Computer Networks 178 (2020) 107344

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Cloud based security on outsourcing using blockchain in E-health systems

T. Benil[a,*], J. Jasper[b]

[a] Department of Computer science, Ponjesly College of Engineering, Anna University, Nagercoil, Tamil Nadu, 629003, India
[b] Department of Electrical and electronic engineering, Anna University, Ponjesly College of Engineering, Nagercoil, Tamil Nadu, 629003, India

ARTICLE INFO

Keywords:
Blockchain
Signature
Integrity
Certificateless
Aggregated
ECC

ABSTRACT

Recently, Electronic Health Records (EHR) plays a significant role in E-health care system that allows data exchange of patient's health records via a portal designated by healthcare professionals. In this, the lack of confidentiality and integrity factors leads to different security issues on sensitive health information and causes a serious impact on a patient's life. To accomplish the security of medical data, we propose a new scheme named Elliptical Curve Certificateless Aggregate Cryptography Signature scheme (EC-ACS) for the public verification and auditing in the Medical Cloud Server (MCS) to secure EHR using authorized blockchain technology. In this, we use Elliptic Curve Cryptography (ECC) to encrypt medical data and the Certificateless Aggregate Signature scheme (CAS) to generate the digital signature for sharing and storing data in the cloud storage. This proposed scheme ensures security, privacy and safeguards the confidential information from unauthorized access in the cloud health system. Furthermore, the blockchain technique guarantees the integrity, traceability and secure storage of medical records in the cloud environment.

1. Introduction

The modern technologies have profoundly influenced traditional health care practices in monitoring the health of a person by using various technologies like sensors, smartphones, wearable devices, PDAs (Personal Digital Assistant) and so on [1]. Tobacco usage, reduced physical activity, hypertension, change in lifestyle, unhealthy eating habits, and improper diet may lead to several health problems [2]. A recent study conducted among 2061 candidates showed that 32% of them suffer from hypertension, 8% have diabetes, and when 85 men were tested, reports showed that 55% of them have high cholesterol [3]. Due to the increase in health crises, the heath of a person needs to be monitored periodically. In the traditional system, the medical staff should periodically check the condition of the patient and report it in a paper. But the advancement in technologies enables the user to monitor the health in real-time using various health care applications [4]. The sensor is the most commonly used devices to track a patient's health by placing the sensors in the body either internally or externally. By using this, one can monitor the number of calories consumed, rate of heartbeat, number of steps taken, blood pressure, amount of glucose level in the blood, cardiac monitoring and so on [5]. Personal Health Records (PHR) are controlled only by the user to monitor the fitness level [6]. The data monitored by the user may be flawed and cannot be relied on for major health issues. On the other hand, Electronic Health Records are used by the health care providers, therapists, nurses, radiologists and physicians for monitoring their patients who are suffering from long term illness like chronic illness, cancer, diabetes, asthma, etc. [7].

The potential solution to efficiently share data in real-time by using cloud computing [8]. The data collected from the patient becomes available in the "cloud", from there, the data is distributed to the radiologist and then to a doctor. The test result is then achieved at the hospital which may be accessed by a doctor in another hospital. Through cloud computing, the errors in collecting the data manually are eliminated and also real-time data collection is provided with a low cost [9]. The data stored in the cloud may face security issues and data integrity [10]. The concerned doctor may tamper the medical data of a patient to cover up his mistake or for political reasons. In addition to that, the cloud service provider may be dishonest by deleting the file which is not frequently used, to save the memory space [11]. In order to mitigate these security issues and tampering, a blockchain system can be employed [12]. The concept of blockchain was first implemented in Bitcoin to transfer money without any centralized infrastructure (bank) [13]. This technique can also be implemented in sharing EHR more securely. Here, even if the adversary tries to tamper the data, the blockchain technique creates a new block for the alteration made with time stamp and hash. The previous timestamp is added to the new timestamp, creating a chain. By doing this, the original data can always be retrieved and hence tampering of data becomes nearly impossible for the adversary.

When a patient consults a doctor, the treatment depends upon the accurateness of the health information received by querying the patient. In the case of long term illness, the patient may forget the details regard-

* Corresponding author.
E-mail address: benilt2020@gmail.com (T. Benil).

https://doi.org/10.1016/j.comnet.2020.107344
Received 16 December 2019; Received in revised form 26 March 2020; Accepted 31 May 2020
Available online 3 June 2020
1389-1286/© 2020 Published by Elsevier B.V.

ing the disease or may find hard to describe the previous diagnosis to the doctor due to a lack of medical knowledge. This affects further treatment procedures. Hence, in order to overcome these issues, the health records of the patient are outsourced to the cloud server so that the entire health record of the patient can be accessed by the doctors anywhere and at any time. Hence, to avoid the forgery of the document, the EHR should be kept more securely. Thus a suitable technique has to be implemented to provide confidentiality, privacy, and security of the data.

The Significant contribution of the proposed work:-

- This work proposes a new public EC-ACS verification and auditing schemes using authorized blockchain technology in MCS to detect the data corruption in the cloud as soon as possible. Therefore, we designed a modified ECC digital signature algorithm with a message authentication code to encrypt the confidentiality of medical data to be stored in the cloud.
- Then bilinear pairing of ECC is used to generate session keys between the data owner, data user and MCS to securely share data to access and store the private healthcare data. Here the authorized blockchain-based data storing in the cloud to provide an auditing, data integrity, and secure data trailing to shows the modification records on medical data.
- Finally, performance analysis and security analysis are performed with the proposed scheme to conform is well suitable for data sharing and data storing in the cloud.

The rest of the section is as follows: - In Section 2, some of the related work to our work are presented. In Section 3, some of the preliminaries corresponding to our proposed model are presented. In Section 4, a main system model for the proposed design constraints is explained. Section 5 describes the proposed scheme. Finally, Sections 6 and 7 end with performance evaluation and conclusion.

2. Related work

Recently, several works have been carried out in enhancing the security of EHR by using blockchain technology. Gai et al. [14] presented a combination of edge computing together with an permission blockchain to map and govern the node of the network to detect energy-related attacks in the Smart Grid Network (SGN). Then used a blockchain with a voting function to validate users and some supernodes in the blockchain to organize resource allocation. The three elements taken into consideration are energy consumption, execution times and security performance.

Guo et al. [15] proposed the Multiple Authority secure Attribute-Based Signature (MA-ABS) scheme using blockchain technology. Initially, the data block is signed by multiple authorities or doctors. In the next stage, the private key is shared for every two doctors. Each private key shared is embedded on to the private key of the patient. And finally, the data is verified by a third-party verifier. This method restricts collusion attack and prevents tampering of data but, as the number of authorities increases, the cost of the protocol also increases linearly.

Gai et al. [16] presented a blockchain-based Internet-of-Edge (BloE) differential privacy preservation system to prevent information in blocks from data mining based attacks and to shield raw allocation data. The feature of blockchain uses an alias function to produce scalable and immutable performance to support task allocation in edge computing. Each of the blocks in blockchain saves allocation information to adjust the tolerance errors. The three designed objectives achieved by this method are tamper-proof, task allocation function and privacy prevention.

Cao et al. [17] proposed Tamper-Proofing Electronic Health Record (TP-HER) system using blockchain to share the medical data in a secure way. The protocol consists of four phases: setup, appointment, store, and audit. In the setup phase, the physician and the cloud storage server creates an account in the Ethereum and share them to the auditors and other doctors. In the appointment phase, the person makes an appointment with a physician and in return, the hospital sends the time of appointment. In the store phase, the doctor outsources the health records of the patient to the cloud server. Finally, in the auditing phase, the auditor checks the believability of the outsourced data. In this method, the forgery of the doctors is avoided and security is improved against password guessing attacks with decreased communication overhead, and less delay, but lacks Quality of Service (QoS).

Zhang et al. [18] proposed the identity-based proxy-oriented outsourcing scheme where the medical data is encrypted by the proxy using a light-weight symmetric algorithm. The encrypted data is then authenticated by a digital signature algorithm and is uploaded into the cloud. This encryption makes it difficult for the adversary to forge the medical data block. Finally, the Third-Party Auditor (TPA) appointed by the original owner verifies the stored data in the cloud to obtain confidentiality. This algorithm proves to be more efficient and lightweight but the delay in performance is not optimum.

Zhang et al. [19] proposed a Certificate-less Public Verification scheme against Procrastinating Auditors (CPVPA) scheme. This scheme consists of two phases. In the first phase, the auditing is conducted where the third party auditor examines the outsourced data. In the second phase, the believability of the auditor is verified by the user itself. This two-step verification increases the reliability of the outsourced health records compared to most of the existing methods. However, the system needs further improvement in terms of functionality, performance, and efficiency.

Gai et al. [20] presented a privacy-preserving blockchain-enabled Trading (PBT) model to prevent adversarial activities. In order to ensure accurate open record transactions and prevent data mining based attacks. The approach uses account mapping techniques to prevent attackers from directly touching data and a consortium blockchain to publicly store data on the chain. Therefore, the proposed approach covers the two objective of trading storage and privacy protection.

Zhang and Lin [21] proposed a Blockchain-based Secure Privacy-Preserving (BSPP) protocol to securely outsource the Physical Health Information (PHI). Here, the proposed method consists of three phases: setup phase, data generation phase and data access phase. In the first phase, the patients and the doctors create their private and public keys by choosing random numbers. In the second phase, the patient makes an appointment with the doctor. After examination, the physician updates the health record of the patient to the server and encrypts the data with the public key of the user. Additionally, the doctor generates proof in his private key to establish proof of conformance in the blockchain. The block generated by the doctor for the patient is now broadcasted to the hospital's private blockchain. On receiving the new data block, the auditor verifies the data and gives a validation confirmation message. In the data access phase, if the dishonest doctor tries to modify the data, the original data can always be accessed by decrypting. This protocol employs secure hunt, high data security, and time-controlled withdrawal. As the package length increases, the time cost, verification, and transmission process also increase. The comparison and analysis of the existing methods are given in Table 1.

3. Preliminaries

Before moving on to the proposed scheme, some preliminaries are introduced to assist the following description.

3.1. E-healthcare data

The term health data is defined as the health status of patients in the form of data flow or EHR. An EHR is a type of electronic version consisting of patient histories maintained by the provider. Each of the health data flows is assigned in a unique ID with a time label. This data is cut and saved into multiple blocks in terms of time interval. All data blocks with the same data flow share the same data ID with different indexes that have a start and end time label.

Table 1
Comparisons and analyses of the proposed and existing methods.

Author / Reference	Methods	Properties			Experimental Platform
		Security model	*Security Assumption*	*Resisting collusion attack*	
Guo et al. [15]	MA-ABS scheme	Random model	CBDH	Yes	-
Cao et al. [17]	TP-EHR scheme	Random model	Distributed Denial of Service (DDoS)	Yes	MIRACL library
Zhang et al. [18]	IBPOPA scheme	Random model	ECDLP	No	MIRACL library
Zhang et al. [19]	CPVPA scheme	Random model	CDH	No	MIRACL library
Zhang and Lin [21]	BSPP method	Random model	q-BDHI	No	JPBC library
Proposed method	EC-ACS method	Random Oracle model	BCDH	Yes	MATLAB R-2018 b

3.2. *Key generation*

There are many types of public key cryptography (PKC) including RSA, Diffie-Helman, etc. Among them, Elliptical curve cryptography (ECC) is an easy way to encrypt data and only specific people can decrypt the data. It uses two keys of public and private key for encryption and decryption, so one can easily look at the encrypted data during transmission and impossible to read the message content. Each of the PKC algorithms has its unique trapdoor function. Comparing RSA with ECC, both generate a public and private key and securely establish communication between two parties. The main difference between them is the size of the bit key. In RSA, the size of the bit key is 3072 and lower for ECC than a key of 256 bits. ECC trapdoor function is same as a mathematical game of pool. It starts with a certain point in the elliptical curve and use a dot function to view the next point. This dot function process is repeated to hop around the curve until finding the endpoint. The advantage is, it is impossible to find how many hops it took.

In this implementation, ECC key generation method are used for generating a private-public key pair for presenting the security and privacy for the system model. In the proposed method, the following keys are used to perform the role of user authentication, user verification, request generation, auditing, storing and request authentication. The types of keys generated are (i) Data owner (DO) public-private key pair (ii) Data user (DU) public-private key pair (iii) Medical Cloud server public-private key pair and (iv) Transaction over blockchain public-private key pair.

3.3. *Blockchain based medical data in cloud environment*

Nowadays, storing and processing of healthcare data is considered as the main problem for humans. Cloud computing and blockchain technology shows there usability separately. Therefore, in our proposed system model, these two technologies are combined together for storing and managing the electronic medical records in the cloud environment. The way of using the cloud in healthcare system presents a great achievement for storing medical reports and records by collaborating doctor through video calling, health care mobile, etc. But, demerits is, users need to trust the third party to ensure their data. Moreover, the data stored in cloud are not encrypted. Therefore, the main concern to be noticed is privacy and security, because the data owners and cloud service provider carries sensitive information of the patient. Hence, blockchain is employed with cloud, so it is impossible for attackers to find a new way to enter the system.

3.3.1. *Internal workflow of blockchain in cloud*

A blockchain has a distributed shared ledger network in which the blocks are sorted in the form of a chain. The blocks in the blockchain consist of the header along with the body. Each block in the blockchain, the header carries the hash value of the previous blocks and the body maintains the initial blocks hash value of the data, in this way the blocks are connected. So in each transaction, the data stored in blocks are identified by its initial and pervious block hash value. Thus the transactions of the medical data are permanently recorded and stored in a block in the form of ledger. Once a new block is created, a hash value is calculated and validated by the miners on that network. Therefore, anybody who tries to make any changes within the data inside the individual block can change the hash value of the specific block and indicates the changes made within data. It makes the corresponding specific block to be invalid and will not add the blocks that already exist in the blockchain network. Hence, it is impossible to make any alternation in an attached block, this makes the blockchain remain secure and managed. These properties make blockchain flexible in the MCS and security is further enhanced by the encryption process. Using blockchain in the cloud for health data leads to privacy and security in data and no need for the third party separately.

3.3.2. *Transaction of blockchain*

The main components of the transaction procedure used in the blockchain are based on the address. Each of the blocks in the blockchain carries the data and address of the next blocks as an output. The transaction carried out by each block is protected using a series of signatures called e-stamps. The recipient will confirm the data sent by the owner looking at this signature, finally, the identity is verified using this signature and the public key of the previous node. After confirmation, the validated transaction blocks are linked from the genesis block to the most current block with each block linked to its previous block using the cryptographic hash of the previous block.

3.3.3. *Membership services*

These services manage the privacy, confidentiality, audibility and identity of the blockchain. In the blockchain, membership is applied only to authorization blockchains. In this proposed work, authorized blockchains are used to allow specific actors for doing transactions and validate the network.

3.3.4. *Encryption services*

The component encryption services in blockchain carry the hash function and the digital signature to allow the addition of new data in the blockchain. Using the hash function in every signature block can often protect the ledger from other changes by any doctor or attacker. Any information changes to be made in ledger are performed by a computed hash nearby the previously computed hash and stored to ledger. A new hash is computed for each time as a new transaction is added to the ledger. Digital signatures confirm that the recipient receives the transactions without intermediate parts changing or distorting the contents of the transactions. Also confirm that the transactions made from senders signed with private keys are not done by imposters.

4. Methodology of proposed system model

The proposed framework model has five different entities such as Key Generation Center (KGC), DO, Data User, Medical Cloud server (MCS), and Third Party Auditor (TPA) as shown in Fig. 1. Here, KGC is the trusted entity that generates the system parameters and partial public-private key pairs to all other four entities according to their registration. In this, the DO represents the patient who can encrypt his/her data, and uploads the cipertext of the physical information to the cloud server

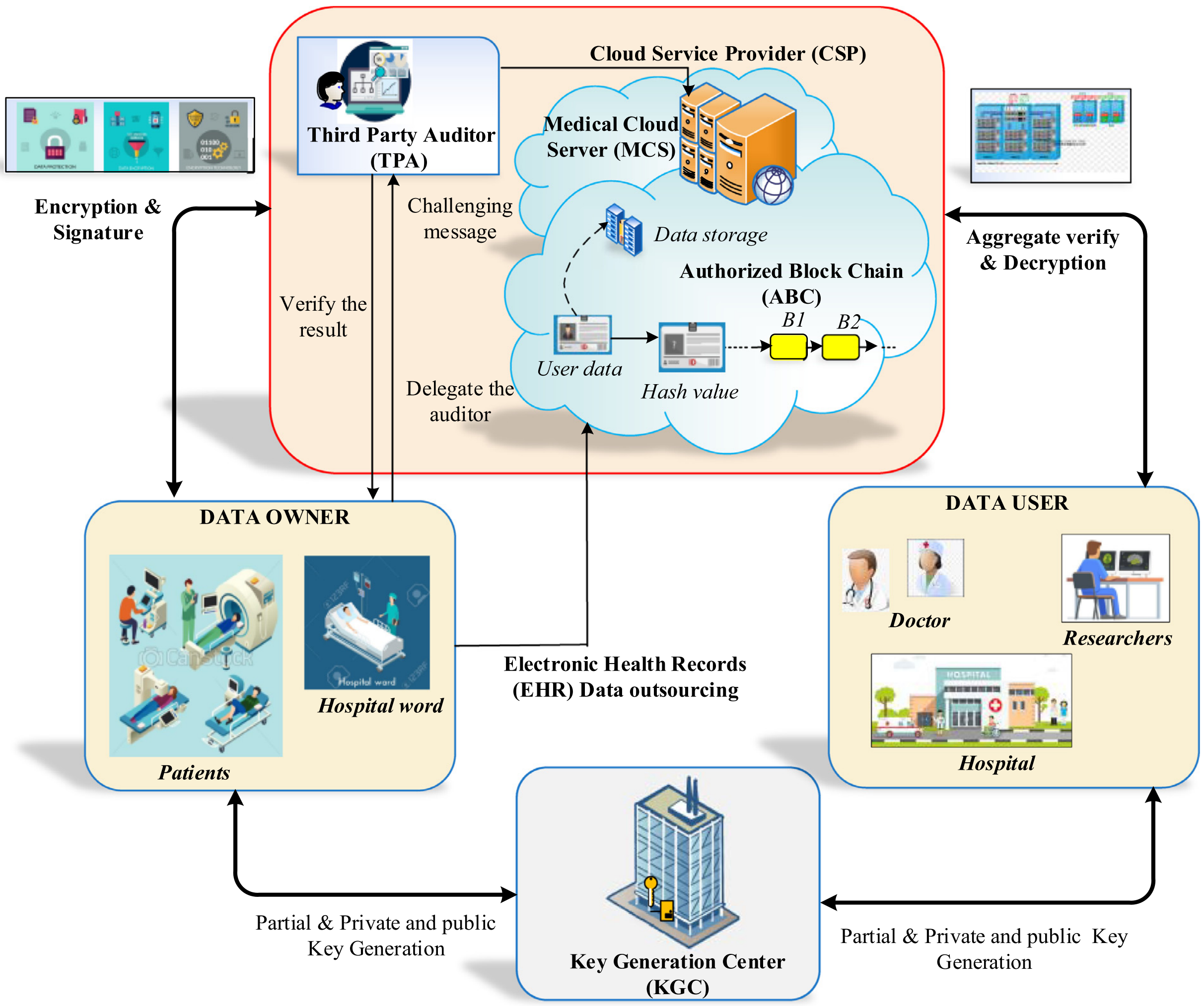

Cloud Service Provider (CSP)
Third Party Auditor (TPA)
Medical Cloud Server (MCS)
Encryption & Signature
Challenging message
Verify the result
Data storage
Authorized Block Chain (ABC)
B1
B2
User data
Hash value
Delegate the auditor
Aggregate verify & Decryption
DATA OWNER
Hospital word
Patients
Electronic Health Records (EHR) Data outsourcing
DATA USER
Doctor
Researchers
Hospital
Key Generation Center (KGC)
Partial & Private and public Key Generation
Partial & Private and public Key Generation

Fig. 1. A system model for the proposed approach.

after processing the keyword and signature tag on the document. Data user also termed as requester who may be a doctor search the encrypted data stored by the DO with some secret value (trapdoor). The MCS is managed by the cloud service provider with blockchain technology has the strong computing power and stores large amount of information's sent by the user by maintaining the actual healthcare data of the patients. TPA is the semi-trusted member, which is assigned by the user after the data outsourcing to estimate the data integrity verification process, by means of sending the audit challenge message to the server and then the server replay back with proof message.

4.1. Role of proposed approach

In this section, we consider data sharing among Owner -Cloud / Cloud- User and data storing in cloud as main problem to develop a new method. The data stored in the cloud has also been shared by many users in the cloud storage application of iCloud, Google Drive and Drop-box. However, data stored in the cloud also addresses the problem of corruption and data loss due to hardware failures, software bugs and human errors in the cloud. The purpose of the new method is to keep the sensitive information of the patients and hospital in cloud as a safe manner, and not to be explored by other external users, and not making any changes in data to corrupt their mistake.

In our proposed work, we consider Electronic Health Records (EHR) to be stored in the cloud. It contains both the patient's sensitive information (Name of patients, Phone number, ID number, etc.) and hospital sensitive information (Name of hospital, Patient ID, etc.). Therefore, to check if the data is correctly stored in the cloud, there need to check the integrity of the data in the cloud [22,23]. Hence, data sharing also considered as a common features in cloud storage. The way to solve this problem is to encrypt the data with suitable keywords before storing in cloud and verify the integrity of the encrypted file by generating a signature. Therefore, it make the whole shared files unable to see by other external users.

The role of blockchain in cloud storage receives user data in terms of an additional level of security using blockchain functionality such as public/private encryption key, a hashing function, and transaction ledgers. The origin of blockchain uses a cryptographically signed transaction to operate in each block. Where the authentication of a transaction is validated and evaluated for every single point of failure. The main component of hashes in the blockchain encrypts the data present inside the block and compute any size of data, ledger compose a set of transactions and the blocks receive a transaction ID given by end-users. The invalid transaction is rejected by the authorized blockchain mechanism. This process confirms the rigidness of data with the generated hash. To provide blockchain security, an ECC mechanism is used for blockchain transactions in terms of a block of keys such as public and private keys for the entire transaction. Here the private key is used for validating the signature on each transaction and public key check the generated signature by the private key. After completing the transactions, the data is stored in a decentralized form. By this, if attackers or hackers try to hack it, they have no information or ideas about the functionality of the blockchain that secures documents in the blockchain [24]. Whenever a user tries to recover data, the data is first validated, and if changes are made to the data, the changed data is removed from the network by the miner and created as another redundant copy near the original blocks. Therefore, users can simultaneously identify the data of the original and identical copies.

5. Design of secure EC-ACS scheme of public auditing and verification in medical cloud server using blockchain

Our proposed method includes 17 algorithms, **Setup, Partial Private Key Generation, Set Secret-Value, Set Private Key, Set Public Key, Sign, Sign-Verify, Aggregate, Aggregate-Verify, Store, Audit, Challenge, Proof generation, proof verify, Log generation, Check log and verify.** By using the first 5 algorithms, the DO, DU and MCS can able to get her partial private key pair, generate secrete key and system parameter using KGC. In 7 and 8 algorithm, collects the signature of the users and verify it. In remaining algorithm, the cloud server can able to generate a proof of possession of data and the third party provider can able to check the correctness of the proof before using cloud data. The overall flow of the EC-ACS scheme are shown in Fig. 2.

Composite order bilinear elliptical curve

Let E/F_p denote the elliptical curve E over a prime finite field F_p, the curve point $E_p(a, b)$are determined by the following equation $y^2 = x^3 + ax + b \bmod p$, where a and b are the elliptical curve parameter and x and y are the point at infinity lies on the curve. Let G be a generator group. The algorithm consider the security parameter q as input and bilinear groups of Gas an output, whereG_1, G_2 and G_T are the finite cyclic group. The order of these three groups are the composite integer$N$having large prime factor of$P_1, P_2, P_3, \ldots\ldots, P_n$. So the output of G is$(N = P_1, P_2, P_3, G, G_T, e)$, where$P_1, P_2, P_3$ is the distinct prime, $G$and $G_T$are the cyclic groups of order N and $e = G_1 \times G_2 \rightarrow G_T$ is an efficient mapping. The problem considered is, Bilinear Computational Diffe-Hellman (BCDH) which is hard to compute in the Probabilistic Polynomial Time (PPT).

5.1. Certificateless public verification (CPV) scheme

A. **Setup**

i) ***Select the ECC:*** Let us assume $G$and G_T are the generator of the respective cyclic group, G_{p_1}, G_{P_2} and $G_{P_3}$are the subgroups of order $P_1, P_2, P_3, \ldots\ldots, P_n$ in$G$. Then, give the security parameter (q), a set a master key (x) and system parameter (S_{para}). Initially, KGC compute its master public key$P_{pub} = x\,p$. Where p is the generator of cyclic group (G).

ii) ***Select the secure Cryptography hash function:*** Choose three cryptography hash function:$H_1 : \{0.1\}^* \times G \times G \rightarrow Z_q^*$, $H_2 : \{0.1\}^* \times G \times G \rightarrow Z_q^*$, and $H_3 : \{0.1\}^* \times G \times G \rightarrow Z_q^*$, where $x \in Z_q^*$is the master key select by KGC.

iii) Finally, it keep the master key x as secret and publish the system parameter as$S_{para} = \{G,\ E/F_p,\ q,\ P,\ P_{pub},\ H_1,\ H_2,\ H_3\}$.

Key generation

The key generation algorithm takes an input of system parameter, master key and set of attributes to generate a private key to encrypt and decrypt the data.

B **Partial private-key generation**

i) ***Server Partial private-key generation***(id_S)***:*** Initially take the system parameter (S_{para}), the identity of cloud server (id_S) and the master key parameter(x). Then randomly select the value $r_{id\ S} \in Z_q^*$ and compute $R_{id\,S} = r_{id\ S}\ P$ and $h_{id\,S} = H_1(R_{id\,S},\ id_S,\ P_{id\ S})$. Finally, the KGC compute the output of Partial private-key server $d_{id\,S} = r_{id\ S} + h_{id\ S} \cdot x$ and send$d_{id\ S}$, $R_{id\ S}$ to the cloud server through the secure channel.

ii) ***Owner Partial private-key generation***$(id\ _O)$***:*** During the owner extraction process the KGC takes the identity of DO$(id\ _O)$, system parameter (S_{para}), and the master key parameter(x) as the input. Initially, select the $r_{id\ O} \in Z_q^*$ random value and compute$R_{id\,O} = r_{id\ O}\ P$, $h_{id\,O} = H_1(R_{id\,O},\ id\ _O,\ P_{id\,O})$ and $d_{id\,O} = r_{id\,O} + h_{id\,O} \cdot x$. Next the KGC sends$d_{id\ O}$ and $R_{id\ O}$ to the DO.

iii) ***User Partial private-key generation*** $(id\ _U)$***:*** The KGC takes S_{para} and $id\ \ _U$ as input, then choose the random value $r_{id\ U} \in Z_q^*$ and compute the private key of user $R_{id\,U} = r_{id\ U}\ P$, $h_{id\ U} = H_1(R_{id\ U},\ id\ _U,\ P_{id\ U})$ and $d_{id\ U} = r_{id\ U} + h_{id\ U} \cdot x$. Next the KGC sends$d_{id\ U}$ and $R_{id\ U}$to the data user. Finally, check the correctness of the user private key identity by verifying$d_{id\ U}\,P = (R_{id\,U} + h_{id\ U}\ P_{pub})$.

C. **Set secret value**

i) ***Secret value of server:*** The server takes the S_{para} and id_S as input and randomly choose its secret value$Y_{id\ S} \in Z_q^*$, where $Y_{id\ S}$ is the secret value of server.

ii) ***Secret value of Owner:*** The DO takes the S_{para} and id_O as input and randomly choose its secret value$Y_{id\ O} \in Z_q^*$, where $Y_{id\ O}$ is the secret value of data owner.

iii) ***Secret value of user:*** The user takes S_{para} and id_U values as input and randomly choose its secret value$Y_{id\ U} \in Z_q^*$, where $Y_{id\ U}$ is the secret value of user.

D. **Set private key**

i) ***Set the private key of server:*** The sever takes the S_{para}, $Y_{id\ S}$ and $d_{id\ S}$ as input. Then the private key of server with identity id_S is verified by$dK_{id\ S} = id_s,\ x_{id\ S}$.

ii) ***Set the private key of owner:*** The owner takes the S_{para}, $Y_{id\ O}$ and $d_{id\ O}$ as input. Finally, the private key of the DO with identity id_O is verified by$dK_{id\ O} = id_O,\ x_{id\,O}$.

iii) ***Set the private key of user:*** The user takes the S_{para}, $Y_{id\ U}$ and $d_{id\ U}$ as input. Then check the private key of the data owner with identity id_U is$dK_{id\ U} = id_U,\ x_{id\,U}$.

E. **Set public key**

i) ***Set the public key of server:*** During the public key setting, the cloud server takesS_{para}, and $Y_{id\,S}$ as input, then compute$P_{id\ S} = Y_{id\ S} \cdot P$. Finally check the public key of the server by$C = P_{id\ S}, Y_{id\ S}$.

ii) ***Set the public key of owner:*** The input of public key setting for the data owner isS_{para}, and $Y_{id\ O}$, then compute$P_{id\ O} = Y_{id\ O} \cdot P$. Finally, verify the public key of the data owner by$PK_{id\ O} = P_{id\ O}, Y_{id\ O}$.

iii) ***Set the public key of user:*** The input of public key setting for the data owner isS_{para}, and $Y_{id\ U}$, then compute$P_{id\ U} = Y_{id\ U} \cdot P$. Finally, verify the public key of the data owner by$PK_{id\ U} = P_{id\ U}, Y_{id\ U}$.

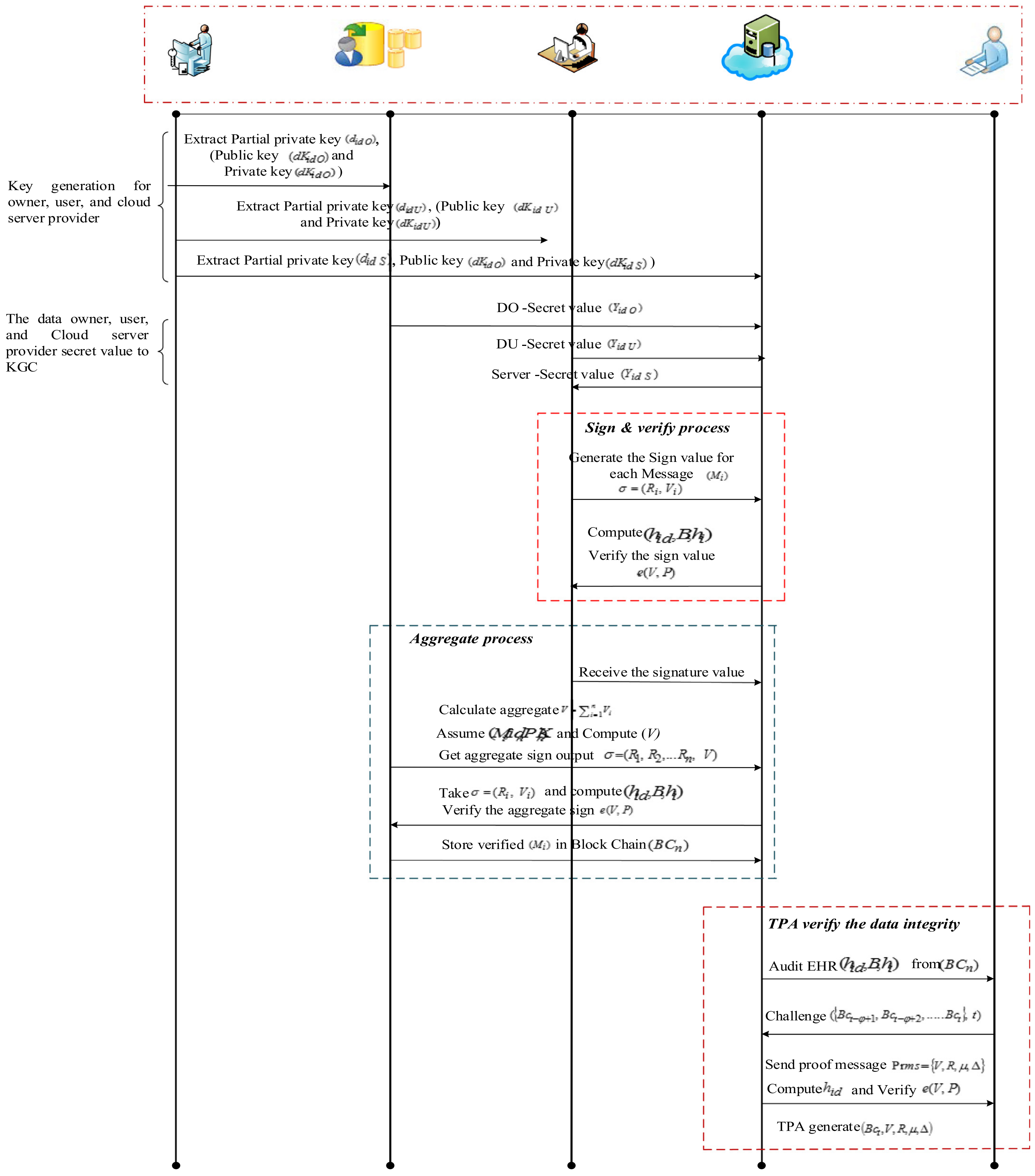
Pseudo code for key generation process

Input : System parameter (S_{para}), identity of cloud server (id_S) and master key (x)
Output: User's public, private, key Partial Private Key and secret value.

```
1. Begin
2. For each $S_{para}$ with the cloud server identity $(id_S)$
// Partial Private Key extraction
3. Perform Partial Private Key extraction for server, owner and user.
4. Extract the private key for server, owner and user.
// Public Key extraction
5. Extract the private key for server, owner and user.
6. Extract the Secret value for server, owner and user.
7. Return private, key Partial Private-Public Key (server, owner and user).
8. End for
9. End
```

F. **Sign**

The identity of user id_U takes the system parameter$S_{para} = \{G,\ E/F_p,\ q,\ P,\ P_{pub},\ H_1,\ H_2,\ H_3\}$, secret value (Y_i) the state information (Δ) and the identity of user private key$dK_{id\ U} = id_U,\ x_{id\,U}$. After that the message (M) with the abstract index as



Extract Partial private key ($d_{id\,O}$), (Public key ($dK_{id\,O}$) and Private key ($dK_{id\,O}$))
Key generation for owner, user, and cloud server provider
Extract Partial private key ($d_{id\,U}$), (Public key ($dK_{id\,U}$) and Private key ($dK_{id\,U}$))
Extract Partial private key ($d_{id\,S}$), Public key ($dK_{id\,O}$) and Private key ($dK_{id\,S}$))
The data owner, user, and Cloud server provider secret value to KGC
DO -Secret value ($Y_{id\,O}$)
DU -Secret value ($Y_{id\,U}$)
Server -Secret value ($Y_{id\,S}$)
Sign & verify process
Generate the Sign value for each Message (M_i) $\sigma = (R_i, V_i)$
Compute (h_{id}, B, h)
Verify the sign value $e(V, P)$
Aggregate process
Receive the signature value
Calculate aggregate $V = \sum_{i=1}^{n} V_i$
Assume (M_i, id, PK) and Compute (V)
Get aggregate sign output $\sigma = (R_1, R_2, ...R_n, V)$
Take $\sigma = (R_i, V_i)$ and compute (h_{id}, B, h)
Verify the aggregate sign $e(V, P)$
Store verified (M_i) in Block Chain (BC_n)
TPA verify the data integrity
Audit EHR (h_{id}, B, h) from (BC_n)
Challenge ($\{Bc_{t-\varphi+1}, Bc_{t-\varphi+2}, ...Bc_t\}, t$)
Send proof message $Prms = \{V, R, \mu, \Delta\}$
Compute h_{id} and Verify $e(V, P)$
TPA generate (Bc_t, V, R, μ, Δ)

Fig. 2. Workflow of EC-ACS of public verification and auditing in MCS using authorized blockchain technology.

input, then the signer generate the signature (σ)of the message as some following steps:

i) Select the random value $r_i \in Z_q^*$and then compute$R_{id} = r_{id} \; P$.
ii) Then compute$B = H_2(\Delta)$, $h_i = H_3(id_i, \; M_i, \; PKid_i, \; R_i)$.
iii) Compute the$V_i = d_{id} + r_i \, B, \; M + h_i, \; Y_{id}, \; P_{pub}$.

Let $\sigma = (R_i, \; V_i)$ is the output of the signature on the message(M_i, Δ , PK_{id} , T). Then the user upload the message and the signature to the cloud sever. After receiving the signature and message, the cloud server verify the computing messages and signatures.

G. **Verify**

The verifier first compute the (h_{id}, B, h_i).Then verify whether $e(V, P) = e(h_{id}, \mathrm{R_{id}} + h_i P_{pub}, P_{id}) \cdot e(r_i P, B, M)$. If the signature verification equation hold then accept the signature otherwise reject. Therefore, we improve the signature by using the user identity correctly by using V_i.

$$\begin{aligned}
e(V, P) &= e(d_{id} + r_i BM + h_i Y_{id} P_{pub}, P)\\
&= e(d_{id}, P) \cdot e(r_i BM, P) \cdot e(h_i Y_{id} P_{pub}, P)\\
&= e((\mathrm{r}_{id} + h_{id} \cdot x), P) \cdot e(r_i BM, P) \cdot e(h_i Y_{id} P_{pub}, P)\\
&= e(h_{id}, \mathrm{r}_{id} \cdot P) \cdot e(x \cdot P) \cdot e(r_i P, B, M) \cdot e(h_i Y_{id} \cdot P, P_{pub})\\
&= e(h_{id}, \mathrm{R_{id}}) \cdot e(P_{pup}) \cdot e(r_i P, B, M) \cdot e(h_i P_{id}, P_{pub})\\
&= e(h_{id}, \mathrm{R_{id}}, P_{pup}) \cdot e(r_i P, B, M) \cdot e(h_i, P_{pub}, P_{id})\\
&= e(h_{id}, \mathrm{R_{id}} + h_i P_{pub}, P_{id}) \cdot e(r_i P, B, M)
\end{aligned} \tag{1}$$

Because $d_{id} = r_{id} + h_{id} \cdot x$, $R_{id\,S} = r_{id\,S} P$, $P_{id} = Y_{id} \cdot P$ and $P_{pub} = x p$ and are substituted in $e(V, P)$. Therefore, the correctness of the proposed public verification scheme has been proved.

H. **Aggregate**

An aggregator collects all signatures on the message and runs after having run verifyS_{para}, the signature of the message $(M_1, M_2, ...M_n)$, the user $id(id_1, id_2, ...id_n)$, public key $(PK_1, PK_2, ...PK_n)$ and the aggregate signature to obtain true result. The aggregate compute $V = \sum_{i=1}^{n} V_i$ and the output of aggregate signature is generated by$\sigma = (R_1, R_2, ...R_n, V)$.

i) Initially assume the message$(M_1, M_2, ...M_n)$, the user id$(id_1, id_2, ...id_n)$, and public key$(PK_1, PK_2, ...PK_n)$. Then, compute$V = \sum_{i=1}^{n} V_i$.

ii) Finally, the last output $\{\{Y_{id}, M_i, P_{id}, R_i\}_{i=1}^{n}, \Delta, V\}$as an aggregate the signature for n message.

I. **Aggregate verify**

This algorithm is used to verify and aggregate the signature for n message. It aggregates n message signatures by following steps:

i) Initially take the signature value $\sigma = (R_i, V_i)$ of message (M) on identity (id_i) with the public key$(PK_{id\ i})$ and state information (Δ).

ii) Then compute$h_{id} = H_1(R_{id,} id, P_{id})$, $B = H_2(\Delta)$, and $h_i = H_3(id_i, M_i, PKid_i, R_i)$.

iii) Verifies the

$$e(V_i, P) = e\left(\sum_{i=1}^{n} (h_{id}, \mathrm{R_{id}} + h_i P_{pub}, P_{id})\right) \cdot e\left(\sum_{i=1}^{n} (r_i P, B, M)\right) \tag{2}$$

If the above equation holds, the algorithm returns true. Else, it returns false.

The correctness of verification of above equation can be proved as follows:

$$\begin{aligned}
e(V, P) &= e\left(\sum_{i=1}^{n} (d_{id} + r_i BM + h_i Y_{id} P_{pub}, P)\right)\\
&= e\left(\sum_{i=1}^{n} (d_{id}, P)\right) \cdot e\left(\sum_{i=1}^{n} (r_i BM, P)\right) \cdot\\
&\quad e\left(\sum_{i=1}^{n} (h_i Y_{id} P_{pub}, P)\right)\\
&= e\left(\sum_{i=1}^{n} ((\mathrm{r}_{id} + h_{id} \cdot x)), P\right) \cdot e\left(\sum_{i=1}^{n} (r_i BM, P)\right) \cdot\\
&\quad e\left(\sum_{i=1}^{n} (h_i Y_{id} P_{pub}, P)\right)\\
&= e\left(\sum_{i=1}^{n} (h_{id}, \mathrm{r}_{id} \cdot P)\right) \cdot e\left(\sum_{i=1}^{n} (x \cdot P)\right) \cdot\\
&\quad e\left(\sum_{i=1}^{n} (r_i P, B, M)\right) \cdot e\left(\sum_{i=1}^{n} (h_i Y_{id} \cdot P, P_{pub})\right)\\
&= e\left(\sum_{i=1}^{n} (h_{id}, \mathrm{R_{id}})\right) \cdot e\left(\sum_{i=1}^{n} (P_{pup})\right) \cdot\\
&\quad e\left(\sum_{i=1}^{n} (r_i P, B, M)\right) \cdot e\left(\sum_{i=1}^{n} (h_i P_{id}, P_{pub})\right)\\
&= e\left(\sum_{i=1}^{n} (h_{id}, \mathrm{R_{id}}, P_{pup})\right) \cdot e\left(\sum_{i=1}^{n} (r_i P, B, M)\right) \cdot\\
&\quad e\left(\sum_{i=1}^{n} (h_i, P_{pub}, P_{id})\right)\\
&= e\left(\sum_{i=1}^{n} (h_{id}, \mathrm{R_{id}} + h_i P_{pub}, P_{id})\right) \cdot\\
&\quad e\left(\sum_{i=1}^{n} (r_i P, B, M)\right)
\end{aligned}$$

Therefore, the verification Eq. (2) is correct.

Pseudo code for sign generation and aggregation process

Input: User's Public (P_{id}), Private (dK_{id}), key Partial Private Key (d_{id}) and secret value $(Y_{id\ S})$.
Output: Sign value, Aggregate result

```
1. Begin
2. For each cloud user identity $(id_U)$
3. Generate signature $\sigma = (R_i, V_i)$
4. Verify the sign
5. End for
6. Begin
7. Aggregate the Sign$\sigma = (R_i, V_i)$, Messages$(M_1, M_2, ....M_n)$, User id$(id_1, id_2, ....id_n)$, and public key$(PK_1, PK_2, ....PK_n)$.
8. Verify the Aggregate Eq. (2)
9. End for
10. End
```

5.2. Certificateless Public Auditing (CPA) scheme

In this section, we describe the efficiency of certificate less public auditing scheme based on the aggregate signature scheme. This scheme consist of six algorithm: setup, store, Audit, challenge, Proof and verify. The first algorithm is same as those in Section 5.1, so we only show the details about last five algorithm as follows.

Phase 1: *TPA in cloud verify the data integrity*

J. **Store**

i) The messages (M) are stored in to block (n):$M = \{m_i\}_{1 \le i \le n}$. Then select the random elements for each user $U_{id} \in Z_q^*$ and randomly select the state of information$\Delta \in Z_q^*$. Compute$T = H(U_{id} \parallel m \parallel \Delta \parallel PK_{id})$.

ii) The user randomly choose $r \in Z_q^*$ and compute$h_{id} = H_1(R_{id,} id, P_{id})$, $B = H_2(\Delta)$, and $h_i = H_3(id_i, M_i, PKid_i, R_i)$.

iii) The user outsource $\tilde{M} = \{M, \{Y_i\}_{i=1,2,..n}, \Delta\}$ to cloud server(C_s). After that check the correctness of the data by verifying the equation,

$$e(V_i, P) = e\left(\sum_{i=1}^{n} (h_{id}, \mathrm{R_{id}} + h_i P_{pub}, P_{id})\right) \cdot e\left(\sum_{i=1}^{n} (r_i P, B, M)\right) \tag{3}$$

If the verified equation is true, the cloud server (C_s) accept the Message (M).

K. **Audit**

Initially, we give the EHR (h_{id}, B, h_i), the auditor is able to check the correctness and timeline as follows:

i) Analysis the EHR and get (h_{id}, B, h_i).

ii) Extract the corresponding transaction from the Blockchain (Bc) and get the corresponding information and verify the transaction.

Then, the data integrity in cloud server is verified by three algorithms Challenges, Proof-gen and proof-verify. In the real cloud, there is usually more than one user in a data group. Therefore, we extend our scheme to support the multiuser group and make the verification scheme more efficient when we carry out the batch verification in this document. Then we show the details of the other three different algorithms. Here, the TPA first generates the challenging message as follows:

L. **Challenge**

To check the integrity of outsource data, TPA first generate the challenging messages as follows:

i) Initially, we extract $\{Bc_{t-\phi+1},\ Bc_{t-\phi+2},\ .....\ Bc_t\}$ from the Ethereum Blockchain (EBC) based on the agreed verification time and extract the hash value of the data from the transaction. Whereϕ denote the number of block chain used to confirm a transaction, t represents the height of the block that is latest confirmed at the present time.

ii) Then, set $(\{Bc_{t-\phi+1},\ Bc_{t-\phi+2},\ .....\ Bc_t\},\ t)$ as the challenging message and send it to the cloud server(C_s). After receiving the message $(\{Bc_{t-\phi+1},\ Bc_{t-\phi+2},\ .....\ Bc_t\},\ t)$ from TPA, the C_s first verify its validity. Whether the message sets are correct or not in the blockchain. If the validation is fail, the C_s will reject the data; otherwise it generate the proof information.

M. **Proof generation**

After receiving $(\{Bc_{t-\phi+1},\ Bc_{t-\phi+2},\ .....\ Bc_t\},\ t)$from TPA. Then check and validate whether $(\{Bc_{t-\phi+1},\ Bc_{t-\phi+2},\ .....\ Bc_t\},\ t)$ are the correct one in the blockchain. If the checking fail, the C_s will reject or the cloud server will create the proof information and its steps as follows:

i) First, the cloud server first compute the message $g_1 = h_1\,(Bc_{t-\phi+1},\ Bc_{t-\phi+2},\ .....\ Bc_t)$ and $g_2 = h_2\,(Bc_{t-\phi+1},\ Bc_{t-\phi+2},\ .....\ Bc_t)$, where h_1 denotes the secure of hash function to the key space of$\pi_g\,(\cdot)$, $h_2$denotes the secure of hash function to the key space of$f_g\,(\cdot)$.

ii) Compute $i_\xi = \pi_{g\,1}\,(\xi)$ and $v_\xi = f_{g\,2}\,(\xi)$, ξ=*1, 2, 3... n.* where n denotes the number of information block that should be checked and it is determined by$q$.

iii) Then, the $C_s$computes the value $V = \sum_{\xi i\in 1}^{C_s} v_{i\xi}\cdot s_{i\xi}$ and $\mu = \sum_{\xi i\in 1}^{C_s} v_{i\xi},\ m_{i\xi}$. After this process, send the proof message $\mathrm{Pr}_{Ms} = \{V,\ R,\ \mu,\ \Delta\}$ to TPA.

N. **Proof verify**

Upon getting the proof message fromC_s, TPA check data integrity are as follows:

i) Initially, the verifier compute $g_1 = h_1\,(Bc_{t-\phi+1},\ Bc_{t-\phi+2},\ .....\ Bc_t)$ and $g_2 = h_2\,(Bc_{t-\phi+1},\ Bc_{t-\phi+2},\ .....\ Bc_t)$.

ii) Compute the hash value $h_{id} = H_1(R_{id},\ id,\ P_{id})$ and check whether

$$e(V,\ P) = e\left(\sum_{\xi=1}^{C_s}(h_{id},\ \mathrm{R}_{\mathrm{id}} +\ h_i\,P_{pub},\ \ P_{id})\right)\cdot e\left(\sum_{\xi=1}^{C_s}(r_i\,P,\ B,\ M)\right) \tag{4}$$

Then hold it down to check the correctness and timeline of the EHR is guaranteed or not. If the above equation does not hold, the TPA will verify and accept the proof, if the elements make the equality hold; otherwise, TPA take the checking result as reject. If the verification is successful, the correctness and timing of the EHR are guaranteed.

Then hold it down to check the accuracy and timing of the EHR by checking the transaction time, if the check fails, it will be rejected, where the transaction time is derived from the EBC. Calculate the hash function and check if it is same as the data transaction information. Finally, TPA generate the entry as follows:

$$\left(Bc_t,\ V,\ R,\ \mu,\ \Delta\right) \tag{5}$$

Then store the generated entry in to a log file(Λ).

Phase 2: ***The user verify the behaviour of TPA***

O. **Log Gen**

The TPA generate the log file as follows:

For each task verification, it generate an entry as$(\{Bc_{t-\phi+1},\ Bc_{t-\phi+2},\ .....\ Bc_t\},\ t,\ S,\ \mu)$. The entry log file$(\Lambda)$stored in sequential order is provided in Table 2. Then, compute the hash function as,

$$\theta_{t(1)} = H\left(Bc^1_{t-\phi+1}\middle\|Bc^1_{t-\phi+2}\|.....\ \middle\|Bc^1_t\ \middle\|t^1\ \middle\|S^1\ \middle\|\mu^1\right) \tag{6}$$

Then generate the data transaction(T_{x1}), where the data field is set to(θ_{t1}). Then, upload it to the EBC. During this process, T_{x1}perfectly recorded in the block whose hash value is$Bc^1_{t-\phi+1}$ and the data height is$t-\phi+1$.

Table 2
Log file in the proposed method.

R			
$Bc^{(1)}_{t-\phi+1},\ Bc^{(1)}_{t-\phi+2},\ .....\ Bc^{(1)}_t)$	$t^{(1)}$	$\mu^{(1)}$	$V^{(1)}$
$Bc^{(1)}_{t-\phi+1},\ Bc^{(1)}_{t-\phi+2},\ .....\ Bc^{(1)}_t)$	$t^{(2)}$	$\mu^{(2)}$	$V^{(2)}$
.....	.....	.....	.....
$Bc^{(1)}_{t-\phi+1},\ Bc^{(1)}_{t-\phi+2},\ .....\ Bc^{(1)}_t)$	$t^{(c)}$	$\mu^{(c)}$	$V^{(c)}$

P. **Check log**

It check the validity of entry log file(Λ). Initially, we check the single entry in(Λ). Here, the user first acquire $t^{(1)}-\phi+1$ and $t^{(1)}$, then derive the physical time when the verification process is performed form$t^{(1)}$ to$t^{(1)}-\phi+1$. If the processing time is not matched with the agreed one, then the user set the result as reject.

Then, the user acquire $Bc_t^1$from EBC and extract the (θ_{t1}) form the EBC whose hash value is$Bc^1_{t-\phi+1}$. If the user extraction is failed, the result of user sets are verified and reject the unwanted sets. Again the user check whether θ_{t1} matches the entry in the first row of log file(Λ). User compute time sensitivity of data as,

$$T = H(U_{id}\,\|\,m\,\|\,\Delta\,\|\,PK_{id}\ ),i_\xi = \pi_{g\,1}(\xi),v_\xi = f_{g\,2}\,(\xi) \tag{7}$$

where, $g_1 = h_1\ Bc^1_{t-\phi+1},\ \|Bc^{\,1}_{t-\phi+2}\|....\ \|Bc^{\,1}_{t},$ $g_2 = h_2\,(Bc^1_{t-\phi+1},\ \|Bc^{\,1}_{t-\phi+2}\|....\ \|Bc^{\,1}_{t})$. Then, the user checks,

$$e(V^1,\ P) = e\left(\sum_{\xi=1}^{C_s}(h^{(\mu 1)}{}_{id},\ \mathrm{R}^{(\mu 1)}{}_{\mathrm{id}} + h_i\,P_{pub},\ \ P^{(\mu 1)}{}_{id})\right)\cdot e\left(\sum_{\xi=1}^{C_s}(r^{(\mu 1)}{}_i\,P,\ B,\ M)\right) \tag{8}$$

If the result fails, the user sets the auditing result as accept.

In practice, a user must check multiple entries to ensure the correctness of TPA's behaviour during a data check period. This process is used to reduce the auditing cost, multiple entries in the log file can be verified at the same time. Specifically the user pick a random (b_e) element subset of the set$(l, 1)$. Let $\{l_1,\ l_2,\ ..,l_{b_e}\}$ be the picked subset. The user then audit the behaviour of TPA by checking the result. If the result is failed, the user sets the auditing result as reject, otherwise the auditing result is accepted. The correctness of the proposed work is depended on the correctness of Eqs. (4) and (8), and provide the correctness proof in the following:

$$\begin{aligned} e(V,\ P) &= e\left(\sum_{\xi\in 1}^{b_e} V^{vi},\ P\right)\\ &= e\left(\sum_{\xi=1}^{C_s}(h^{(vi)}{}_{id},\ \mathrm{R}^{(\mathrm{vi})}{}_{\mathrm{id}} +\ h_i\,P_{pub},\ \ P^{(vi)}{}_{id})\right)\\ &\quad\times e\left(\sum_{\xi=1}^{C_s}(r^{(vi)}{}_i\,P,\ B,\ M)\right)\\ &= e\left(\sum_{\xi=1}^{C_s}(h^{(\mu vi)}{}_{id},\ \mathrm{R}^{(\mu\mathrm{vi})}{}_{\mathrm{id}} +\ h_i\,P_{pub},\ \ P^{(vi)}{}_{id})\right)\\ &\quad\times e\left(\sum_{\xi=1}^{C_s}(r^{(\mu vi)}{}_i\,P,\ B,\ M)\right) \end{aligned}$$

Pseudo code for Auditing

```
Input: Initialize $h_{id}$,$B$,$h_i$
Output: $e(V^1,\ P)$

1. Begin
2. Verify $h_{id}$,$B$,$h_i$
3. Extract the $\{Bc_{t-\phi+1},\ Bc_{t-\phi+2},\ .....\ Bc_t\}$
4. Proof and verify the message in Eq. (5).
5. Generate the log file in Eq. (6).
6. Check the log file in Eq. (8)
7. End for
8. End
```

5.3. Security model

For the security model, we consider three different threads such as misbehave auditor, semi-trusted sever and malicious attack.

(i) Misbehave auditor:
The TPA can hide data corruption in the cloud server, moreover, it can derive from the prescribed verification period and cannot perform verification within the program.

(ii) Semi trusted server:
The cloud server is also referred to as a semi-trusted entity, which can hide corruption data by creating a piece of test information to cheat the TPA. Therefore, we follow the previous thread model with the new EC-ACS model. Proof of work is conducted under two types of adversaries.

Type 1: Adversaries 1 (A1):- To model outsider attacks. It cannot able to obtain the user partial key or master key from the system, but it can compromise the secret value of users.

Type 2: Adversaries 1 (A2):- To model outsider attacks. It can generate each user's partial private based on system parameters and specification, but it can't able to compromise user's secret value.

(iii) **Malicious attack:**

This attack affects user-uploaded tags and loads an incorrect verification tag to save on the cloud server. Thus in this work, we take care of protecting security, privacy and integrity in the cloud server against malicious data attack. For this, the security model is defined as a game played between the Adversary (A) and the challenges under the random oracle model. The game contains three steps to play are as follows:

Initialization setup: In this first step, the challenger ($\wp$) will create the public parameter and the private key of the system. Then, the challenger send the parameters to adversaries.

Oracle query: In this step, the adversaries create user public keys, secret values and extract the partial key and sign oracle queries in various order and time based on the game rule of challenges.

Forgery: Here, challengers create the fake signature based on the oracle demand.

In this game, adversaries only break the signature scheme (Ω) if it is able to create a valid request message. Suppose that an adversary advance sign is the probability, A will break the signature scheme in this game.

5.3.1. Security analysis and proof

In this section, we analyse the security of the proposed scheme. The testing of security analysis is conducted in secret key which is used to ensure the privacy of patient information to access the information. Then, the MCS ensure the useful information, which does not accessed by any hacker or un-requesting users who does not have the required attributes. This ensure the security of EHR in medical cloud server. The EC-ACS is assessed on the security under the random oracle model.

Lemma 1. *The signature $\sigma_i = (R_i,\ V_i)$ in proposed model is existentially unforgivable if attacks on messages have been chosen in an adaptive way.*

To develop this lemma, first we define two games corresponding to the two Adversaries.

Theorem 1. *Assume there is a Probabilistic Polynomial-Time (PPT) adversary A1 who can win Game I with non-negligible probability, therefore must exist a challenger ($\wp$) who can solve the BCDH problem.*

Initialization setup: Initially, the challenger ($\wp$) run the setup algorithm and obtain the public parameters. Then, the challenger send the public parameters to Adversary A1.

Oracle query: Initially the A1 create the oracle queries in PPT and collect the answer from $\wp$ as following steps.

- Request of public key query: when A1 asks a query on user identity, the challenger search for (id, d_{id}, R_{id}) and (id_i, p_{id}, x_{id}) respectively. Then $\wp$ provide the output for $PKr(id_U,\ PK_{id\ U}{}')$.
- Replacement queries of public key $PKr(id_U,\ PK_{id\ U}{}')$: A1 is able to select a new public key $PK_{id\ U}{}' = \{h_{id\,U,0},\ h_{id\,U,1},\ PK_U\}$and sets $PK_{id\ U}{}'$as the new public key of user. The challenger ($\wp$) will record this replaced key.
- Extract the secret value: Once received the query, the challenger check the users list. If there exists entry with any user ID, the $\wp$ will return message to A1.
- Extract the partial private key: Once the query received onid_U, then the challenger check the users list, if the$id_U = id_{\ O}$, $\wp$ returns$\perp$. Else if there exists entry with any user ID, then the $\wp$ check returns d_{id} to A1, else $\wp$ execute the created id and returns d_{id} to A1.
- Sign queries $S\,(id_U, PK_{id\,U},\ \Delta_i, M_i)$: A1 can request the user signature on a message M_i under the state dataΔ_i. Based on the receiving query, the challenger create the corresponding signature σ_i for that and send it to A1.

Forgery: For the identity of user id_U, the Adversary A1 generate the output for this created public key $PK'_{id\ U}$, message M^*, state of informationΔ^* and a signature σ^*. Let's say that AI wins game I if and only if:

- The signature $\sigma^*$is valid on message M^* with the information Δ^*under user identity and new public key.
- But the new message $M^*$is not submitted during the sign queries.

Proof: Let challenger be a BCDH attackers who receive a random instance (C, C^a, C^b) of the BCDH problem in group G and need to compute C^{ab}. Here, we illustrate that if A1 is able to forge the signature with the probability, $\wp$ is able to solve the CHP problem by using the output of A1 with same probability. Because, to space limitation, we only provide the proof sketch and neglect some interaction details.

In the initial stage, the challenger set the $P_{pub} = C^a$ as an instance of BCDH problem, and simulates $H_1(\cdot), H_2(\cdot) \sim H_3(\cdot)$as random oracles. In game 1, the $\wp$ sets $h_{id} = C^{\varsigma *}$, $B = C^{\beta *}$ and $h_i = C^{\gamma *}$, where ς^*, β^* and γ^* are the randomly chosen variables. For the above replaced keys, the challenger's responses for the corresponding signature are as follows:

- Initially, $\wp$ sets$H(R) = (m\varsigma_u^*)/\varsigma_u^*$.
- Then, $\wp$ randomly select $r \in Z_q$, and compute $R_{id}\ = r_{id}\ \ P$

$$V = C^{(\varsigma^* m)xu} - \ C^{-((\varsigma'\,u^*)/\,(\beta' u^*)} \times C^{-((\gamma'^*\ mu)\,/\,(\gamma'^*\ u,1)xu} . C^{\varsigma^* r}$$

- The challenger response with (V, R).

Finally, the tuple output of A1 is$\{\sigma^*, M^*,\ \Delta^*, PK_{id\ U}{}^*\}$, where $\sigma^* = \{V^*,\ R^*\}$ denotes the valid message under Δ^* and $PK_{id\ U}{}^*$.

We have the valid signatureσ^*,

$$e\,(V^*, P) = e\left(h^{m^* u}{}_{id}, \mathrm{R}^{\mathrm{H}^* u}{}_{\mathrm{id}}\right) + e\left(h_i P_{pub}{}^{m^* u}, P^*{}_{id}\right) \times e\left(r^*{}_i P, B^*, M^*\right)^{H^*}$$

where, $H^* = H(R^*)$, and $B^* = H_2(R^*)$. Here, we discussed about the security of the signature, hence the form of $H^*$is little different from the one in EC-ACS. But this difference should not reduce the security. By these way, the challenger solve the BCDH problem.

Theorem 2. *Suppose there is an opponent in PPT A2 who can win game 2 with a not negligible probability, therefore there must be a challenger able to solve the BCDH problem.*

Initialization setup: The challenger ($\wp$) execute the setups and obtain the public and secret parameters. Then, the $\wp$send the public parameters and the KGC master key to A2.

Oracle query: Initially the A2 create the oracle queries in polynomial time and collect the answer from $\wp$ as following steps.

- Request of public key query: when A2 asks a query on user identity, the challenger answers according to the original public key.

- Replacement queries of public key: A2 is able to select a new public key and the challenger record and replaced it.
- Extract the secret value: Once received the query, the challenger check the users list. If there exists entry with any user ID, the $\wp$ will return message to A2. Else, if$id \neq id'$, then $\wp$ selects the $r_{id\ S} \in Z_q^*$ as the id secret value, then return r to A2.
- Extract the partial private key: Once the query receive onid_U. Then, the challenger ($\wp$) maintain the users list. If there exists entry with any user ID, then $\wp$ check and return d_{id} to A2, else $\wp$ execute the created id and returns d_{id} to A2.
- Sign queries$S\,(id_U, PK_{id\,U}, \Delta_i, M_i)$: A2 can request the user signature on a message M_i under the state dataΔ_i. Based on the receiving query the challenger create the corresponding signature σ_i for that and send it to A2.

Forgery: For the identity of user id_U, the Adversary A1 creates the public key$PK'_{id\ U}$, messageM^*, state of informationΔ^* and a signatureσ^*. Let's say that AI wins game I if and only if:

- σ^* Signature is a valid sign on message M^* with the state of information Δ^*under new public key and user identity.
- But the new message M^* is not submitted during the sign queries.

Proof:

Let challenger be solved BCDH attacker, who receive a random instance (C, C^a, C^b) and needs to compute C^{ab} by using A2's outcome. In initial stage, the game 2 challenger randomly select $\alpha, r_{id\ S} \in Z_q^*$ and obtains the public parameters. Then, the challenger send $\alpha$and the public parameter to A2. In game 2, $\wp$ sets$h_{id} = C^{\beta *}$, $B = C^{\gamma * \cdot \gamma' * a}$ and $h_i = C^{\varsigma' * a}$, and $PK_{id\ U} = C^{xu\, b'}$, whereβ^*,ς^*, ς'^*, $\gamma^*\ \gamma' * \in Z_q^*$ are randomly selected. For any signature query on any message m under any state information. Then the corresponding signatures are as follows:

- The challengers $\wp$ sets $H(R) = -(\gamma'^* m)/(\varsigma'^*)$.
- Then, $\wp$ randomly select $r \in Z_q$, and compute $R_{id} = r_{id}\ \ P$.

$$V = H_1(u)^\alpha \cdot\ \ H_1(u)^\alpha\ \cdot\ \ C^{\gamma^*\, mxu} \times C^{-((\gamma' * \varsigma * m/\varsigma' *)\, xu}\,.\,C^{\beta^* r}$$

- The output of $\wp$ is (V, R) as the response.

Finally, the result of A2 is tuple$\{\sigma^*, M^*, \Delta^*\}$, where $\sigma^* = \{V^*,\ R^*\}$ is valid signature of M^*under Δ^*.

Since, valid the signature σ^*, it satisfies the $(eV\ *, P)$, by our setting.

$$e(V^*, C) = e(C^{xu,bm^*\gamma^*} \times C^{xu,bm^*\gamma' a}.C^{xu,bH^*\varsigma^*} \times C^{xu,bH^*\varsigma' * a}.(R^*)^{\beta *}, C) \tag{9}$$

These above steps are used to solve the CPH problem. This conclude the proof of lemma.

Claim 1. If the cloud server passes the TPA's verification, it must possess truly the specified data intact.

Proof: To prove the authenticity of the proposed model, we define the games with interleaved analysis.

Game 0: The game 0 simply challenge the game between TPA and cloud server in BC.

Game 1: The function of this game is same as game 0, with the exception of one different that is the above adversaries are trained to be able to forge the part of the proof data in Audit. Since $\sigma_i = (R_i,\ V_i)$ in proposed model is existentially unforgeable. Here the challengers store each proof data crated by the adversary, declare the failures and aborts if. Initially, the proof information is a valid one. $\mu'$is the proof data that is different from the expectedμ.

Analysis: Initially, the game 1 is defined in the event that the challenger interrupts, and the test information's are generated by the adversary is$\{V, R, \mu, \Delta\}$. By the correctness, we have

$$e\,(V,\ P) = e\left(\sum\nolimits_{\xi=1}^{C_s} (h^{(\mu v i)}{}_{id},\ \mathrm{R}^{(\mu \mathrm{vi})}{}_{\mathrm{id}} +\ \ h_i\, P_{pub},\ \ P^{(vi)}{}_{id})\right) \times e\left(\sum\nolimits_{\xi=1}^{C_s} (r^{(\mu v i)}{}_i\, P,\ B,\ \ M)\right) \tag{10}$$

and the output of adversary's, we have

$$e\,(V,\ P) = e\left(\sum\nolimits_{\xi=1}^{C_s} (h^{(\mu' v i)}{}_{id},\ \mathrm{R}^{(\mu' \mathrm{vi})}{}_{\mathrm{id}} +\ \ h_i\, P_{pub},\ \ P^{(vi)}{}_{id})\right) \times e\left(\sum\nolimits_{\xi=1}^{C_s} (r^{(\mu' v i)}{}_i\, P,\ B,\ \ M)\right) \tag{11}$$

Since then $\mu \neq \mu'$, $\bar{\mu} = \mu - \mu' \neq 0$. Moreover, we have $B^\mu \neq B^{\mu'}$, and also,

$$e\,(h^{(\mu)}{}_{id},\ \mathrm{R}^{\mu}{}_{\mathrm{id}}) \cdot e\ (r^{\mu}{}_i\, P,\ B,\ \ M)) =\ e\,(h^{\mu'}{}_{id},\ \mathrm{R}^{\mu'}{}_{\mathrm{id}} \cdot e(r^{\ \mu'}{}_i\, P,\ B,\ \ M) \tag{12}$$

Rearrange the above equation

$$e(h^{(\mu)}{}_{id}, \mathrm{R}^{\mu}{}_{\mathrm{id}}) \cdot e(r^{\mu}{}_i P, B, M), C = e(h^{\mu'}{}_{id}, \mathrm{R}^{\mu'}{}_{\mathrm{id}} \cdot e(r^{\mu'}{}_i P, B, M), P \tag{13}$$

That is$(h^{\alpha}{}_{id},\ \mathrm{R}^{\alpha}{}_{\mathrm{id}}) \cdot\ (r^{xu}{}_i\, P,\ B,\ \ M)^{\mu} =\ (h^{\alpha}{}_{id},\ \mathrm{R}^{\alpha}{}_{\mathrm{id}}) \cdot (r^{\ xu}{}_i\, P,\ B,\ \ M)^{\mu'}$. We set$\omega = (h^{\alpha}{}_{id}, \mathrm{R}^{\alpha}{}_{\mathrm{id}})\ \cdot\ (r^{xu}{}_i P, B, M)$, and for the ωarbitrary, we represent as $\omega = (C^*)^{\varsigma *}.(C'^*)^{\varsigma' *}$, where$\varsigma'^*$, $\varsigma' * \in Z_q$, $C^*, C'^* \in G$ are the random elements. Similarly, there exists $\mathrm{X} \in Z_q$ such that $C = (C'\ *)^{\mathrm{X}}$. Here, the CPH problem gives $C'^*$and $C = (C'\ *)^{\mathrm{X}}$, then compute X. By our security setting, the solution CPH problem is expressed by$\mathrm{X} = -(\varsigma'\ *\ /\varsigma\ *)$.

Game 2: The function of game 2 is same as the game 1, but it has one difference. The adversary is prepared to have the option to fashion any piece of evidence data in Audit. That, is the challenger store all proof information's, which is created by the adversary and declare the failure process.

- The proof data $\{V', R', \mu', \Delta\}$ is a valid one.
- The above mentioned proof information is different form the expected information$\{V, R, \mu, \Delta\}$.

Analysis: In this analysis, we considered that the user as the challenger. In the initial stage of this game, set the challenger $P_{pub} = C^V$ as an instance of the CPH problem. After this process, the challenger randomly selects $\lambda,\ \lambda' \in Z_q$ and sets$H_i = -m_i\, \lambda'\, \lambda'^{-1}$. For the random value r and xu, then the challenger compute(V_i, R_i):

$$\begin{aligned} R_i &= C^{ri} \\ V_i &= \left(R_{id} \cdot B^{xu}\right)^{mi}.\left(R_{id} \cdot h_i{}^{xu}\right)^{mi}.\,h_{id}{}^{ri} \\ &= C^{\left(m_i\lambda - m_i\lambda^{-1}\right)^s}.\ \ B^{m_i\, xu} \times h_i{}^{\left(m_i\lambda - m_i\lambda'^{-1} xu\right)^s},h_{id}{}^{ri} \end{aligned} \tag{14}$$

Then the challengers continues to interact this process with adversaries. When the challenger's aborts, the received proof data is$\{V', R', \mu', \Delta\}$, otherwise we expect$\{V, R, \mu, \Delta\}$. By the correctness, we have Eq. (8) and

$$e\,(V',\ P) = e\left(\sum\nolimits_{\xi=1}^{C_s} \left(h^{(\mu' v i)}{}_{id},\ \mathrm{R}^{(\mu' \mathrm{vi})}{}_{\mathrm{id}} +\ \ h_i\, P_{pub},\ \ P^{(vi)}{}_{id}\right)\right) \times e\left(\sum\nolimits_{\xi=1}^{C_s} \left(r^{(\mu' v i)}{}_i\, P,\ B',\ \ M\right)\right) \tag{15}$$

By our setting, $e\,(V', P) \neq (V, P)$. Clearly,$\mu \neq \mu'$, then we describe $\bar{\mu} = \mu' - \mu$. Here, the CPH problem is that provide $(C, C', C^s)$and compute$(C')^s$. From this process we have$e\,(V'/\ V, p)$.

$$e(V'/V, p) = (h^{\bar{\mu}}_{id}, \mathrm{R}^{\bar{\mu}}_{\mathrm{id}}) \cdot ((r^{xu\bar{\mu}}{}_i P, B, M)/B^{ri}_{\xi}, C)$$

Further we get$V'/\ V = C^{s\bar{\mu}\lambda}.(C')^{s\bar{\mu}\lambda'}.\ h_{id}{}^{xu\bar{\mu}}.\ \sum\nolimits_{\xi=1}^{C_s}((B'{}_{i\xi})^{r'}/\ B^{r}{}_{i\xi})^{vi_\xi}$. Finally, the solution of CPH problem is

$$\left(C'\right)^s = (C'.(C^{-1}.\ \left(C^{s\bar{\mu}\lambda}\right)^{-1}.\ \left(r_i{}^{xu\bar{\mu}}\right)^{-1} \times\ \sum\nolimits_{\xi=1}^{C_s}((B'{}_{i\xi})^{r'}.\ \ B^{r}{}_{i\xi})^{vi_\xi})^{(\bar{\mu}\lambda')^{-1}} \tag{16}$$

This conclude the proofs of Theorems 1 and 2, the proposed scheme can be safe against the two types of Adversaries under the EC-ACS hypothesis.

Lemma 2. *The proof of this work is based on the Authorized block chain system, an adversary cannot pre-determine the hash value of the block generate at a future moment.*

Proof: We first introduce the collision resistance of the ECC hash function. The hash function is a collision resistance, if the given function is uniform, it is infeasible for PPT adversary, which is used to find the valueα that is expressed as$H(\alpha) = y$. With the collision resistance of the hash function, the confirmation of this lemma is clear.

Assume the blocks under the authorized block chain (Bc_t) and hash function $H(\cdot)$is secure with collision resistance. If the adversary can pre-determine the (Bc_t) and collision resistance of $H(\cdot)$ is broken. This conclude this lemma 2.

Claim 2. The EC-ACS can foil malicious auditors who can perform two assaults to break the security:

1. The forging an entry the log file to pass the auditor.
2. Sampling challenging bias messages to create bias verification results.

Proof: This proof analysis contain two types of cases. In the first case, we described it in two features.

In the first feature, the TPA create the forges an entry $\{(Bc_{t-\phi+1},\ Bc_{t-\phi+2},\ \ldots..\ Bc_t),\ V',\ R',\ \mu'\}$ that sends the users auditing. The authenticity of proposed method is computationally infeasible to create the entry. In particular, since we have demonstrated that if the information loss occur, the cloud server can pass the auditor to check the data's with the negligible probability. On the off chance that TPA can produce the entry that passes the users auditing, this makes the cloud server to break the authenticity of EC-ACS by following the TPA's strategy. In the second feature, the TPA forge an entry to convince the user's for creating the entry at the block time. In this attack, the TPA need to produce a data exchange and record it to a block that has been chained in the Authorized blockchain. However, due to Authorized blockchain security, this attack is not computationally computable.

In the second case, we analyse it in two aspects. First part, the hash values of blocks are used to compute the challenging message, but it cannot predetermined and specified by any entitles. Secondly, blocks compute the challenging message would not be generated by the adversary, because the chain quality of block chain is strong. Finally, we successfully compute the probability of adversary Bc_t. This settings illustrates, that keys of any corrupted data block from a set ε. A wins whenever anyone of keys of challenged blocks generated by Bc_t is not fall in the sets ε.

6. Performance evaluation

In the following evaluation, we utilize the ECC to simulate the encryption operations in our proposed scheme, and all the experiments are tested on Intel core i5 processor, RAM 8GB, Windows 10 with Platform 64. The algorithms are implemented with the help of MATLAB R-2018 b. In this section, we provide the efficiency of signatures generation and auditing under different conditions. The signature verifications and auditing process for a different numbers of blocks are taken into consideration. The comparison of proposed EC-ACS scheme and existing Secure Certificateless Public Verification (SCLPV) [19], Certificateless Public Verification (CLPV) [25], Strategic Workforce Planning (SWP) [26], Identity Based Proxy Oriented Data Uploading and Remote Data Integrity Checking in Public Cloud (ID-PUIC), Identity Based Data Outsourcing (IBDO) [18], K-CLAS and Improved Certificateless Aggregate Signature Scheme (iCLAS) scheme [27] are taken for (i) Verification time under different numbers of blocks, (ii) Verification delay at auditing side, (iii) Auditing time taken for each blocks (iv) Computational time for both sign and verify / aggregation and aggregation verify.

Fig. 3 shows the analysis of TPA verification time for different numbers of blocks. The analysis of the proposed method is compared with the CLPV and SCLPV method [24]. Here, our proposed method has a slightly larger verification overhead in the TPA side. However, this additional verification overhead ensures that the security weakness of the CLPV and SCLPV are clearly shown in figure. In this proposed work, the TPA need not to download the entire outsourced data, generate signatures for each of the data blocks, and convince the user that it has computed the parameters correctly. Therefore, in that sense, the proposed method is more practical than the existing method. The proposed work can avoid the additional verification cost and the certificate management problem.

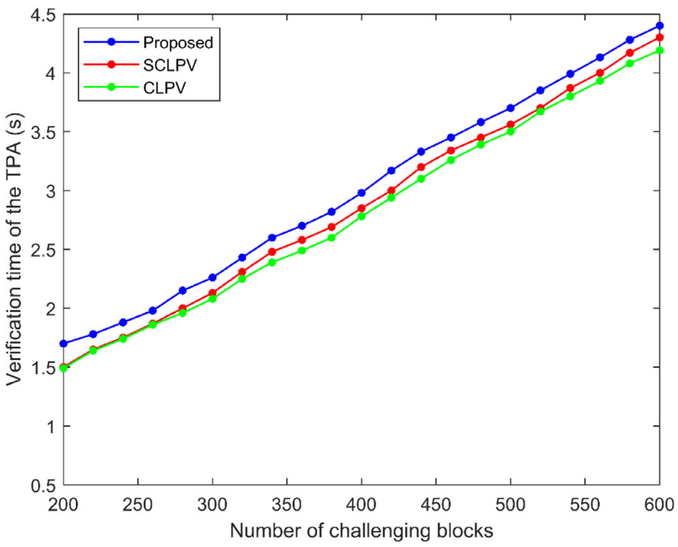


Fig. 3. Verification time of TPA under different blocks.

Table 3
Comparison of TPA Performance.

Algorithm	Verification overhead
SWP	$3Pair_{G_T} + C \cdot Hash_G + (C+S)\,Mul_G + (C+s)\,Add_G$
SCLPV	$4Pair_{G_T} + (2C+5) \cdot Hash_G + (2C+4)\,Mul_G + (2C+2)\,Add_G$
Proposed	$1Pair_{G_T} + C \cdot Hash_G + (C+1)\,Mul_G + C\,Add_G$

Table 4
Notation of the operations.

Notation	Operation
T_{SM}	The running time of a scalar multiplication
T_{PA}	The running time of a point addition
T_{Bp}	The running time of a bilinear pairing
T_{MP}	The running time of a Map - point function operation
T_{ha}	The running time of a hash function
T_{ME}	The running time of a scalar multiplication of ECC
T_{AM}	The running time of a point addition of ECC

The Table 3 illustrate the comparison of TPA performance between the proposed and the SWP and SCLPV existing method. From the analysis, we verify the overhead of TPA performance cloud stored data.

Fig. 4 shows the comparison of verification delay on the auditor side with proposed method and two existing methods such as the SWP [26] and SCLPV [19]. According to the comparison, we can see the comparatively high performance of the TPA in our proposed work. Note that, the SWP and SCLPV private verification scheme, the verification costs are high. This means that even when compared with the private verification scheme of SWP and SCLPV, the verification overhead of the proposed method is lower.

The calculation of the auditing time for various challenging blocks is shown in Fig. 5. Performance analysis is compared with two existing ID-PUIC and IBDO methods [18]. Moreover, when the number of healthcare data blocks are increased to 600, the auditing time of ID-PUIC is more than 20 times than the proposed method, and also the auditing time of IBDO is higher than 17 of our scheme. With the growing of challenging health data blocks, the case will be much higher. On

Table 5
The computation cost comparison of the proposed and existing schemes.

Algorithm	Sign	Verify	Aggregate	Aggregate verify
K-CLAS	$3T_{SM} + 2T_{PA} + 1T_{MP}$ $+1T_{ha}$	$3T_{Bp} + 2T_{SM} + 1T_{PA}$ $+1T_{MP} + 1T_{ha}$	$(n-1)T_{Bp}$	$3T_{Bp} + nT_{SM} + 2(n-1)T_{PA}$ $+nT_{MP} + nT_{ha}$
(iCLAS)	$1T_{ME} + 1T_{ha}$	$4T_{ME} + 3T_{AM} + 1T_{ha}$	$(n-1)T_{Bp}$ le	$(3n+1)T_{ME} + 3nT_{AM}$ $+(3n+1)T_{ha}$
Proposed	$1T_{ME} + 3T_{ha} + 1T_{SM}$	$3T_{ME} + 2T_{AM} + 3T_{ha}$	Can be negligible	$(2n+1)T_{ME} + 2nT_{AM}$ $+(3n+1)T_{ha}$

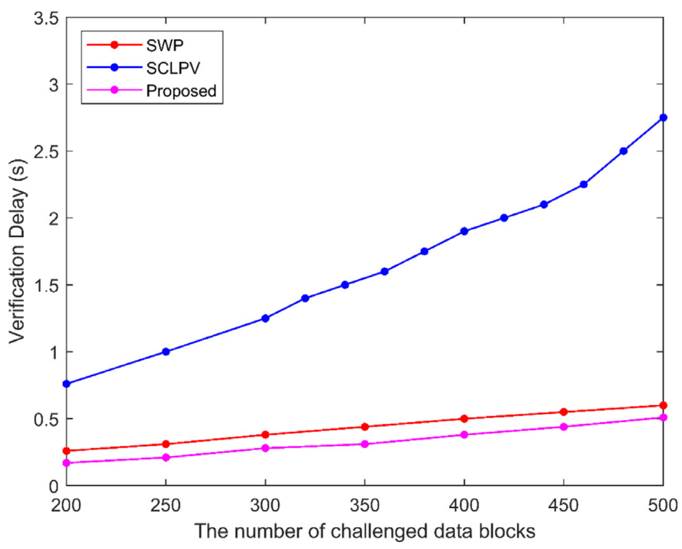


Fig. 4. Verification delay under auditing side.

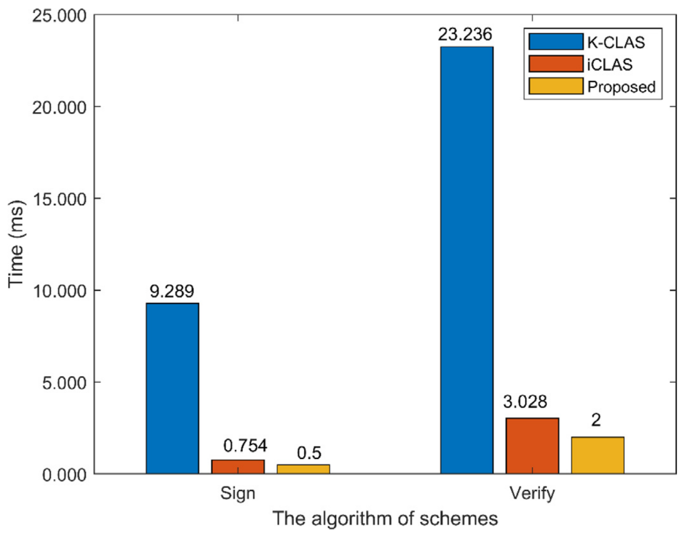


Fig. 6. Computation cost taken for sign and verify.

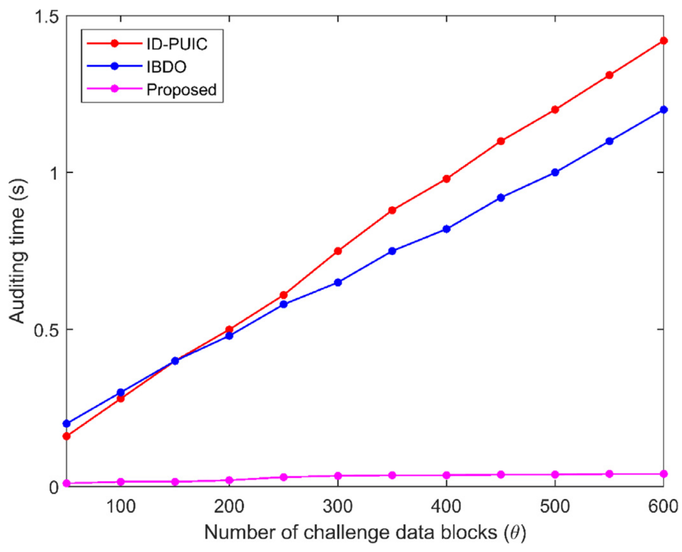


Fig. 5. Auditing time taken for various blocks.

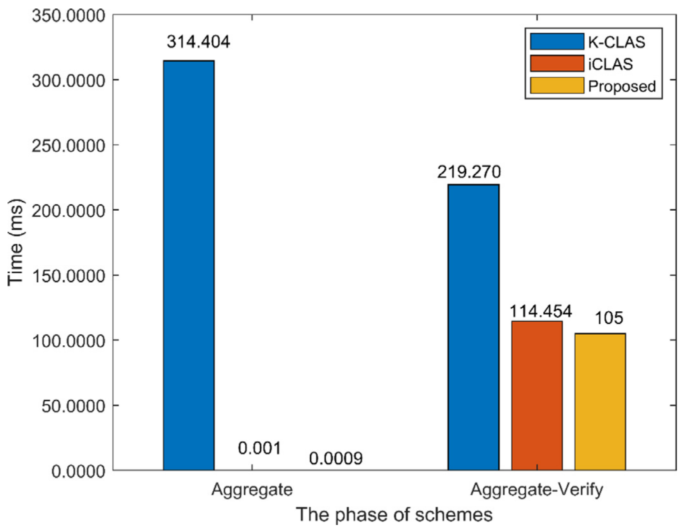


Fig. 7. Computation cost on Aggregate and Aggregate-verify.

the other hand, the growth of the proposed auditing time is very low. This is because the analysis of our proposed work is based on ECC.

The computational cost for the prosed and the existing methods in the sign, verify, Aggregate and Aggregate verify algorithm are shown in Table 5 and its operations are given in Table 4.

Fig. 6 clearly shows the computational time under the sign and verify algorithm. The performance of the proposed work is compared with the existing K-CLAS and (iCLAS) [27] scheme. During this analysis, our proposed method greatly exploits the computational costs than two algorithms in 0.50 ms and 2.00 ms.

Comparisons of calculation costs under the Aggregate and Aggregate-Verify algorithm are shown in Fig. 7. This shows that our proposed method takes a great advantage over calculation costs compared to the K-CLAS and iCLAS [27] in 0, 0009 ms in Aggregate and 105.0 ms in Aggregate - Verify.

7. Conclusion

Healthcare medical records include personal details of patients that should not be shared with other third parties. Thus medical cloud is introduced to keep safe the medical records, but it might be misused by some other parties of false doctors by deleting the original data and make modifications. Therefore, in this paper, we proposed an EC-ACS public verification and auditing scheme in the MCS using authorized blockchain technology. First, the secure certificateless public verification scheme is designed using ECC for the key generation to encrypt and the signature on the data to verify the aggregate signature. Then, we design the secure certificateless public auditing scheme to check the data integrity of data outsourced by CSP using auditor is integrated into a transaction on the blockchain. We simulate our proposed method in the MATLAB platform. In this, we consider metrics related to computation cost, verification, aggregate and auditing delay. The evaluation of performance comparison shows that our scheme is more efficient at the side of the TPA, which is more light-weight and suitable in cloud-based EHR.

Funding

There is no funding for this study.

Ethical approval

This article does not contain any studies with human participants and/or animals performed by any of the authors.

Informed consent

There is no informed consent for this study.

Authors' contribution

All the authors have participated in writing the manuscript and have revised the final version. All authors read and approved the final manuscript.

Declaration of Competing Interest

Authors declares that they have no conflict of interest.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.comnet.2020.107344.

References

[1] I. Korhonen, J. Parkka, M. Van Gils, Health monitoring in the home of the future, IEEE Eng. Med. Biol. Mag. 22 (3) (2003) 66–73..
[2] M. Ketteler, M.L. Gross, E. Ritz, Calcification and cardiovascular problems in renal failure, Kidney Int. 67 (2005) S120–S127.
[3] J.F. Thayer, S.S. Yamamoto, J.F. Brosschot, The relationship of autonomic imbalance, heart rate variability and cardiovascular disease risk factors, Int. J. Cardiol. 141 (2) (2010) 122–131.
[4] K.D. Mandl, J.C. Mandel, I.S. Kohane, Driving innovation in health systems through an apps-based information economy, Cell Syst. 1 (1) (2015) 8–13.
[5] A. Milenković, C. Otto, E. Jovanov, Wireless sensor networks for personal health monitoring: issues and an implementation, Comput. Commun. 29 (13–14) (2006) 2521–2533..
[6] D. Lupton, The digitally engaged patient: Self-monitoring and self-care in the digital health era, Soc. Theory Health 11 (3) (2013) 256–270.
[7] K. Häyrinen, K. Saranto, P. Nykänen, Definition, structure, content, use and impacts of electronic health records: a review of the research literature, Int. J. Med. Inf. 77 (5) (2008) 291–304.
[8] N. Sultan, Making use of cloud computing for healthcare provision: opportunities and challenges, Int. J. Inf. Manage. 34 (2) (2014) 177–184.
[9] C.O. Rolim, F.L. Koch, C.B. Westphall, J. Werner, A. Fracalossi, G.S. Salvador, A cloud computing solution for patient's data collection in health care institutions, in: 2010 Second International Conference on eHealth, Telemedicine, and Social Medicine, IEEE, 2010, p. 9599.
[10] B. Yüksel, A. Küpçü, Ö Özkasap, Research issues for privacy and security of electronic health services, Future Gen. Comput. Syst. 68 (2017) 1–13.
[11] K. Yang, X. Jia, An efficient and secure dynamic auditing protocol for data storage in cloud computing, IEEE Trans. Parallel Distrib. Syst. 24 (9) (2012) 1717–1726.
[12] P. Mytis-Gkometh, G. Drosatos, P.S. Efraimidis, E. Kaldoudi, Notarization of knowledge retrieval from biomedical repositories using blockchain technology, in: Precision Medicine Powered by pHealth and Connected Health, 2018, pp. 69–73.
[13] Nakamoto, S.Bitcoin: a peer-to-peer electronic cash system (2008).
[14] K Gai, Y Wu, L Zhu, L Xu, Y Zhang, Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks, IEEE Internet Things J. 6 (5) (2019) 7992–8004.
[15] R. Guo, H. Shi, Q. Zhao, D. Zheng, Secure attribute-based signature scheme with multiple authorities for blockchain in electronic health records systems, IEEE Access 6 (2018) 1167611686.
[16] K Gai, Y Wu, L Zhu, Z Zhang, M Qiu, Differential Privacy-based blockchain for industrial Internet of Things, IEEE Trans. Ind. Inf. (2019).
[17] S. Cao, G. Zhang, P. Liu, X. Zhang, F. Neri, Cloud-assisted secure eHealth systems for tamper-proofing EHR via blockchain, Inf. Sci. 485 (2019) 427–440.
[18] X. Zhang, J. Zhao, L. Mu, Y. Tang, C. Xu, Identity-based proxy-oriented outsourcing with public auditing in cloud-based medical cyber–physical systems, Pervasive Mob. Comput. 56 (2019) 18–28.
[19] Y. Zhang, C. Xu, X. Lin, X.S. Shen, Blockchain-based public integrity verification for cloud storage against procrastinating auditors, IEEE Trans. Cloud Comput. (2019).
[20] K Gai, Y Wu, L Zhu, M Qiu, M Shen, Privacy-preserving energy trading using consortium blockchain in smart grid, IEEE Trans. Ind. Inf. 15 (6) (2019) 3548–3558.
[21] A. Zhang, X. Lin, Towards secure and privacy-preserving data sharing in e-health systems via consortium blockchain, J. Med. Syst. 42 (8) (2018) 140.
[22] W. Shen, J. Yu, H. Xia, H. Zhang, X. Lu, R. Hao, Light-weight and privacypreserving secure cloud auditing scheme for group users via the third party medium, J. Netw. Comput. Appl. 82 (2017) 56–64.
[23] S.G. Worku, C. Xu, J. Zhao, X. He, Secure and efficient privacy-preserving public auditing scheme for cloud storage, Comput. Electr. Eng. 40 (5) (2014) 1703–1713.
[24] Y Zhang, R Deng, X Liu, D Zheng, Outsourcing service fair payment based on blockchain and its applications in cloud computing, IEEE Trans. Serv. Comput. (2018).
[25] Y. Zhang, C. Xu, S. Yu, H. Li, X.SCLPV Zhang, Secure certificateless public verification for cloud-based cyber-physical-social systems against malicious auditors, IEEE Trans. Comput. Soc. Syst. 2 (4) (2015) 159–170.
[26] Y. Zhang, C. Xu, X. Liang, H. Li, Y. Mu, X. Zhang, Efficient public verification of data integrity for cloud storage systems from indistinguishability obfuscation, IEEE Trans. Inf. Forensics Secur. 12 (3) (2016) 676–688.
[27] Y. Xie, X. Li, S. Zhang, Y Li, $ iCLAS $: an improved certificateless aggregate signature scheme for healthcare wireless sensor networks, IEEE Access 7 (2019) 15170–15182.

Benil.T was born in Nagercoil, India, in 1987. He received a B.E. Degree in Computer Science and Engineering from Anna University, and the M.E degrees in Computer Science and Engineering from Anna University Chennai. In 2012, he joined the Department of Computer Science, at Ponjesly College of Engineering, as an Assistant Professor. Currently, he is doing Ph.D. Research in Computer Science and Engineering at Ponjesly College of Engineering, Tamil Nadu, India. His current research interests includes Cloud Computing, Cloud Security, and Network Security.

Jasper.J was born on February 28, 1981. He received his B.E degree in Electrical Engineering from *Manonmaniam* Sundaranar *University, India in 2003 and M.E in Electrical Engineering from Annamalai University, Tamil Nadu India in 2005. He got his Ph.D. degrees from* Anna University, India in 2014. His major research interest includes Artificial Intelligence, Distributed Generation, Cloud Computing and computational intelligent techniques.