ПРОГРЕСИВНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ, ВИСОКОПРОДУКТИВНІ КОМП'ЮТЕРНІ СИСТЕМИ

UDC 62-50
DOI: 10.20535/SRIT.2308-8893.2023.3.02

AUGMENTED SECURITY SCHEME FOR SHARED DYNAMIC DATA WITH EFFICIENT LIGHTWEIGHT ELLIPTIC CURVE CRYPTOGRAPHY

DIPA D. DHARMADHIKARI, SHARVARI C. TAMANE

Abstract. Technology for Cloud Computing (CC) has advanced, so Cloud Computing creates a variety of cloud services. Users may receive storage space from the provider as Cloud storage services are quite practical; many users and businesses save their data in cloud storage. Data confidentiality becomes a larger risk for service providers when more information is outsourced to Cloud storage. Hence in this work, a Ciphertext and Elliptic Curve Cryptography (ECC) with Identity-based encryption (CP-IBE) approaches are used in the cloud environment to ensure data security for a healthcare environment. The revocation problem becomes complicated since characteristics are used to create cipher texts and secret keys; therefore, a User revocation algorithm is introduced for which a secret token key is uniquely produced for each level ensuring security. The initial operation, including signature, public audits, and dynamic data, are sensible to Sybil attacks; hence, to overcome that, a Sybil Attack Check Algorithm is introduced, effectively securing the system. Moreover, the conditions for public auditing using shared data and providing typical strategies, including the analytical function, security, and performance conditions, are analyzed in terms of accuracy, sensitivity, and similarity.

Keywords: ciphertext, user revocation, data sharing, CC, ECC, security issues.

INTRODUCTION

Innovative developments have made it possible to implement progressive solutions to improve the nature of human existence. In order to gather information and address challenges relating to wellbeing, analysts who are thinking about the growth of innovation have collected and evaluated wellbeing data from these sources. Accordingly, the development of integrated medical care innovation has the potential to improve efficiency and results comprehension at every level of the medical care framework [1]. By utilizing robust patient well-being controls, pervasive information access, remote patient checking, quick clinical intervention, and decentralized electronic-medical care records, new electronic health (e-Health) application frameworks are being developed that can address specific issues related to traditional medical care frameworks [2]. These systems can manage patient and well-being data, boost individual satisfaction, foster teamwork, enhance outcomes, cut costs, and generally improve the effectiveness of e-medical care administrations [3].

IoT usage and the advancement of wireless communication technologies enable real-time streaming of patients' health conditions to caregivers [4]. Additionally, a number of readily available sensors and portable devices may measure particular human physiological parameters with a single touch, including blood pressure (BP), respiration rate (RR), and heart rate (HR) [5]. Although it is still in the early stages of development, businesses and industries have quickly incorporated the power of IoT into their current systems and seen gains in both user experiences and production [6]. However, the use of IoT technology in the healthcare sector poses a number of difficulties, including those related to data management, storage, and transmission between devices as well as issues with security and privacy. Among the potential answers to these is Cloud Computing technology [7].

Cloud Computing is incredibly beneficial since it has many distinctive characteristics. Cloud computing makes a variety of services easily available [8]. Some advantages of this technology include suppleness, cost consumption, pay-as-you-go, increased effectiveness, and nimbleness. Through dedicated CS, cloud computing services offer client-specific applications and data storage [9]. Through the use of the cloud, businesses are able to avoid upfront framework costs and investments, allowing them to launch their applications more quickly, more intelligently, and with less maintenance. To provide the best services to mobile clients, portable distributed computing combines cloud computing and mobile devices [10].

Cloud Computing becoming increasingly important for the rapid growth of technology, particularly in the industry of the health care system [11]. Cloud storage is many times less costly than server storage, storage, equipment materials, and HR training to strengthen required operations [12]. Because in the cloud all the information of patients is stored, and retrieve the prescription from the cloud at any time and from any location. Because the data is stored in the cloud, healthcare professionals and identified patients will be able to access it via mobile devices such as smartphones and PDAs [13].

Cloud Computing enables internet-connected devices to access healthcare information from anywhere in the world. Furthermore, medical practitioners may exchange their resources and medical knowledge with other famous researchers in the same area from across the world [14]. Healthcare organizations, medical medication producers, pharmacists, medical insurance providers, researchers, and patients must all exchange EH records [15]. This presents a significant challenge in terms of keeping sensitive patient data secure. While there are numerous benefits, there are also some risks, particularly with regard to data security in the cloud, which is the most difficult issue at all times. It becomes more difficult in cloud computing because the actual data is stored in another location. As a result, providing security for data in the cloud is a time-consuming task for cloud computing organizations [16].The current healthcare sector is beset with computational and processing challenges. Inherent issues in the traditional healthcare sector [17; 18]. Patients' records include sensitive information that must be kept secure at all times. The current method has several irregularities in securing patient data. Medical data takes up more memory space, which is inefficient [19]. As a consequence, the purpose of this study is to enhance the performance and strengthen the operations of the existing cloud system in the healthcare industry. The proposed solution's key contributions include security and privacy, low-cost access policies for SHRs (Smart Health Records), a lightweight IoT detection system, rapid detection of Sybil attacks to reduce their impact on the network, and Sybil node identification using node properties. The proposed system in the given paper includes:

- A reversible and efficient no-pairing data-sharing technique for cloud storage systems based on Elliptic Curve Cryptography (ECC) with Identity-based encryption (IBE) approach.
- Cipher text Based Encryption (CBE), which presents an encryption access control (EAC) technique to satisfy User revocation that includes both user revocation and attribute, is seen here as a means of ensuring data security.
- A threat detection model to identify resource depletion attacks and Sybil attacks detection in the cloud system ensuring the safe storage of data.

LITERATURE SURVEY

In their study, Yan et al. [20] created the retrieval and storage-based indexing framework (RSIF) to enhance concurrent user and service provider access to healthcare data stored in the cloud. Concurrent access to stored data was made possible through continuous, replication-free indexing and time-constrained retrieval. Deep learning is used for all storage instances to categorize the restrictions for data augmentation and update. The learning process determines the approximate indexing and ordering for storage and retrieval, respectively, through conditional assessment. As long as the processes are independent, this helps to shorten the time for access and retrieval occurring at the same time. This data analysis should be carried out using various indexing techniques and storage and processing techniques.

PRCL, a privacy-aware, and resource-saving collaborative learning protocol was proposed by Hao et al. [21]. They created a unique model splitting technique that divides the network into three pieces, which offloads the intensive center half for CS to decrease the overhead. The original data, labels, and model parameters are all kept private by PRCL through the use of a mild perturbation of data and filled partly homomorphic encryption. Additionally, they examined the suggested protocol's security and showed how PRCL performed better with correctness and effectiveness. Future work will need to focus on developing adaptive data perturbation techniques, lowering the system's overhead of communication.

For a system of exchanging personal health records, Zhang et al. [22] suggested an effective identity-based distributed decryption technique. They may easily share their data with different parties without having to reassemble the decryption private key. They demonstrated the scheme's resistance to chosen-ciphertext attacks (CCA). Additionally, they used an Android phone and a laptop to implement the plan using the Java pairing-based cryptography (JPBC) package. The outcomes of the trial demonstrated the system's viability in an electronic personal health record system. In the future, it will be necessary to look into some more effective strategies, such as removing the zero-knowledge proof from the scheme and dispersing the secret without the need for a secret channel.

Using JavaScript-based smart contracts, Singh et al. [23] proposed a patient-centric architecture for a decentralized healthcare management system with a blockchain-based EHR. Additionally, a functional prototype based on the composer and Hyper ledger fabric technology has been put into place, ensuring the security of the suggested paradigm. Performance metrics including latency, throughput, resource usage, and others are measured in experiments using the hyper ledger calliper benchmarking tool under various situations and control parameters. The outcomes support the effectiveness of the suggested strategy. The authors want to expand their work based on fault tolerance in the future.

For cloud-based WBANs, Yang et al. [24] suggested a brand-new effective and anonymous authentication technique. The security study demonstrated that the system could fix the flaws in earlier ones and satisfy all security criteria. They also demonstrated the benefits of the suggested plan via presentation analysis of functionalities, computing transparency, storage overhead, and communication overhead demonstrating that the plan is better suited for real-world applications in the healthcare industry. The authors intended to create a universal authentication system that may be used in a variety of application scenarios in the future.

Son et al. [25] use blockchain to ensure data integrity and cipher text-User attribute-based encryption (CP-ABE) to create access controls to store data on CS. They used automated validation of internet security protocols and applications to do informal analysis, Burrows-Adabi-Needham (BAN) logic analysis, and formal validation of the proposed protocol to verify its robustness (AVISPA). As a consequence, they proved that the proposed protocol is more secure and performs better than comparable protocols. Future work will include modelling the full network as well as the security protocol in order to develop a new, more practical solution.

From the survey it is observed that for [20] data analysis should be carried out using various indexing techniques and storage and processing techniques, [21] need to focus on developing adaptive data perturbation techniques and lowering the system's communication overhead, for [22] it is necessary to look into some more effective strategies, such as removing the zero-knowledge proof from the scheme and dispersing the secret without the need of a secret channel, [23] work has to be expanded based on fault tolerance, for [24] the authors intended to create a universal authentication system that may be used in a variety of application scenarios and [25] requires simulation of the entire network and the secure protocol in order to build a new and more workable. Hence order to achieve the abovementioned necessities it is essential to develop a model.

AUGMENTED SECURITY SCHEME FOR SHARED DYNAMIC DATA WITH EFFICIENT LIGHTWEIGHT ELLIPTIC CURVE CRYPTOGRAPHY

This paper takes into account a system for auditing cloud storage that consists of the cloud, users, a group, and proxies. The company can host and share data to the cloud. Users who produce data and exchange it with one another make up the group. Users in the group are able to govern the group collaboratively since they trust one another. Here initially a container is created to store Cloudlets in the CloudSim library. The Data centers and Brokers are created in which the VMs and Cloudlets are loaded. Hosts with specific IDs are created and the simulation is run (Fig. 1).

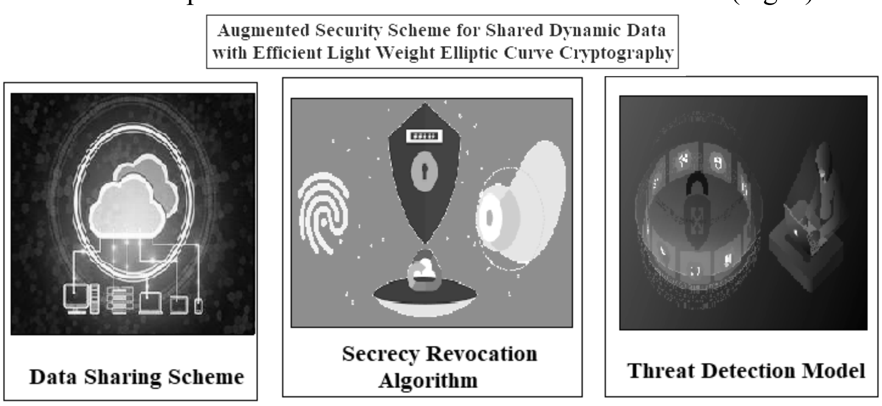

Fig. 1. Framework of the proposed model

Data sharing scheme

Traditional encryption techniques consume a lot of storage space since they require duplicates for all cipher text in every particular user through a unique key. To create a secure, reliable, and accurate model sharing data, the dispute must consider, importantly the key distribution among new users necessitates that data owners remain online constantly. Initially, the system must allow data owners to add or delete users. Then, the system must allow data owners to guarantee that data confidentiality is protected from CS, attribute authorities, and unauthorized users. Consequently, consumers should confirm the data correctness they have received. At last, users have to have mobile access to shared data. Hence, considering the above-mentioned requirements this research presented a reversible and efficient no-pairing data-sharing technique for cloud storage systems based on ECC withIdentity-based encryption (CP-IBE) approach. Cloud computing, smart grids, the Internet of Things, and other distributed systems may all benefit from the one-to-many encryption technique known as CP-IBE, which is based on public keys and enables flexible and fine-grained data access management. Unfortunately, because they rely on pricey bilinear pairing algorithms and have large encryption and decryption computation overhead costs, the majority of modern data-sharing approaches are ineffective for cloud systems with limited resources. The paper here proposed an effective data-sharing scheme for cloud storage systems based on the fact that the ECC algorithm has stronger bit security than exponential-based public key cryptographic algorithms like RSA and can achieve the same level of security with smaller key sizes and higher computational efficiency. The four entities in the proposed system model are a trusted expert, a cloud source, senders, and users which are depicted in Fig. 2.

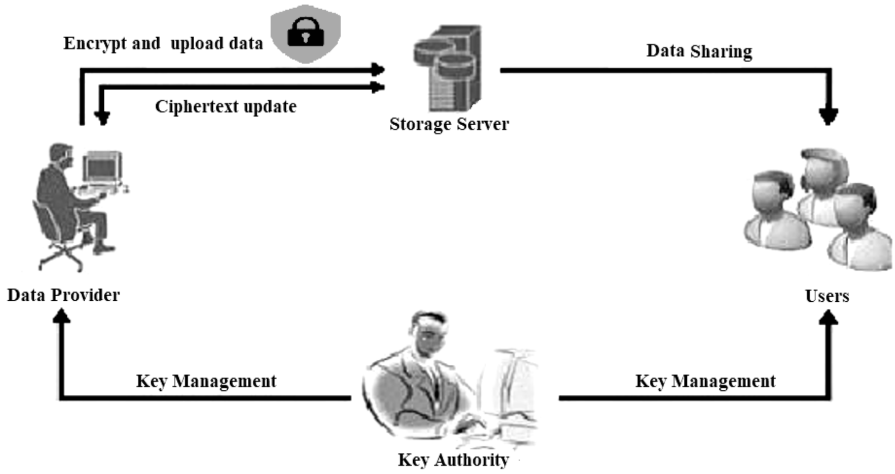

Fig. 2. Entities of the system model [31]

Trusted Expert (TE). The TE is generate both master secret keys and global public parameters, which produce and share out users' matching private keys. Additionally, it is in charge of blocking users. The TE is expected to be trustworthy but inquisitive, which means refuse service for authorized users provide adhere to established procedures appropriately, if interested in the data's substance, and want to learn as much as it can about its customers' private information.

Cloud Source (CS). Data from the owners are gathered and stored by the CS, a potent computing entity with limitless resources. Additionally, computing a sizable amount of decryption overhead aids users in decrypting the cipher text. Like the TE, the CS is expected to be sincere but curious.

Data owner. A data owner is a group that needs to outsource a data file to the CS. Data is encrypted initially under a set of characteristics before transmitting it.

User. It is an entity with unrestricted access to the ciphertext of the cloud server. To achieve this, it first creates a token using its key, and then it asks the CS for access to the data by giving the CS the token.

Proposed CP-IBE-based ECC approach. The algorithms utilized in the data-sharing system are discussed in detail below.

- **Setup** $(S_p\ ,A) \to (M_K, P_G)$

The setup procedure takes a security parameter S_p and an attribute universe A as inputs and produces global public parameters P_G and a master key M_K for the system as outputs.

- **Encryption** $(P_G, M, \omega) \to CT$

The encryption method is given a message M , a collection of descriptive qualities, and the global parameters P_G . It generates cipher text CT .

- **KeyGen** $(P_G, M_K, T) \to SK$

The global parameters P_G, the master secret key M_K , and an access tree T are sent into the key creation method. For each authorized user, it produces a private key P_K .

- **TokenGen** $(P_G, D) \to TK$

The user executes this procedure to create a decryption token TK .

- **Partial Encryption** $(P_G, TK, CT) \to CTPartial$

The partial decryption algorithm accepts the global parameters P_G, the user token, and the ciphertext as input. It returns ciphertext that has been partly decrypted.

- **Decryption** $(CTPartial) \to (M, AC_M)$

The user executes the decryption algorithm, which uses the partly decrypted ciphertext. It displays the message M as well as the message authentication code, AC_M .

- **Elliptic Curve Cryptography**

ECC is public key cryptography. Assume p is a prime number and F_p is the field of integers modulo p . A cubic equation $y^2 = x^3 + ax + b$ defines an elliptic curve (EC) over a finite field (Galois Field) GF and each elliptic curve is formed by a distinct value of a and b. The collection of all locations (x, y) that satisfy the above equation, as well as a point in infinity, lies on the elliptic curve. The private key in the ECC is a random number, while the public key is a point in the curve formed by multiplying the private key by the generator point G in the curve.

Based on the preliminary results and system model, this paper presented a CP-IBE scheme in detail in this section owner O develops a collection of expressive qualities before encrypting a message M with an algorithm and sending cipher text to the CS provider. This work employs lightweight operations of ECC in conjunction with an algorithm of symmetric encryption. If user U_s gets provided authority and wishes to get data stored on CS, user U_s must construct and deliver a decryption token to the CS. When the CS receives the token, it partly gets back the saved cipher text, and transmits consequential communication to the U_s. Ciphertext can then be readily decrypted by the user. Setup, Encryption, KeyGen, TokenGen, Partial Decryption, and Decryption are the six function modules of the proposed data-sharing method. These are their descriptions:

- **Setup** $(S_p, A_u) \to (M_K.P_G)$

The trusted attribute authority executes the setup procedure, input as security parameter S_p and the attribute universe A_u. It generates Global Public Parameters P_G as well as the Master Key M_K. To do this, the authority first chooses a random integer R_i from $Z P^*$ and computes $PK = R_i.G$. Let $A_u = \{1,...,n\}$ be a collection of all attributes in the system; the authority selects a random number $s_i \in Z P^*$ and computes the public key of each attribute I as $P_i = s_i.G$. The CS chooses a random number μ for its secret key and calculates $PKCS = \mu.G$ for its public key. The authority does not know μ, and the CS proves the authority's knowledge of μ using a zero-knowledge proof procedure. The authority assigns the secret key $M_K = \{R_i, \{s_1,...,s_i\}\ i \in A_u$ and publishes the global parameters $P_G = \{M_K, M_K CS,\ \{P_1,...,P_i\}\ i \in A_u\}.o$.

- **Encryption** $(P_G, M, \omega) \to CT$

As input, the encryption method receives a message M, a collection of descriptive qualities ω, and the global parameters P_G. When the owner O wishes to encrypt a message using the set of characteristics, he or she selects a value at random from the set of values Z_P^* and computes K and F as follows:

$$K = d \cdot PK = (k_1, k_2)\,;\quad F = d \cdot PKCS = (f_1, f_2)\,.$$

If $K = O$, the authority re-selects d at random from Z_P^* to compute K until $K = O$. Then use the points (k_1, k_2) as the encryption and integrity keys, and construct ciphertext C and AC_M for message M as follows:

$$C = ENC(M, k_1)\,;$$

$$C' = ENC(C, f_1)\,; \tag{1}$$

$$AC_M = HAC_M(M, k_2)\,. \tag{2}$$

$ENC\,()$ in Equation (1) is a symmetric encryption method such as AES. The message M is encrypted with key k_1 and then re-encrypted with key f_1, obscuring the ciphertext C from the authority's view. $HAC_M()$ is a cryptographic hash function in Equation (2) that creates the hash-based message authentication code for message M based on the integrity key k_2. Finally, the owner O computes $C_i = d.P_i$ for all of the attributes in ω and uploads $CT = (\omega, C', AC_M, \{C_i\}\ i \in \omega, H = d \cdot G)$ to the cloud service provider.

- **KeyGen** $(P_G, M_K, T) \rightarrow SK$

The keyGen algorithm calculates the decryption key for a user's request, U_B, in the manner shown below. In the access tree, it selects a polynomial q_x for each node x. Starting with the root node R, these polynomials are selected from top to bottom. The authority determines the degree dx of the polynomial q_x for each node x in the tree to be one less than the threshold kx of that node, that is, $dx = kx_1$. In order to fully fix qR, it first sets $qR(0) = R_i$ for the root node R (keep in mind that R_i is the authority's secret value). The remaining points are then set at random. It sets $q_x(0) = q_{\text{parent}(x)}(index(x))$ for every other node x and picks dx additional random locations like $qR.\,(x)$. The unique index number assigned to x by its parent is called $Index(x)$.

Let Y represent the collection of leaf nodes in the tree, and $att(y)$ represents the attribute related to leaf node y. The KeyGen method produces the following values once the polynomials are finished for all leaf nodes y:

$$D_y = q_y(0) / s_i,\ i = att(y).$$

Finally, the authority sends $D = (D_x,\ i = att(x),\text{ and } i \in \omega)$ as the private key for U_B.

- **TokenGen** $(P_G, D) \rightarrow TK$

At this phase, a user U_B generates a token TK_B based on his/her private key and sends it to the CS to convey most of the decoding computational load to the CS. For this purpose, U_B first selects a random number b from Z_P^* and computes $D' = \{D_x \cdot\ b = (q_x(0) \cdot\ b) / s_i\} i \in \omega$. After that, the user U_B computes the point $Q = b.PK_{CS} = b.\beta.G = (q_1, q_2)$ and $B = b.G$ and sets the token TK_B as follows:

$$TK_B = \{B, TB = ENCq_1(D')\ i = att(x),\ i \in \omega, T)\}\ .$$

To protect against a DOS attack in this case, a timestamp T is employed. The CS examines the time stamp T and decrypts TB after receiving the token. The CS continues the partial decryption step if it is valid. An symmetric encryption function like $AES\, is\, ENC(.)$.

- **Partial Decryption** $(P_G, TK_B, CT) \rightarrow CTPartial$

When the CS receives the token TK_B, it uses its secret key to compute the decryption key q_1 as $Q = \beta.B = \beta.b.\ G = (q_1, q_2)$ and then decrypts T_B. The CS declines the request for partial decryption if the timestamp $T's$ validation is unsuccessful. Otherwise, the partial decryption process is carried out by the CS as follows. Let x be a node of tree T, first this algorithm defines a recursive algorithm Decrypt Node (CT, D', x). Let $i = att(x)$, if x is a leaf node, then Decrypt Node (CT, D', x) is computed as follows:

$$D'x.Ci = Dx.b.Ci = q_x(0).s-1i.b.d.s_i.G = q_x(0).b.d.G.$$

This recursive procedure returns an element from the ECC group or $\perp$.

The algorithm $Decrypt\,Node(CT,D',x)$ searches for all nodes z that are offspring of x if x is a non-leaf node, and the result is saved as F_z. Let L_x be a random k_x-sized collection of child nodes z that satisfy $Fz \neq \perp$. $Decrypt\,Node\,(CT,D',x)$ returns if there is no such L_x, indicating that the node was not fulfilled. Otherwise, assuming $i = index(z)$ and $L'x = \{index(z),\ z \in Lx\}$ it is possible to calculate $DecryptNode(CT,D',x)$ as follows:

$$\Sigma Xz \in Lx \Delta i, L'_x(0).DecryptNode(CT,D',z) = \sum_{z \in L'_x} \Delta i, L'_x(0).qz(0).b.d.G =$$

$$= \sum_{z \in L'_x} \Delta i, L'_x(0).qparent(z)\big(index\ (z)\big).b.d.G =$$

$$= \sum_{z \in L'_x} \Delta i, L'_x\ (0).q_z(i).b.d.G = q_z(0).b.d.G. \qquad (3)$$

The outcome of $Decrypt\,Node\,(CT,D',R) = qR$ for the root node R of the access tree T is based on the information given $qR(0).R_i.b.d.G = R_i.b.d.G$ The CS calculates F = after computing the $Decrypt\,Node\,(CT,\ TK_B,R).C = DEC(C',f_1)$ and $H = .d.G = (f_1)(f_1,f_2)$. The CS then gives the user $UB\,CT_{Partial} = C,\ AC_M,\ N = Decrypt\,Node(CT,D',R)$. Equation (3) demonstrates that the cloud service provider cannot decipher the ciphertext since they are unaware of the value of b and can only assist the receiving users in doing so.

- **Decryption** $(CT_{Partial}) \rightarrow (M, AC_M)$

The user UB may quickly determine the decryption and integrity keys after receiving the $CT_{Partial}.N = R_i.b.d.G$ means that the decryption method just needs to divide b to retrieve the keys as Equation

$$C'_1 = N/b = \alpha.b.d.G.b^{-1} = \alpha.d.G = (k'_1\ ,k'_2)\ .$$

The points $(k'_1\ ,k'_2)$ serve as the message $M's$ integrity key and decryption key, respectively. When the user enters $M' = DEC(C,k'_1)$, the message M may be decrypted. If $HAC_M(M',k'_2) = AC_M$, the message M is accurate.

In order to revoke a user *UB*, the authority securely transfers all of the revoked user *UB*'s attributes to the *CS*. When the *CS* receives the token, it first determines if it has all of the *UB*'s properties, and if so, it rejects it.

Secrecy Revocation Algorithm

The system has the power to cancel a user's access if they leave the system. In other words, the authority securely transmits to the *CSP* all of a user's *UB*'s revoked characteristics. When the *CSP* receives the token, it first determines if it has all of the *UB*'s properties, and if so, it rejects it. Even if the user still possesses a working secret key after the revocation procedure, they will not be able to access the saved data. The revocation procedure is effective since no need to be updated. Cipher text Based Encryption (CBE), which presents an encryption access control (EAC) technique to satisfy *User* revocation that includes both user revocation and attribute, is seen here as a means of ensuring data security. The authorized users should be updated right once if the data owner modifies one of the attributes in an access *User* since the revoked users who had previously received access to the *User* can see the ciphertext. Four different sorts of update *User* levels are established, specifically for data owners. All secret token keys are uniquely produced at all levels by categorizing those levels. As a result, the secret token key is hashed to create a new secret key (Fig. 3).

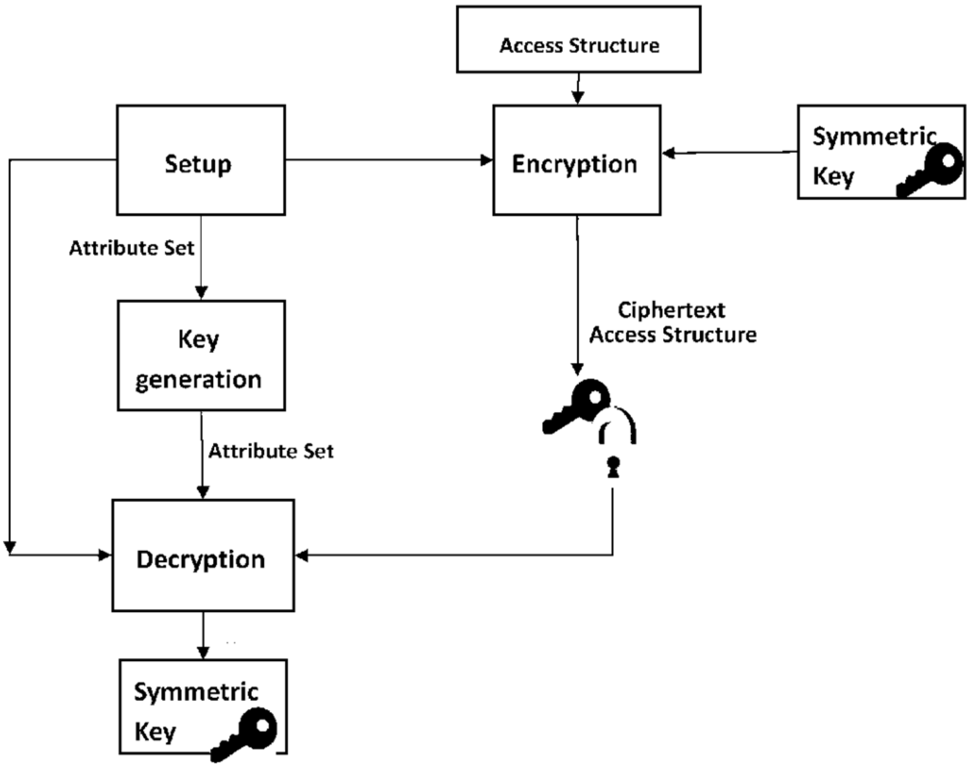

Fig. 3. Flow chart encrypting *User* algorithm

The proposed method has four basic algorithms to handle user secrecy revocation. They are, in order, the encrypting *User* algorithm, re-encryption algorithm, update key generation algorithm and decryption algorithm.

Encrypting *User* algorithm. Each access *User* is represented by a distinct identity, which stands for the traits that are crucial for identifying authorized individuals. The data owner's *(DO)* present *User* identity is documented in the algorithm for updating policies as $User_{id}$, and the new *User* identity the *DO* wants to modify is marked as $New\ User_{id}$. The four status numbers are specified for each level of the individual updating *User*. The status number should be. The identification of the material for plaintext is recorded in the system and is specified as α in accordance with each Upd_{level}.

Input: $User_{id}$, New_User_{id}.

Output: Upd_{level}

1) perform one of four *User* levels for updating;

2) recognize Upd_{level} based on choice in step 1;

3) obtain α by formative all Upd_{level};

4) locate New_User_{id} as a present *User* identity and cancel the old $User_{id}$;

5) go again Upd_{level}.

Re-Encryption Algorithm. The plaintext message (M) related to new_User_{id} is was re-encrypted using a special method. The Upd_{SK} is the encryption key used by the DO. The generated cipher text is C.

Input: $M\ New\_User_{id}$, Upd_{SK}.

Output: C

1) perform updates $User_{id}$ to, New_User_{id} for M;

2) obtain $Upd_{level} = UpdateUser(Userid, New\_User_{id})$;

3) acquire Upd_{SK} from the TA;

4) $C = ReEncrypt\ (M, New\_User_{id}\ Upd_{SK})$;

5) go back C.

Update Key Generation Algorithm. *TA* generates a default key string () and uses *M'*s unique identity, which is to be accessible as, in the update key creation procedure. The status number (α) is originally determined by *TA* using the Upd_{level}. The TA then changes the numbers for both strings. The *TA* then joins the strings of the key with the other keys. As a secret token key, this concatenation is transformed into the *Base64String format* (SK_{token}). By obtaining the hash code from the MD5 function for the SK_{token}, the *TA* then constructs Upd_{SK}. The *Base64String format* output is likewise consistently applied to the final Upd_{SK}. According to each user's request identification, the *TA* also creates $a\$$ for them.

Input: $¥, \beta, \gamma, Upd_{level}, New\_User_{id}$.

Output: Upd_{SK}, \$

1) start α based on Upd_{level};

2) TA obtains the and β since the users need a directory;

3) TA gets $¥$, and converts both β and α to strings;

4) $SK_{token} = Concatenate(¥, \beta, \alpha)$;

5) $SK_{token} = ToBase64String(SK_{token})$;

6) $.Upd_{SK} = GetHashCode(SK_{token})$;

7) $Upd_{SK} = ToBase64String(Upd_{SK})$;

8) TA generates \$;

9) records the $New_{User_{id}}$ to filter unauthorized users;

10) return Upd_{SK}, \$.

Decryption Algorithm. The data user (*DU*) uses his $User_{id}$ to prove his identification as a *User holder* in the decryption procedure to decode *C*. The *TA* will offer the *DU* the Upd_{SK} and if his $User_{id}$ is acceptable to the New $User_{id}$ of the *DO*. In order to obtain the original ciphertext C, the *DU* uses it to decode the proxy ciphertext (*C'*) from the cloud. The *DU* can then use this Upd_{SK} to decode the *C*.

Input: *C*, \$, Upd_{SK}, $User\_User_{id}$.

Output: *M*

1) *DU* enters the system by demonstrating his abilities;

2) the system records his $User\_User_{id}$ according to his attributes;

3) *DU* needs the. Upd_{SK} and \$ to the TA;

4) the system records his $User\_User_{id}$ based on this attributes;

5) *DU* requests the Upd_{SK} and \$ to the TA;

6) *DU* verifies inbox and acquires the Upd_{SK}, \$ from TA;

7) *DU* obtains the C' from the cloud;

8) $C = Decrypt\ (C', \$, User\_User_{id})$;

9) $M = Decrypt\ (C, Upd_{SK}, User\_User_{id})$;

10) return M .

Threat detection model

Resource depletion attack detection algorithm. In order to distinguish between legitimate and illicit sources, first specify a set of parameters (or rules) for a source $S_1(y \le j \le x)$. The first argument is *atj* $(atj \le h)$, which stands for the active time of a source S_1. This field demonstrates that source S_1 has not delivered any attack traffic to the system while acting as a cloud provider, in addition to indicating how long a source has been connected to a cloud service. As a result, *atj* is regarded as a crucial factor in making informed choices about attack detection. The second parameter, N_{in}, indicates how many incoming flows have been established to a cloud service and is the number of flows from a source S_1, where $N_{in} \ge 1$. Whenever return traffic constantly takes a different route for a better service response or for other objectives, it is fair to just take into account request flows. In the Internet protocol suite, the *ICMP* protocol is referred to as a supporting protocol and is only used to determine whether a requested service is unavailable or whether a host or router could not be reached. As a result, a typical source S_1 typically transmits a small number of request packets to a destination, and each source S_1 only creates one flow in an *SDN* switch's flow tables. A malicious source S_1, on the other hand, can produce one or many flows in an *SDN* switch. In order to distinguish an abnormal source S_1 from its regular counterparts based on the aforementioned studies, define the third parameter, Pf (the average number of packets per flow of the source S_1), as follows:

$$Pf = tpktj; \quad Attack\, Type\, I\ ;$$

$$Pf = \frac{tpktj}{nconj}; \quad Attack\, Type\, II\ .$$

Where $tpktj$ represents the source $S_1's$ transmitted packets to the cloud and Nin represents the source $S_1's$ flow number. The source $S_1's$ traffic protocol is a crucial factor in dividing the two primary assault methods. The following assessed parameter is the priority of a source S_1, indicated as $Prij$, which makes a distinction between reliable $(Prij=1)$, typical $(Prij=2)$, and unidentified $(Prij=3)$ sources. The $flag, Flag_j$, which displays the status of the source S_1, is the final parameter introduced. Source S_1, has two statuses: attack and normal, which correspond to $Flag_j = -1$ and 1, respectively. At first, a new source is designated as a standard source. It should be noted that the Update Agent updates a database after collecting values from each active source $S_1's\, atj, Nin, Pf, Proj$, and $Prij$ at each observation. The tuple of parameters for a source S_1 is constructed using the definitions given above and is an entry in the database. $S_1 = (atj, Nin, Pf, Proj, PrijFlag_j)$. The search engine uses S_1 and $Proj$ matches to separate the *IP* sources for each search operation. Then examine the effective classification of normal and attack sources using the *IP-based* technique utilizing a history of *IP* database (*HIP Database*). Based on the statistics gathered at each observation, first extract and update all active source *IP* addresses by protocol to the *IP* Database using the Update Agent when the system is functioning. Utilize the Ini sets to categorize normal and abnormal sources for each sort of traffic protocol for our first observations and keep the Ini sets separate until the proposed system picks up any attack sources. The *IP* Database could provide some more sources for the observation t. Using pre-made Ini sets, these sources are further validated to determine their $Flag$. The suggested Algorithm will replace the value of the associated Ini set for the subsequent observation $(t+1)$. Then, using the $B(tC_1)ICMP$ comparison, identify these sources as either regular sources or attack sources using the Anomalous Source Detection Algorithm.

Step 1: $B(t+1)i = \{(ati; w1i); (Pfi; w2i); (nconi; w3i)\} \leftarrow e$ The boundary set of the protocol I have given at the t^{th} observation

Step 2: $(atj, Nin, Pf, Proj, PrijFlag_j)$ A set of attributes of a source $S1$ that is collected at the (tC_1) observation

Step 3: $Xi\, D\, ati * w1i + C\, Pfi * w2i + C\, nconi * w3i$

Step 4: $Xj\, D\, atj * w1i + C\, Pf * w2i + C\, Nin * w3i$

Step 5: if $Xi \le Xj$ **then**

Step 6: $Flag_j = -1$ {Attack source}

Step 7: else

Step 8: $Flag_j = 1$ {Normal source}

Step 9: end if

Step 10: return: $Flag_j$

Normally attacks create a large number of flows to the target with a small number of packets. In order to counteract this attack, the Mitigation Agent notifies the *SDN* controller's forwarding engine to ignore packets in messages from attacking sources that demand the installation of new flows at the Open Flow switch and sends a flow mod message with a delete action to the edge Open Flow switch. In the event of an assault, this strategy eliminates all anomalous flows and stops fresh attack flows. As a result, it prevents the system from being overloadedand guarantees efficient cloud operation. Additionally, the attack flow deletion from the switch will result in a considerable drop in the number of collected flows at the time of the subsequent observation. Therefore, this approach can conserve as many computing resources as feasible from the cloud control plane.

Sybil attacks detection. Sybil Attack Check employs encryption in order to offer secure communication, and it also safeguards the data against Sybil assaults by utilizing a detection algorithm. The cluster's nodes are given secure communication during the secure phase. Utilizing one-time authentication (OTA), the secret key and the node id provide authentication. A node needs a one-time authentication from the Cluster head (CH) in order to communicate, and the CH node only sends the OTA following a successful verification. The CEA cryptography algorithm is a novel one that is suggested to offer secrecy (Cipher encryption algorithm). Traditional encryption techniques are resource-intensive; they use a lot of storage space, power, and processing time.

CEA is a straightforward encryption algorithm with promising security. It just requires a small number of rounds and basic operations. The precise stages of the CEA algorithm are shown in the algorithm below. It accepts 64-bit plain text as input, and 32-bit random keys, and outputs a 64-bit encrypted text. Either eight or sixteen rounds of operation can be used to implement CEA. Four fundamental operations—*XOR, pair swap, encoding, and 1s complement*—are carried out in each round. In the encryption procedure, the plaintext is divided into the left plaintext (L_i) denotes the round, and the right plaintext (R_i), and an *XOR* operation is performed with two 32-bit random keys. On the resulting L_i and R_i, a paired exchange is performed in the second phase. Two-bit pair swapping is done after the plain text has been divided into two-bit pairs. These two-bit pairs are then substituted in the following phase with the equivalent encoding bits. As an illustration, the bit 00 is encoded as 01, 01 as 10, and so on. The final step involves performing a 1s complement operation to obtain the necessary 64-bit cipher text. The operations are carried out backward throughout the decryption process; the first stage involves doing the 1s complement. The next phase involves decoding using the decoding bits followed by a pair swap operation. To obtain the necessary plain text, an *XOR* operation is performed in the last stage.

The cryptographic symbols used in Sybil Attack Check include: G_g — Group generator of prime order; G_g — a bilinear group output; A — set of all features; SK_s — secret key; CA — cipher text with access User. Set of integer integers A and Z_p. While security typically rises, composite-order group pairing performance sharply declines. Since the assault, in this case, is selective, there is a need to utilize a prime-order group since it is less difficult and more affordable. Prime-order groups can only provide selective security in their respective security models. Definition: Assume that G_g operates on the input variable as a bilinear prime order group generator. $G_g 1$, and $G_g 2$, which have additive and multiplication properties, are cyclic groups of prime order.

The output of G_g is defined as $(p, G_g 1, G_g 2, G_g T, e)$.

Key generation $(PK,\ M_K, S) \rightarrow SK_s$: The secret key SK_s are produced using the key generation algorithm using the inputs of the public key PK, master key M_K, and set of characteristics S, as illustrated in Fig. 4.

$SK_s = (S, K, K', \{K_i\}\ i \in Is)$, where $S = (Is, S\ )$ with $Is \subseteq Zp$ and $S = \{S_i\}$ $i \in Is$, $K = g^{\alpha} g^{at} R$ choose $tR\,Zp, R, R', Ri \in R\,G2, K' = gt\,R'$ and $Ki = (Ui\,si\,h\ )t\,Ri$.

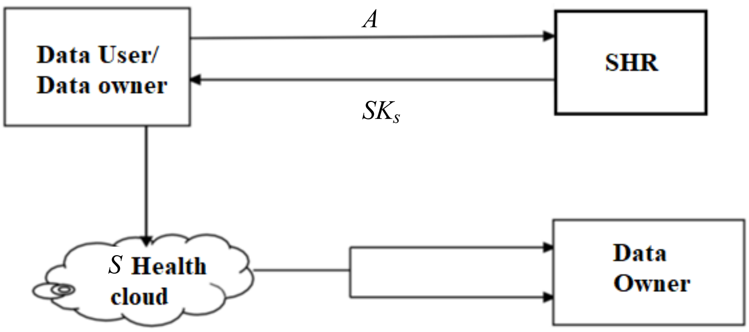

Fig. 4. Smart Health Records (SHR) encryption and decryption process in the cloud

Where $e: G_g 1 \times G_g 2 \rightarrow G_g T$ is a bilinear map with the following property.

- Non-degenerate: $e(g, g)$ has the order p where $g \in G_g$.
- Bilinear: $e(ga, hb) = e\ (ga, hb)$ for all $a, b \in Zp$ and $g \in G_g 1,\ h \in G_g 2$.

The following set of algorithms is used in the proposed approach.

Setup $(1\lambda) \rightarrow (PK, M_K)$*:* The setup algorithm takes the input parameter λ as input and runs the group generator $\mathbf{G}(1\lambda)$ to get $(p, G_g 1, G_g 2, G_g T, e)$. It outputs the Master Key M_K and Public Key PK of the system. The master key is $M_K = (,h,)$ where $R\,G2$ and the system public key is $PK = (p, g, U_1, U_2, ..., U_p, ga, Y)$ where Y is defined as $e\,(g, g)$, $H = $ $= YZ, g, h, U_1, U_2, ..., U_p$ and an $R\,Zp$.

SW. Encryption $(PK, M, A) \rightarrow CA$: The message M, the access structure A, and the system's public parameters PK are all inputs to the encryption method (A, P, T). The cipher text CA is produced.

$$CA = (A, P), C', \{TI\}\ i \in S\},$$

where

$$C' = SW.LEA(M)\ .Y^S \ \text{ and } \ \{TI\} = g^{t1}, g^{t2}, \ldots, g^{t|u|}.$$

SW. Decryption $(PK, CA, SKS) \rightarrow M$ or $\perp$: the decryption algorithm takes the public key, plain text, and secret key as inputs and produces M if the access structure is satisfied otherwise. The real *SHR* is only decrypted if the characteristics match the access structure defined for that specific *SHR*; otherwise, access is denied to that particular user. This is done at the initial stage of the decryption process. From (A, P) derive IA, P, where IA, P stands for the minimal subset of

$1,...,l$ that meets the (A,P). Here, examine the possibility of a $I\ IA,P$ satisfying $C1=e\ (SKS,I$ over (A,P).

Where $\Sigma\omega_i A_i = (1,0,\ldots,0)$ otherwise it returns $\perp$. Here (A,P,T) is hidden, and in which (A,P), is revealed. If $i \in x$ satisfied the plain text M is given as output. $M = SW.LEA(C_1)/Y^S$, where M is in plain text, and C_1 is M's encrypted form.

PERFORMANCE ANALYSIS

This section addresses the results of the implementation and the performance of our proposed system. The proposed system is implemented using the Cloud Sim framework. The fundamental goal of the suggested technique is to ensure security and privacy. Novel algorithms are employed to accomplish this goal of policy concealing and the performance of the model is discussed here (Fig. 5).

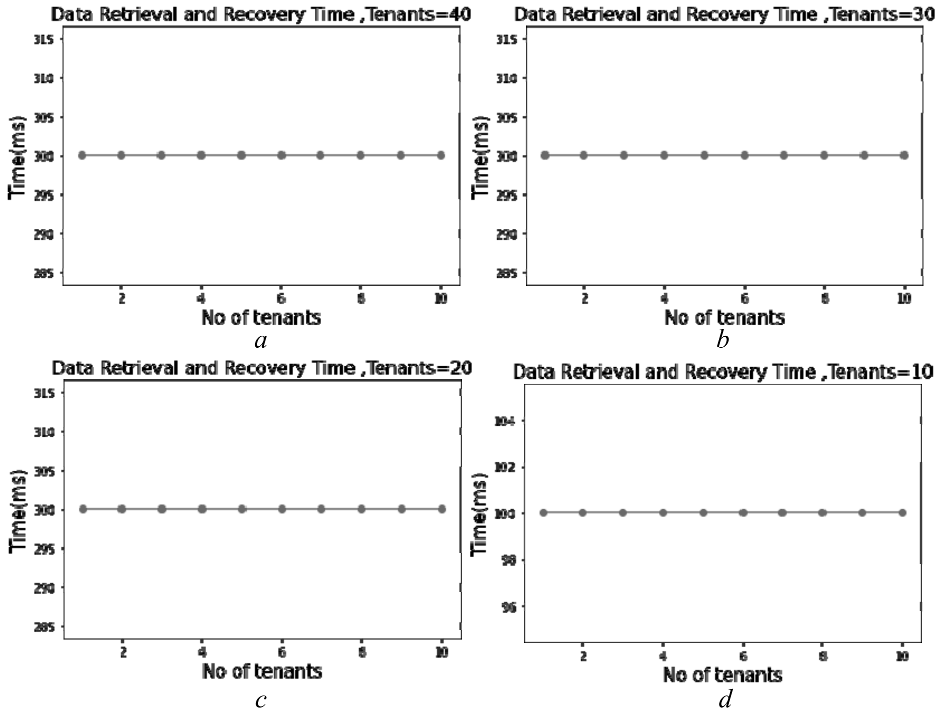

Fig. 5. Data retrieval and recovery time: *a* — tenants 40; *b* — tenants 30; *c* — tenants 20; *d* — tenants 10

The user's terminal will return and decrypt a variety of matching files using a data sharing scheme. There are 0 to 10 files that have been matched. Tenants range in number from 10 to 40. The value of recovery time, which ranges from 100 to 300, is discovered to remain constant for any number of renters.

The amount of time a user must wait after making a keyword token query in order to receive the retrieved health records must be carefully considered. All the data are calculated for 10 tenants. The test time of 1800 ms is observed for 10 tenants. The trace of the system is range from 50 to 500 ms. The time for token generation is observed to be constant at 30 ms. The decryption time for the model was found as 25 ms. and the encryption time for the model is estimated as 50 milliseconds. The key generation time for the model is nearly 25 ms and it increases to 140 ms for 10 tenants (Fig. 6).

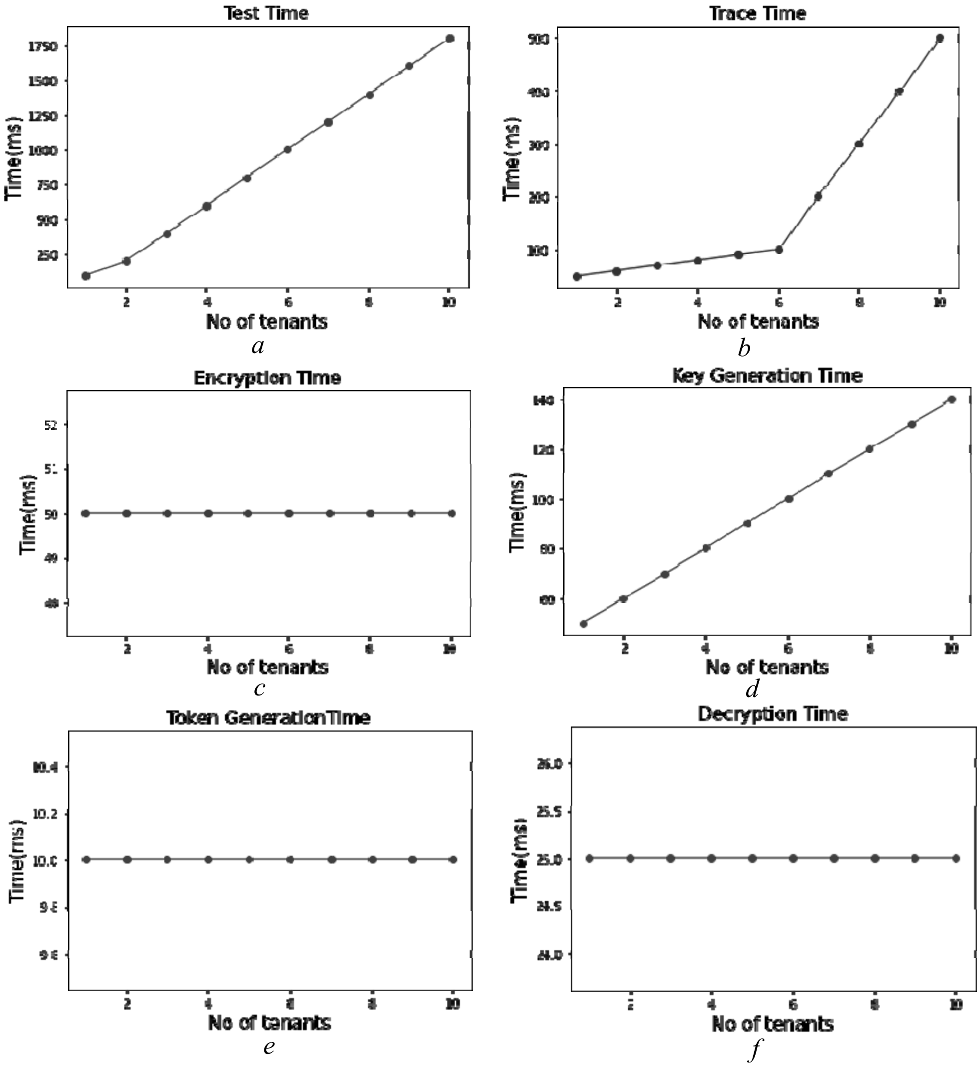

Fig. 6. Computation efficiency

To evaluate the storage and transmission overhead, there is a need to identify the performance based on public and private keys and the token size in Fig. 7. The research found that for the public parameter 7, *a* requires substantially limited space and communication expenses. The key parameter size is 5000 bits, regardless of how many tenants are allowed in the system. For *b* secret key size is increasing from 1000 to 10000 bits with an increase in the number of tenants.

It is shown that 7, *c* requires the least amount of cipher text storage, which is especially useful for saving money under the pay-for-use cloud model. Additionally, by transferring the ciphertext to the public cloud using less battery power, the data owner may increase the lifespan of the user's mobile devices. The token size obtained in 7, *d* illustrates that the size remains constant as still figure of tenants rises.

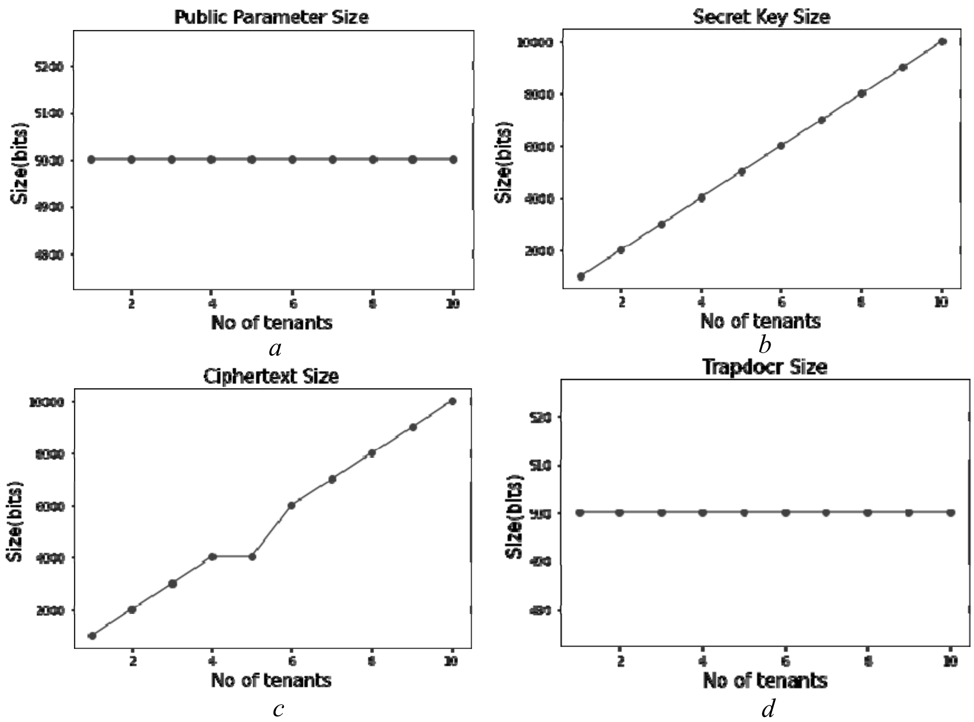

Fig. 7. Storage and transmission efficiency

The performance of the scheme in terms of accuracy, similarity, sensitivity, error, and false positive rate is depicted in the above graph. A heterogeneous, time-series set of data with ten classes of data was utilized. It is observed that the accuracy is obtained to be 90 %. The similarity range is obtained as 21%. The sensitivity of the system for any input change is depicted to be 98% for any given number of attributes. The calculation of the false positive rate (FPR) with the following formulae. The error is determined to be low as 10% and the FPR was found to be 1% for the proposed system (Fig. 8).

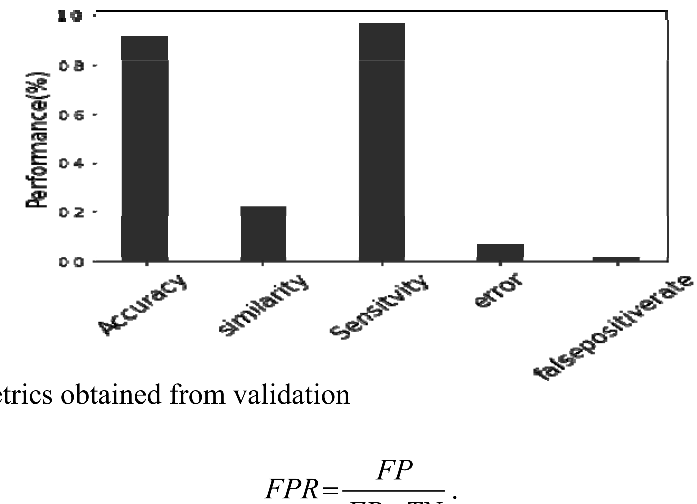

Fig. 8. Metrics obtained from validation

$$FPR = \frac{FP}{FP + TN}.$$

Comparison Metrics

In this part, the validity of the recommended methodology is assessed with the working of various traditional methodologies (Fig. 9).

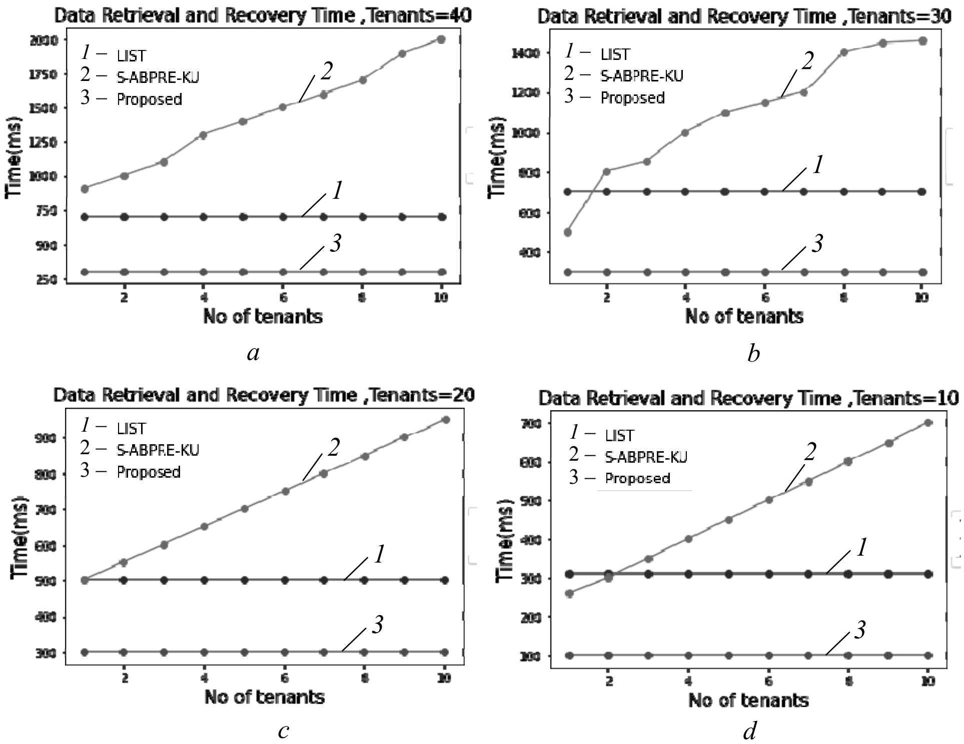

Fig. 9. Comparison of data retrieval and recovery time: *a* — Tenants=40; *b* — Tenants = 30; *c* — Tenants=20; *d* — Tenants=10

The data retrieval and recovery time of the proposed method is evaluated with other existing methods like LIST [30] and SABPRE [27]. For all the other methods the value of time with respect to the increase in tenants is observed to be comparatively high. The maximum time observed for 10 tenants is 710 ms for the LIST model whereas for the proposed model it is 100. The recovery time for 20 tenants is found to range from 500 to 1000 ms for the other methods and for the proposed method it is found to be 300 ms. The time is constant for 30 tenants for LIST as 700 and for SABPRE it is ranging from 500 to 1500 ms. For 40 tenants also the proposed method achieved less recovery time comparatively which is 350 ms.

The computation time for the model is compared for test time, trace time, token generation time, decryption time, encryption time and key generation time with existing models like LIST [30], SABPRE [27], ABKS [26], TCPABE [29] and LUCP ABE [28] etc.

For test and trace time the comparison is with LIST, SABPRE and ABKS. It is observed that the test duration from 1 to 1750 ms and trace duration from 1 to 200 ms which is minimum for the proposed model. The maximum values for the test time ranges from 1 to 2500 for the ABKS method and for trace the value ranges from 1 to 5000 ms for LUCPABE (Fig. 10).

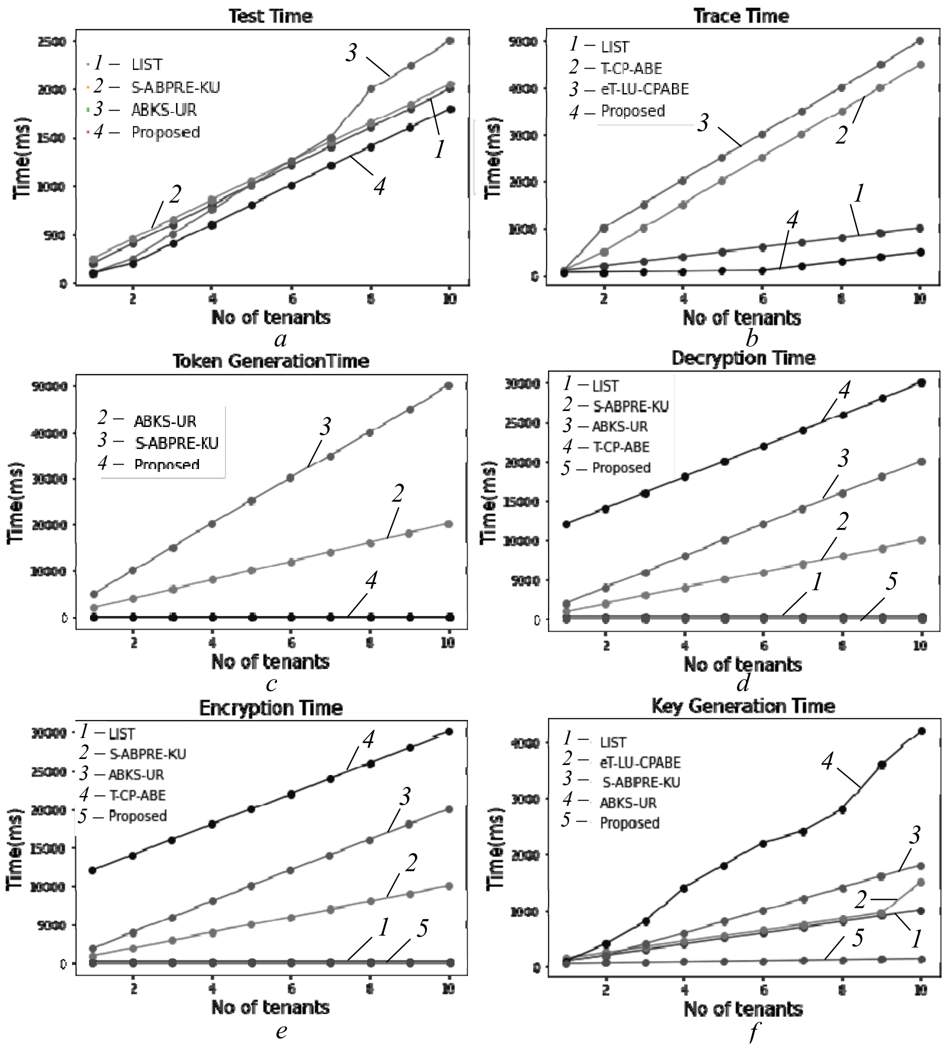

Fig. 10. Comparison of computation time

The token generation time for the proposed method is minimum when compared with others. The maximum values for the same with 10 tenants are found to be 50000 ms for the SABPRE method whereas for the proposed method it is 1 ms. The encryption and decryption times are compared in which the encryption and decryption time for the proposed method is constant for any number of tenants for other methods it linearly varies. The maximum value is obtained as 30000ms whereas for the proposed method it is 1 ms. For the key generation time, the value is 1 ms for any tenant value. But for other methods, it is constantly increasing with the increase in tenant number. The maximum value is 4500 ms for the ABKS method.

Storage and transmission overhead is compared with existing models in which the proposed model makes constant size irrespective of the tenant number, whereas all other methods vary with tenant size. The proposed method has the least size for all the storage and transmission sizes. All the size values range from 1 to 300000 bits with the proposed system (Fig. 11).

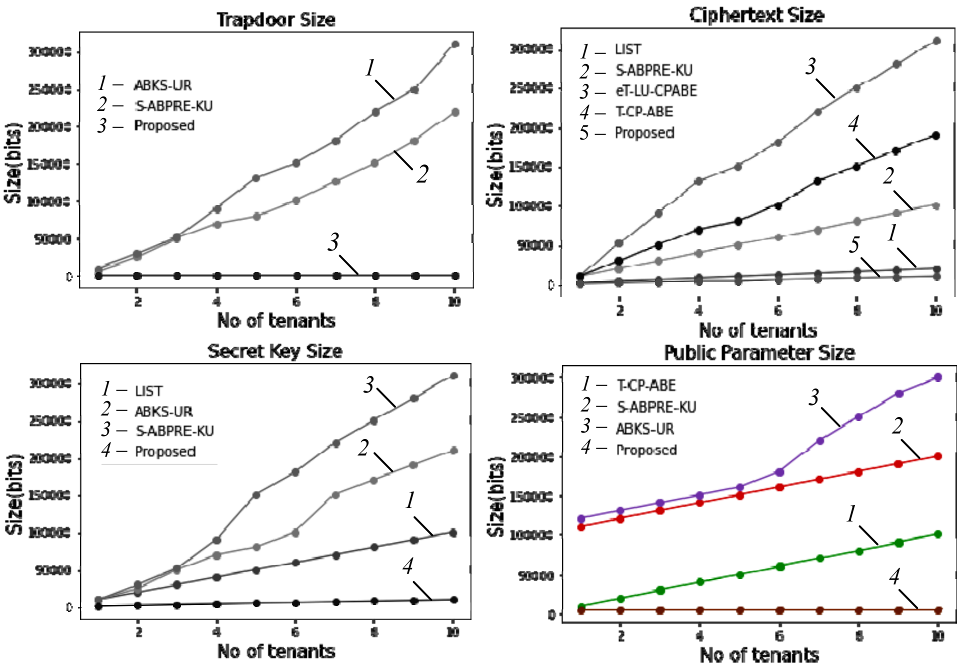

Fig. 11. Storage and transmission comparison

CONCLUSION

This study presented a data-sharing and revocation system based on ciphertext and ECC. Without giving any personal information to the cloud service provider, the majority of the decryption computational burden was transferred. Each authorized user employs a one-time password to decode the cipher text and a private secret key to decode the original cipher text. Thus, only an authorized user's secret key and session key may be used to view the plaintext communication. To tackle Sybil and resource depletion threats in a cloud context, another novel methodology was presented. This method not only prevents Sybil attacks within the planes of control and data from overwhelming cloud infrastructure, but it in turn enhances the level of service delivered to cloud users. Identification of Sybil nodes created as a result of the Sybil attack is done, and all authorized entities detected in the smart health system are informed of the updated revocation list. The system performed 90% accurately, according to the results with 21% of similarity found. The sensitivity of the system for any input change is depicted to be 98% for any given number of attributes. The error is determined to be low as 10% and the false positive rate is found to be 2% for the system and the system performance was compared with other existing techniques.

REFERENCES

1. Remya Sivan and Zuriati Ahmad Zukarnain, "Security and privacy in cloud-based e-health system," *Symmetry,* vol. 13, no. 5, pp. 742, 2021.
2. Umar Zaman et al., "Towards Secure and Intelligent Internet of Health Things: A Survey of Enabling Technologies and Applications," *Electronics,* vol. 11, no. 12, pp. 1893, 2022.
3. Tuan Minh Nguyen Dang, Nhan Nguyen Thanh, and Loc Le Tuan, "Applying a mindfulness-based reliability strategy to the Internet of Things in healthcare–A busi-

ness model in the Vietnamese market," *Technological Forecasting and Social Change,* vol. 140, pp. 54–68, 2019.

4. Kavita Jaiswal and Anand Veena, "A survey on IoT-based healthcare system: potential applications, issues, and challenges," *Advances in Biomedical Engineering and Technology*. Springer, Singapore, pp. 459–471, 2021.
5. Ying Jin et al., "Identifying human body states by using a flexible integrated sensor," *npj Flexible Electronics,* vol. 4, no. 1, pp. 1–8, 2020.
6. L. Minh Dang et al., "A survey on internet of things and cloud computing for healthcare," *Electronics,* vol. 8, no. 7, pp. 768, 2019.
7. Sherali Zeadally et al., "Smart healthcare: Challenges and potential solutions using internet of things (IoT) and big data analytics," *PSU research review,* 2019.
8. Shahidinejad Ali, Mostafa Ghobaei-Arani, and Mohammad Masdari, "Resource provisioning using workload clustering in cloud computing environment: a hybrid approach," *Cluster Computing,* vol. 24, no. 1, pp. 319–342, 2021.
9. M. Saleh Altowaijri, "An architecture to improve the security of cloud computing in the healthcare sector," *Smart Infrastructure and Applications.* Springer, Cham, pp. 249–266, 2020.
10. L. Pallavi, A. Jagan, and B. Thirumala Rao, "Mobility Management Challenges and Solutions in Mobile Cloud Computing System for Next Generation Networks," *International Journal of Advanced Computer Science and Applications,* vol. 11, no. 3, 2020.
11. Li Xiaoqing et al., "Building the Internet of Things platform for smart maternal healthcare services with wearable devices and cloud computing," *Future Generation Computer Systems,* vol. 118, pp. 282–296, 2021.
12. G. Lavanya et al., *IoT Enabled Assisting Device for Seizures Monitoring*. 2019, pp. 10–14.
13. M. Anuradha et al., "IoT enabled cancer prediction system to enhance the authentication and security using cloud computing," *Microprocessors and Microsystems*, vol. 80, pp. 103301, 2021.
14. Thilakarathne Navod Neranjan, Mohan Krishna Kagita, and Thippa Reddy Gadekallu, "The role of the internet of things in health care: a systematic and comprehensive study," *Available at SSRN,* pp. 3690815, 2020.
15. Mehrotra Preeti, Preeti Malani, and Prashant Yadav, "Personal protective equipment shortages during COVID-19—supply chain–related causes and mitigation strategies," *JAMA Health Forum*, vol. 1, no. 5, 2020.
16. Verma Deepak Kumar and Tanya Sharma, "Issues and challenges in cloud computing," *Int. J. Adv. Res. Comput. Commun. Eng.,* vol. 8, pp. 188–195, 2019.
17. Siyal Asad Ali et al., "Applications of blockchain technology in medicine and healthcare: Challenges and future perspectives," *Cryptography,* vol. 3, no. 1, pp. 3, 2019.
18. Kumar Y. Kiran and R. Mahammad Shafi, "An efficient and secure data storage in cloud computing using modified RSA public key cryptosystem," *International Journal of Electrical and Computer Engineering,* vol. 10, no. 1, pp. 530, 2020.
19. Hans-Georg Eichler et al., "Data rich, information poor: can we use electronic health records to create a learning healthcare system for pharmaceuticals?," *Clinical Pharmacology & Therapeutics,* vol. 105, no. 4, pp. 912–922, 2019.
20. Yan Shengguang et al., "Concurrent healthcare data processing and storage framework using deep-learning in distributed cloud computing environment," *IEEE Transactions on Industrial Informatics,* vol. 17, no. 4, pp. 2794–2801, 2020.
21. Hao Meng et al., "Privacy-aware and resource-saving collaborative learning for healthcare in cloud computing," *ICC 2020-2020 IEEE International Conference on Communications (ICC).* IEEE, 2020.
22. Yudi Zhang et al., "Efficient identity-based distributed decryption scheme for electronic personal health record sharing system," *IEEE Journal on Selected Areas in Communications,* vol. 39, no. 2, pp. 384–395, 2020.
23. Akhilendra Pratap Singh et al., "A novel patient-centric architectural framework for blockchain-enabled healthcare applications," *IEEE Transactions on Industrial Informatics,* vol. 17, no. 8, pp. 5779–5789, 2020.

24. Xu Yang et al., “Efficient and anonymous authentication for healthcare service with cloud based WBANs,” *IEEE Transactions on Services Computing,* 2021.
25. Seunghwan Son et al., “Design of secure authentication protocol for cloud-assisted telecare medical information system using blockchain,” *IEEE Access,* vol. 8, pp. 192177–192191, 2020.
26. Wenhai Sun et al., “Protecting your right: Verifiable attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud,” *IEEE Transactions on Parallel and Distributed Systems,* vol. 27, no. 4, pp. 1187–1198, 2014.
27. Kaitai Liang and Susilo Willy, “Searchable attribute-based mechanism with efficient data sharing for secure cloud storage,” *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 1981–1992.
28. Jianting Ning et al., “White-box traceable ciphertext-policy attribute-based encryption supporting flexible attributes,” *IEEE Transactions on Information Forensics and Security,* vol. 10, no. 6, pp. 1274–1288, 2015.
29. Liu Zhen, Zhenfu Cao, and S. Duncan Wong, “White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures,” *IEEE Transactions on Information Forensics and Security,* vol. 8, no. 1, pp. 76–88, 2012.
30. Yang Yang, et al., “Lightweight sharable and traceable secure mobile health system,” *IEEE Transactions on Dependable and Secure Computing,* vol. 17, no. 1, pp. 78–91, 2017.
31. Q. Zhang, Q. Liu, and G. Wang, “PRMS: A personalized mobile search over encrypted outsourced data,” *IEEE Access*, 6, pp. 31541–31552, 2018.

Received 21.10.2022

INFORMATION ON THE ARTICLE

Dipa D. Dharmadhikari, ORCID: 0000-0001-7206-5573, CSIT Dr. Babasaheb Ambedkar Marathwada University, Aurangabad, India, e-mail: scholar.dipaddharmadhikari@gmail.com

Dr. Sharvari Chandrashekhar Tamane, ORCID: 0000-0001-9350-1549, MGM’s Jawaharlal Nehru Engineering College, MGM University, Aurangabad, India

РОЗШИРЕНА СХЕМА БЕЗПЕКИ ДЛЯ СПІЛЬНИХ ДИНАМІЧНИХ ДАНИХ З ЕФЕКТИВНОЮ ЛЕГКОЮ ЕЛІПТИЧНОЮ КРИПТОГРАФІЄЮ / Діпа Д. Дхармадхікарі, Шарварі Чандрашекхар Тамане

Анотація. Технологія хмарних обчислень прогресує, тому Cloud Computing (CC) створює різноманітні хмарні сервіси (CS). Користувачі можуть отримати простір для зберігання від постачальника, оскільки послуги хмарного зберігання досить практичні; багато користувачів і компаній зберігають свої дані у хмарному сховищі. Конфіденційність даних стає більшим ризиком для постачальників послуг, коли більше інформації передається CS. У роботі підхід до шифрування тексту та еліптичної кривої (ECC) із шифруванням на основі ідентифікації (CP-IBE) використовується у хмарному середовищі для забезпечення безпеки даних у середовищі охорони здоров’я. Проблема відкликання стає складною, оскільки характеристики використовуються для створення шифрованих текстів і секретних ключів, отже, вводиться алгоритм відкликання користувача, для якого секретний ключ маркера унікально створюється для кожного рівня, що забезпечує безпеку. Початкова операція, включаючи підпис, публічні перевірки, динамічні дані, чутливі до атак Sybil, для подолання цього вводиться алгоритм перевірки атак Sybil, який ефективно захищає систему. Крім того, умови для публічного аудиту з використанням спільних даних і типових стратегій, включаючи аналітичну функцію, безпеку та умови продуктивності, аналізуються щодо точності, чутливості та подібності.

Ключові слова: зашифрований текст, відкликання користувача, обмін даними, CC, ECC, питання безпеки.