ISSN: 2633-4828 Vol. 5 No.4, December, 2023

International Journal of Applied Engineering & Technology

ARS-PRE: AUTHENTICATED REVOCABLE SYMMETRIC PROXY RE-ENCRYPTION FOR BLOCKCHAIN-BASED EHR SHARING

[1*]Gajala Praveen, [2]Piyush Kumar Singh and [3]Prabhat Ranjan
[1,2,3]Department of Computer Science, Central University of South Bihar, Gaya, India
*praweengajala@gmail.com

ABSTRACT
Nowadays, we are seeing a lot of data breaches in the healthcare industry. In comparison to other industries, it has risen at a quick rate because health insurance needs legal health records. Every citizen must have insurance because it is mandated in the majority of countries. Sharing data between healthcare organizations is difficult due to security and privacy concerns, although each hospital uses distinct terminology, employs different methodologies, and may have varied functional capabilities. In this paper, we have explored the revocable symmetric proxy re-encryption scheme that make data sharing among doctors from different hospital. The main purpose of this chapter is to provide high security using the proxy re-encryption technology.

4.1 INTRODUCTION
Medical health records are a necessary part of the healthcare sector as they store and share information. These records were traditionally handwritten, turned out to be burdensome when the records needed to be shared [1]. To resolve this problem, electronic health records (EHRs) introduced a novel way to interchange information [2-3]. However, as the hospitals had independent databases, it was again inconvenient to share or access information. This was solved by cloud-based EHRs. As the information is stored in a centralized space, it makes the data vulnerable [4]. To help both the hospital and the patient, this project will use blockchain technology to remove barriers that could harm both. It will also make information sharing easier and safer for everyone. The management of the medication supply chain entails the transfer of drugs from pharmaceutical corporations to pharmacies [5-6]. To prevent tampering and counterfeiting of the drug, adequate monitoring and tracking mechanisms should be used throughout the supply chain. The mechanism is used to ensure that all stakeholders, including end users, can verify the drugs constituents. Prescription management necessitates a well-defined method for delivering healthcare services. Misuse of the medication results in a serious problem. The Blockchain-based system enables a safe prescription process in which all transactions are recorded in the network. Health care providers and medical research benefit from personal health data collected through mobile and wearable devices. One of the most prominent benefits of blockchain technology is the ability to provide ownership of data for user-centric healthcare systems via decentralised and permissioned storage. Nearly every company requires a method for handling sensitive data. Even though there are various techniques for storing and transmitting electronic documents, users frequently encounter missing or manipulated data when distributing it over a public network [7]. The traceability and regulation of data when it is disseminated around the world is impractical. Blockchain technology will be critical in maintaining the integrity of digital data. For immutability and transparency, blockchain maintains data in a decentralized, distributed ledger.

People share a variety of EHR files with different service providers in their daily lives for verification purposes. If these personal EHR files are not treated ethically, they could be misused. Such HER files can be shared with service providers via a cloud storage platform [8]. Security breaches could result from cloud EHR file sharing of plaintext files. As a result, these materials ought to be encrypted and distributed only to approved service providers. For this, hybrid encryption techniques can be applied. In hybrid encryption techniques, files are encrypted with symmetric keys, which are then encrypted with public key algorithms. These plans guarantee file confidentiality. The integrity of the shared files should, however, be able to be confirmed by the service providers [9]. Additionally, in order to prevent unauthorized use, EHR files should only be shared with service providers for a brief period of time. Therefore, it is necessary to modify the file's encryption key so that the service provider can no longer access the file after the time limit has passed [10]. Strictly symmetric key proxy re encryption can do

this. An authorized and revocable proxy re encryption technique for personal file sharing using cloud storage has been presented in the study under consideration. The proposed system is designed using key homomorphic encryption and AONT transformation.

Paper organization: The contents of this paper are organized as follows: The first Section 4.1 discusses the introduction. Section 4.2 provides the previous work. Section 4.3 presents the preliminaries. Section 4.4 discusses proposed method. Section 4.5 gives implementation details. Lastly, we have presented the conclusion in section 4.6.

4.2 RELATED WORK

The current AONT schemes and proxy re-encryption techniques have been covered in this section along with their respective benefits and drawbacks. Rivest [11] put out the initial AONT plan. In his plan, decrypting a single message block requires an opponent to collect every block of the ciphertext. They demonstrated how brute force key searches on symmetric encryption slow down by a factor of the ciphertext's block count when utilising AONT. A later proposal for an unconditionally secure all-or-nothing transform came from Stins on [12]. Rivest AONT was extended by V. Card and T. Van[13], demonstrating that AONT's security is unaffected by the quantity of blocks in the ciphertext. A novel and effective AONT system with the ability to withstand exhaustive key search properties was proposed by Anand Desai [14]. Public key proxy re encryption was first proposed by Mabo et al. [15]. However, Blaze et al. were the first to formally define it [16]. The secret key that was used to encrypt the file can then be re-encrypted for numerous users using their public key thanks to a number of public key proxy re-encryption techniques that have been developed [17], [18], and [19]. Nevertheless, the files encrypted using the symmetric key is not decrypted again. Users can still decode the file if they have downloaded the encrypted symmetric key.

Cook et al. [20] presented the first symmetric proxy re-encryption technique as a solution to this issue. It was suggested that two encryption keys be used. The file owner encrypts the file using the first key. The proxy re-encrypts the file using the second key. When key revocation occurs later, the proxy first uses the second key to decrypt the file before reencrypting it using the new key. There are two expensive steps involved in this: encryption and decryption. Horose [21] attempted to enhance the previously mentioned approach by offering a quicker way to convert encryption from second key to new key. Nevertheless, this approach is inefficient since it necessitates two encryption processes.

A symmetric proxy re-encryption system with poor encryption (converting one permutation to another) was proposed by Amril Syalin et al. They started by altering Rivest's AONT system [1].

They recommended utilising the file with the altered AONT. Next, they employed a mapping function, which is capable of converting one permutation into another. They demonstrated the security of their plan by assuming that the output of AONT is always random [22]. However, this presumption could not be accurate if a user had access to earlier encryption keys.

A hybrid proxy reencryption system combining AONT transformation, symmetric encryption, and public proxy re-encryption was proposed by Steven Myers and Adam Shull [23]. To create the randomness in the suggested approach, a key homomorphic encryption is applied to the AONT blocks. The encryption's key homomorphic characteristic is later used to alter the encryption key during re-encryption.

4.3 PRELIMINARIES

Blockchain is a type of distributed database that combines data blocks in chronological order. It primarily addresses trust and security concerns related to transactions. Proxy reencryption and Delegated Proof-of-Stake being the most prominent and important for implementation of this framework

4.2.1 All-or-Nothing Transformation

All-or-Nothing transform (AONT) is a pre-processing method used before to encrypting the message in order to make it more difficult for a brute force search attack to crack the encryption algorithm used to encrypt the data

with symmetric keys. Let's divide a message m into blocks b1, b2, ….bs. From the key space 2n, where n is the number of bits in key K, a random key K is selected. The output block order is produced as follows:

$$b_i' = b_i \oplus E(K, i) \text{ for i=1,2,..s}$$
$$h_i = b_i' \oplus E(K_0, i) \text{ for i=1,2,..s}$$
$$b_{s+1}' = K \oplus h_1 \oplus h_2..h_s$$

K0 is here made available to the authorised user. The key is retrieved as follows during decryption:

$$\text{K} = b_{s+1}' \oplus h_1 \oplus h_2..h_s$$

The following is how the file blocks are retrieved using the key obtained in the previous step.

$$b_i = b_i' \oplus E(K, i)$$

The original file is recovered by combining the blocks obtained in above step.

4.2.2 Key Homomorphic Encryption

Let F: K*X be a pseudo random function (PRF). F is called Key-Homomorphic PRF [24] if:

- Given F (K1, x) and F (K2, x), there is an efficient polynomial time algorithm to compute F (K1 + K2, x).
- Further, F (K1 + K2, x) = F (K1, x)*F (K2, x)

4.2.3 Delegated Proof-of-Stake (DPOS)

Blockchain makes use of the consensus process to guarantee that the same global ledger is maintained by all authorised nodes. Delegated Proof-of-Stake, or DPOS, is a dependable and effective consensus process. Coin holders choose certain nodes to vote on behalf of all DPOS users, just like in a board vote. It might increase how quickly a consensus is reached. Every coin owner participates in the DPOS process by casting a vote and selecting delegates first. It is possible to think of the delegates as super nodes with reciprocal rights. Next, it is up to these super nodes to take turns creating new blocks. The network will choose new super nodes to replace delegates that do not fulfill their obligations (for example, by failing to compute the correct value when it is their turn).

4.2.4 Proxy re-encryption

To ensure the security in the data sharing, the proxy re-encryption is used as shown in Figure 38. These techniques involve one party, patient A, entrusting a semi-honest agent or a trustworthy third party to convert ciphertext encrypted with its public key into ciphertext encrypted with patient B's public key. After then, B might realise data sharing by using his or her own private key to decrypt the ciphertext. A's private key is kept secret the entire time, and the encrypted data is extremely safe. The following is a list of the precise steps:

1) A uses its own public key, CA=EA (M), to encrypt the plaintext M, where M is the information that A wishes to send to B and E is an asymmetric encryption technique like the traditional RSA.
2) After B sends A the request, A (or the agent) creates one PKA-B conversion key.

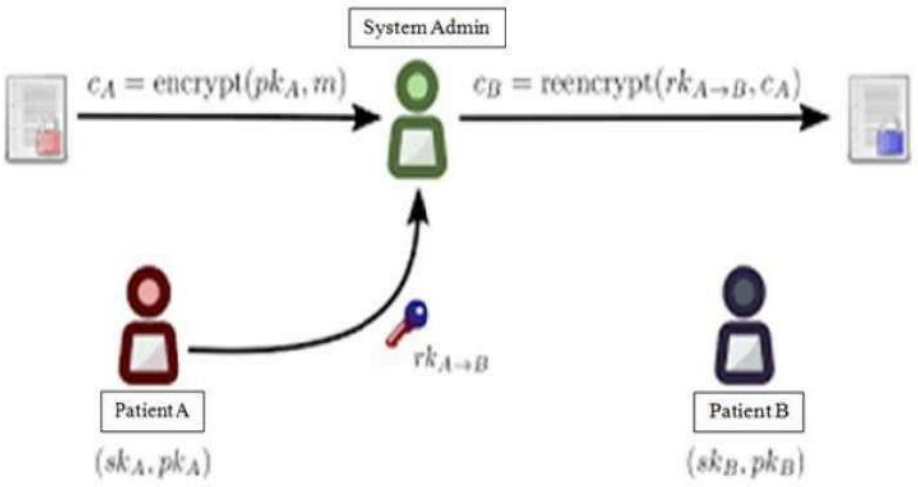

Figure 38: Proxy re-encryption

3) A forwards PKA-B and CA to the agent.
4) Using PKA-B, the agent translates the encrypted text CA into CB. Here, M's ciphertext, encrypted using B's public key, is represented as CB. The agent can only receive the transformation service in this stage; it is unable to obtain the plaintext.
5) The ciphertext CB is sent by the agent to B.
6) Using his own private key, B decrypts CB to obtain the plaintext M.

4.4 PROPOSED SCHEME

The three parties that make up the overall network model of the blockchain-based medical data exchange and protection scheme are the system manager, the user (patient), and the hospital as shown in Figure 39.

Hospital creates its private and public keys after initially registering with SM. A user must register with the hospital and set their private and public keys before seeing a doctor there. The physician will share the results in the blockchain after the diagnostic is complete. The patient's medical results will be recorded in the hospital's blockchain if they have successfully passed the server's verification process.

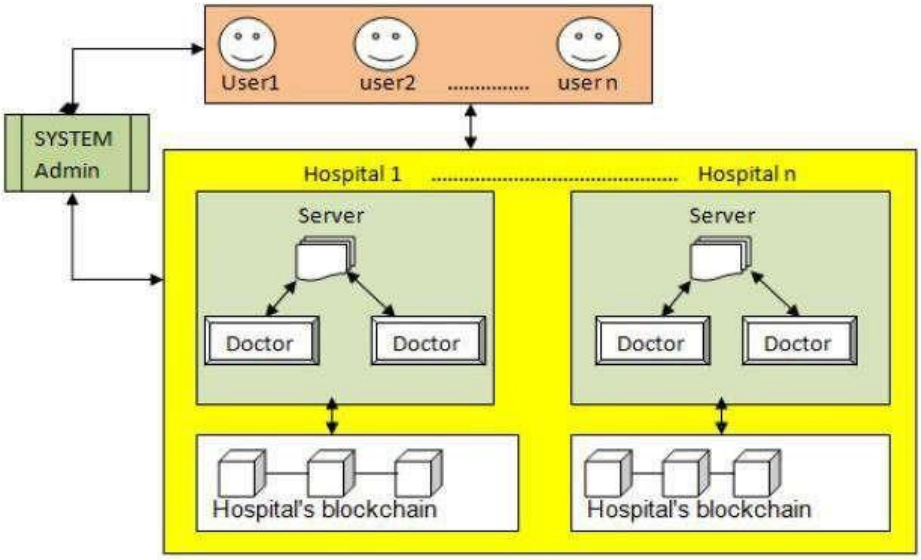

Figure 39: General network model of proposed framework

Both the patient and the doctor at any hospital should apply to the SM at the same time if they wish to view the patient's medical history. The previous records will be re-encrypted using the doctor's public key, and SM will find the conversion key and produce the ciphertext for them. Next, SM forwards the ciphertext to the physician. Lastly, any two patients might establish a session key for their upcoming session and carry out a mutual authentication. Our scheme includes the following six phases:

(i) **System Manager:** Usually performed by a few reliable authorities, the system manager (abbreviated as SM) is in charge of creating the master key and system parameters. It is in responsibility for overseeing the entire system.

(ii) **Hospital join:** Hospital creates its private and public keys after initially registering with SM.

(iii) **User Join:** A user initially registers with the hospital and sets their private and public keys before needing to see a doctor. The hospital will then set up a physician to diagnose him or her.

(iv) **Data join in blockchain:** Once the diagnosis has been completed, if the medical results have been verified by the verifiers, the doctor will put them on the blockchain.

(v) **Data search and sharing:** Both the patient and the doctor at any hospital should apply to the SM at the same time if they wish to view the patient's medical history. The history records will be re-encrypted using the doctor's public key, and SM will calculate the conversion key and produce the ciphertext for them. Next, SM forwards the ciphertext to the physician. In particular, a patient's historical data saved in the blockchain can be accessed by any doctor with consent when needed.

(vi) **Patient session:** Lastly, any two patients might establish a session key for their upcoming session and carry out a mutual authentication.

4.3.1 System Overview

The following steps make up the suggested approach.

1) **Setup Phase:** During this stage, the system admin issues a public key and private key to each system user.

2) **Encryption Phase:** There are two sub phases in this phase.

a) **AONT Phase:** In this stage, the physician encrypts its health records, which are files R of any type, using the AONT encryption Algorithm 2 (Rivest, 1997). The owner determines, using the number n, how many AONT blocks it needs. Asymmetric key K is used by AONT to perform the encryption after dividing the record into n blocks. As a result, the canary block bn and the AONT filele blocks b0, b1,..., bn-1 are calculated as

$$b_n = H(H(b_0)||H(b_1)...||H(b_{n-1}) \oplus K$$

To verify the integrity of the individual blocks and share the key.

Algorithm 2: Homomorphic PRF

Require: Elliptic curve E, public parameter X, key K
Ensure: G(X,K)
P = H(X)
Compute T = K*P
Output T

b) **Homomorphic Encryption Phase:** Each AONT file block is put via the key homomorphic encrypter (KHE) in this subphase, which is seen in Figure 40 and accepts a key called K1 as input. Blocks in the AONT file are transformed by the KHE into a collection of points on the selected elliptic curve E [25].

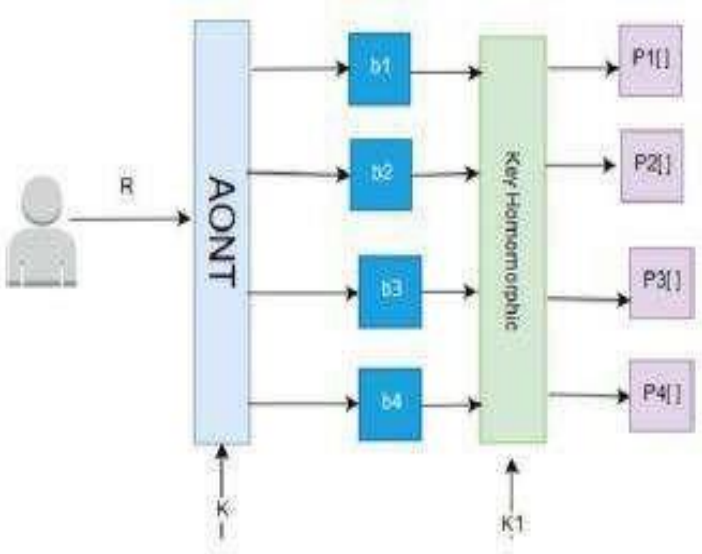

Figure 40: Encryption Phase

To accomplish this, PKCS-7 padding is applied to each block of an AONT file (Kaliski, 1998). The padded block is then split into l message blocks, each with a size of p, where p is the bitsize of the selected elliptic curve. In Algorithm 3, the encryption algorithm is described in depth.

Algorithm 3: Encryption

Require: Elliptic curve E, public parameter X, AONT Block File b_i, key K_1
Ensure: A set of random points on the elliptic curve E
$b_i' = \text{Pad}(b_i)$
m[] = Fragment(b_i')
for i: 1 to l **do**
Q = MessageToPoint($m[i]$)
P[i] = Q ⊕ G(X⊕ i,K_1)
Output P
end for

3) **Re-encryption Phase:** In this phase, the system admin re-encrypt the files with a new homomorphic key K2 if it wants to revoke access to its personal. A proxy key must be created by the patient A and given to the patient B. The system must use the re-encryption key K=K1-K2 to re-encrypt the ciphertexts. In algorithm, the comprehensive reencryption Algorithm [1] is presented.

Algorithm 4: Re-Encryption

Require: Elliptic curve E, public parameter X, data points $R_i[]$,Re-Encryption key ΔK
Ensure: Set of Elliptic Curve Points $R_i'[]$
for i: 1 to l **do**
$R'[i]$ = R[i] ⊕ G(X,ΔK)
end for
Output $R'[]$

[1]) **Decryption Phase:** The patient A retrieves the patient's B health records during the decryption step. It makes use of the same homomorphic pseudorandom function and an asymmetric proxy re-encryption scheme with the key K1 shared to it. To get the appropriate AONT block, the output of the KH-PRF is subtracted from the encrypted point. All of the AONT blocks that were subsequently acquired were combined to recreate the health record as depicted in Figure 41. Algorithm 5 provides a complete decryption procedure algorithm.

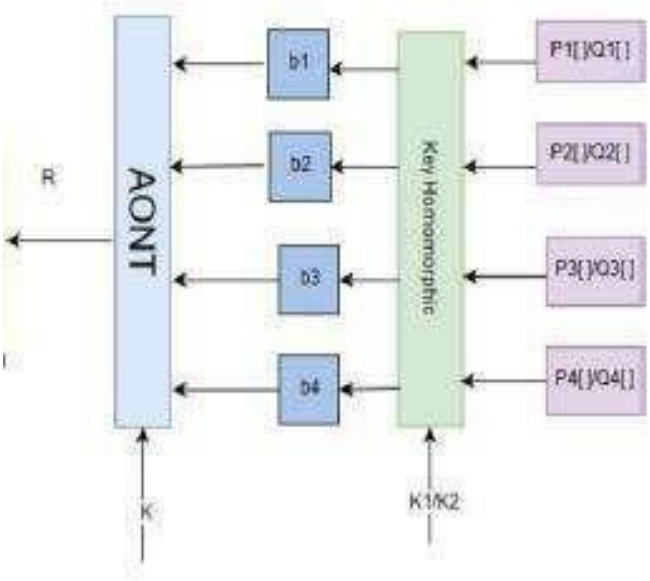

Figure 41: Decryption phase

Algorithm 5: Decryption

Require: Elliptic curve E, public parameter X, data points R, key K_1

Ensure: AONT Block File b_i

for i: 1 to l **do**

$Q = R[i] \oplus G(X \oplus i, K_1)$

$m[i] = \text{PointToBlock}(Q)$

end for

Output $b_i = m[1]||m[2]..m[l]$

4.3.2 Flow of the Application

Flow diagram of the application is shown in Figure 42.

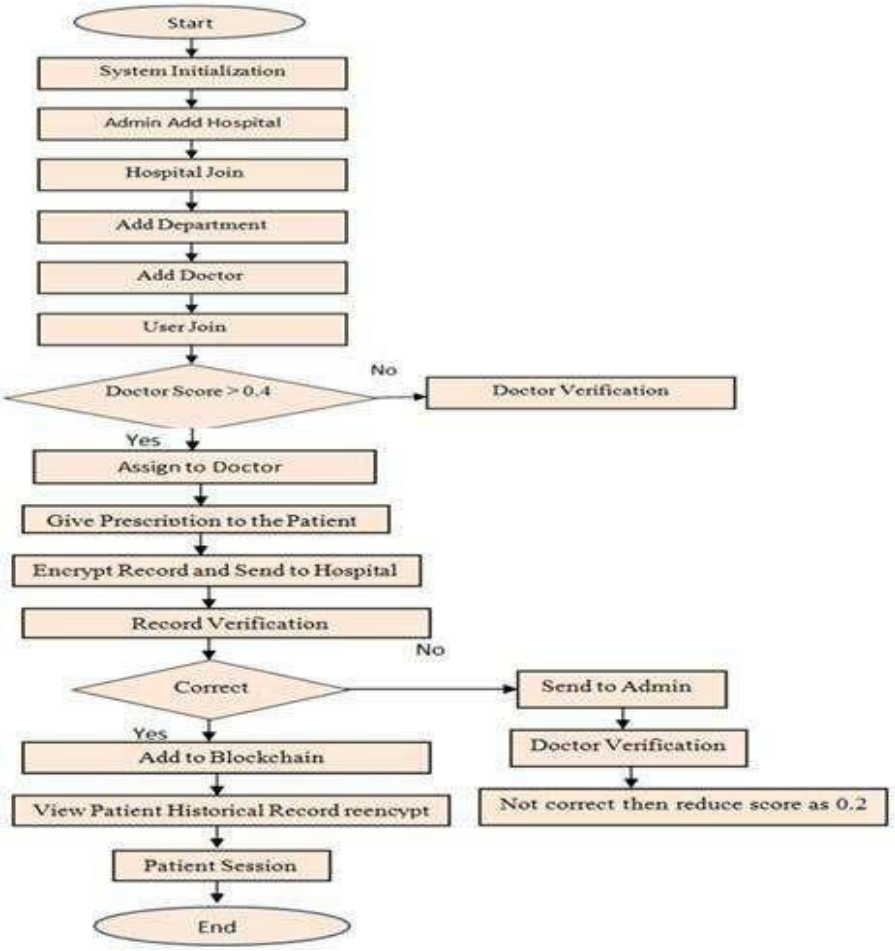

Figure 42: Flow diagram of the application

4.5 IMPLEMENTATION DETAILS

The next subsections provide the technical specification, security and performance analyses.

4.4.1 Technical specification

We have tested the suggested framework's functionality through experiments using the following configurations:
Language: Java

Platform: Netbeans 8.2

Back End: Mysql (SQLYog)

4.4.2 Result and Analysis

Some implementation steps and screen shorts are shared in this section as shown in Figure 43-55.

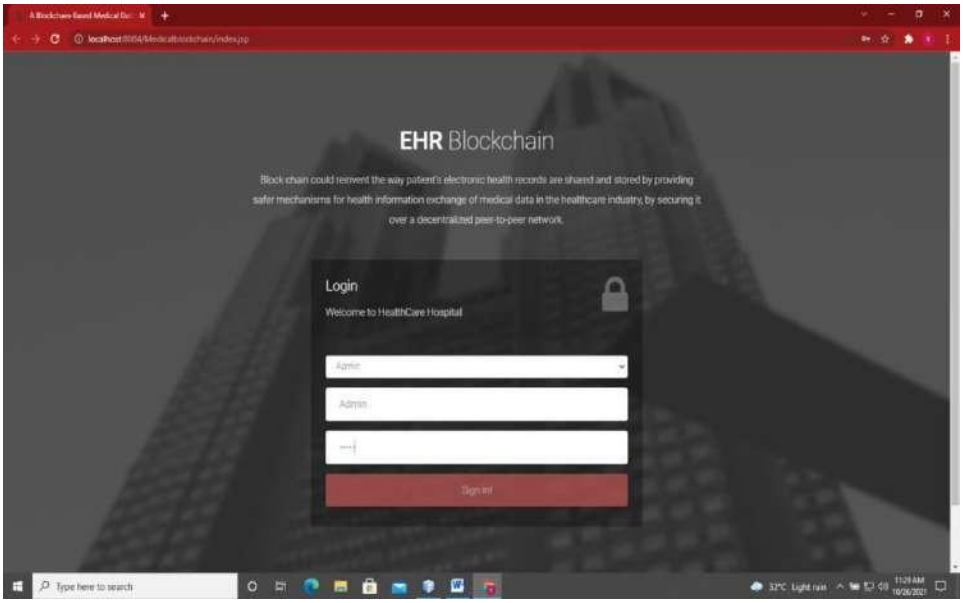

Figure 43: Admin page

Figure 44: Dashboard

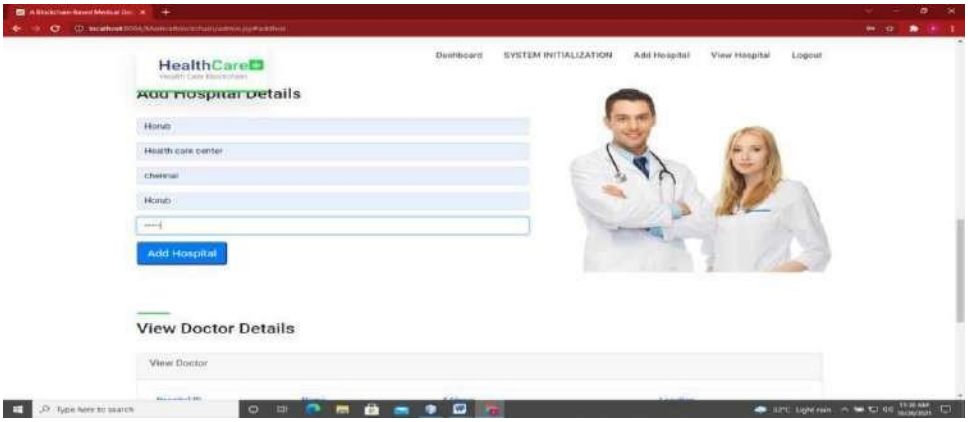

Figure 45: Add Hospital

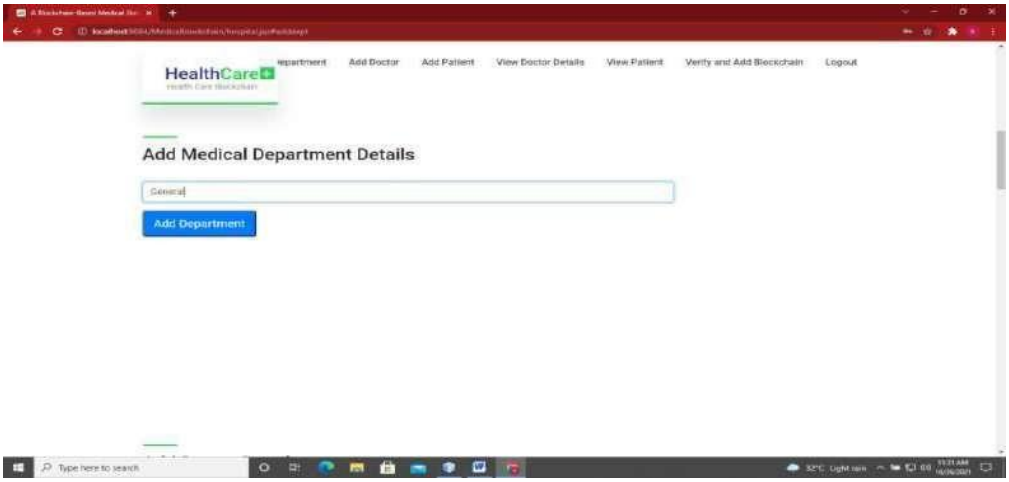

Figure 46: Add medical department

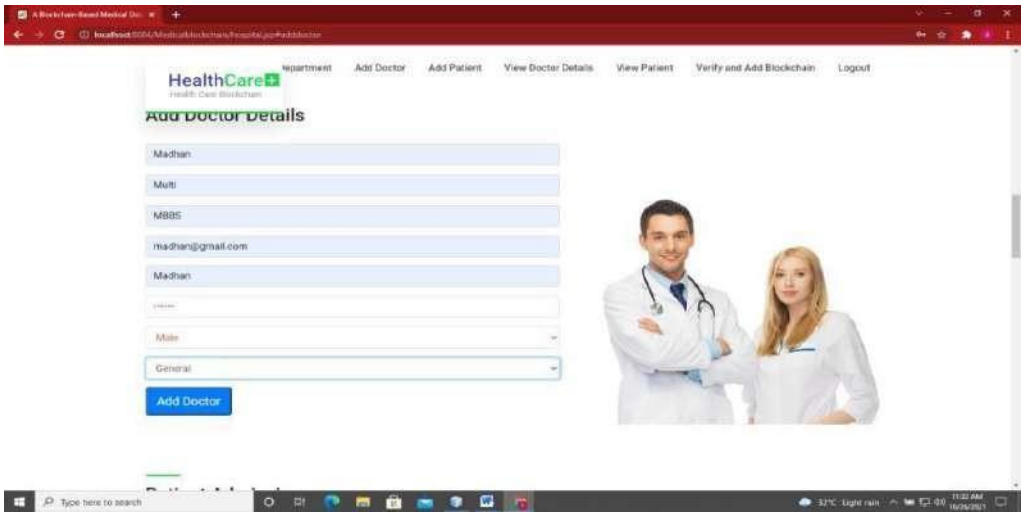

Figure 47: Add Doctor

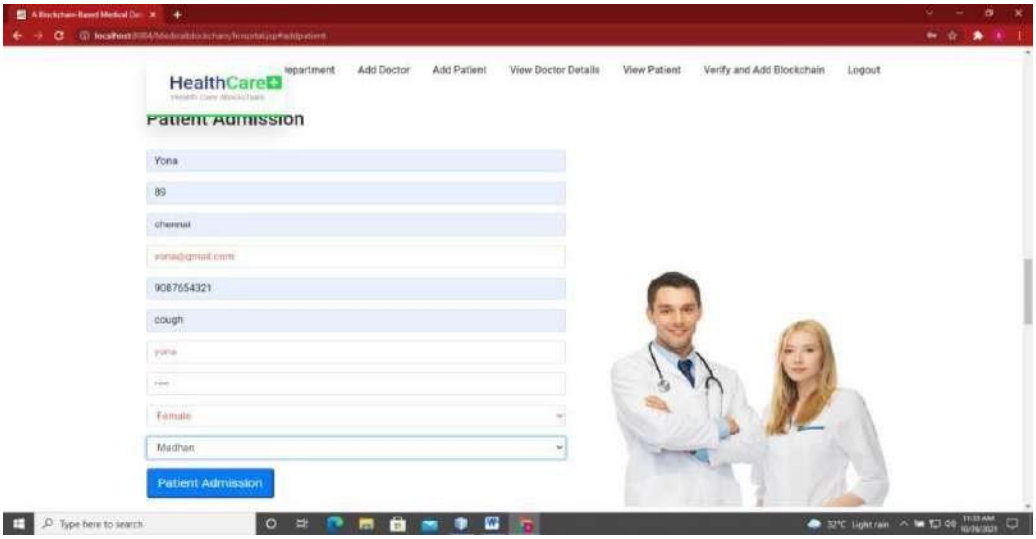

Figure 48: Add patient details

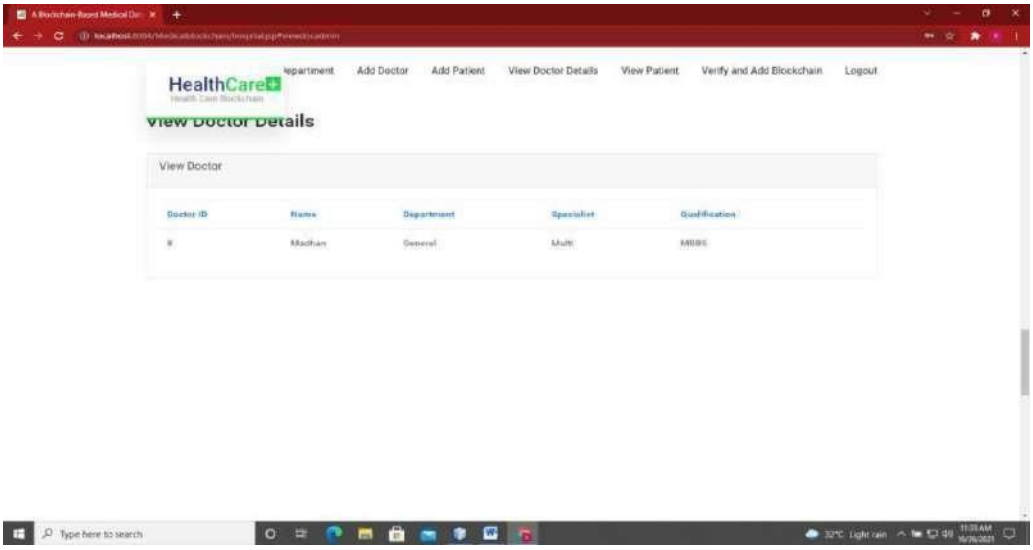

Figure 49: View doctor details

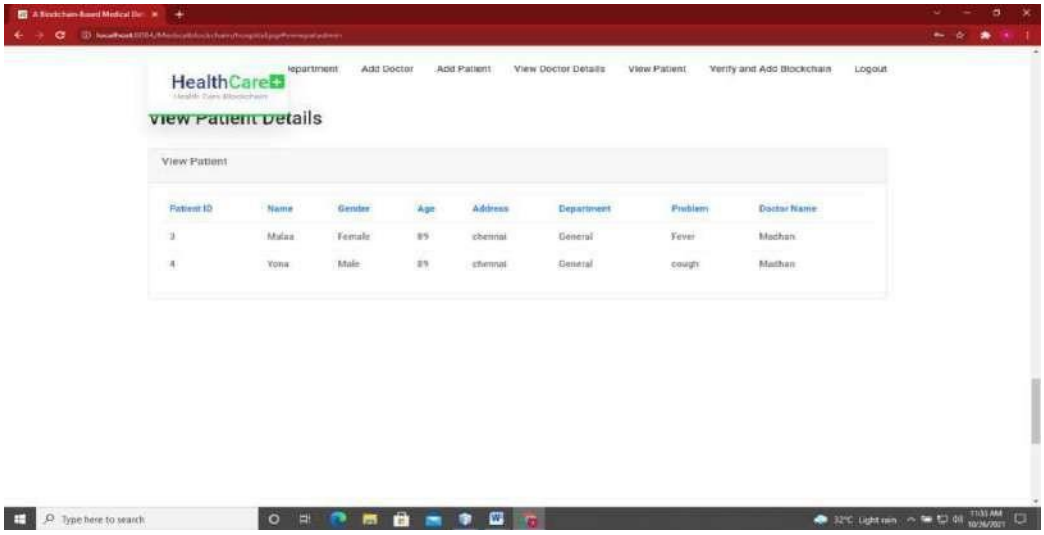

Figure 50: View Patient Details

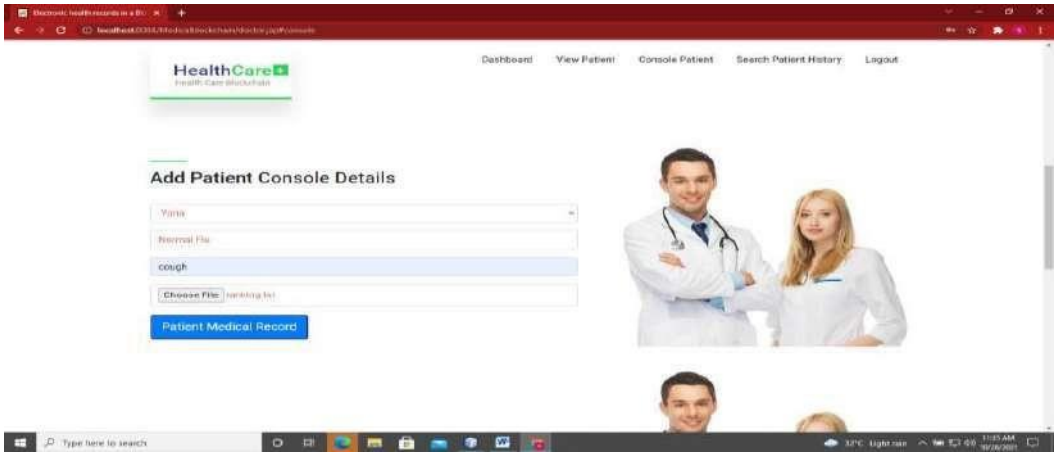

Figure 51: Add Patient Console

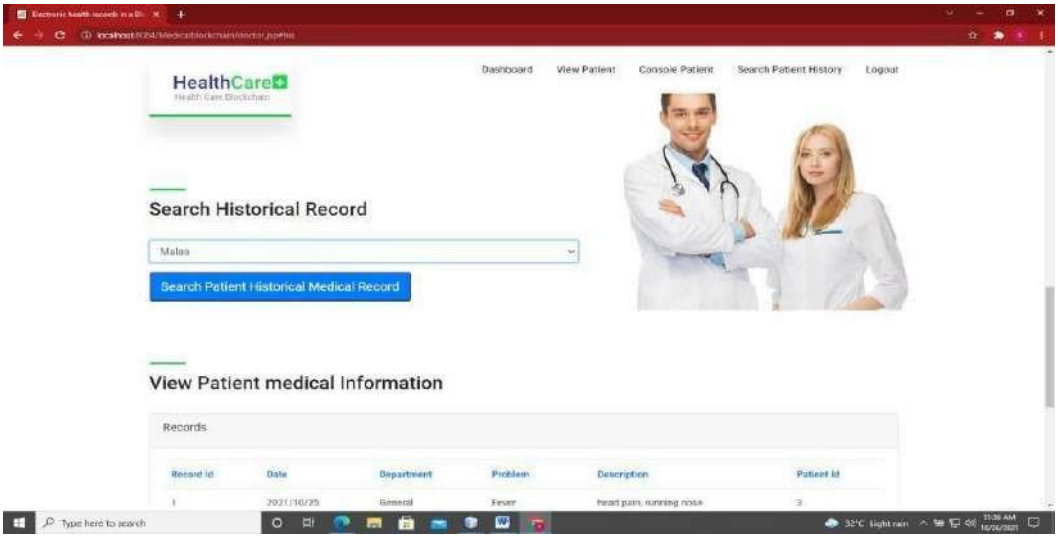

Figure 52: Search patient historical record

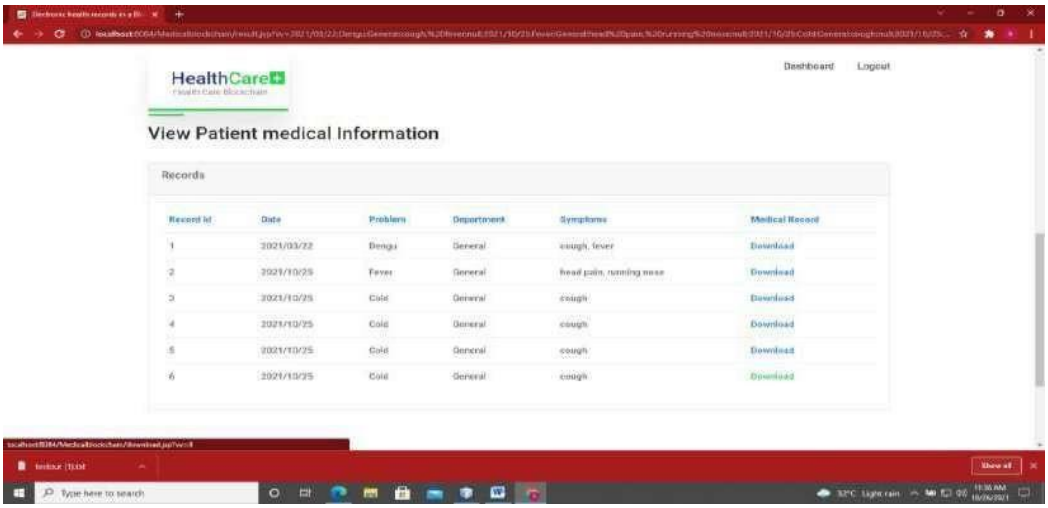

Figure 53: download all historical record

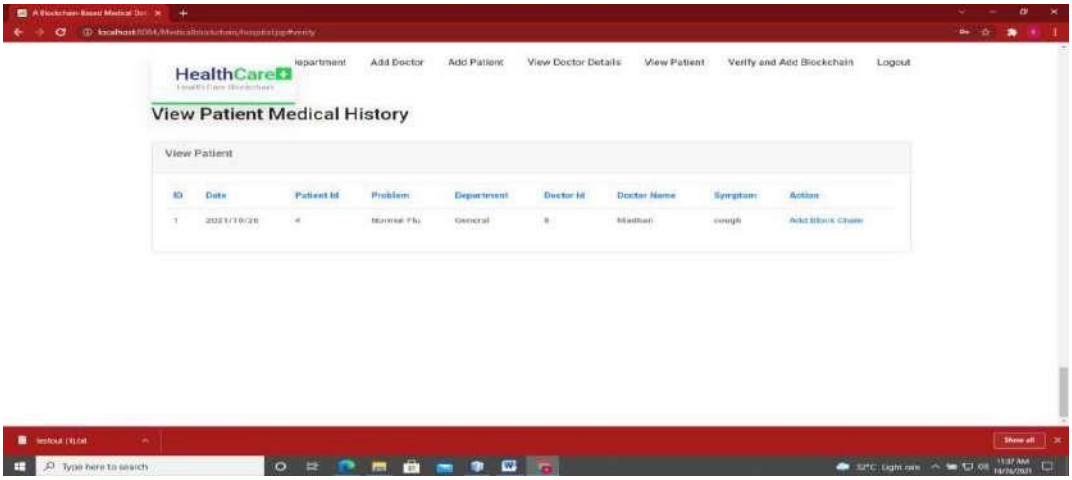

Figure 54: All health record added to blockchain

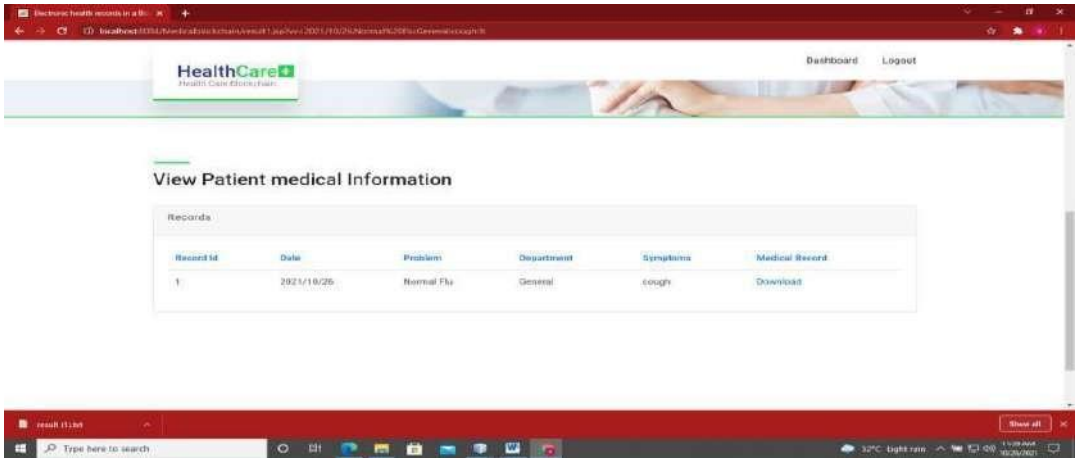

Figure 55: All health record added to blockchain

4.4.2.1 Security Analysis

File Confidentiality: Here, the demand for file secrecy is examined in relation to the threat model's three antagonists: the patient, the doctor, and the system admin. The symmetric key as well as the KH encryption key is not in the possession of an unauthorised user. Therefore, it is unable to restore the original file or the AONT blocks. Key homomorphic PRF is used to encrypt the AONT blocks. However, PRF's output is arbitrary. Adversary is unable to decrypt the block.

The Decision Diffie Hellman assumption holds in the chosen EC group G. Both the symmetric key and the revoked KH Encryption key Ki may be in the possession of the revoked usage r. Let ki+t be the current KH Encryption key. Because the KH Encryption keys are created at random, the only algorithm to deduce ki+t from Ki is a brute force search or breaking the PRF's security. Both the symmetric key and the PRF key are encrypted before being placed in the cloud storage. Only the re encryption token, which is a component obtained by deducting two random numbers from the field Zp, is accessible to the system admin.

File Integrity: The PRF key is initially obtained by the authorised user. To generate the hash output hF, it concatenates the AONT blocks and uses SHA 256. The original symmetric key K used for AONT is then obtained by performing an XOR operation on the hF and the encrypted symmetric key. To generate the hash output hF, it concatenates the AONT blocks and uses SHA 256. The encrypted symmetric key is then XORed with the hF to produce the original symmetric key K used for AONT. If the key turns out to be invalid. As a result, it is impossible to reconstruct the original file from a modified ciphertext. As a result, the approved service provider can check the file's integrity. Once more, AONT encryption makes certain that none of the original AONT blocks are missing.

Re encryption Indistinguishably: Let C1 and C2 be the respective ciphertext blocks and K1 and K2 be the two keys used to encrypt the file block "b". Allow G(X, K1) and G's outputs, r1 and r2, to be (X, K2). Since r1 and r2 are random numbers, they will produce two random blocks when added to the data block. As a result, the adversary is unable to determine the encryption keys that match the encrypted blocks.

4.4.2.2 Performance Analysis

In Table 11, the symbols utilized in the study of the suggested system are included along with a brief description. It takes time to map each block of K bits to a point on an elliptic curve, run the homomorphic function (scalar point multiplication), add points, and then map the encrypted point back to a block during the encryption procedure. The identical set of operations is carried out during the decryption process by admin using the additive inverse of the homorphic key. The methods for proxy re encryption using admin are identical to those for encryption, with the exception that the key to be utilized is the re encryption key.

Symbol	Meaning
Tbp	It's time to connect a data block with an EC point.
Tpb	It's time to connect an EC point with a data block
Tm	Time to multiply a point using a scalar
Ta	Time to add a point
Tz	It's time to add or subtract in the ZP group.

Table 11: Description with symbols

The proposed FileReCrypt scheme's time requirements for the Encrypt, Decrypt, Re Encrypt, and Decrypt operations are shown in Table 12.

Operation	File size		
	1KB	1MB	1GB
Encrypt	4.24 ms	4.92 s	1.10 hrs
Decrypt	4.31 ms	6.31 s	1.84 hrs
Re-encrypt	4.32 ms	6.62 s	1.86 hrs

Table 12: Time taken by different operations

For implementation, the SECG curve "prime256v1" is utilized. It has been found that the time required for certain operations increases linearly with the number of blocks in the file. Different procedures have longer durations than AES encryption. This is because ECC uses public key cryptography, which has more expensive cryptographic primitives than its symmetric key counter-part. The suggested system is therefore appropriate for exchanging files up to a few MB. Typically, files that are shared for verification purposes are under a few MB in size. The suggested method thus satisfies the application scenario's performance requirements.

4.6 CONCLUSION

Personal EHR file sharing is crucial for document verification in many real-world situations. It is crucial that these documents remain confidential. Files need to be encrypted in order to accomplish this security characteristic. The files must be proxy re encrypted in order to achieve safe sharing and revocation. The AONT transformation with homomorphic proxy re-encryption approach has been used in the suggested study. According to the performance analysis, this technique can be used to share files up to a few MBs in size. These methods can be enhanced in future work to handle huge volume files.

REFERENCES

[1] A. K. Jha, D. Doolan, D. Grandt, T. Scott, and D. W. Bates, ``The use of health information technology in seven nations," *Int. J. Med. Inform.*, vol. 77, no. 12, pp. 848_854, 2008.

[2] Y. Guo and C. Liang, ``Blockchain application and outlook in the banking industry," *Financial Innov.*, vol. 2, no. 1, p. 24, 2016.

[3] Y. Yuan and F.-Y. Wang, ``Blockchain: The state of the art and future trends," *Acta Autom. Sinica*, vol. 42, no. 4, pp. 481_494, 2016.

[4] B. Shickel, P. J. Tighe, A. Bihorac, and P. Rashidi, ``Deep EHR: A survey of recent advances in deep learning techniques for electronic health record (EHR) analysis," *IEEE J. Biomed. Health Inform.*, vol. 22, no. 5, pp. 1589_1604, Sep. 2018. [5] G. S. Birkhead, M. Klompas, and N. R. Shah, ``Uses of electronic health records for public health surveillance to advance public health," *Annu. Rev. Public Health*, vol. 36, pp. 345_359, Mar. 2015.

[6] F. G. Li, Y. N. Han, and C. H. Jin, ``Cost-effective and anonymous access control for wireless body area networks," *IEEE Syst. J.*, vol. 12, no. 1, pp. 747_758, Mar. 2018.

[7] M. M. Hassan, K. Lin, X. Yue, and J.Wan, ``A multimedia healthcare data sharing approach through cloud-based body area network," *Future Gener. Comput. Syst.*, vol. 66, pp. 48_58, Jan. 2017.

[8] J. J. P. C. Rodrigues, I. de la Torre, G. Fernández, and M. López-Coronado, ``Analysis of the security and privacy requirements of cloud-based electronic health records systems," *J Med. Internet Res.*, vol. 15, no. 8, pp. 418_426, 2013.

[9] M. Preethi and R. Balakrishnan, ``Cloud enabled patient-centric EHR management system," in *Proc. IEEE Int. Conf. Adv. Commun., Control Comput. Technol.*, Ramanathapuram, India, May 2014, pp. 1678_1680.

[10] F. Xhafa, J. Feng, Y. Zhang, X. Chen, and J. Li, ``Privacy-aware attributebased PHR sharing with user accountability in cloud computing," *J. Super comput.*, vol. 71, no. 5, pp. 1607_1619, 2015.

[11] R. L. Rivest, "All-or-nothing encryption and the package transform," in International Workshop on Fast Software Encryption. Springer, pp. 210–218, 1997.

[12] D. R. Stinson, "Something about all or nothing (transforms)," Designs, Codes and Cryptography, vol. 22, no. 2, pp. 133–138, 2001.

[13] Canda, Valer, and Tran Van Trung. "A new mode of using all-or-nothing transforms." In Proceedings IEEE International Symposium on Information Theory, p. 296. IEEE, 2002.

[14] Desai, Anand. "The security of all-or-nothing encryption: Protecting against exhaustive key search." In Annual International Cryptology Conference, pp. 359-375. Springer, Berlin, Heidelberg, 2000.

[15] M. Mambo and E. Okamoto, "Proxy cryptosystems: Delegation of the power to decrypt ciphertexts," IEICE transactions on fundamentals of electronics, Communications andcomputer sciences, vol. 80, no. 1, pp. 54–63, 1997.

[16] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in International Conference on the Theory and Applications of Cryptographic Techniques. Springer, pp. 127–144, 1998.

[17] M. Jakobsson, "On quorum controlled asymmetric proxy re-encryption," in International Workshop on Public Key Cryptography. Springer, pp. 112–121, 1999.

[18] Y. Dodis, "Proxy cryptography revisited," in Proc. 10th Annual Network and Distributed System Security Symposium-NDSS'03, 2003.

[19] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," ACM Transactions on Information and System Security (TISSEC), vol. 9, no. 1, pp. 1–30, 2006

[20] Cool, D. L., and Angelos D. Keromytis. "Conversion and proxy functions for symmetric key ciphers." In International Conference on Information Technology: Coding and Computing (ITCC'05)-Volume II, vol. 1, pp. 662-667. IEEE, 2005.

[21] S. Hirose, "On re-encryption for symmetric authenticated encryption," in Computer Security Symposium (CSS), 2010

[22] A. Syalim, T. Nishide, and K. Sakurai, "Realizing proxy re-encryption in the symmetric world," in International Conference on Informatics Engineering andInformationScience. Springer, pp. 259–274, 2011.

[23] S. Myers and A. Shull, "Efficient hybrid proxy re-encryption for practical revocation and key rotation." IACR Cryptology ePrint Archive, vol. 2017, p. 833, 2017.

[24] D. Boneh, K. Lewi, H. Montgomery, and A. Raghunathan, "Key homo-morphic prfs and their applications," in Annual Cryptology Conference. Springer, pp. 410–428, 2013.

[25] B. Kaliski, "Pkcs# 7: Cryptographic message syntax version 1.5," Tech. Rep., 1998.