http://doi.org/10.64771/jsetms.2024.v01.i01.bFt9ZQ

SESPHR: A METHODOLOGY FOR SECURE SHARING OF PERSONAL HEALTH RECORDS IN THE CLOUD

Dr A AVANI[1], Dr T BHARATH KRISHNA[2], Dr D PRANEETH[3], N SUSHMA[4]

[1, 2,3]Associate Professor, Department of Computer Science and Engineering, Anu Bose Institute of Technology for Women's, KSP Road, New Paloncha, Bhadradri Kothagudem District, Telangana (TS), 507115

[4]Assistant Professor, Department of Computer Science and Engineering, Anu Bose Institute of Technology for Women's, KSP Road, New Paloncha, Bhadradri Kothagudem District, Telangana (TS), 507115

Submitted: 05-09-2024 Accepted: 15-10-2024 Published: 22-10-2024

Abstract

The broad use of cloud-based services in the healthcare industry has made it possible for various participating entities of the e-Health systems to exchange personal health records (PHRs) at a low cost and with ease. However, putting the private health data on cloud servers leaves it open to theft or disclosure, necessitating the creation of procedures that protect the PHRs' privacy. Consequently, we suggest a technique named SeSPHR for PHR cloud sharing that is secure. The SeSPHR system makes sure that PHRs are controlled from a patient-centric perspective and maintains the maintaining the PHRs' privacy. Patients keep encrypted PHRs on unreliable cloud servers and only give certain people access. on various PHR sections, to distinct categories of users. a setup and re-encryption server, a semi-trusted proxy (SRS) is used to create the re-encryption keys and to create the public/private key pairings. Additionally, the process is safe. It implements a forward and backward access control and protects against insider risks. Additionally, we officially evaluate and confirm the use of the SeSPHR approach using high-level petri nets (HLPN). Evaluation of performance in relation to time Consumption suggests that the SeSPHR approach may be used for safely transferring PHRs to the cloud.

I. INTRODUCTION

In order to provide ubiquitous and on-demand availability of different resources in the form of hardware, software, infrastructure, and storage, LOUD computing has developed as a significant computing paradigm [1, 2].

As a result, the cloud computing paradigm helps enterprises by relieving them of the time-consuming task of developing infrastructure and encouraging them to rely on outside Information Technology (IT) services [3]. Additionally, the cloud computing architecture has shown tremendous promise for improving coordination among many healthcare stakeholders and for guaranteeing scalability and ongoing availability of health information [4, 5]. Additionally, the cloud computing connects a number of significant healthcare domains, including patients, hospital staff, including physicians and nurses, pharmacists, and clinical laboratory staff, as well as insurance companies and service providers [6]. As a consequence, a collaborative and cost-effective health ecosystem where patients may easily establish and maintain their Personal Health Records (PHRs) develops as a result of the integration of the aforementioned organisations [7].

The PHRs often include data like: (a) demographics, (b) medical history, including diagnoses, allergies, surgeries, and treatments, (c) laboratory results, (d) information on health insurance claims, and (e) patient-only notes regarding specific significant observed health issues [8].

More technically, PHRs are controlled through Internet-based technologies, allowing individuals to manage their health information as permanent records that can be accessed by those who need it [9]. As a result, PHRs make it possible for people to successfully communicate with medical professionals in order to describe their symptoms, ask for guidance, and maintain their health records for proper diagnosis and treatment.

Despite the benefits of the scalable, adaptable, affordable, and widespread services provided by the cloud, a number of issues linked to the privacy of health data also come up. The use of the cloud to distribute and store PHRs is a crucial factor in patients' concerns about the confidentiality of such records [10]. Private health information stored on cloud servers run by third parties is vulnerable to intrusion. Particularly jeopardised is the privacy of PHRs kept in public clouds run by for-profit service providers [11]. The PHRs' privacy may be under danger in a number of ways, including theft, loss, and leaking [12]. Because of the malevolent actions of other entities, the PHRs in cloud storage, in transit from the patient to the cloud, or from the cloud to any other user, may be vulnerable to unauthorised access. Additionally, there are occasional threats made against the data by real insiders [13]. For instance, due of the nefarious actions of other organisations, the PHRs in cloud storage, in transit from the patient to the cloud, or from the cloud to any other user, may be vulnerable to illegal access [10]. People who work for the cloud service provider may act maliciously. The episode in which a U.S. Department of Veterans Affairs employee took home without authority the private health information of over 26.5 million people is a well-known illustration of that [14].

The Health Insurance Portability and Accountability Act (HIPAA) requires that patients' consent and the terms of use and disclosure be followed in order to maintain the integrity and confidentiality of electronic health information kept by healthcare providers [15]. Additionally, the PHRs should be encrypted when being kept on third-party cloud storage so that neither the cloud server providers nor unauthorised parties may access the PHRs. The PHRs should only be accessible to entities or people who have the "right-to-know" privilege. To prevent unauthorised alterations or abuse of data when it is transferred to the other stakeholders in the health cloud environment, the mechanism for granting access to PHRs should be managed by the patients themselves.

The privacy of PHRs kept on cloud servers has been protected in a variety of ways. Confidentiality, integrity, authenticity, accountability, and audit trial are ensured by privacy-preserving methods.

While integrity concerns with preserving the originality of the data, whether in transit or in cloud storage, confidentiality guarantees that the health information is completely hidden from unauthorised parties [14].

Accountability refers to the need that data access regulations follow the established protocols, while authenticity ensures that the health data is only accessible by authorised parties. The term "audit trial" refers to the process of observing how health data is used even after access to it has been given [6].

We provide a way for managing the PHR access control system that is controlled by patients themselves, dubbed Secure Sharing of PHRs in the Cloud (SeSPHR).

The approach limits unauthorised users to protect the PHRs' confidentiality. In the suggested approach, there are typically two categories of PHR users: (a) patients or PHR owners; and (b) users of PHRs who are not owners, such as patients' family members or friends, physicians, health insurance company representatives, pharmacists, and researchers.

By selectively providing people access to certain PHR sections, patients who are the PHRs' owners are allowed to upload encrypted PHRs to the cloud. Depending on their job, each member of the group of users of the latter kind is given access to the PHRs to a certain degree by the PHR

owners. The PHR owner defines the degrees of access given to different user groups in the Access Control List (ACL).

For instance, the owner of the PHRs may provide complete access to the patient's family members or acquaintances. Similar to this, insurance company personnel would only be allowed to see the PHR sections that include information concerning health insurance claims, with access to other personal medical information, such the patient's medical history, being blocked for these users.

The SeSPHR methodology avoids the overhead by delegating the SRS for setting up the public/private key pairs and producing the decryption keys for the authorised users only. In contrast to the approach proposed in [10], which suggests that the PHR owners manage multiple keys, this approach avoids overhead by proposing that the PHR owners produce the decryption keys. This ultimately leads to overhead at the PHR owner's end. The Setup and Reencryption Server (SRS), a semi-trusted server, is used as the proxy since the approach views cloud servers as an untrusted entity. For the SRS to generate the re-encryption keys for safe sharing of PHRs across users, a proxy reencryption-based technique is utilised.

Patients or PHR owners encrypt the PHRs, and only authorised users with keys provided by the SRS may decode the PHRs. Additionally, the users are given access to the PHRs' particular sections that the PHR owner deems to be crucial. The proposed method is secure compared to previous constructs since the PHR data is never sent from the SRS in the proposed framework. Instead, it is the SRS's duty to maintain the keys, with PHR owners handling encryption tasks, and requesting users handling decryption tasks, provided they have access to valid decryption keys.

The forward and backward access controls are likewise enforced by the suggested method. The keys are obtained by the newly joining members of a certain user group from the SRS. Only the owner's keys are used to encrypt the shared data.

After receiving the PHR owner's consent, newly joining members are given access to the data. The corresponding keys for a departing user are also destroyed, and that user is also removed from the ACL. Any unauthorised access attempts made after the user has left are denied access to the PHR due to the deletion of the user keys and removal from the ACL. We also used High Level Petri Nets (HLPN) and the Z language to do a formal examination of the suggested design.

The HLPN is used to both imitate the system and to provide the mathematical characteristics that are later utilised to analyse the behaviour of the system. The Z3 solver and the Satisfiability Modulo Theories Library (SMT-Lib) are used to carry out the verification. To carry out the work of verification using the SMT, the petri net model is first translated into the SMT together with the specified properties, and then the Z3 solver is used to check whether the properties are true or false.

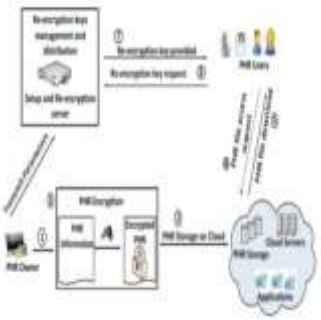

Fig-1: Architecture of the proposed SeSPHR methodology

The following are the main contributions of the suggested work:

1. SeSPHR, a technique we offer, enables patients to control the sharing of their own PHRs in the cloud.

2. To maintain PHR secrecy, the SeSPHR approach uses proxy re-encryption and El-Gamal encryption.
3. Based on the access level established in the ACL for various user groups, the approach enables PHR owners to selectively provide users access to users over the sections of PHRs.
4. To provide access control and to produce the reencryption keys for various user groups, a semi-trusted proxy named SRS is implemented, removing the burden of key management from the PHR owner's end.
5. The suggested technique also uses forward and backward access control.
6. The suggested approach is formally analysed and verified to ensure that it operates in accordance with the requirements.

II.RELATED WORK

The existing works that are related to the proposed work are presented in this section. By sending the Personally Identifiable Information individually, the authors in [28] developed a public key encryption-based technique to maintain the anonymity and unlinkability of health information in a semitrusted cloud (PII). The Cloud Service Provider (CSP) saves the health record and the location of the file (index), and later encrypts them using symmetric key encryption.

The patients encrypt the PHRs by the patients using the public key of the CSP, and the CSP decrypts the record using the private key. By associating the location and the master key, the administrative control of the patient on the PHRs is kept in place. The approach's drawback is that it enables the CSP to decrypt PHRs, which may then be used maliciously. The SRS, on the other hand, is a semi-trusted authority that decrypts the ciphertext created by the PHR owner and provides keys to the users that request access to the PHRs.

In a multi-user cloud setting, Chen et altechnique .'s [12] uses the SKE and the Lagrange Multiplier to dynamically exercise access control on PHRs. The approach's primary characteristics include automatic user revocation. A partial order link between the users is kept in order to get around the challenges of other key management. However, the system requires the PHR owners' online presence in order to give or cancel access. Our suggested technique does not need the PHR owners to be online in order to offer the access over PHRs, in contrast to the plan described in [12]. Instead, the semi-trusted authority chooses the users' access rights and, upon successful authorisation, decides the re-encryption keys for the users making the request.

To provide patient-centric access control, the authors in [29] developed a Digital Right Management (DRM)-based solution. The writers used Content Key Encryption (CKE) to encrypt the data, and only users with valid licences are allowed access. The first proxy re-encryption technique was put out in [33]. Unlike our policy, which is based on keys and has no effect on the size of the ciphertext, the policy in [33] is based on ciphertext, and the size of the ciphertext rises linearly with multi-use usage. This is because the [33] needs a step that is missing from our methodology—re-encryption. Li et al. [14] offer a method for sharing PHRs in multi-owner settings that are separated into several domains that uses attribute-based encryption (ABE).

The technique initially presented in [33] serves as the foundation for the suggested methodology. After a given user's access has been revoked, the method re-encrypts the PHRs using the proxy re-encryption mechanism (s). The method successfully reduces the complexity and expense of key management while also improving the phenomena of on-demand user revocation. Despite being scalable, the method is unable to handle situations when granting access permissions based on users' identities is necessary.

To guarantee user responsibility, Xhafa et al. [30] also applied Ciphertext Policy ABE (CPABE). In addition to preserving user privacy, the suggested method has the ability to track down users who misbehave and illegitimately share their decryption keys with other users.

Presents a method for ensuring both the secrecy and fine-grained access to the healthcare data that has been contracted out to cloud servers. By using proxy re-encryption, Key Policy ABE (KP-ABE), and lazy re-encryption, the expensive duties of re-encrypting data files, updating secret keys, and preventing users whose access has been revoked from learning the contents of the data are handled. The re-encryption of data files and subsequent storage in the cloud environment are responsibilities assigned to the cloud servers. However, the data owner is also expected in the proposed framework to be a reliable authority who controls the keys for several owners and users.

Therefore, managing several keys for various attributes for many owners would be inefficient at the PHR owners' end. Because the functions of key creation and key distribution to various user types are carried out by the semi-trusted authority, our technique eliminates overhead. In order to provide fine-grained access control, the authors in [31] and [32] also employed proxy re-encryption-based techniques. The system we provide allows owners to encrypt PHRs before putting them in the cloud and adds a semi-trusted authority that re-encrypts the ciphertext without knowing what's within the PHRs. The PHRs can only be decrypted by authorised users who possess decryption keys issued by the semi-trusted authority.

III.THE SESPHR PROPOSED METHODOLOGY

The recommended method makes use of proxy re-encryption to provide PHR confidentiality and exchange security across public clouds. The architecture of the suggested SeSPHR technique is shown in Fig. 1.

Persons The recommended method for exchanging PHRs in a cloud environment involves the Setup and Re-encryption Server (SRS), the cloud, and the users. An overview of each of the entities is provided below.

The cloud The strategy advises PHR owners to save their data in the cloud so they may subsequently safely share it with other users. Users assume that the cloud is an unreliable source when they upload or download PHRs to or from cloud servers. No changes to the cloud are necessary since both types of users are the only ones that upload and download PHRs in the way stated.

Setting up and installing the SRS: Every system user's public/private key pairs must be generated by the SRS, a semi-trusted server. The SRS further generates the re-encryption keys in order to safely transfer PHR among several user groups. The SRS is regarded as a semitrusted entity in the recommended method. In light of this, we draw the conclusion that it is honest and generally follows the law, but odd. The SRS monitors the keys, but it never gets PHR information. Operations for encryption and decryption are completed at the endpoints of the users. The SRS offers key management in addition to access control for the shared data.

Due of the public cloud's unreliability, the SRS is a standalone server that cannot be installed there. The SRS may be managed by a group of institutions or by a respectable third-party organisation for the benefit of the patients. It could also be maintained by a group of connected patients. However, SRS maintained by hospitals or a group of patients might inspire higher trust due to the involvement of medical specialists and/or the patients' self-control over SRS.

Users: Patients (owners of PHRs who wish to securely share their PHRs with others) and patients' family members or friends, doctors, representatives of health insurance companies, pharmacists, and researchers are the two main groups of users of the system. Friends and relatives are classed as private domain users under the SeSPHR methodology, whereas all other users are categorised as public domain users.

PHR owners may provide users access to PHRs in the public and private domains to varying degrees. Users who come under the private domain, for example, may have complete access to the PHR, but those who fall under the public domain, such as physicians, scientists, and pharmacists, would only have access to a limited number of PHR parts. The aforementioned users may also be

granted full access to the PHRs if the PHR owner determines it is required. In other words, the SeSPHR approach allows patients to impose precise access control over PHRs.

Every system user must register with the SRS in order to access the SRS's services. As a doctor, researcher, or pharmacist, for example, the registration procedure is dependent on the user's responsibilities.

HR Partitioning

The four sections listed below are logical divisions of the PHR:

Personal information, health-related information, insurance information, and information on prescription drugs;

It is crucial to note that the aforementioned division is flexible. The PHR may be divided into fewer or more divisions at the user's discretion. The PHRs are represented in a number of formats, including XML, and may be simply separated into pieces. The PHR owner also has the option of giving many partitions the same level of access control. Some PHR partitions may include user restrictions, meaning that a particular user may not have full access to the health data.

For instance, a pharmacist may not have access to personal or medical information, but they may be given prescription and insurance-related information. Full access to the PHR may also be given to family members and friends. A researcher could only need access to the patient's medical records once the personal data has been deleted. The PHR owner grants the SRS access rights to each of the various PHR partitions when data is uploaded to the cloud.

The Proposed Methodology's Approach Functions

The suggested SeSPHR technique consists of the following steps: setup, key creation, encryption, and decryption. The parts that follow go through each action:

Setup

The offered approaches work well with the G1 and G2 groups with the prime order q. G1 G1 G1 and G2 are bilinearly mapped to form G2. A random number generator where g G1 has g as a parameter. Z is used as a second random number generator using the formula Z = e(g, g) G2.

Key Generation

Public/private key pairs are created by the SRS for the set of authorised users.

Encryption

Imagine that patient P is required to upload their PHR to the cloud. The PHR partitions that the user has allocated to the different access level groups are represented by a random number or numbers that are generated by the patient client application. In our case, we take into consideration that the access levels for each of the four partitions specified in Section 3.2 vary. As a consequence, four random variables are created in our example: r1, r2, r3, and r4 (Zq). The variable ri is used to encrypt the i-th partition of the PHR. Each partition is encrypted separately by the client programme. Thanks to the XML structure, the application can quickly perform encryption and decryption on the PHR's logical partitions. The partitions stated above in the PHR are encrypted as seen below.

$Cper = Zr1$. PHRper (6), where PHRper only refers to the personal partition of the PHR and Cper stands for the partly encrypted file that contains the personal partition's encrypted data.

$Cins = Zr2$. PHRins (7) is the semi-encrypted file that also contains the Cper that was encrypted in the prior stage together with the insurance partition as encrypted text. PHRins only refers to the insurance partition of the PHR.

$Cmed = Zr3$. PHRmed (8), where PHRmed only denotes the PHR's medical information partition and Cmed is a partially encrypted file that also contains the Cper and Cins that were encrypted in earlier stages, together with the insurance partition in encrypted text.

$C = Zr4$. PHRpres (9) only refers to the PHR's section for prescription information. In this case, the letter C stands for the whole encrypted file, which also contains all of the partitions. As a result, we

skipped the subscript in the final encryption phase. It is crucial to remember that the encryption sequence may be changed and that the preceding order is not strictly adhered to.

The client also chooses the following settings in addition to the aforementioned encryptions.

$RperP = gr1xp$ (10)

$RinsP = gr2xp$ (11)

$RmedP = gr3xp$ (12)

Where xp is the patient's private key used to upload the PHR, Rpres P = g r4xp (13). The parameter R is used to produce the reencryption key for the partition indicated in each R's subscript. The user P is the one who created the parameter R, as indicated by the P in the subscript. When the encryption process is complete, the whole encrypted file C is uploaded to a public cloud. The parameters Rper P, Rins P, Rmed P, and Rpres P are provided to the SRS along with the file identification for which they were formed.

Remember that in order to get the aforementioned parameters after registering with the SRS, a patient must provide at least the following information.

The quantity of PHR partitions; the titles of each partition, such as "Personal Health Record" (PHR), "Medical Record," "Insurance Information," and "Prescription Information; (any role may be granted access to more than one partition, such as doctors may be granted access to medical information).

• The first close relative or acquaintance to provide access • If there is any default access for new members

Decryption

Let's assume user U requests access to the patient P's provided encrypted PHR (C). User U downloads the C directly from the cloud after completing the cloud authentication process. The user U requests the SRS to determine and deliver the correct R parameters required for decryption at that point. By looking at the asking user's ACL, the SRS determines if the PHR owner has granted access to the partition for which the user has requested R. Based on the access rights specified in the ACL, the SRS will create and provide the necessary parameters to the requesting user. We shall show how R is produced for each division in the text that follows in order to provide a comprehensive explanation of the procedure. Therefore, we assume that user U has complete access to all partitions. The SRS computes R and sends it to the user U together with the re-encryption key.

The computations for R and the re-encryption keys are listed below:

RKPU is the re-encryption key given by patient P to user U, and it has the mathematical formula RKPU = g xU xP (14) where xU and xP are the private keys of users U and P, respectively. The parameters R for each partition that belongs to user U are then computed using the following formulae.

$RperU = e\ (RKP{\rightarrow}U, RperP$ \s) = $e\ (g$ \s$xU$ \s , g \sr1xP) = $e(g, g)$

The option Rper U, which is pertinent for the user U, is used to decrypt the partition labelled "personal information." Its equation is Z r1 x U = r1 x U. (15). According to Equations 16, 17, and 18, the R parameters for further partitions corresponding to User U are established.

$RinsU = (RKP{\rightarrow}U, RinsP) = e\ (g$ \s$xU$ \s , g \sr2$xP) = e(g, g)$

r2xU = Z sr2 xU (16), sRmed U = e(RKPU, Rmaed P) = e (g sxU sxP, g sr3xP) = se(g, g).

$r3xU = Zr3\ xU$ (17) , \s$RpresU = e(RKP{\rightarrow}U, RpresP) = e\ (g$ \s$xU$ \s$xP$, g \sr4xP) = $e(g, g)$

$r4xU = Z$ \sr4 xU (18)

The aforementioned parameters are given to user U, and U uses them together with the formulas provided below to decrypt each partition.

$PHRper = Cper$ \s$RperU$ \s1 \s$xU$ (19)

$PHRins = Cins$ \s$RinsU$ \s1 \s$xU$ (20)

$PHRmed = Cmed$ \s$RmedU$ \s1 \s$xU$ (21)

$PHRpres = Cpres$ \s$RpresU$ \s1 \s$xU$ (22)

When the last partition has been decrypted, the whole PHR will be available in plain form. As previously stated, the user will only be able to get the R parameter from the SRS for the partition(s) to which access is granted to the asking user.

freshly admitted students

A new member may join the group by signing up with the SRS. After the SRS registers new members in the system in line with their responsibilities, the PHR owner confirms their registration. The SRS creates the public/private key pairings.

Users (new members) securely get their keys.

New members initially get the default access permission set by the PHR owner at the time of registration, depending on the kind of group the member is enrolled in. However, enhanced access capabilities to PHRs are only granted with the PHR owner's permission if a particular user requests them. Additionally, the PHR owner's permission is required in order to add a user to the family/friends category. The ACL is altered once the new user registers and includes their joining date. The joining user is granted access to the files as of the joining date, unless the PHR owner indicates otherwise.

The roles of a doctor, researcher, or pharmacist may be identified using X.509 role-based certificates, also known as X.509 attribute certificates.

X.509 attribute certificates link the identity of the individual with the role, such as a doctor, engineer, or pharmacist, to carry out the authorization function once the authentication mechanism has been successfully employed. Therefore, in the context of SeSPHR, X.509 attribute certificates may successfully identify the role of the aforementioned users in an open connectivity system. Nevertheless, family members and friends may sign up for the system and may be granted access with the PHR owner's consent, as was previously noted.

Dismissing User

When a PHR user is asked to leave for whatever reason, the PHR owner instructs the SRS to revoke the permitted access. The SRS deletes the keys related to the departing user and removes the user from the ACL. The system does not have to update the keys for every user or re-encrypt all of the data.

IV. DISCUSSION OF THE SESPHR METHODOLOGY

The suggested technique gives the PHRs shared through the public cloud the following services.

Backward and forward access control; confidentiality; secure PHR sharing across groups of authorised users; protection of PHRs from unlawful access by legitimate insiders;

The cloud is not regarded as a trustworthy entity in the suggested technique. The shared pool of resources, multi-tenancy, and virtualization that are part of the cloud computing paradigm might lead to a variety of insider and outsider risks to the PHRs that are shared across the cloud. It is crucial that the PHRs be encrypted before being stored on the third-party cloud server. Prior to being transferred to the cloud, the PHR is first encrypted at the PHR owner's end.

The cloud \smerely functions as a storage service in the suggested methodology. Never are the encryption keys or any other control data kept on the cloud. As a result, the secrecy of the data is effectively ensured at the cloud's end. Because the control data does not exist in the cloud and the PHR is guaranteed to remain secret, even if an unauthorised user at the cloud somehow manages to get the encrypted PHR file, the file cannot be decrypted.

The owner encrypts the PHRs before uploading them, and the other users may access the plain data by using the re-encryption key that the SRS calculates. Only the permitted partitions that belong to the requesting user are generated by the SRS when creating the re-encryption settings.

Therefore, a compromised valid group member cannot undermine the privacy of the whole system.

The PHR owner provides the ACL, which details all the privileges relating to each user. The privileges are established depending on the user categories and are increased or decreased with the PHR owner's consent. Based on the provided privileges on the partitions, the SRS determines and delivers the re-encryption settings. As a result, even authorised users are unable to access the unlawful partition.

The SRS gives the keys to the newly joined member. Only the owner's keys are used to encrypt the shared data. With the SRS's agreement, newly joining members are given access to the data. Additionally, changing the system's key does not need re-encrypting all of the data. The keys associated with a leaving user are also erased and they are removed from the ACL. Any further unauthorised access attempts will be denied access to the PHR due to the deletion of the user keys and removal from the ACL. Since the suggested approach limits access for leaving users (forward access control) while allowing incoming users to view historical data, it is effectively secure (backward access control).

The SRS is seen as an honest yet inquisitive semi-trusted authority. In general, it is anticipated that the SRS will honestly follow the process. The data, whether encrypted or plain, is never transferred to the SRS, despite the fact that it produces and keeps the key pair for each user. Only key management and generating re-encryption parameters are within the purview of the SRS. Additionally, the SRS also enforces access restriction. However, the suggested methodology's shortcoming and difficulty is maintaining the SRS.

V. RESULTS AND OUTPUTS

Fig- 2: Home page

Fig-3: Public key Results

Fig-4 View Access Control

Fig-5 Clinical Report

VI CONCLUSIONS

We suggested a mechanism for transmitting and storing PHRs in the cloud securely to authorised parties. The technique upholds a patient-centric access control to various PHR subsystems based on the access granted by the patients, while maintaining the privacy of the PHRs.

We put in place a form of fine-grained access restriction so that not even authorised users of the system could access restricted areas of the PHR. Only authorised users with legitimate re-encryption keys supplied by a semi trusted proxy are able to decrypt PHRs, which are stored encrypted by PHR owners in the cloud. The tractor trailer proxy's job is to create and maintain public and private key pairs for the system's users.

The methodology also manages forward and backward identity management for leaving and newly joining users, correspondingly, in addition to maintaining confidentiality and guaranteeing patient-centric access control for PHRs. Additionally, we officially assessed and validated the SeSPHR methodology's operation using the HLPN, SMT-Lib, and Z3 solver. The time it took to generate keys, the activities involved in both encryption and decryption including timely delivery were all taken into account when evaluating performance. The outcomes of the experiment show that the SeSPHR approach may be used to safely exchange PHRs in a cloud context.

REFERENCES

[1] K. Gai, M. Qiu, Z. Xiong, and M. Liu, “Privacy-preserving multi-channel communication in Edge-of-Things,” Future Generation Computer Systems, 85, 2018, pp. 190-200.
[2] K. Gai, M. Qiu, and X. Sun, “A survey on FinTech,” Journal of Network and Computer Applications, 2017, pp. 1-12.

[3] A. Abbas, K. Bilal, L. Zhang, and S. U. Khan, "A cloud based health insurance plan recommendation system: A user centered approach, "Future Generation Computer Systems, vols. 43-44, pp. 99-109, 2015.
[4] A. N. Khan, ML M. Kiah, S. A. Madani, M. Ali, and S. Shamshirband, "Incremental proxy re-encryption scheme for mobile cloud computing environment," The Journal of Supercomputing, Vol. 68, No. 2, 2014, pp. 624-651.
[5] Nandigama, N. C. (2021). A Hybrid Approach for Feature Selection Analysis on the Intrusion Detection System Using Naive Bayes and Improved BAT Algorithm. Research Journal of Nanoscience and Engineering, 5(1), 15–19. https://doi.org/10.22259/2637-5591.0501003.
[6] A. Abbas and S. U. Khan, "A Review on the State-of-the-Art Privacy Preserving Approaches in E-Health Clouds," IEEE Journal of Biomedical and Health Informatics, vol. 18, no. 4, pp. 1431-1441, 2014.
[7] M. H. Au, T. H. Yuen, J. K. Liu, W. Susilo, X. Huang, Y. Xiang, and Z. L. Jiang, "A general framework for secure sharing of personal health records in cloud system," Journal of Computer and System Sciences, vol. 90, pp, 46-62, 2017.
[8] Nandigama, N. C. (2023). Enhanced Fingerprint Recognition system using hybrid Feature Fusion with Deep learning and Machine learning Optimization. Research Journal of Nanoscience and Engineering, 6(1), 9–15. https://doi.org/10.22259/2637-5591.0601003.
[10] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable and fine-grained data access control in cloud computing," in Proceedings of the IEEE INFOCOM, March 2010, pp. 1-9.
[11] S. Kamara and K. Lauter, "Cryptographic cloud storage," Financial Cryptography and Data Security, vol. 6054, pp. 136–149, 2010.
[12] T. S. Chen, C. H. Liu, T. L. Chen, C. S. Chen, J. G. Bau, and T.C. Lin, "Secure Dynamic access control scheme of PHR in cloud computing," Journal of Medical Systems, vol. 36, no. 6, pp. 4005–4020, 2012.
[13] K. Gai, M. Qiu, "Blend arithmetic operations on tensor-based fully homomorphic encryption over real numbers," IEEE Transactions on Industrial Informatics, 2017, DOI: 10.1109/TII.2017.2780885.
[14] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," IEEE Transactions on Parallel and Distributed Systems, 2013, vol. 24, no. 1, pp. 131–143.
[15] "Health Insurance Portability and Accountability,"
http://www.hhs.gov/ocr/privacy/hipaa/administrative/pr ivacyrule/, accessed on October 20, 2014.
[16] Z. Xiao and Y. Xiao, "Security and privacy in cloud computing," IEEE Communications Surveys and Tutorials, vol. 15, no. 2, pp. 1–17, Jul. 2012.