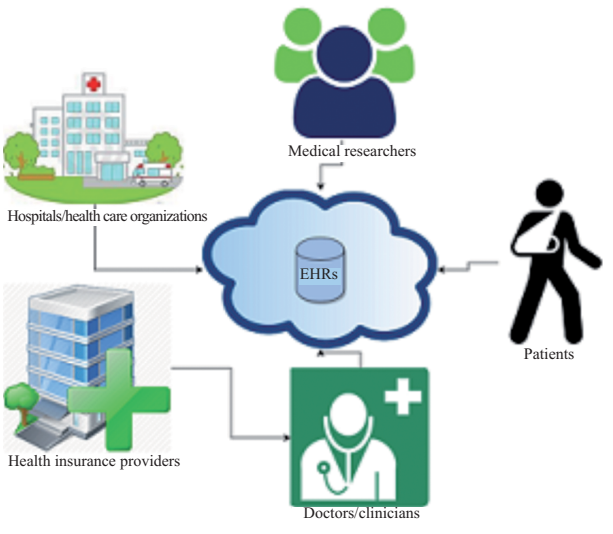

Fig. 1. Secure e-health cloud model using HE.

and decisions. The following scenarios explain the usage of this secure health cloud better. If a patient of the PHC goes to the government hospital for further treatment, patient's EHRs can be retrieved quickly by the doctor for analysing the patient's history. For example, if a health worker records dengue virus in particular area, the results of predictive analytics can trigger the Department of Health and Family Welfare to make quick decisions such as ordering necessary drugs from pharmaceutical companies, raising the awareness among people and cleaning the sewage stagnations in the affected area by the municipal corporation.

We have evaluated the performance of our system with the dataset of 10,000 records from UCI machine learning repository. For medical data analytics, 3000 records were used as training dataset and 800 records as test dataset. To summarize:

(1) encryption of EHRs uses Boneh, Goh, and Nissim (BGN) cryptosystem and stores data in the cloud – secure storage of data;
(2) re-encryption of EHRs is made by CSPs using AFGH algorithm – secure sharing of data;
(3) ticket generation by the data owner for granting access to EHRs – secure access to data;
(4) retrieval of EHRs by the trusted user of the e-health cloud – secure data retrieval.

The rest of the paper is organized as follows. Section 4 describes the crux of the entire system (homomorphic encryption). Section 5 discusses the overview of the secure privacy-preserving health cloud framework. Section 6 presents the BGN cryptosystem and AFGH re-encryption algorithm which we have used to build our system. Section 7 discusses security analysis of our system through a threat model. Section 8 presents the experimental results of our system, and the paper ends with conclusions.

4. CRYPTOGRAPHICAL BACKGROUND

4.1. Homomorphic encryption

Homomorphic encryption (HE) [4] is the scheme that allows convoluted calculations to be performed on encrypted data without compromising the encryption. In mathematics, HE describes the transformation of one dataset into another while preserving the relationship between elements in both sets. The term is derived from the Greek words for "same structure". Hence the records contained in homomorphic encryption system remain the similar schema and logical calculations, they are performed on encoded or decoded records.

Homomorphic encryption plays a major role in cloud computing, allowing the users to store encrypted data in public clouds and perform computations in the cloud by CSPs with no fear of losing privacy or security. The concept of Gentry's breakthrough introduced computation on convoluted computations to be performed on enciphered data without ever having to decrypt it or accord the encryption. Such technique requires vast amounts of computational power (up to a trillion times more compared to what is currently used). Confidential data need to be analysed without fear of compromising. That could make companies and agencies which now refuse to let such data off their servers more comfortable outsourcing high-value work.

Homomorphism is not a new word in the field of cryptography. In 1978 Rivest, Adleman and Dertouzos [16] published a paper which discussed how homomorphic properties can be used for securing data and allowing untrusted parties to work with the data. Later, in 1982, Rivest, Shamir and Adleman founded RSA Data Security.

In mathematics, a homomorphisms are maps between two algebraic objects (such as two rings, or two fields). This means that homomorphism between two algebraic objects A, B is a function f: $A \rightarrow B$ which preserves the agebraic structure on A and B. If the operations on A and B are both addition, then the homomorphism condition is $f(a+b) = f(a) + f(b)$. If A and B are both rings, with addition and multiplication, there is also a multiplicative condition: $f(ab) = f(a)f(b)$ [3].

To understand HE better, let us assume that encryption is applied by multiplying the plaintext by 2, and decryption operation is the reverse operation, dividing the ciphertext by 2, it works as described below: $y = \text{Enc}(a) = 2a$, $a = \text{Dec}(b) = b/2$, where a and b are plaintext and ciphertext, respectively. The arithmetic operations can be applied on the ciphertexts directly, the obtained result can be decrypted to get the actual result.

The different types of HE schemes available are as follows: fully homomorphic encryption (FHE), partial homomorphic encryption (PHE) and somewhat homomorphic encryption (SHE). A cryptosystem is said to be partially homomorphic if it demonstrates either additive or multiplicative homomorphism but not both. Some examples of partially homomorphic encryption systems are: RSA cryptosystem (exhibits multiplicative homomorphism), ElGamal cryptosystem (exhibits multiplicative and exponentiation homomorphism) and Paillier's cryptosystem (exhibits additive homomorphism).

A cryptographic system that promotes random calculations on encrypted texts are said to be fully homomorphic encryption and known to be much stronger. Such a scheme allows to build programs for certain functions which can be run on encrypted inputs to generate the result in encrypted form. Homomorphic encryption system doesn't need to decrypt its inputs. It can be run by a entrusted party without revealing its inputs and internal state. Fully homomorphic

Table 1. List of homomorphic algorithms

Type of encryption	Schemes available
Partial homomorphic encryption	1. Unpadded RSA 2. ElGamal 3. Paillier 4. Benaloh 5. Goldwasser–Micali cryptosystem 6. Naccache–Stern 7. Okamato–Uchiyama cryptosystem 8. Damgard–Jurik cryptosystem
Fully homomorphic encryption	1. Gentry's cryptosystem 2. DGHV homomorphic encryption scheme 3. BGV homomorphic encryption scheme 4. FV homomorphic encryption scheme 5. YASHE homomorphic encryption scheme

cryptosystems have great practical implications in the outsourcing of private computations, for instance, in the context of cloud computing.

Gentry [4] used lattice-based cryptography for demonstrating the first FHE scheme announced by IBM on June 25, 2009. He started with the construction of a somewhat homomorphic scheme where he limits the number of operations that can be performed on the ciphertext which is referred to as evaluation circuits. Then he introduced bootstrapping where the system can evaluate its own decryption algorithm circuit. Then he finally proved that any bootstrappable SHE can be converted into a fully homomorphic encryption by squashing the decryption circuits and through recursive self-embedding.

Table 1 shows various homomorphic schemes available. Halevi [17] has categorized the FHE constructions into three generations. The first generation of FHE constructions suffered from a problem of fast-increasing noise. The second-generation FHE constructions enable to control noise growth in better ways. These techniques rely on limiting the number of operations which result in 'levelled' schemes of partial homomorphism. This can be further converted into FHE schemes through bootstrapping. The third generation FHE schemes had asymmetric multiplication, in the sense that the homomorphic multiplication $c1 \otimes c2$ results in a different ciphertext compared to $c2 \otimes c1$ (both of which encrypt the same product $b1 \cdot b2$). The important factor is that the noise growth is also asymmetric: the noise in the left multiplicand has greater influence on the result than the noise in the right multiplicand.

4.2. Re-encryption

Re-encryption allows proxy to convert ciphertexts of the data owner's key into ciphertexts of authorized user's key, without revealing the plaintext or using the data owner's private keys. Therefore, data owner can share EHRs to any trusted users like other doctors or researchers, without sharing the data owner's private key. This method also eliminates data owner's need to perform any special encryption for the trusted user whom he wishes to grant access.

The concept of proxy re-encryption was proposed in 1998. It allows proxy (a semi-trusted entity) to transform the ciphertext of one user (A) into a ciphertext of another user (B). Now, user B can decrypt the ciphertext without A's private key. We have used the AFGH algorithm for re-encryption. In addition to the standard functions like key generation, encryption and decryption, we have also defined re-encryption ticket generation and re-encryption.

Ticket generation is made by the data owner who wishes to grant access to the trusted user. Tickets are generated by the data owner with the public key of trusted user and private key of the data owner. Now, the generated ticket is used by the proxy to re-encrypt the EHRs which can be decrypted using the private key of trusted user.

5. FRAMEWORK OVERVIEW

The proposed secure e-health cloud framework mainly focuses on the storage of patient's EHRs, which are sensitive in nature, and on the issue how to share EHRs securely between various entities within the system. The combination of homomorphic encryption and re-encryption algorithms is used to securely store and share EHRs in the cloud. Figure 2 shows the general architecture of the system. The entities which comprise the system are the following: (1) key authority (KA); (2) cloud service provider (CSP); (3) primary health centres/community health centres/sub-health centres (PHCs/CHCs/SHCs); (4) government hospitals (GHs); and (5) Department of Health and Family Welfare (DoHFW).

(1) **Key authority (KA)** is a trusted entity which generates and distributes key pairs for other entities involved in the system. We assume that KA performs authentication and checks roles and policies of the user.
(2) **Cloud service provider (CSP)** provides unlimited storage space to other parties for outsourcing so that they can store and manage their data. CSPs are able to do computations with the stored data in addition to offering storage features.
(3) **PHCs/CHCs/SHCs** are set up by the initiative of the government to provide public healthcare services. It generates medical data which are encrypted and can be further used for health data analytics to bring out significant references.
(4) **Government hospitals (GHs)** also provide healthcare services and have more sophisticated facilities compared to PHCs, CHCs or SHCs. Government

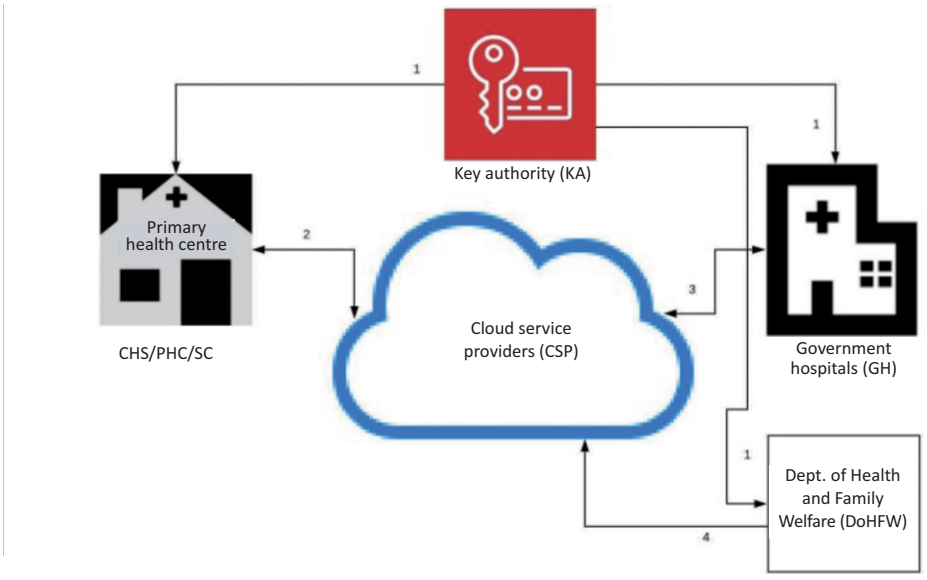

Fig. 2. General architecture of the system.

hospitals provide more advanced treatment and better services, higher level of making diagnosis and medication supply.

(5) **Department of Health and Family Welfare (DoHFW)** is a department under the government of Tamil Nadu, and is responsible for ensuring access to basic public health services. It also instigates many family welfare programs.

EHRs are generated in health centres in one of the following ways:

- health worker or health assistant uses a mobile application to collect information at patient's home during periodical visits or the census;
- when a patient visits HC;
- when patients are enrolled to medical camps, etc.

These EHRs are homomorphically encrypted and moved to the cloud.

If the doctor wishes to share patient's EHRs with other HCs or GHs, the EHRs are re-encrypted using AFGH scheme and then moved to the cloud. If the EHRs are not meant for sharing, data are encrypted using BGN algorithm and stored in the cloud repository. The shared EHRs can be used by GHs to quickly diagnose the patient. Authorities in DoHFW can perform data analytics to generate reports on the performance of HCS which can be used for ranking the PHCs and issuing sensible commands to the relevant government departments. For example, drug stocks can be updated according to the requirements of PHCs. Government hospitals can also perform secure medical data analytics with EHRs to predict diseases or to classify patients depending on the severity of disease, frequency of visits, etc.

6. ALGORITHMS

This section explains the background of the encryption process made in our system. Somewhat homomorphic encryption (SHE) is used to encrypt the EHRs and then the encrypted EHRs are moved to the cloud repository for later access.

6.1. Storage of EHR

We have implemented the SHE using Boneh, Goh and Nissim (BGN) homomorphic public key cryptosystem [18]. This algorithm is built on the properties of a class of subgroup decision problem. The strength of the algorithm relies on the difficulty of deciding whether an element, say 'x' belongs to a group G of some order p, where p = n1n2, also belongs to a subgroup of order n1. The diagram shown below (Fig. 3) explains how the EHRs from primary health centres are stored securely through BGN cryptosystem.

Key generation:

(1) Choose two random s-bit primes p1 and p2 and set n = p1p2 ∈ Z, where s is the security parameter.
(2) Let G be the bilinear group of order n with generator g.
(3) e: GXG → G1 is the bilinear map.
(4) Choose two random generators g, u ← G and set $h = u^{p2}$.
(5) Public key = (n,G,G1,e,g,h).
(6) Private key = p.

Encryption:

(1) Message space consists of integers in the set {0, 1, 2, ... X} with X < p2.
(2) Choose a random r, 0<r<n-1.

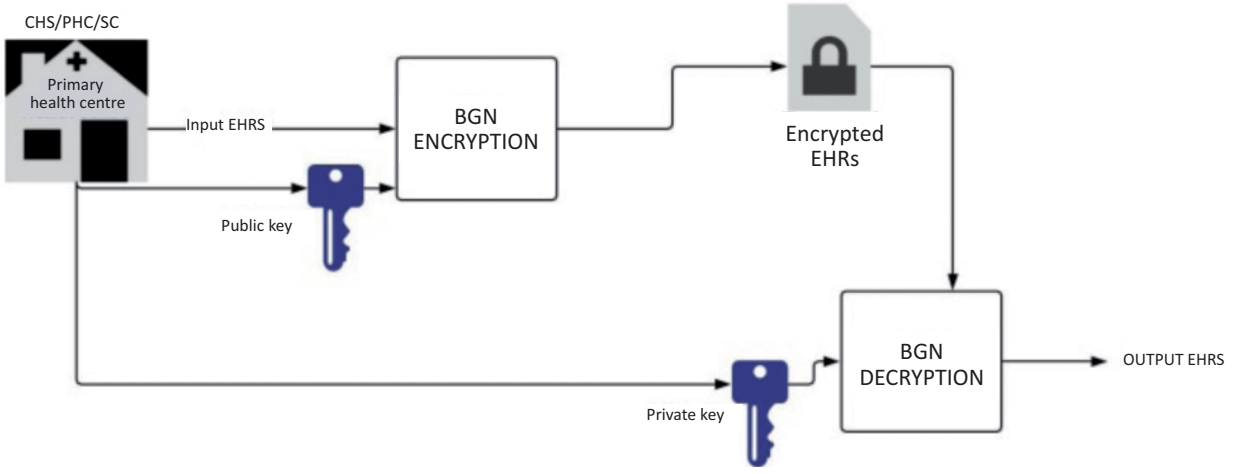

Fig. 3. Storage of EHRs by CHC/PHC/SC.

(3) encrypt (m) = C = $g^m h^r \in G$.
(4) Return C, the generated ciphertext.
Decryption:
(1) The plaintext can be generated by decrypting the ciphertext using private key p.
(2) decrypt (C)= $C^p = (g^m h^r)^p = (C^p)^m$.
(3) Plaintext m is retrieved by computing the discrete log of C^p base $\hat{g}$, where $\hat{g} = g^p$.
(4) Return m, the decrypted message.

6.2. Secure EHR sharing

To achieve the sharing of EHRs among various health centres (HCs), government hospitals (GHs) and other government organizations (e.g. DoHFW), Ateniese, Fu, Green and Hohenberger (AFGH) [19,20] algorithm is used. Figure 4 shows how secure sharing of EHRs is accomplished among various entities of the secure e-health cloud.

The construction of the AFGH system is also based on bilinear groups similar to the BGN algorithm. This AFGH cryptosystem is built on the nature of ciphertexts which can be moved from one group to another group with bilinear transformation. We make use of the AFGH algorithm to share the encrypted EHRs to the authorized user by CSPs without the need of revealing the owner's private key. CSPs re-encrypt [19] the encrypted EHRS with the help of a ticket generated by the owner and public key of the user requesting the EHR. This re-encrypted EHR can be decrypted by the authorized user using his private key. The algorithm description is as follows.

Key generation:
(1) Let G1 be a group of prime order q <g> = G1.
(2) Let α be the random number chosen from Z_q^*.
(3) Private key = k1 =PrK1.
(4) Public key = g^{k1} = PuK1.
(5) For User2, Private key = k2 = PrK2.
Public key = g^{k2} = PuK2.
Encryption (data owner):
(1) Message m ε G2, G2 is the group of prime order q.
(2) Let r is the random number, $r \in Z_q^*$.
(3) encrypt (m) = C1 = $(Z^r \cdot m, g^{rk1})$.
Ticket generation (data owner):
$T = (g^{k2})^{1/k1} = g^{k2/k1}$.
Decryption (data owner):
decrypt (c1) = $m = Z^r \cdot m / e(g^{rk1}, g^{1/k2}) = Z^r \cdot m / Z^r$.
Re-encryption (CSP):
C1 → CSP → C2.
$C2 = (Z^r \cdot m, e(g^{rk1}, T)) = (Z^r \cdot m, e(g^{rk1}, g^{k2/k1}))$
$= (Z^r \cdot m, Z^{rk2})$.
Decryption (authorized user accessing EHR – User2):
Decrypt (C2) = $m = Z^{r \cdot m} (Z^{rk2})^{1/k2}$.

7. DISCUSSION

We assume that the CSP is honest-but-curious, as the same assumption is made by many researchers according to the literature [14,22,23]. CSP honestly follows a procedure to store the data using a standard protocol but may curiously deduce relevant sensitive private information belonging to the patients from the stored data. Moreover, unauthorized users or hackers may intrude to access the private information without the necessary privileges by compromising the key.

The threat model is constructed based on the following cases.

Case 1. The intruder may observe all ciphertexts stored in the cloud. He can also observe the posted queries.

Case 2. The intruder may even arbitrarily construct some random EHRs and push to the cloud for disease

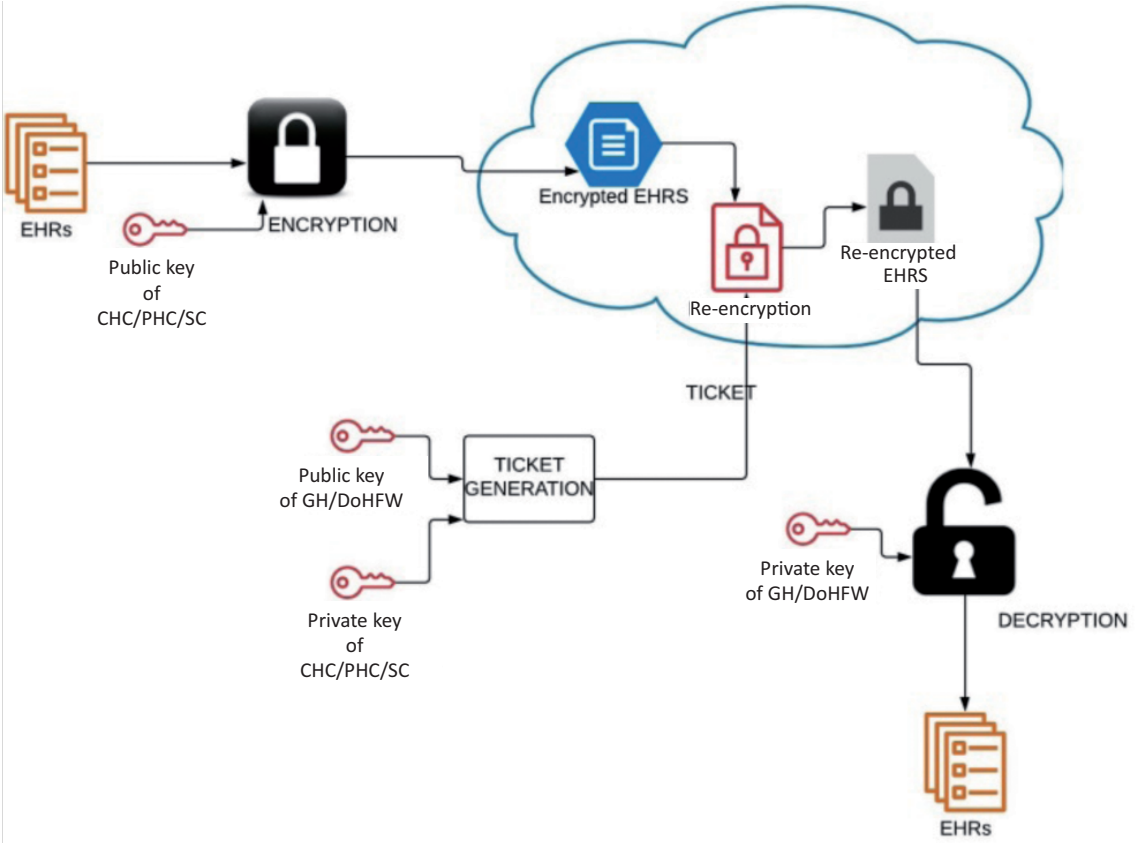

Fig. 4. Secure EHR sharing between CHCs/PHCs/SCs and GHs/DoHFWs.

prediction. This is something similar to the chosen-ciphertext attack (CCA).

Case 3. The intruder may know EHRs of few patients before they are pushed to the cloud. So, this may lead to known-plaintext attacks (KPA). But still, the intruder has no knowledge about the corresponding ciphertext stored in the cloud.

Our framework protects the confidentiality of EHRs stored in the cloud by encrypting EHRs homomorphically. The EHRs of particular clients may be lost only if the client's (data owner) device is compromised. Confidentiality of the data is maintained, though it is shared in the public cloud. The entities of the system may not leak any information as they move only encrypted EHRs to the cloud. It is assumed that key authority verifies and authenticates that all the participants are legitimate trusted users. Our secure EHR storage and sharing framework stands well against passive attacks preserving the confidentiality of the patients' sensitive EHRs.

Key generation.

Key generation takes some average setup time in the data owner's devices. BGN takes approximately 0.4 ms for key generation in 80-bit security mode. Compared to Paillier, it is much lower, as Paillier takes more than 1600 ms. However, BGN cryptosystem takes additional time to find the prime numbers and congruences for CRT operation to optimize the trade-off between the performance of decryption operations.

Key revocation.

To grant access to the EHRs, the data owner generates a ticket for the trusted user. Generation of this ticket is the function of the data owner's private key and trusted user's public key. Cloud on behalf of the data owner re-encrypts the EHR with the generated ticket, which in turn is decrypted by the trusted user. If the data owner wants to revoke the granted access, he can simply change the key and start using new keys for encryption. This key update makes the already generated tickets obsolete and it can no longer be accessed. If the valid sharing relationships need to be maintained, new tickets have to be generated.

8. EXPERIMENTAL RESULTS

We have evaluated our framework in the Amazon Web Service (AWS) environment with multiple clients. Client systems are PCs configured with Intel Core i3-2120 3.30

GHz CPU and 8 GB RAM running Fedora Linux or Lenovo K8 with 2.3 GHz dual core processor and 4 GB RAM running Android 5.1.1. Amazon T3 instances are configured in cloud setup. The AWS accounts provide a default amount of storage capacity and one instance of Intel Xenon 3.3 GHZ with 8 GB RAM.

Plaintexts are generated uniformly at random for 16-bit, 32-bit and 64-bit integers. We have run the encryption and decryption operations several times repeatedly and recorded the execution time. All the noted numbers are calculated with a single running thread. Tables 2 and 3 show the execution time for encryption and decryption operations in the data owner's device (mobile devices or personal computers) and cloud.

We have compared our framework with the standard Paillier cryptosystem [24] and plotted the results. It is visible that the BGN cryptosystem outperforms the standard Paillier system. In case of Paillier cryptosystem, the time for encryption increases exponentially when the size of the plaintext increases. Though the BGN scheme also suffers in the case of large plaintexts, the performance can still be enhanced when we implement batch processing by splitting up the plaintext into blocks using the CRT algorithm.

Figure 5 shows the execution time for encryption and decryption operations with BGN vs. Paillier over 16-bit, 32-bit and 64-bit integers, with both 80-bit and 128-bit security level. Paillier has huge computation time increase due to its large byte stuffing. However, BGN cryptosystem outperforms Paillier cryptosystem approximately 3 or more times in case of smaller plaintexts. For decryption, BGN has best running time in all settings except for 80-bit security and 64-bit integers. Paillier has severe drop down in performance with 128-bit security probably due to big key sizes (from 1048-bit to 3064-bit) subsequently resulting in big number of operations.

The size of the generated ciphertext after re-encryption causes a huge impact on the storage capacity and network communication constraints. In this regard, AFGH performs better than standard Paillier cryptosystem. The framework supports 16-, 32-, and 64-bit integers in storage mode. In the 80-bit security mode, increasing integer size includes a different number of congruences, leading to a ciphertext size ranging between 48 and 128 bytes. Paillier, in comparison, requires 256 bytes, no matter of the size of the integer. The gap is even more noticeable as the protection level rises and the network capacity and cloud have negative impact. Figure 6 shows the 16-bit, 32-bit and 64-bit ciphertext sizes in the Paillier and AFGH cryptosystems, respectively.

Table 3. Performance summary – cloud

Type	Security mode	e-Health cloud	
		Addition (μs)	Sharing (ms)
Storage of EHRs – repository (BGN)	80	60	–
	128	93	–
Secure EHR sharing – (AFGH)	80	68	1.9
	128	75	2.5

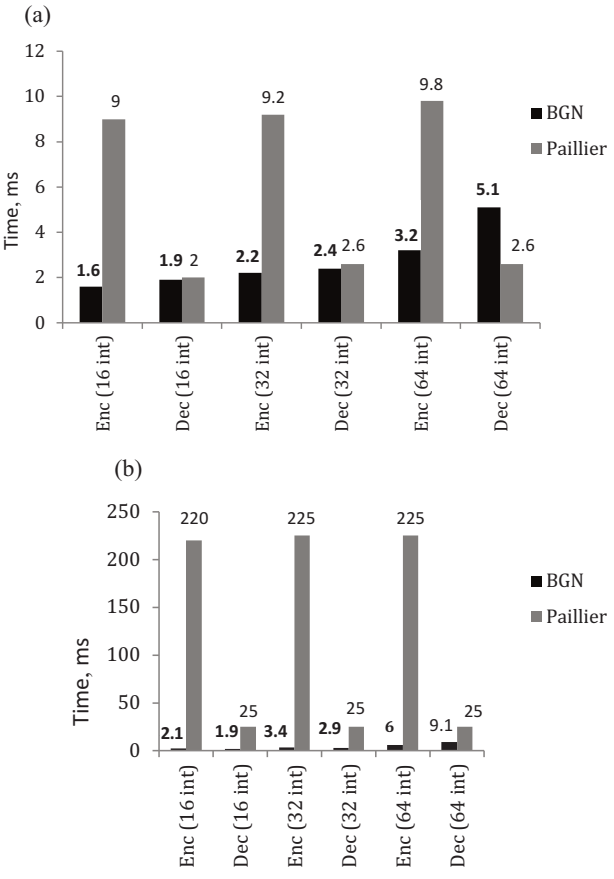

Fig. 5. (a) Execution time comparison – BGN vs. Paillier – 80 bit; (b) Execution time comparison – BGN vs. Paillier – 128 bit.

Table 2. Performance summary – data owner's device

Type	Security mode	Data owner's device		
		Encryption (ms)	Decryption (ms)	Re-encryption (ms)
Storage of EHRs – repository (BGN)	80	2.5	2	NA
	128	4.9	3.9	NA
Secure EHR sharing – (AFGH)	80	9.7	11.9	9.2
	128	15.1	17.3	13.1

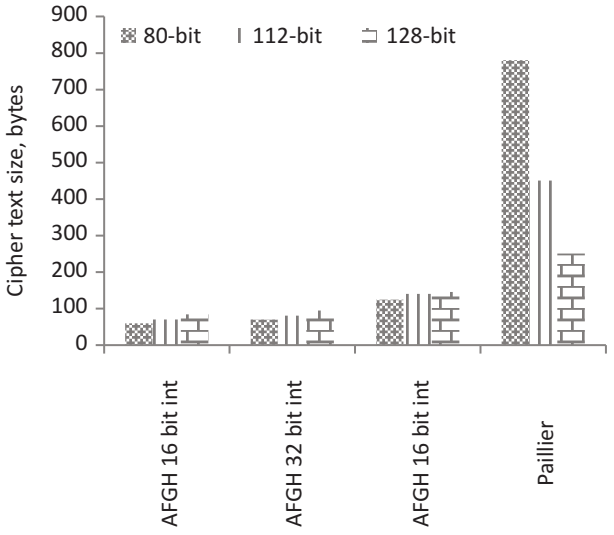

Fig. 6. Ciphertext size – Paillier vs. AFGH algorithm. Healthcare industry in India.

9. CONCLUSIONS

We have presented a secure framework for storing and sharing EHRs in the cloud using homomorphic encryption algorithms. The results are shown and compared with standard Paillier algorithm. Our framework has shown better results. As we are moving only encrypted data to the cloud, it guarantees the confidentiality of sensitive patient's data. The curious CSP may not meaningfully read the outsourced medical data as it is encrypted before moving it to the third party's CSPs. Our secure framework stands well against the passive attacks. For the future work, key authority must be implemented with the additional capabilities of ensuring the roles, policies and trust in the process of authorization. Various statistical methods and machine learning approaches [21] can be applied on the health data for disease prediction and to derive other meaningful inferences. Role-based access control (RBAC) policies [24] may be introduced. Also, bootstrapping techniques can be explored for providing full homomorphism.

ACKNOWLEDGEMENTS

The authors are grateful to the faculties and the management of R.M.K. College of Engineering and Technology for their support and guidance.The publication costs of this article were partially covered by the Estonian Academy of Sciences.

REFERENCES

1. Healthcare Industry in India. https://www.ibef.org/industry/healthcare-india.aspx
2. Usage of HIPPA Act in US. https://en.wikipedia.org/wiki/Health_Insurance_Portability_and_Accountability_Act
3. Fundamentals of the crux of our health cloud framework - Homomorphic Encryption. https://en.wikipedia.org/wiki/Homomorphic_encryption
4. Gentry, C. A fully homomorphic encryption scheme. In *Proceedings of the 41st Annual ACM Symposium on Symposium on Theory of Computing (STOC '09), May 31–June 2, 2009, Bethesda, Maryland, USA*. Association for Computing Machinery, New York, 2009, 169–178. https://doi.org/10.1145/1536414.1536440
5. Acar, A., Aksu, H., Uluagac, A. S., and Conti, M. *A Survey on Homomorphic Encryption Schemes: Theory and Implementation*. 1–35. https://doi.org/10.1145/3214303
6. Abbas, A. and Khan, S. U. A review on the state-of-the-art privacy-preserving approaches in the e-Health clouds. *IEEE J. Biomed. Health Inf.*, 2017, **18**(4), 1431–1441. https://doi.org/10.1109/JBHI.2014.2300846
7. Bocu, R. and Costache, C. A homomorphic encryption-based system for securely managing personal health metrics data. *IBM J. Res. Dev.*, 2018, **62**(1), 1:1–1:10. https://doi.org/10.1147/jrd.2017.2755524
8. Li, Z., Ma, C., and Wang, D. Towards Multi-Hop Homomorphic Identity-Based Proxy Re-Encryption via Branching Program. *IEEE Access*, 2017, **5**, 16214–16228. https://doi.org/10.1109/ACCESS.2017.2740720
9. Sethi, K., Majumdar, A., and Bera, P. 2017. A novel implementation of parallel homomorphic encryption for secure data storage in cloud. In *Proceedings of the International Conference on Cyber Security and Protection of Digital Services (Cyber Security 2017), June 19–20, 2017, London, UK*. https://doi.org/10.1109/CyberSecPODS.2017.8074851
10. Chen, M., Qian, Y., Chen, J., Hwang, K., Mao, S., and Hu, L. Privacy Protection and Intrusion Avoidance for Cloudlet-based Medical Data Sharing. *IEEE Trans. Cloud Comput.*, 2016, 1. https://doi.org/10.1109/TCC.2016.2617382
11. Zhang, C., Zhu, L., Xu, C., and Lu, R. PPDP: An efficient and privacy-preserving disease prediction scheme in cloud-based e-Healthcare system. *Future Generation Computer Systems*, 2018, **79**(1), 16–25. https://doi.org/10.1016/j.future.2017.09.002
12. Zhang, Z., Plantard, T., and Susilo, W. Reaction attack on Outsourced Computing with Fully Homomorphic Encryption Schemes. In *Proceedings of the International Conference on Information Security and Cryptology (ICISC 2011), November 30 – December 2, 2011*, Seoul, Korea. Springer-Verlag, Berlin, Heidelberg, 2012, 419–436. https://doi.org/10.1007/978-3-642-31912-9_28
13. Hassan. N. A. *Data Hiding Techniques in Windows OS: A Practical Approach to Investigation and Defense, 1st ed*. Syngress, Rockland, 2016.
14. Chen, M., Hao, Y., Hwang, K., Wang, L., and Wang, L. Disease prediction by machine learning over Big Data from healthcare communities. *IEEE Access*, 2017, **5**, 8869–8879. https://doi.org/10.1109/ACCESS.2017.2694446
15. https://en.wikipedia.org/wiki/Primary_Health_Centre_(India)
16. Paar, C., Pelzl, J., Paar, C., and Pelzl, J. The RSA Cryptosystem. In *Understanding Cryptography*. Springer,

Berlin, Heidelberg, 2009, 173–204. https://doi.org/10.1007/978-3-642-04101-3_7
17. Halevi, S. Homomorphic Encryption. In *Tutorials on the Foundations of Cryptography. Information Security and Cryptography* (Lindell, Y., ed.). Springer, Cham, 2017, 219–276. https://doi.org/10.1007/978-3-319-57048-8_5
18. Freeman, D. M. *Homomorphic Encryption and the BGN Cryptosystem*. 2011. http://theory.stanford.edu/~dfreeman/cs259c-f11/lectures/bgn
19. Ateniese, G., Fu, K., Green, M., and Hohenberger, S. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Trans. Inf. Syst. Secur.*, 2006, **9**(1), 1–30. https://doi.org/10.1145/1127345.1127346
20. Shao, J., Lu, R., Lin, X., and Liang, K. Secure bidirectional proxy re-encryption for cryptographic cloud storage. *Pervasive Mob. Comput.*, 2016, **28**, 113–121. https://doi.org/10.1016/j.pmcj.2015.06.016
21. Aslett, L. J. M., Esperança, P. M., and Holmes, C. C. *A review of homomorphic encryption and software tools for encrypted statistical machine learning*. arXiv:1508.06574, 2015.
22. Premarathne, U., Abuadbba, A., Alabdulatif, A., Khalil, I., Tari, Z., Zomaya, A., and Buyya, R. Hybrid cryptographic access control for cloud-based EHR systems. *IEEE Cloud Comput.*, 2016, **3**(4), 58–64. https://doi.org/10.1109/MCC.2016.76
23. Liu, X., Lu, R., Ma, J., Chen, L., and Qin, B. Privacy-preserving patient-centric clinical decision support system on naïve Bayesian classification. *IEEE J. Biomed. Health Inf.*, 2016, **20**(2), 655–668. https://doi.org/10.1109/JBHI.2015.2407157
24. Galbraith, S. D. Elliptic curve Paillier schemes. *J. Cryptol.*, 2002, **15**, 129–138. https://doi.org/10.1007/s00145-001-0015-6
25. Electronic health record standards for India. https://www.nhp.gov.in/data-privacy-and-security_mtl
26. Privacy in Cloud Computing. ITU-T Technology Watch Report. ITU Telecommunication Standardization Bureau, 2012. https://www.itu.int/dms_pub/itu-t/oth/23/01/T23010000160001PDFE.pdf
27. Health and Privacy. Privacy India. https://cis-india.org/internet-governance/health-privacy.pdf/view
28. Shen, J., Deng, X., and Xu, Z. Multi-security-level cloud storage system based on improved proxy re-encryption. *EURASIP J. Wireless Commun. Networking*, 2019, 277. https://doi.org/10.1186/s13638-019-1614-y