Hindawi
Security and Communication Networks
Volume 2021, Article ID 9363824, 17 pages
https://doi.org/10.1155/2021/9363824

Research Article

A Lightweight Proxy Re-Encryption Approach with Certificate-Based and Incremental Cryptography for Fog-Enabled E-Healthcare

Junaid Hassan,[1] Danish Shehzad,[1] Insaf Ullah,[2] Fahad Algarni,[3] Muhammad Umar Aftab,[1] Muhammad Asghar Khan,[2] and M. Irfan Uddin[4]

[1]*Department of Computer Science, National University of Computer and Emerging Sciences, Islamabad Chiniot-Faisalabad Campus, Chiniot 35400, Pakistan*
[2]*Hamdard Institute of Engineering & Technology, Hamdard University, Islamabad 44000, Pakistan*
[3]*College of Computing and Information Technology, The University of Bisha, Bisha, Saudi Arabia*
[4]*Institute of Computing, Kohat University of Science and Technology, Kohat 26000, Pakistan*

Correspondence should be addressed to Insaf Ullah; insafktk@gmail.com and Muhammad Umar Aftab; ms.umaraftab@yahoo.com

Received 27 July 2021; Revised 1 October 2021; Accepted 9 October 2021; Published 19 November 2021

Academic Editor: Chinmay Chakraborty

Cloud computing aims to provide reliable, customized, and quality of service (QoS) guaranteed dynamic computing environments for end-users. However, there are applications such as e-health and emergency response monitoring that require quick response and low latency. Delays caused by transferring data over the cloud can seriously affect the performance and reliability of real-time applications. Before outsourcing e-health care data to the cloud, the user needs to perform encryption on these sensitive data to ensure its confidentiality. Conventionally, any modification to the user data requires encrypting the entire data and calculating the hash of the data from scratch. This data modification mechanism increases communication and computation costs over the cloud. The distributed environment of fog computing is used to overcome the limitations of cloud computing. This paper proposed a certificate-based incremental proxy re-encryption scheme (CB-PReS) for e-health data sharing in fog computing. The proposed scheme improves the file modification operations, i.e., updation, deletion, and insertion. The proposed scheme is tested on the iFogSim simulator. The iFogSim simulator facilitates the development of models for fog and IoT environments, and it also measures the impact of resource management techniques regarding network congestion and latency. Experiments depict that the proposed scheme is better than the existing schemes based on expensive bilinear pairing and elliptic curve techniques. The proposed scheme shows significant improvement in key generation and file modification time.

1. Introduction

Health monitoring has become a leading issue in modern paradigms all over the world. According to the United Nations report [1], by 2050, the world's 22% population will consist of vulnerable people who can be victims of various diseases. Nowadays, many devices are being developed to monitor vital signs such as human blood pressure, glucose level, heartbeat, and oxygen level. Readings are collected from the medical devices and then shared with the authorized health teams. These medical devices generate a large amount of data, and managing those data using traditional hardware or software has become a huge challenge [2, 3]. Since the advent of COVID-19, medical big data (MBD) has become important for the effective health monitoring systems. MBD by now has become irresistible because of its diversity and volume [4–7]. People suffer from many diseases in the world but Parkinson's disease (PD) is a serious neurodegenerative disorder. Some recent studies have suggested different techniques for diagnosing Parkinson's disease [8, 9]. These studies have improved the automated Parkinson's disease detection using neural network techniques.

Over time, in healthcare, electronic health records (EHR) have replaced the paper record system, so that data

can be handled and maintained efficiently. It further improves the availability, sharing, and cost of data maintenance. A hybrid machine learning framework is proposed to predict mortality in paralytic ileus patients using (EHRs) [10]. Previously, if a person wanted to be treated by another doctor, he had to bring his report to the doctor in paper format. The EHR system has saved people from this hassle because it is abysmal to save paper reports for a long time. Now, the patient record is stored on a centralized cloud database from which all the doctors can check the patient's medical reports and history. Therefore, the patient no longer needs to bring his reports to the doctor in paper form. All the users who register to a healthcare application can share their medical reports with all the healthcare authorities, i.e., doctors, physicians, pharmacists, and laboratory technicians.

The widespread adoption of IoT devices in business and healthcare has raised serious concerns about evaluating IoT architectures regarding data communication, storage, security, and processing. In the healthcare field, a large number of IoT devices and sensors are currently being used producing a vast amount of data [11]. Therefore, we required an infrastructure that can process, store, and analyze this large amount of data.

Across the world, cloud infrastructure is used to process a large variety of data. Currently, cloud infrastructure is the only feasible option for managing and communicating between IoT devices in the healthcare field [12, 13]. All computational heavy and battery drain works are being offloaded from IoT devices to cloud infrastructure to reduce the load on IoT devices, which increases the battery life of IoT devices [14].

Healthcare IoT devices require minimum communication latency to gain expected performance. On the other hand, healthcare devices produced a massive amount of data. This large amount of data leads to high data traffic that causes network congestion and further delays. Healthcare-sensitive data becomes inadequate and meaningless for end-users due to large hop counts and large data transmission between IoT devices and cloud servers; as a result, we face two-way delays. Real-time data are required for time-sensitive healthcare applications. End users and healthcare IoT devices expect minimal latency delays, but cloud servers cannot meet these expectations. All kinds of delays, such as network delays, connection delays, and computational delays, need to be minimized during data transformation among healthcare IoT devices. The fog computing paradigm has emerged to address the challenges mentioned earlier. Fog computing has a fog node layer between IoT devices and cloud servers, as shown in Figure 1. Distributed fog devices can also contribute to cloud scalability by reducing centralized communication and processing. Compared to the cloud, fog nodes in the edge network are much closer to end devices and have lower latency [15]. The 20% percent user response time is reduced in fog computing, and 90% of data traffic between cloud and end devices is reduced. The fog computing can minimize response latency for real-time applications by up to 50% [16]. While fog computing can serve IoT devices and applications more efficiently, data protection remains a critical issue in fog computing [17].

Risk problems become even more critical as IoT devices share sensitive data because they are directly connected to the Internet [18]. Many research articles on fog computing and the Internet of things have focused on infrastructure and application development issues without providing adequate support for privacy and security protection mechanisms [19]. However, more complex and intricate security mechanisms cannot be performed on resource-constrained IoT devices, which shorten the device's battery life and lead to high computational costs and significant delays in real-time applications that require an immediate response [20]. We avoid this by offloading these complex security tasks to fog rather than resource-limited IoT devices. When the user's private data are stored on a fog node, they have no physical control over their data and face numerous privacy and security attacks. We cannot consider fog nodes as reliable storage. Fog computing can perform some security operations while transmitting and retrieving data to address IoT security and privacy concerns. We can achieve data privacy and confidentiality by encrypting our data. But encryption is a very complex and heavy function that requires more computation time and power. Therefore, we offload these cryptographic functions on fog nodes to achieve efficiency. However, fog nodes must also provide convenient, manageable, and easy communication and accurate access control.

Some incremental cryptographic schemes for the limited resource devices are proposed in [21]. Because there is a lot of literature out there that is useful for data sharing. Still, our best research shows that certificate-based incremental proxy re-encryption is the most important and secure scheme. This paper proposes a certificate-based incremental proxy re-encryption scheme (CB-PReS) for e-healthcare data sharing in fog computing. The key escrow problem is also eliminated in certificate-based proxy re-encryption. In contrast, a key escrow problem is present in the identity-based proxy re-encryption scheme [22]. The efficiency and security hardness of certificate-based proxy re-encryption and identity-based proxy re-encryption scheme (IB-PRE) is based on the standard cryptosystems such as bilinear pairing (BP), elliptic curve (EC), and Rivest, Shamir, and Adleman (RSA). The 1024-bit key is used in RSA, while 160-bit key is used in ECC. The experimental results show that the bilinear pairing is 13.65 ms worse than that of Rivest, Shamir, and Adleman (RSA) as well as 13.93 ms worse than E.C. [23], and Rivest, Shamir, and Adleman (RSA) is 14.42 ms worse than the hyperelliptic curve [24]. The proposed scheme uses a hyperelliptic curve (HEC), and the 80-bit key is used in HEC to provide the same level of security and low communication and computational cost.

1.1. Motivation. There are some serious concerns about the security and privacy of IoT devices. Given these concerns, security measures need to be taken immediately. IoT devices are very resource-constrained devices, and the implementation of resource-intensive and complex security tasks on these devices is not feasible. This usually shortens the device's battery life and leads to higher computational costs

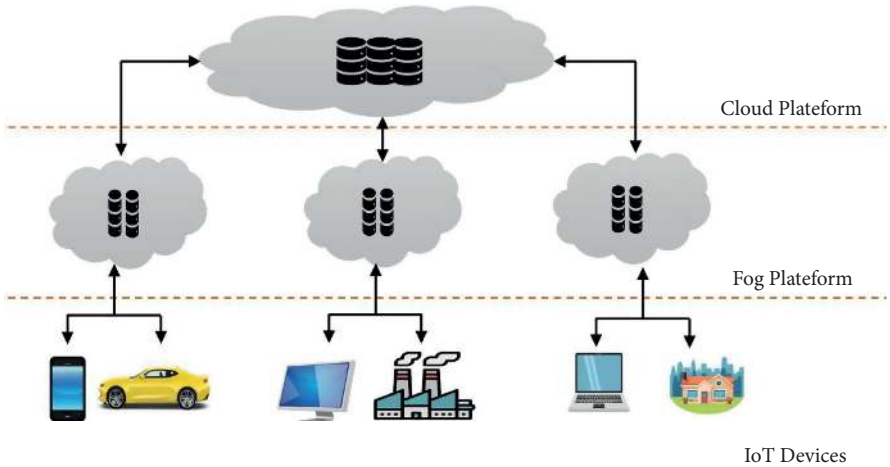

Figure 1: Cloud, fog, and end-device architecture.

and significant delays in real-time applications that require immediate response [20]. Therefore, we can avoid these problems by offloading these complex security tasks to the fog instead of processing them on resource-constrained IoT devices. The communication and computational cost of the traditional crypto system is very high because of the use of traditional cryptographic functions such as RSA that uses a 1024-bit key size. The bilinear pairing scheme is 13.65 times worse than RSA, and RSA is 14.42 times worse than the hyperelliptic curve [23]. When we need to share encrypted data with multiple participants, we can achieve significant performance by using the proxy re-encryption scheme. In some cases, when we update the EHR data, we need to calculate the hash value from scratch to reflect the updation. If we want a minor update to a large amount of EHR data, then the entire hash must be recalculated from scratch, which is not a good thing in practice. Therefore, there is a need to implement incremental cryptography that reduces the overhead of hash value recalculation from scratch.

1.2. Contribution. The contribution of our proposed scheme is listed in the following steps:

(i) We have proposed a certificate-based incremental proxy re-encryption scheme for E-healthcare data sharing on fog computing, which deals with the problem of overhead and delay of previously proposed PRE schemes

(ii) Our scheme reduces the commutation cost and communication overhead of the resource-constraint IoT devices

(iii) Our scheme is based on the hyperelliptic curve, which uses the 80-bit key instead of elliptical curves, and bilinear pairing, which uses a 160-bit key and 1024-bit key, respectively

(iv) Our scheme provides block base data modification by using the concept of incremental cryptography

(v) Our scheme also fixes the key escrow problem by using the certificate-based proxy re-encryption

(vi) We have also provided a security model and security analysis of our scheme

(vii) Our scheme provides some security services such as integrity, confidentiality, unforgeability, and anti-replay attack, respectively

2. Related Work

In this part, we will review some of the publications that are related to our proposed work. In [25], the integrity-enforcement scheme is proposed that takes advantage of trusted computing and incremental cryptographic primitive and ensures the integrity of electronic health records for mobile device users stored in the cloud computing database. However, this scheme ignores data confidentiality. The proxy re-encryption (PRE) scheme is proposed by Blaze et al. in 1998 [26], various PRE schemes have been introduced in the literature so far. Proxy re-encryption scheme may be divided into two different classes, namely, bidirectional and unidirectional. In the unidirectional PRE schemes, there are further different types of PRE schemes, i.e., (1) identity-based PRE, (2) conditional PRE, (3) attribute-based PRE, and (4) time-based PRE. In the bidirectional scheme, there are also two types of PRE schemes, i.e., (1) threshold-based PRE and (2) type-based PRE) [27]. The re-encryption scheme can provide different kinds of features, such as key optimality, collusion resistance, proxy invisibility, non-transferability, noninteractivity, and unidirectionality, and this has been debated extensively in the literature [28]. Based on a set of these features, advanced proxy re-encryption schemes with advanced features were developed. Based on the presence of these features, the proxy re-encryption schemes are evaluated.

Fine-grain access control is provided on the user's private data in the attribution-based PRE (AB-PRE) [29, 30]. In AB-PRE, user A's ciphertext is transformed under some certain set of attributes into user B's ciphertext under some other set of attributes. Conditional PRE (C-PRE) is proposed in [31]. In the C-PRE scheme, the ciphertext is only transformed into another ciphertext by the proxy when it

fulfils certain conditions. In the paper [32], the author defines another type of conditional PRE in which a ciphertext is transformed into another ciphertext by a proxy that has been received from a specific sender. Identity-based encryption (IB-PRE) is defined by the author [33] in which the proxy under Alice identity transforms the ciphertext into another ciphertext under Bob's identity. Later another type of identity-based encryption is defined by the author [34] without random oracles. An extended type of IB-PRE is defined in [35], which facilitates the conditional re-encryption features. This conditional re-encryption feature protects the user's data from conditional attacks and also from identity chosen-ciphertext attacks. First time collusion-resistant unidirectional IB-PRE scheme is proposed in [36] which is secure from quantum attacks. The author in [37] proposed a type-based proxy re-encryption scheme in which ciphertexts of every delegator are associated with a specific type, and proxy transformed the ciphertext only when the public key of a delegate has the same type and through which the owner of the data gains proper delegation control. Another version of proxy re-encryption with certificate-based encryption is suggested in [38], which provides resistance to chosen-ciphertext attacks. Keyword search-based PRE scheme is proposed in [39], in which a proxy transforms the ciphertext if the ciphertext contains some keywords that match with specific information about the re-encryption key. In the paper [40], the author proposed a group-based PRE scheme in which a ciphertext is transformed by a proxy to be broadcast to a group. Therefore, all users belonging to the group can decrypt this ciphertext. Another scheme, proposed in [41], relies on conditional broadcast PRE. In this scheme, users are dynamically added to a group at runtime, and there is no need to change the public key for encryption every time.

However, many existing schemes for secure data sharing and communication in cloud computing have limitations, i.e., delays between user requests and responses from the cloud due to a drastic increase in data volume. A large number of users are involved in outsourcing and cryptographic operations [42, 43]. The purpose of fog computing is to solve cloud computing-related problems by offloading expensive computational tasks to the network's edge. Fog computing is also used to perform costly computational tasks that have reduced the computational overhead required on resource-constrained devices. However, some of the studies only focus on the privacy and security of fog computing [44]. Security issues and threats to fog computing are discussed in [16]. The PRE schemes proposed in the fog computing literature are the same ones that have been addressed in the cloud computing literature. Ciphertext-policy attribute-base-encryption (CP-ABE) is proposed in the literature [42, 45, 46] that provides secure access control to data encrypted in fog computing, which allows the data owner to define access policy to one of the features that the user needs to decrypt the ciphertext. Privacy-preserving proxy re-encryption scheme for access control is proposed by [47] which secures against chosen-plaintext attack (IND-CP-CPA). An improved version of proxy re-encryption scheme is proposed [48] in which at every hop ciphertext is transformed into a different ciphertext. We can secure the sensitive data with the usage of secure technologies mentioned in [49] such as access control, steganography, watermarking, and cryptography.

The main disadvantage of using ID-based encryption, attribute-based encryption, and CP-ABE schemes in fog computing is the expensive computational cost of the decryption, which includes the complexity of the policy as well as various pairing operations [42, 46]. The first improved scheme for Internet of Vehicles is proposed [50] in which all the vehicles communicate with each other securely. Dynamic and simultaneous data communication between IoT devices occurs in fog computing. Due to the limited resources of fog computing and IoT networks, public-key encryption and traditional key management practices to secure communications between connected devices are incompatible with fog computing. Traditional encryption mechanisms require extensive computing resources for most objects and do not meet the operational requirements of the Internet of Things in real time [43].

In [44], the authors proposed a scheme that is based on the random oracle model and uses bilinear paring cryptography. In 2013, the authors [51] proposed a random oracle-based CCA-secure proxy re-encryption scheme. In 2018, Bhatia et al. [45] proposed a certificateless proxy re-encryption scheme which is better that the scheme in [44] in computational cost because it uses elliptic curve cryptographic techniques. Another CL-PRE scheme is proposed by Xu and Chang [46] in 2012. This scheme is based on the key management and encryption-based access control for the data distributed using cloud infrastructure. Another single hope certificateless proxy re-encryption and CCA-secure unidirectional scheme is proposed in [52]. The first pairing-free certificateless proxy re-encryption scheme is proposed in 2014 by Lee and Han [53]. In 2015, another CL-PRE scheme for data distribution in cloud is proposed by Qin et al. [54]. This scheme does not provide any kind of security analysis.

Distributed secure accessibility in mobile cloud computing is proposed in [55]. Some incremental cryptographic schemes, such as sharing-based scheme (ShS), coding-based scheme (CoS), and encryption-based scheme (EnS) [56], are proposed in [21] for the limited resource devices. To ensure data security, the mobile user enters a password that is converted into a key, and then, we encrypt the data through this key. More mobile resources are used in the process of encrypting and uploading complex and heavy tasks. The incremental proxy re-encryption scheme (I-PReS) is proposed in [56]. I-PReS is a block-based data sharing scheme [57]. This I-PReS improve the file modification operations that provide data integrity and confidentiality by using advanced cryptographic functions.

The industry did not accept the incremental hash functions due to the following flaws. Firstly, a certain level of security is achieved through the use of a large number of expensive pairing operations by using known incremental hash functions [58, 59] (for example, 2^{128} or 2^{256}), which can degrade their performance with respect to hash

function. Secondly, hash value size is disproportionate to the incremental hash functions security level, which is calculated in several thousand bits, unlike SHA_1 producing 160-bit message digest output [60], SHA_2 producing 512-bit message digest output [61], and SHA_3 producing 512-bit message digest output [62]. Incremental hashing is proposed in [63], which requires all hash values of intermediate nodes to be stored. Recently, the authors of [64] proposed a CL-IPRE scheme for the healthcare system based on the elliptic curve improved version of [56]. In the CL-IPRE scheme, the author compares the results of the block modification with the previous scheme and uses an elliptical curve algorithm instead of bilinear pairing for key generation.

3. Materials and Methods

3.1. Preliminaries. Hyperelliptic curve (HEC) is a family of public-key cryptosystems, and its structure is based on algebraic curves. That is why, it is also called a special class of algebraic curves and can also be seen as a generalized form of elliptic curve cryptography (ECC) [65]. HEC differs from ECC in terms of obtaining curve points from the group as shown in Figure 2. The additive abelian group is obtained from the devisor and computed by the HEC. We use a divisor class group of the curve, and there is no group law on the points of the HEC. HEC can implement all major operations of the public-key cryptosystem, such as signature, encryption, decryption, and key exchange. It is also called a successor of the RSA cryptosystem because HEC has the same security level as RSA and uses a smaller signature and key. It also provides a fast signature and fast key generation.

The hyperelliptic curve is defined over a finite field Fq (where q is a prime and $q > 3$). We can also denote the field Fq as $F2m$ (where the $q = 2m$); this means that field size is $q \times q$, and it is a square matrix, and curve points are limited to integer coordinates. When we perform any algebraic operation such as addition or multiplication, then as a result, we get another point on the curve within the field. The genus of the curve over the Fq field is denoted by "g," and the curve of genus one is called elliptic curve with the field Fq having the following values $|Fq|g.\ \log 2\ q \approx 2^{160}$ and the genus two curve is called the hyperelliptic curve with the field Fq having the following values $|Fq|g.\ \log 2\ q \approx 2^{80}$.

HEC is a special type of projective and nonsingular curve. Hyperelliptic curve over the field Fq with the points $(U, V) \in Fq$ satisfies the following equation:

$$V^2 + h(U)V = f(U), \tag{1}$$

where f and h both are polynomial in the field Fq with deg $(f) = 2g + 1$ and deg $(h) \leq g$. It also satisfies both equation (1) and partial derivative equation $h\prime(U) = 0$ and $h\prime(U)V + f'(U) = 0$.

3.2. Complexity Assumptions. In this section, we have made some assumption as follows:

(i) We can denote the field Fq as F_{2m} (where $q = 2m$); this means that field size is q x q, and it is a square matrix.

(ii) D is denoted as a divisor of the curve, and it is a formal sum of the points $P \in$ HEC as shown in the following equation:

$$D = \sum p_{\epsilon HEC}\, n_i p\,, \tag{2}$$

where $n_{\mathrm{i}} \in Fq$.

Definition 1. We have $\partial, \varphi \epsilon \{1, 2, 3, 4, \ldots., q-1\}$, and ∂, φ picks a random value from the given range, and then we calculate the value of N using the following equation:

$$N = \partial. D, \tag{3}$$

where D is a divisor from the Jacobian group.

After that, we find the value of Λ from the following equation:

$$\Lambda = \varphi. D, \tag{4}$$

where D is a divisor from the Jacobian group.

The probability of computing the values of ∂ and φ from equations (3) and (4) is negligible due to the Hec–Deffie–Hellman problem (HEC-DHP).

Definition 2. We have $\acute{E} \epsilon \{1, 2, 3, 4, \ldots., q-1\}$, and $\acute{E}$ picks a random value from the given range, and then we calculate the value of N from the following equation:

$$N = \acute{E}. D, \tag{5}$$

where D is a divisor from the Jacobian group.

The probability of computing the values of $\acute{E}$ from equation (5) is negligible due to the Hec-discrete-logarithm problem (HEC-DLP).

3.3. Syntax of Certificate-Based Incremental Proxy Re-Encryption Scheme. Our proposed scheme is an extension of Khan et al. [56] and Bhatia et al. [64]. Our certificate-based incremental proxy re-encryption scheme consists of seven phases that are (system setup, public and private key generation, certificate generation, proxy re-encryption key generation, encryption, proxy re-encryption, and decryption). The basic symbols which are used in the construction of the proposed scheme are given in Table 1. All these phases are explained as follows.

3.3.1. Setup. In this phase, the certificate authority sets public parameters ψ, then randomly selects a master secret key δ, and computes a master public key MP_k.

3.3.2. Public and Private Key Generation. In this phase, each participant (sender, receiver, and certificate authority) first randomly selects three numbers such as α, β, and γ and calculates $X = \alpha + \beta$, $\xi = \alpha.\ (\gamma\text{-}\alpha)$ and $\eta = \gamma.\ (\beta + \gamma)$. After that,

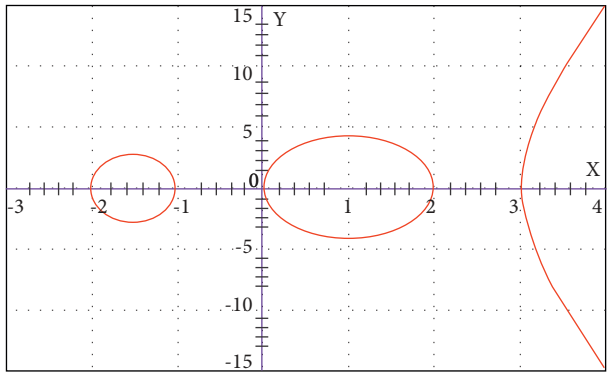

Figure 2: Hyperelliptic curve with genus = 2 [65].

each participant further computes public and private key pair (P_U, K_U).

3.3.3. Certificate Generation. In this phase, CA takes the participant identity ID_U, participant public key P_U, and the public parameters ψ as input and generates a certificate for each participant (sender and receiver) and sends to each participant.

3.3.4. Re-Encryption Key Generation. This algorithm is executed by the sender to generate re-encryption key such as $R_{A\longrightarrow B} = K^b/K_a$ and send to the resident fog proxy server without revealing any secret information about any user.

3.3.5. Encryption. In this phase, the user outsources a file F on the fog node. The user divides the file F into z blocks and encrypts each block using his public key P_U. The user chooses a random number $\chi \in \{1, 2, \ldots .. .., q\text{-}1\}$ to encrypt the file with his public key P_U. To achieve integrity, the user applies the SHA-3 algorithm on each block of the FILE such as $\text{MAC}_k = \text{HSHA-3}\ (B_k)$, where $1 \le k \le z$. Then again, the hash function is applied to the concatenated hash values to get a final single hash value that verifies the integrity of the file as $\text{MAC}_{\text{final}} = \text{HSHA-3} \parallel (\text{MAC}_k)$, where $1 \le k \le z$.

3.3.6. Proxy Re-Encryption. In this phase, the proxy server transformed the first level ciphertext of the sender ($C_A \in C_{\text{sender}}$) to the second level ciphertext of the receiver ($C_B \in C_{\text{receiver}}$). Proxy checks the access policy control list, if user B has access rights to the uploaded file, and then the proxy server re-encrypts that file and allows user B to download it.

3.3.7. Decryption. In this phase, the receiver downloads the final hash value ($\text{MAC}_{\text{final}}$), second-level ciphertext (C, C_B), and total numbers of blocks 'z' and decrypts each block by using his private key K_B. After decrypting all the blocks, the receiver concatenates all the blocks to get the original file and checks the integrity of the FILE by calculating the hash of the FILE and matching the calculated hash with $\text{MAC}_{\text{final}}$.

4. Construction of the Proposed Algorithm

In this paper, we have proposed a certificate-based incremental proxy re-encryption data sharing scheme for fog computing. Our scheme deals with the computation and communication problem of previous PRE schemes due to the use of expensive cryptographic functions and cloud computing. In our scheme, to reduce the overhead of resource constraints IoT devices, complex and resource-intensive cryptographic functions are offloaded on fog nodes. We show that our scheme has less communication cost as compared to previous PRE schemes that are necessary for real-time devices. Therefore, each fog node is considered as a proxy node, and all the resource-intensive tasks involved in the process of re-encryption are performed on fog nodes. The structure of our proposed scheme is shown in Figure 3.

Our proposed certificate-based incremental proxy re-encryption data sharing scheme for fog computing is in [64].

4.1. System Setup. Certificate authority (CA) runs this algorithm, and it chooses a security parameter E and hyperelliptic curve (HEC) over a finite field Fq of order q. CA selects an integer $\wp$ as a divisor that is the generator point of order q on the curve. Then, CA randomly selects a master secret key $\delta \in \{1, 2, \ldots .., q\text{-}1\}$ and further calculates the master public key as $\text{MP}_k = \delta.\wp$. Finally, CA announces some public parameters for encryption, decryption, and proxy re-encryption such as $\psi = \{\text{Cert}_U, Fq, \text{ID}_U, \wp, \text{MP}_k, h1, h2, h3\}$.

4.2. Public and Private Key Generation. In this phase, each participant (sender, receiver, and certificate authority) executes this algorithm, the participant first randomly selects three numbers such as $\alpha \in \{1, 2, \ldots .., q{-}1\}$, $\beta \in \{1, 2, ..., q{-}1\}$, and $\gamma \in \{1, 2, ..., q{-}1\}$, and then calculates $X = \alpha + \beta$, $\xi = \alpha.(\gamma{-}\alpha)$, and $\eta = \gamma.(\beta + \gamma)$. The participant with identity IDU computes the public and private key pair (P_U, K_U). Each participant computes its private key such as $K_U = \beta.(\xi{-}\eta) + \text{MP}_k$, after computing private key each participant further computes its public key such as $P_U = K_U.\ \wp$. The public key of each participant is publicly announced.

4.3. Certificate Generation. In this phase, CA takes the participant identity ID_U, participant public key P_U, and the public parameters ψ as an input (ID_U, P_U, ψ) and generates a certificate (Cert_U) for each participant (sender and receiver) and sends to each participant. CA generates the certificate through the following process: first, CA takes participant public key P_U and calculates hash of the P_U such as $H_{PU} = h1(P_U||\text{ID}_U) + \text{MP}_k$ after computing the hash of P_U, and CA digitally signs H_{PU} with its private key such as $S = \text{K}_{\text{CertA}}(H_{PU})$.

4.4. Re-Encryption Key Generation. If user A wants to share data with user B, then user A runs this algorithm to generate a re-encryption key such as $R_{A\longrightarrow B} = K^b/K_a$ using a two-party secure integer division algorithm [65] and sends it to the

Table 1: Notations of the proposed scheme.

Serial no.	Notation	Description
1	ID_U	Unique identity of user
2	E	Security parameter
3	CA	Certificate authority
4	Q	It is a large prime number
5	δ	Generator point
6	δ	Master secret key
7	MP_k	Master public key
8	Ψ	Set of public parameters
9	$Cert_U$	User certificate
10	α, β, γ	Random values
11	S	Digital signature
12	$H_{P\ U}$	Public key hash
13	K_U	Private key
14	P_U	Public key
15	$R_{A\longrightarrow B}$	Re-encryption key
16	HER	Electronic health record
17	Z	Total number of blocks of a file
18	B_k	k_{th} block of the FILE
19	D_j	Size of the j_{th} block
20	FS	Total size of the FILE
21	C_k	Ciphertext of k_{th} block
22	C	Ciphertext of entire file
23	C_A	A number χ PK_A
24	C_B	A number χ PK_B
25	MAC_k	MAC value of B_k block
26	MAC_{final}	Final mac value of all z blocks
27	B_{update}	Ciphertext of updated block
28	C_{update}	Block that user want to update
29	$h1$	Hash function
30	K_{CertA}	Certificate authority private key

resident fog proxy server without revealing any secret information about the K_a and K_b.

4.5. Encryption. In this phase, when a user wants to outsource a file F on the fog node, first, he divides the file F into z blocks, and each block has a constant size of d bits except the last block. To achieve confidentiality, the user encrypts each block using his public key P_U. The following condition should be satisfied while dividing the file F into z blocks as follows:

$$\text{FILE} = \|_{k=1}^{z}(B_k), \tag{6}$$

$$D_j = \left[\frac{FS}{Z}\right], \tag{7}$$

where $1 \le j \le z-1$ and

$$D_z = FS - \left(\left[\frac{FS}{Z}\right] * z - 1\right), \tag{8}$$

where B_k represents the k th block of the file, D_j represents the size of the jth block of the file, F_S denotes the total size of the file, $[FS/Z]$ represents the mathematical floor function that removes the fraction part, and D_z denotes the size of the last block of file. Before outsourcing file on the fog node, the user encrypts all the file blocks with his public key P_U to achieve confidentiality. The user chooses a random number $\chi \in \{1, 2, \ldots, q-1\}$ to encrypt the FILE with his public key P_U. Equations (9) and (10) depict the encryption process.

$$\begin{aligned} C_A &= \chi.P_A, \\ C_K &= B_K + \chi.\wp, \end{aligned} \tag{9}$$

$$C = \|_{k=1}^{z}(C_k), \tag{10}$$

where $1 \le k \le z$.

To achieve integrity, the user applies the recently proposed algorithm SHA-3 on each block of the file. Afterwards, hash values of all the blocks are concatenated together. Then again, the hash function is applied to the concatenated hash values to get a final single hash value that verifies the file integrity as follows.

$$MAC_k = H_{\text{SHA}-3}(B_k), \tag{11}$$

where $1 \le k \le z$, and

$$MAC_{\text{final}} = H_{\text{SHA}-3}\|_{k=1}^{z}(MAC_k), \tag{12}$$

where $1 \le k \le z$.

The hash value of each block (MAC_k), the encrypted file (C, C_A), total number of blocks "z," and the final hash value (MAC_{final}) are stored on the fog node. Only the total number of blocks "z" and the file's name are saved on the local storage. The user saves the local information (z, filename) for every file.

4.6. Proxy Re-Encryption. User B requests the proxy server for the re-encryption process to get the file uploaded on the fog node by user A. In the re-encryption process, the proxy server transforms the first level ciphertext of user A ($C_A \in$ C1) to the second level ciphertext of user B ($C_B \in$ C2) and also chooses a fresh nonce. The proxy first checks the access control list if user B has access rights to the uploaded file, then the proxy server re-encrypts that file and attached a Nonce value with the file as depicted in equations (13) and (14), and then allows user B to download it.

$$\begin{aligned} C_B &= C_A R_{A\longrightarrow B} = \chi.K_a.\wp\left(\frac{K_b}{K_a}\right) = \chi.K_b.\wp, \\ C_k &= B_k + \chi.\wp, \end{aligned} \tag{13}$$

$$C = \|_{k=1}^{z}(Ck\|\text{Nonce}), \tag{14}$$

where $1 \le k \le z$.

The proxy server transfers the final hash value (MAC_{final}), second-level ciphertext (C, C_B), and the total numbers of blocks "z" to user B for checking the integrity of the file and decryption.

4.7. Decryption. User B downloads the final hash value (MAC_{final}), second-level ciphertext (C, C_B), and the total numbers of blocks "z." Then, user B decrypts the file by using C_B and his private key K_B as shown in the following equation:

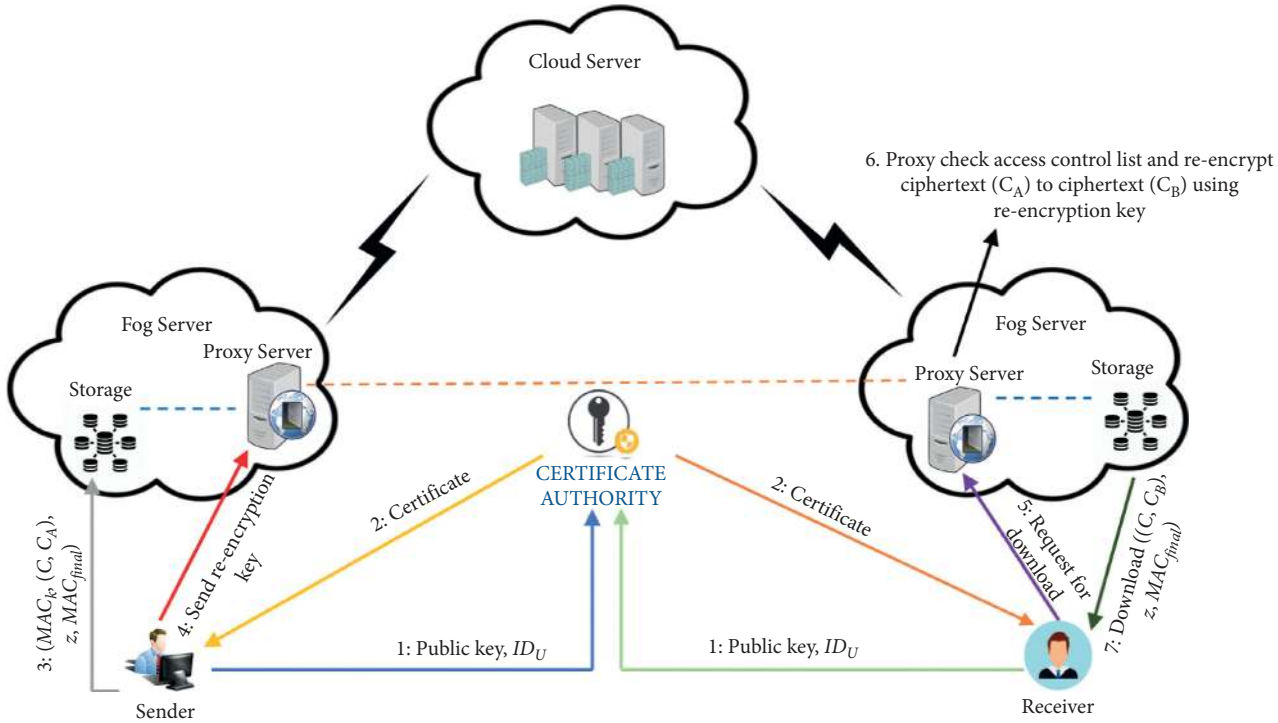

Figure 3: Proposed scheme architecture for secure data sharing in fog computing.

$$B_k = C_k - \left(\frac{1}{K_B}\right) C_B, \tag{15}$$

where $1 \le k \le z$.

User A can decrypt the file by using C_A and his private key K_A as shown in the following equation:

$$B_k = C_k - \left(\frac{1}{K_A}\right) C_A, \tag{16}$$

where $1 \le k \le z$.

After decrypting all the blocks, user B concatenates all the blocks to get the original file. User B checks the integrity of the file depicted as follows:

$$\text{FILE} = \|_{k=1}^{z} (B_k), \tag{17}$$

where $1 \le k \le z$,

$$\text{MAC}_k = H_{\text{SHA-3}} (B_k), \tag{18}$$

where $1 \le k \le z$, and

$$\text{MAC}_{\text{final}} = H_{SHA-3} \|_{k=1}^{z} (\text{MAC}_k), \tag{19}$$

where $1 \le k \le z$.

If the value of the calculated hash function is equal to the value of the final hash function ($\text{MAC}_{\text{final}}$), then the FILE integrity is confirmed; otherwise, the original file is changed by some intruder.

5. Incremental Block Modification

In this phase, we use incremental cryptographic technique shown in Figure 4. The block modification operations are performed by using the incremental cryptographic technique. The block modification consists of three phases: (1) block deletion, (2) block updation, and (3) block insertion. Dividing the FILE into blocks and calculating the hash value of each block increases the overhead, but this overhead can be overcome by using the incremental cryptographic technique during the block modification operation. There is no need to calculate the hash value from scratch while performing the block modification operations. Different phases of block modification are given in [56].

5.1. Block Deletion Operation. In this phase, the user wants to delete some blocks r of the uploaded file from different locations Idx. The user downloads hash values of all blocks (MAC_k) of the corresponding file from the fog node. The user can delete the required blocks from the file by updating the final hash value. Final hash is updated by computing the hash of the concatenated hashes of the remaining blocks as depicted in (20).

$$\text{MAC}_{\text{final}} = H_{\text{SHA-3}} \left(\|_{k=1}^{I\ dx-1} \text{MAC}_k \|_{L=I\ dx+r}^{z} \text{MAC}_L \right). \tag{20}$$

The user sends the location information (Idx) of the deleted blocks and updates the final hash value ($\text{MAC}_{\text{final}}$), total number of deleted blocks r, with delete request to the

proxy server. The proxy server updates the final hash value $\mathrm{MAC}_{\mathrm{final}}$ and deletes all the requested blocks from the file given in (20) and (21).

$$\mathrm{MAC}_{\mathrm{final}} = \left(\left\|_{k=1}^{I\ dx-1} \mathrm{MAC}_k \right\|_{L=I\ dx+r}^{z} \mathrm{MAC}_L \right), \tag{21}$$

$$C = \left(\left\|_{k=1}^{I\ dx-1} \mathrm{MAC}_k \right\|_{L=I\ dx+r}^{z} \mathrm{C}_L \right). \tag{22}$$

The proxy server also updates all the values into the fog storage, and the value of the total number of blocks is also changed from z to $z = z\text{-}r$.

5.2. Block Modification Operation. In this phase, if the user wants to modify some blocks r of the corresponding file at different locations *Idx*, the user downloads the hash value of all the blocks (MAC_k) and the encrypted file (C, C_U) from the fog node. The user then encrypts all the blocks that need to be modified using the following process:

$$B_{\mathrm{update}_k} = C_{\mathrm{update}_k} + \left(\frac{1}{K_A} \right) C_A, \tag{23}$$

where $1 \le k \le z$.

The user also calculates the hash values of all the modified blocks, and afterwards, the user calculates the final hash value $\mathrm{MAC}_{\mathrm{final}}$ as follows:

$$\mathrm{MAC}_{\mathrm{update}} = H_{\mathrm{SHA-3}}\left(B_{\mathrm{update}} \right), \tag{24}$$

$$\mathrm{MAC}_{\mathrm{final}} = \left(\left\|_{k=1}^{I\ dx-1} \mathrm{MAC}_k \right\|_{L=I\ dx}^{I\ dx+r} \mathrm{MAC}_{\mathrm{update}_L} \right\|_{M=I\ dx+r}^{z} \mathrm{MAC}_M \Bigg). \tag{25}$$

The user sends the location information (*Idx*) of the modified blocks and the final hash value ($\mathrm{MAC}_{\mathrm{final}}$) obtained by applying hash function on the concatenated hash values of all the blocks, total number of modified blocks r, with modification request to the proxy server. The proxy server updates the final hash value $\mathrm{MAC}_{\mathrm{final}}$ and updates all the requested blocks of the FILE as follows:

$$\mathrm{MAC}_{\mathrm{final}} = \left(\left\|_{k=1}^{I\ dx-1} \mathrm{MAC}_k \right\|_{L=I\ dx}^{I\ dx+r} \mathrm{MAC}_{\mathrm{update}_L} \right\|_{M=I\ dx+r}^{z} \mathrm{MAC}_M \Bigg), \tag{26}$$

$$\mathrm{C}_{\mathrm{final}} = \left(\left\|_{k=1}^{I\ dx-1} \mathrm{C}_k \right\|_{L=I\ dx}^{I\ dx+r} \mathrm{C}_{\mathrm{update}_L} \right\|_{M=I\ dx+r}^{z} \mathrm{C}_M \Bigg). \tag{27}$$

The proxy server updates all the values into the fog storage, and the value of the total number of blocks z remains same.

5.3. Block Insertion Operation. In this phase, if the user wants to insert some new blocks r at different locations *Idx* of the corresponding file. Then, the user downloads the hash value of all the blocks (MAC_k) and encrypted file (C, C_U) from the fog node. The user then encrypts all the new blocks that need to be inserted in the file using the following procedure:

$$C_{\mathrm{new}} = \chi \cdot P_A C_U = B_{\mathrm{new}} \cdot \chi \cdot \wp. \tag{28}$$

The user also calculates the hash values of all the newly inserted blocks in the file, and afterwards, the user calculates the final hash value $\mathrm{MAC}_{\mathrm{final}}$ using the following equations:

$$\mathrm{MAC}_{\mathrm{insert}} = H_{\mathrm{SHA-3}}\left(B_{\mathrm{insert}} \right), \tag{29}$$

$$\mathrm{MAC}_{\mathrm{final}} = \left(\left\|_{k=1}^{I\ dx-1} \mathrm{MAC}_k \right\|_{L=I\ dx}^{I\ dx+r} \mathrm{MAC}_{\mathrm{update}_L} \right\|_{M=I\ dx+r}^{z} \mathrm{MAC}_M \Bigg). \tag{30}$$

The user sends the location information (*Idx*) of the newly inserted blocks in the file and final hash value ($\mathrm{MAC}_{\mathrm{final}}$) obtained by applying hash function on all the concatenated hash values of all old and new blocks, total number of newly inserted blocks r, with insertion request to the proxy server. The proxy server updates the final hash value $\mathrm{MAC}_{\mathrm{final}}$ and all the newly inserted blocks in the file using the following procedure:

$$\mathrm{MAC}_{\mathrm{final}} = \left(\left\|_{k=1}^{I\ dx-1} \mathrm{MAC}_k \right\|_{L=I\ dx}^{I\ dx+r} \mathrm{MAC}_{\mathrm{update}_L} \right\|_{M=I\ dx+r}^{z} \mathrm{MAC}_M \Bigg), \tag{31}$$

$$\mathrm{C}_{\mathrm{final}} = \left(\left\|_{k=1}^{I\ dx-1} \mathrm{C}_k \right\|_{L=I\ dx}^{I\ dx+r} \mathrm{C}_{\mathrm{update}_L} \right\|_{M=I\ dx+r}^{z} \mathrm{C}_M \Bigg). \tag{32}$$

The proxy server updates all the values into the fog storage, and the value of the total number of blocks also changes from z to $z = z + r$.

6. Security Analysis

In this phase, we present the security analysis is of our certificate-based incremental proxy re-encryption scheme. In this security analysis, we ensure various security requirements such as integrity and confidentiality.

6.1. Confidentiality. Data confidentiality means that sensitive data are protected from unauthorized users and blocks unauthorized users' access to sensitive data.

6.1.1. Level-1 Encryption. In our method, if an intruder wants to steal our sensitive data, then he must know about the level-1 encryption sender private key. An intruder can get the sender's private key by performing the following steps:

Step 1: the intruder can get a level-1 encryption sender private key if he computes equation (33). For this, the intruder needs to find the value of β, and it is not feasible due to the hyperelliptic curve discrete logarithm problem:

$$K_U = \beta \cdot (\xi - \eta). \tag{33}$$

Step 2: if in any way the intruder gets the value of β, then next it wants to find the value of ξ and η from equations (34) and (35). For this, the intruder needs to find the value of α and γ, and it is also not feasible due to

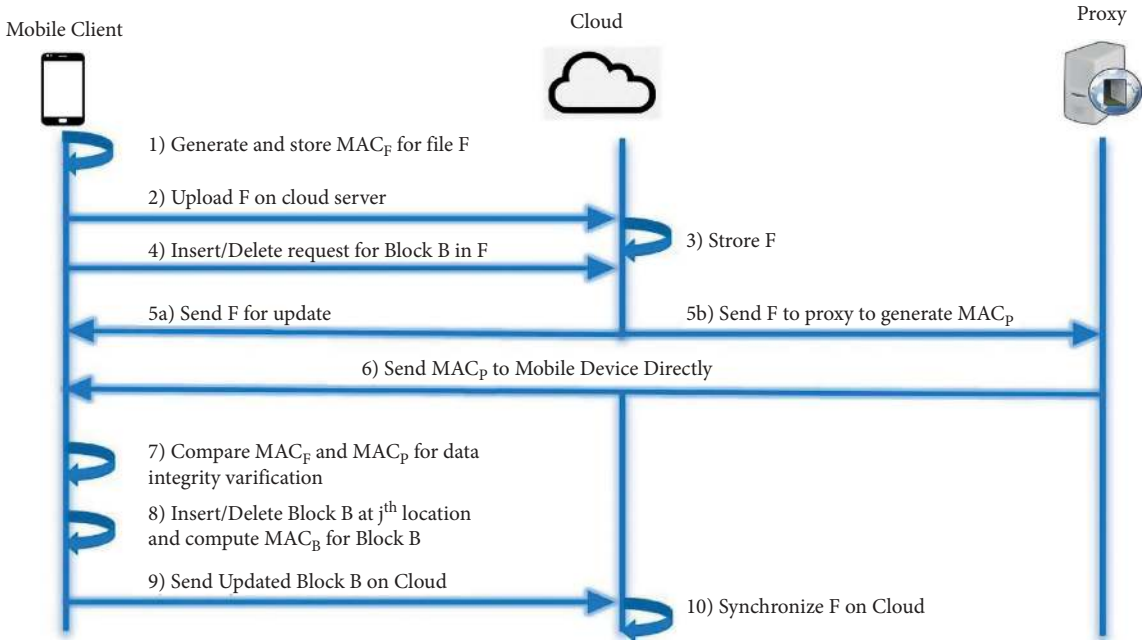

Figure 4: Workflow of the incremental cryptographic scheme [66].

the hyperelliptic curve discrete logarithm problem. In fact, to get the level-1 encryption sender private key, the intruder will have to solve the hyperelliptic curve discrete logarithm problem three times, which is not feasible.

$$\xi = \alpha \cdot (\gamma - \alpha), \tag{34}$$

$$\eta = \gamma \cdot (\beta + \gamma). \tag{35}$$

6.1.2. Level-2 Encryption. In the level-2 encryption phase, we analyse the confidentiality of the proxy re-encrypted data. In this phase, there are two cases, one for the intruder.

Case 1: in the first case, the intruder wants to get the sensitive data. For this, he must know about the level-2 encryption receiver private key. An intruder can get the receiver's private key by performing the following steps:

Step 1: the intruder can get a level-2 encryption receiver private key if he computes equation (33). For this, the intruder needs to find the value of β, and it is not feasible due to the hyperelliptic curve discrete logarithm problem.

Step 2: if in any way the intruder gets the value of β, then next it wants to find the value of ξ and η from equations (34) and (35). For this, the intruder needs to find the value of α and γ, and it is also not feasible due to the hyperelliptic curve discrete logarithm problem.

6.2. Integrity. Data integrity means the overall reliability, completeness, and accuracy of the data. This means that the recipient receives the same data which is sent by the sender, and we use the hash function to ensure integrity. The sender applies the hash function on each data block and then applies final hash on the concatenated hash value shown as follows.

$$\mathrm{MAC}_k = H_{\mathrm{SHA-3}}(B_k), \tag{36}$$

where $1 \le k \le z$, and

$$\mathrm{MAC}_{\mathrm{final}} = H_{\mathrm{SHA-3}}\left\|_{k=1}^{z}(\mathrm{MAC}_k), \right. \tag{37}$$

where $1 \le k \le z$.

If the attacker has made any changes in the ciphertext, then C is converted to C', and after decrypting C', we get the message M', and the MAC value of M' is not equal to $\mathrm{MAC}_{\mathrm{final}}$. So, we' will find out if the ciphertext has changed. Therefore, our proposed scheme verifies the integrity of the data.

6.3. Replay Attack. In our scheme, every time the proxy server generates a Nonce value Nonce when it re-encrypts the blocks and attached this Nonce value to every block such as FILE = $\|(B_k, \mathrm{Nonce})$. Nonce value Nonce is the identity of each block. The Nonce value is attached with each block to avoid the replay attack. If an intruder tries to send the previous block to the recipient, the recipient easily knows that it is the previous block, by the Nonce value, because the Nonce value is renewed in each session. In this regard, we can say that our scheme is safe from the replay attack.

7. Experimental Results

Our scheme is evaluated on the bases of turnaround time for the different file size that is given in Table 2. We have divided our file into eight blocks. Then, we performed four experiments on the given data set, and in the first experiment, we update only one block and compare the results with the

Table 2: Parameters for all scenarios.

File size in bytes	Total files
51 200	50
102,400	50
153,600	50
204,800	50
256,000	50

previously proposed schemes by Guo et al. [51], Khan et al. [21], Khan et al. [56], and Bhatia et al. [64]. Proxy re-encryption scheme proposed in [51] does not use the incremental cryptographic technique; therefore, it encrypts, decrypts, and computes the hash value for a complete file from scratch. The schemes in [21, 56] use the bilinear paring technique which uses a 256-bit key, and [64] uses the elliptic curve technique with 160-bit key. Our certificate-based incremental proxy re-encryption scheme (CB-PReS) uses the hyperelliptic curve technique with 80-bit key.

7.1. Case Scenario 1. In first case, we perform block modification operation in which we modify only one block of a file with different file sizes given in Table 2, and then we compare our results with the previously proposed schemes [21, 51, 56, 64] based on the turnaround time while block deletion, block insertion, and block updation. We can evaluate the turnaround time as follows:

$$\text{turn around time (TT)} = t_{r\,d} + t_{\text{hash}} + t_{e\,d}, \tag{38}$$

where t_{rd} represents the time that is required to read the file, $t_{h\text{ash}}$ represents the time that is required to compute the hash value, and t_{ed} represents the time that is required for encryption/decryption. Figure 5 shows the result in terms of turnaround time while modifying the block. The file size in bytes is shown along the x-axis with the total numbers of files, and the turnaround time is shown along the y-axis in milliseconds. In this experiment, every file is divided into eight blocks. The results of experiment 1 in Figure 5 clearly show that our scheme is more efficient as compared to the previously proposed schemes based on the turnaround time while block deletion, block insertion, and block updation.

7.2. Case Scenario 2. In second case, we perform block modification operation in which we modify two blocks of a file with different file sizes, and then we compare the obtained results with the previously proposed schemes based on the turnaround time while block deletion, block insertion, and block updation. We can evaluate the turnaround time as shown in equation (38).

The results of experiment 2 in Figure 6 clearly shows that our scheme is more efficient as compared to the previously proposed schemes [21, 51, 56, 64] based on the turnaround time while block deletion, block insertion, and block updation.

7.3. Case Scenario 3. In third case, three blocks are modified of a file with different file sizes, and then we compare the results with the previous proposed schemes. We can evaluate the turnaround time as shown in equation (38).

The results of experiment 3 in Figure 7 clearly shows that our scheme is more efficient as compared to the previous schemes.

7.4. Case Scenario 4. In fourth case, we update four blocks of a file with different sizes, and then we compare the results with the previously proposed schemes. We can evaluate the turnaround time as shown in equation (38).

The results of experiment 4 in Figure 8 clearly shows that our scheme is more efficient as compared to the previous proposed schemes.

8. Performance Evaluation

In this phase, we compare our proposed scheme with the previously proposed schemes by Guo et al. [51], Khan et al. [21], Khan et al. [56], and Bhatia et al. [64] and then evaluate our scheme in terms of computation overhead of each block. All the specifications about hardware and software which we use to conduct the experiments are depicted in Table 3. For the development of the fog client application, we use iFogSim simulator [67]. The major operation, i.e., hyperelliptic curve divisor multiplication (HDM), cost is computed by using a GitHub library called libg2hec [68]. This library provides a divisor group operation in the Jacobian of genus 2 (imaginary) hyperelliptic curve. We need to install V. Shoup's NTL library [69] before installing the G2HEC library. G2HEC library results have been tested with NTL 5.5 version on an x86_64 macOS big sur 11.4 operating system. The cost of the elliptic curve scalar multiplication (SM) operation is computed by using a standard cryptography library MIRACL [70] and bilinear paring modular exponential (MEXP), and bilinear paring operation (PBC) cost is computed by using the (pairing-based cryptography) library PBC [71]. PBC is built on the GMP library [72], and all the paring base mathematical operation such as bilinear paring modular exponential and pairing-based point multiplication are performed by using the GMP library. We can achieve 1024-bit RSA level security by using type A bilinear pairing over an elliptic curve with $p = 512$-bit prime and $q = 160$-bit prime number. We can also achieve the same level of security in ECC by using secp160r1 curve over a finite field Fq, and the equivalent level of security can also be achieved by using the genus two hyperelliptic curve over the finite field Fq with the following values $|Fq\,|g$. log2 $q \approx 280$. To achieve the integrity of the data, we have used the SHA-3 algorithm, which is more resistant to collision and preimaging attacks than the previous SHA-2 algorithm.

8.1. Computational Overhead. In this phase, we consider some major operations such as hyperelliptic curve divisor multiplication (HDM), elliptic curve scalar multiplication (SM), bilinear paring modular exponential (MEXP), and bilinear paring operation (PBC). The cost of the major operation in millisecond is given as follows: hyperelliptic curve divisor multiplication cost is 0.36 ms, elliptic curve

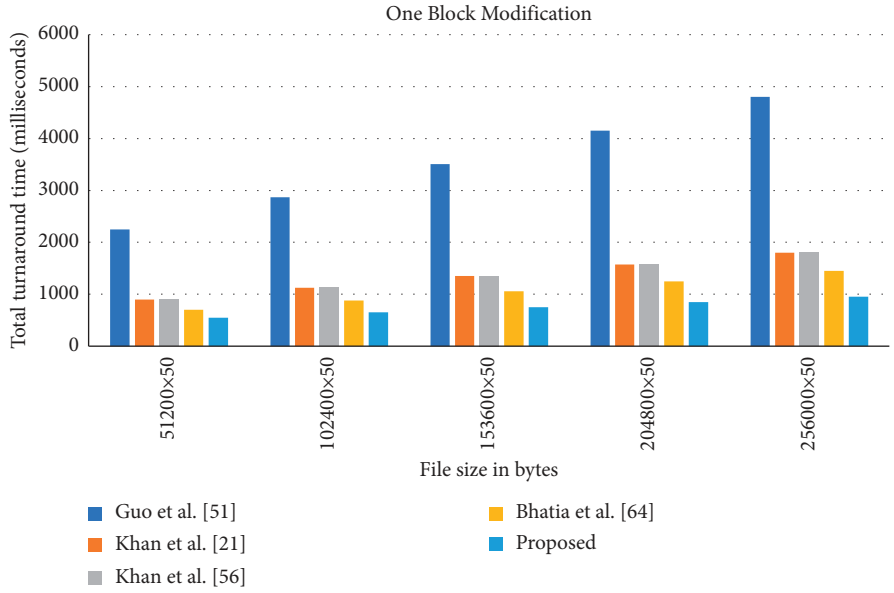

Figure 5: One block modification.

scalar multiplication cost is 0.64 ms, bilinear paring modular exponential cost is 0.84 ms, and bilinear paring operation cost is 1.02 ms. It is clearly shown from the results that the hyperelliptic curve divisor multiplication cost is much lesser as compared to the remaining cryptographic major operations. Our proposed scheme uses hyperelliptic curve divisor multiplication operations. The cost of the all-cryptographic major operations is given in Figure 9 and Table 4.

Due to the use of hyperelliptic curve divisor multiplication, our proposed (CB-PRe) scheme shows significant improvement in the block(s) modification results compared to the previously proposed [21, 51, 56, 64] schemes. The result improvement depends on the number of block(s) modification operations. The comparison of result improvements between our proposed (CB-PRe) scheme and Guo et al.'s [51] scheme is given in Table 5; it shows that the average percentage turnaround time of our proposed (CB-PRe) scheme is 78.24% better than Guo et al.'s scheme while modifying one block, 70.47% better while the modification of two blocks, 62.73% better while modifying three blocks, and 54.54% better while the modification of four blocks. The computational overhead reduction formula is given as follows.

$$\left(\frac{\text{existing scheme } cost \; - \; proposed \; scheme \text{ cost}}{existing \; scheme \text{ cost}}\right) * 100. \tag{39}$$

Table 6 depicts the comparison of result improvements between our (CB-PRe) scheme and Khan et al.'s [21, 56] schemes. The average percentage turnaround time of our proposed scheme is 40.66% better than Khan et al.'s [21, 56] schemes, while modifying one block, 37.51% better while the modification of two blocks, 36.62% better while modifying three blocks, and 33.48% better while the modification of four blocks.

The comparison of result improvements between our proposed scheme and Bhatia et al.'s [64] scheme is given in Table 7; it depicted that the average percentage turnaround time of our scheme is 25.67% better than Bhatia et al.'s scheme while modifying one block, 24.2% better while the modification of two blocks, 22.2% better while modifying three blocks, and 19.92% better while the modification of four blocks.

8.2. Communication Overhead. In this section, we compared the communication cost of our scheme with the existing proposed schemes, i.e., Guo et al. [51], Khan et al. [21], Khan et al. [56], and Bhatia et al. [64]. Communication costs will increase if there are some extra bits with the original message. We suppose some terms as follows:

(i) $|K|$ represents key size of bilinear pairing is equals to 256 bits, elliptic curve is 160 bits, and hyperelliptic curve is 80 bits

(ii) $|M|$ represents ciphertext or plaintext size and equals 100 bits

(iii) MAC represents message digest size of hash function and equals 512 bits

(iv) C_{ERT} represents user certificate size and equals 256 bits in bilinear paring, 160 bits in elliptic curve, and 80 bits in hyper elliptic curve

(v) $|B|$ block size is equal to 13 bits

(vi) $R_{A \longrightarrow B}$ re-encryption key size of bilinear pairing equals 256 bits, elliptic curve is 160 bits, and hyperelliptic curve is 80 bits

Table 8 shows that our scheme is better than the existing related schemes, i.e., [21, 51, 56, 64] in terms of communication cost.

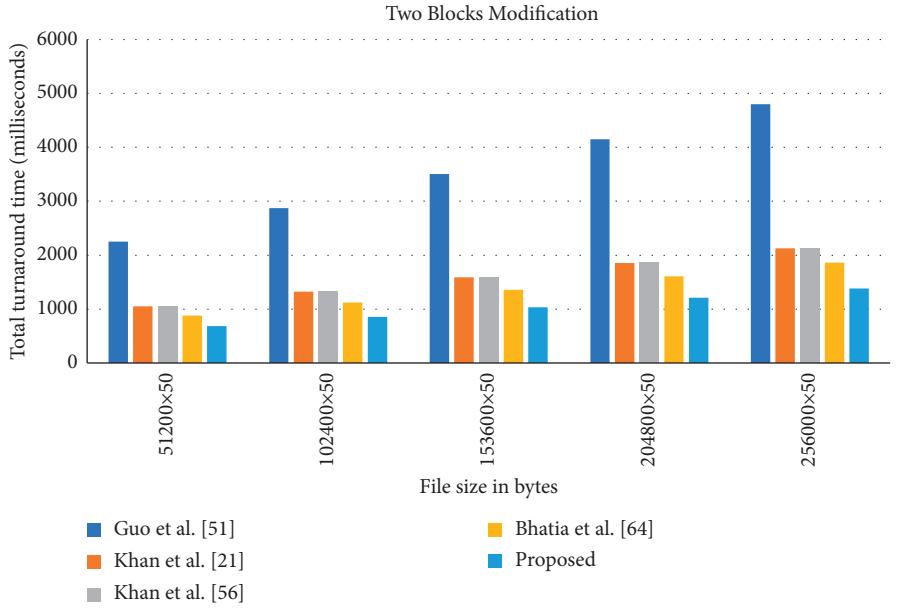

Figure 6: Two-block modification.

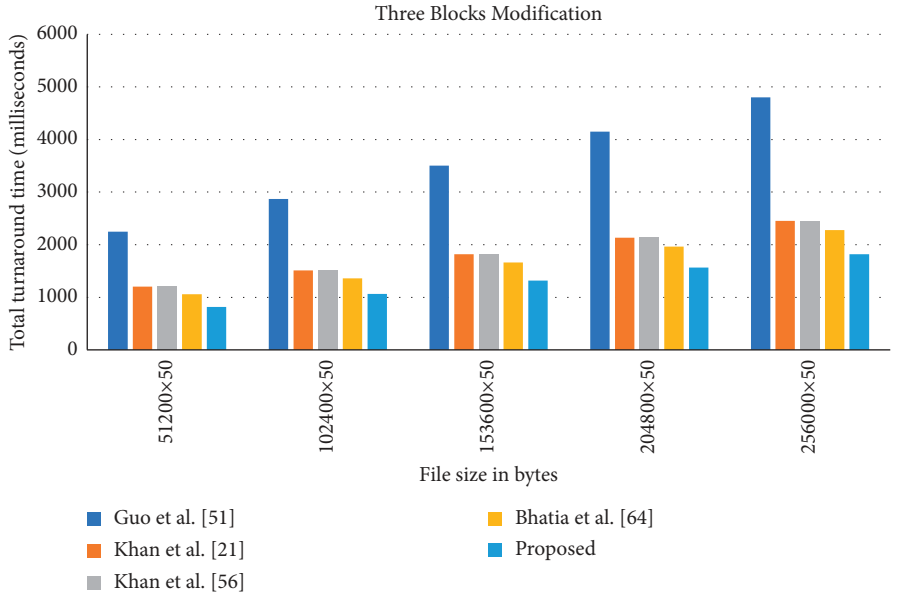

Figure 7: Three blocks modification.

8.3. Communication Cost Reduction. The communication cost reduction of our scheme from the existing schemes can be calculated by the following formula:

$$\left(\frac{\text{existing scheme cost} - \text{proposed scheme cost}}{\text{existing scheme cost}}\right) * 100. \tag{40}$$

Reduction from Guo et al.'s study [51] is

$$\left(\frac{3216 - 2002}{3216}\right) * 100 = 37.75\,\%. \tag{41}$$

Reduction from Khan et al.'s study [21] is

$$\left(\frac{2886 - 2002}{2886}\right) * 100 = 30.63\,\%. \tag{42}$$

Reduction from Khan et al.'s study [56] is

$$\left(\frac{2530 - 2002}{2530}\right) * 100 = 20.86\,\%. \tag{43}$$

Reduction from Bhatia et al.'s study [64] is

$$\left(\frac{2242 - 2002}{2242}\right) * 100 = 9.10\,\%. \tag{44}$$

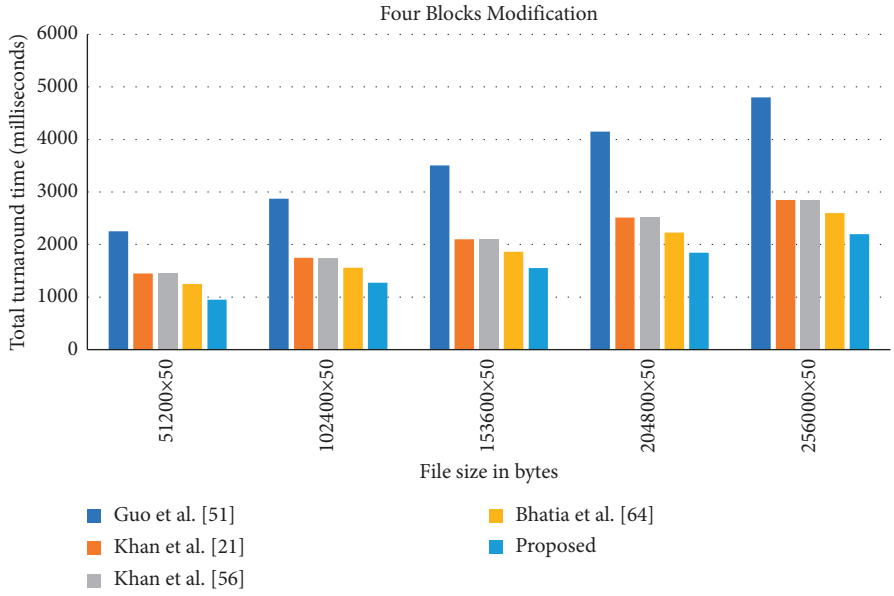

Figure 8: Four blocks modification.

Table 3: Hardware and software specification.

Hardware and software	Specification/version
System	MacBook pro-2015
RAM	16 GB 1600 MHz DDR3
Processor	2.2 GHz quad-core Intel Core i7
Storage	512 GB
OS	macOS big sur 11.4
Application	iFogSim
iFogSim	1.0
libg2hec	2.1
NTL	5.5
MIRACL	7.0.0
PBC	1.0
GMP	6.2.1

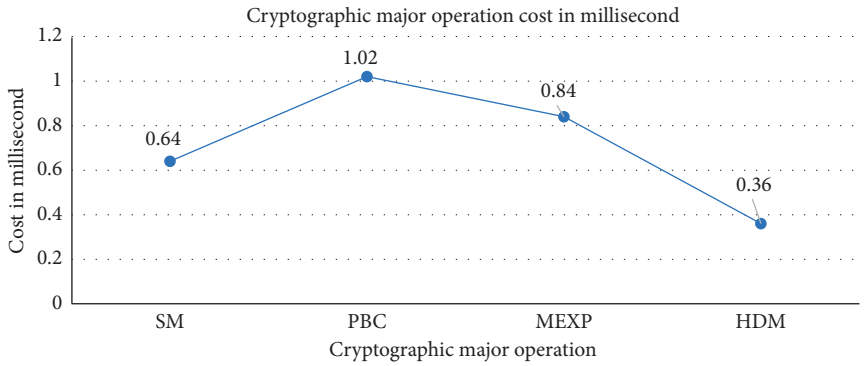

Figure 9: Cryptographic major operation cost in millisecond.

Table 4: Cryptographic major operation cost in millisecond.

Major operations	Cost in milliseconds
Bilinear paring operation	1.02
Bilinear paring modular exponential	0.84
Elliptic curve scalar multiplication	0.64
Hyperelliptic curve divisor multiplication	0.36

Table 5: Result improvements of the proposed scheme in block(s) modification operations as compared to [51].

Number of blocks	51200 × 50 (%)	102400 × 50 (%)	153600 × 50 (%)	204800 × 50 (%)	256000 × 50 (%)	Average (%)
1	75.56	77.35	78.6	79.51	80.2	78.24
2	69.64	70.1	70.52	70.89	71.18	70.47
3	61.89	62.25	62.74	63.09	63.69	62.73
4	53.12	53.61	54.35	55.47	56.16	54.54

Table 6: Result improvements of proposed scheme in block (s) modification operations as compared to [21, 56].

Number of blocks	51200 × 50 (%)	102400 × 50 (%)	153600 × 50 (%)	204800 × 50 (%)	256000 × 50 (%)	Average (%)
1	38.88	39.22	40.49	41.98	42.71	40.66
2	36.09	36.91	37.42	38.12	39.03	37.51
3	34.95	35.61	36.27	37.41	38.88	36.62
4	31.07	32.62	33.88	34.73	35.12	33.48

Table 7: Result improvements of proposed scheme in block(s) modification operations as compared to [64].

Number of blocks	51200 × 50 (%)	102400 × 50 (%)	153600 × 50 (%)	204800 × 50 (%)	256000 × 50 (%)	Average (%)
1	24.21	25.09	25.87	26.34	26.83	25.67
2	23.01	23.67	24.22	24.79	25.31	24.2
3	21.47	21.89	22.78	22.56	22.31	22.2
4	19.09	19.76	20.01	20.43	20.32	19.92

Table 8: Communication cost in bits.

Guo et al. [51]	2 (K) + 4 (M) + 4 (MAC) + 1 ($R_{A\longrightarrow B}$)	2 (256) + 4 (100) + 4 (512) + 1 (256) = 3216
Khan et al. [21]	2 (K) + 3 (M) + 4 (MAC) + 2 (B)	2 (256) + 3 (100) + 4 (512) + 2 (13) = 2886
Khan et al. [56]	2 (K) + 2 (M) + 3 (MAC) + 1 ($R_{A\longrightarrow B}$) + 2 (B)	2 (256) + 2 (100) + 3 (512) + 1 (256) + 2 (13) = 2530
Bhatia et al. [64]	2 (K) + 2 (M + 3 (MAC) + 1 ($R_{A\longrightarrow B}$) + 2 (B)	2 (160) + 2 (100) + 3 (512) + 1 (160) + 2 (13) = 2242
Proposed method	K + 1 (C_{ERT}) + 2 (M) + 3 (MAC) + 1 ($R_{A\longrightarrow B}$) + 2 (B)	1 (80) + 1 (80) + 2 (100) + 3 (512) + 1 (80) + 2 (13) = 2002

9. Conclusion

Secure EHR storage and sharing is a serious issue while using the cloud infrastructure. In certificateless cryptography, we cannot verify the public key of any user that is the major drawback of the certificateless cryptographic schemes. In this paper, we have proposed a lightweight certificate-based incremental proxy re-encryption for e-healthcare data sharing scheme in fog computing. In certificate-based schemes every user can verify the public key of other users. Recently proposed I-PRE schemes involve expensive bilinear pairing and elliptic curve operation which uses 256-bit key and 160-bit key, respectively. In this scheme, we use the hyperelliptic curve technique with 80-bit key and compare the block modification results on the basis of turnaround time with the previously proposed I-PRE schemes. Results clearly show that our scheme is more efficient as compared to the previously proposed schemes. Our scheme also provides data integrity and confidentiality and deals with the latency problem by using the fog computing paradigms. To reduce the overhead of resource constraints IoT devices, complex and resource-intensive cryptographic functions are offloaded on fog nodes. In the future, we will develop a group-based scheme that will be useful for multiple users.

Data Availability

The data used to support the findings of this study are provided in this article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] United Nations, *World Population Ageing, 2014*, Vol. 73, United Nations, Department of Economic and Social Affairs Population Division, , New York, NY, USA, 2014.

[2] M. Chen, J. Yang, Y. Hao, S. Mao, and K. Hwang, "A 5G cognitive system for healthcare," *Big Data and Cognitive Computing*, vol. 1, no. 1, p. 2, 2017.

[3] Frost & Sullivan, *Drowning In Big Data? Reducing Information Technology Complexities and Costs for Healthcare Organizations*, http://www.emc.com/collateral/analyst-reports/frost-sullivan-reducing-information-technology-complexities-ar.pdf, Frost & Sullivan, San Antonio, TX, USA, 2012, http://www.emc.com/collateral/analyst-reports/frost-sullivan-reducing-information-technology-complexities-ar.pdf.

[4] M. Chen, S. Mao, and Y. Liu, "Big data: a survey," *Mobile Networks and Applications*, vol. 19, no. 2, pp. 171–209, 2014.

[5] M. S. Hossain and G. Muhammad, "Healthcare big data voice pathology assessment framework," *IEEE Access*, vol. 4, pp. 7806–7815, 2016.

[6] M. Chen, Y. Hao, K. Hwang, L. Wang, and L. Wang, "Disease prediction by machine learning over big data from healthcare communities," *IEEE Access*, vol. 5, no. 1, pp. 8869–8879, 2017.

[7] M. Chen, P. Zhou, and G. Fortino, "Emotion communication system," *IEEE Access*, vol. 5, pp. 326–337, 2017.

[8] L. Ali, C. Zhu, Z. Zhang, and Y. Liu, "Automated detection of Parkinson's disease based on multiple types of sustained phonations using linear discriminant analysis and genetically optimized neural network," *IEEE Journal of Translational Engineering in Health and Medicine*, vol. 7, pp. 1–10, 2019.

[9] L. Ali, C. Zhu, M. Zhou, and Y. Liu, "Early diagnosis of Parkinson's disease from multiple voice recordings by simultaneous sample and feature selection," *Expert Systems with Applications*, vol. 137, pp. 22–28, 2019.

[10] F. S. Ahmad, L. Ali, R. U. Mustafa et al., "A hybrid machine learning framework to predict mortality in paralytic ileus patients using electronic health records (EHRs)," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 3, pp. 3283–3293, 2021.

[11] A. A. Diro, N. Chilamkurti, and Y. Nam, "Analysis of lightweight encryption scheme for fog-to-things communication," *IEEE Access*, vol. 6, pp. 26820–26830, 2018.

[12] C. S. Nandyala and H.-K. Kim, "From cloud to fog and IoT-based real-time U-healthcare monitoring for smart homes and hospitals," *International Journal of Smart Home*, vol. 10, no. 2, pp. 187–196, 2016.

[13] M. M. Hassan, K. Lin, X. Yue, and J. Wan, "A multimedia healthcare data sharing approach through cloud-based body area network," *Future Generation Computer Systems*, vol. 66, pp. 48–58, 2017.

[14] X. Meng, W. Wang, and Z. Zhang, "Delay-constrained hybrid computation offloading with cloud and fog computing," *IEEE Access*, vol. 5, pp. 21355–21367, 2017.

[15] H. F. Atlam, R. J. Walters, and G. B. Wills, "Fog computing and the internet of things: a review," *Big Data and Cognitive Computing*, vol. 2, no. 2, pp. 1–18, 2018.

[16] J. Ni, K. Zhang, X. Lin, and X. Shen, "Securing fog computing for internet of things applications: challenges and solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 601–628, 2018.

[17] P. Zhang, J. K. Liu, F. R. Yu, M. Sookhak, M. H. Au, and X. Luo, "A survey on access control in fog computing," *IEEE Communications Magazine*, vol. 56, no. 2, pp. 144–149, 2018.

[18] B. J. Mohd and T. Hayajneh, "Lightweight block ciphers for IoT: energy optimization and survivability techniques," *IEEE Access*, vol. 6, pp. 35966–35978, 2018.

[19] N. Farjana, S. Roy, M. J. N. Mahi, and M. Whaiduzzaman, "An identity-based encryption scheme for data security in fog computing," in *Proceedings of the International Joint Conference on Computational Intelligence,Algorithms for Intelligent Systems*, pp. 215–226, Springer, Dhaka, Bangladesh, October 2019.

[20] M. Al-Khafajiy, T. Baker, A. Waraich, D. Al-Jumeily, and A. Hussain, "IoT-fog optimal workload via fog offloading," in *Proceedings of the IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)*, pp. 359–364, IEEE, Zurich, Switzerland, December 2018.

[21] A. N. Khan, M. L. M. Kiah, S. U. Khan, S. A. Madani, and A. R. Khan, "A study of incremental cryptography for security schemes in mobile cloud computing environments," in *Proceedings of the IEEE Symposium on Wireless Technology & Applications (ISWTA)*, pp. 62–67, IEEE, Kuching, Malaysia, September 2013.

[22] C. Cavanagh and U. C. Irvine, "UC irvine electronic Theses and Dissertations," vol. 228, University of California, Irvine, CA, USA, 2016, Thesis.

[23] C. Zhou, Z. Zhao, W. Zhou, and Y. Mei, "Certificateless key-insulated generalized signcryption scheme without bilinear pairings," *Security and Communication Networks*, vol. 2017, pp. 1–17, Article ID 8405879, 2017.

[24] A. Rahman, I. Ullah, M. Naeem et al., "A lightweight multimessage and multi-receiver heterogeneous hybrid signcryption scheme based on hyper elliptic curve," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 5, pp. 160–167, 2018.

[25] W. Itani, A. Kayssi, and A. Chehab, "Efficient healthcare integrity assurance in the cloud with incremental cryptography and trusted computing," *Cloud Technology*, pp. 845–857, 2015.

[26] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," *Lecture Notes in Computer Science*, vol. 1403, pp. 127–144, 1998.

[27] R. Roy and P. P. Mathai, "Proxy re-encryption schemes for secure cloud data and applications: a survey," *International Journal of Computer Applications*, vol. 164, no. 5, pp. 975–8887, 2017.

[28] Z. Qin, H. Xiong, S. Wu, and J. Batamuliza, "A survey of proxy re-encryption for secure data sharing in cloud computing," *IEEE Transactions on Services Computing*, vol. 13, no. 9, p. 1, 2016.

[29] S. Kim and I. Lee, "IoT device security based on proxy re-encryption," *Journal of Ambient Intelligence and Humanized Computing*, vol. 9, no. 4, pp. 1267–1273, 2018.

[30] M. Thangavel, P. Varalakshmi, and C. Abinaya, "A comparative study of attribute-based encryption schemes for secure cloud data outsourcing," in *Proceedings of the 2017 Ninth International Conference on Advanced Computing (ICoAC)*, pp. 261–266, IEEE, Chennai, India, December 2017.

[31] J. Weng, R. H. Deng, X. Ding, C.-K. Chu, and J. Lai, "Conditional proxy re-encryption secure against chosen-ciphertext attack," in *Proceedings of the 4th International Symposium ACM Symposium Information, Computer Communications Security ASIACCS'09*, pp. 322–332, Sydney, Australia, March 2009.

[32] P. Zeng and K.-K. R. Choo, "A new kind of conditional proxy Re-encryption for secure cloud storage," *IEEE Access*, vol. 6, pp. 70017–70024, 2018.

[33] M. Green and G. Ateniese, "Identity-based proxy re-encryption," in *Proceedings of the International Conference on Applied Cryptography and Network Security*, Springer, Zhuhai, China, June 2007.

[34] C. K. Chu and W. G. Tzeng, "Identity-based proxy re-encryption without random oracles," in *Proceedings of the International Conference on Information Security*, Valparaíso, Chile, October 2007.

[35] K. Liang, Z. Liu, X. Tan, D. S. Wong, and C. Tang, "A CCA-secure identity-based conditional proxy re-encryption without random oracles," *Lecture Notes in Computer Science*, vol. 7839, pp. 231–246, 2013.

[36] P. Dutta, W. Susilo, D. H. Duong, and P. S. Roy, "Collusion-resistant identity-based proxy re-encryption: lattice-based constructions in standard model," *Theoretical Computer Science*, vol. 871, pp. 16–29, 2021.

[37] Q. Tang, "Type-based proxy re-encryption and its construction," *Progress in Cryptology-Indocrypt*, vol. 5365, pp. 130–144, 2008.

[38] C. Sur, Y. Park, S. U. Shin, K. H. Rhee, and C. Seo, "Certificate-based proxy re-encryption for public cloud storage," in *Proceedings of the Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pp. 159–166, IEEE, Taichung, Taiwan, July 2013.

[39] A. Braeken, P. Shabisha, A. Touhafi, and K. Steenhaut, "Pairing free and implicit certificate based signcryption scheme with proxy re-encryption for secure cloud data storage," in *Proceedings of the 2017 3rd International Conference of Cloud Computing Technologies and Applications*, pp. 1–7, IEEE, Nanjing, China, June 2017.

[40] M. Sun, C. Ge, L. Fang, and J. Wang, "A proxy broadcast re-encryption for cloud data sharing," *Multimedia Tools and Applications*, vol. 77, no. 9, pp. 10455–10469, 2018.

[41] L. Jiang and D. Guo, "Dynamic encrypted data sharing scheme based on conditional proxy broadcast Re-encryption for cloud storage," *IEEE Access*, vol. 5, pp. 13336–13345, 2017.

[42] P. Zhang, Z. Chen, J. K. Liu, K. Liang, and H. Liu, "An efficient access control scheme with outsourcing capability and attribute update for fog computing," *Future Generation Computer Systems*, vol. 78, pp. 753–762, 2018.

[43] L. Ferretti, M. Marchetti, and M. Colajanni, "Fog-based secure communications for low-power IoT devices," *ACM Transactions on Internet Technology*, vol. 19, no. 2, pp. 1–21, 2019.

[44] Q. Wang, W. Li, and Z. Qin, "Proxy re-encryption in access control framework of information-centric networks," *IEEE Access*, vol. 7, pp. 48417–48429, 2019.

[45] T. Bhatia, A. K. Verma, and G. Sharma, "Secure sharing of mobile personal healthcare records using certificateless proxy re-encryption in cloud," *Transactions on Emerging Telecommunications Technologies*, vol. 29, no. 6, 2018.

[46] J. Xu and E.-C. Chang, "Towards efficient proofs of retrievability," in *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, pp. 79-80, Raleigh, NC, USA, October 2012.

[47] P. Chaudhari and M. L. Das, "PAC: privacy preserving proxy re-encryption for access control in public cloud," *Information Security Journal: A Global Perspective*, pp. 1–16, 2021.

[48] H.-Y. Lin and Y.-M. Hung, "An improved proxy Re-encryption scheme for IoT-based data outsourcing services in clouds," *Sensors*, vol. 21, no. 1, p. 67, 2021.

[49] A. Hamza, D. Shehzad, M. S. Sarfraz, U. Habib, and N. Shafi, "Novel secure hybrid image steganography technique based on pattern matching," *KSII Transactions on Internet and Information Systems*, vol. 15, no. 3, pp. 1051–1077, 2021.

[50] G. Manikandan, R. Perumal, and V. Muthukumaran, "Secure data sharing based on proxy re-encryption for internet of vehicles using seminearring," *Journal of Computational and Theoretical Nanoscience*, vol. 18, no. 1–2, pp. 516–521, 2021.

[51] H. Guo, Z. Zhang, J. Zhang, and C. Chen, "Towards a secure certificateless proxy re-encryption scheme," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8209, pp. 330–346, Springer, Berlin, Germany, 2013.

[52] A. Srinivasan and C. P. Rangan, "Certificateless proxy Re-encryption without pairing: revisited," in *Proceedings of the 3rd International Workshop on Security in Cloud Computing*, pp. 41–52, Singapore, April 2015.

[53] H.-S. Lee and D.-G. Han, "Information security and cryptology-ICISC 2013," in *Proceedings of the International Conference on Information Security and Cryptology*, pp. 67–88, ICISC, Seoul, Seoul Korea, November, 2013.

[54] Z. Qin, S. Wu, and H. Xiong, "Strongly secure and cost-effective certificateless proxy Re-encryption scheme for data sharing in cloud computing," in *Proceedings of the Big Data Computing and Communications*, pp. 205–216, Springer, Taiyuan, China, August 2015.

[55] W. Ren, L. Yu, R. Gao, and F. Xiong, "Lightweight and compromise resilient storage outsourcing with distributed secure accessibility in mobile cloud computing," *Tsinghua Science and Technology*, vol. 16, no. 5, pp. 520–528, 2011.

[56] A. N. Khan, M. L. M. Kiah, S. A. Madani, M. Ali, A. U. R. Khan, and S. Shamshirband, "Incremental proxy re-encryption scheme for mobile cloud computing environment," *The Journal of Supercomputing*, vol. 68, no. 2, pp. 624–651, 2014.

[57] A. N. Khan, M. L. M. Kiah, M. Ali, S. A. Madani, A. u. R. Khan, and S. Shamshirband, "BSS: block-based sharing scheme for secure data storage services in mobile cloud environment," *The Journal of Supercomputing*, vol. 70, no. 2, pp. 946–976, 2014.

[58] M. Bellare, O. Goldreich, and S. Goldwasser, "Incremental cryptography: the case of hashing and signing," in *Proceedings of the Annual International Cryptology Conference*, pp. 216–233, Springer, Santa Barbara, CA, USA, August 2001.

[59] M. Bellare, O. Goldreich, and S. Goldwasser, "Incremental cryptography and application to virus protection," in *Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing*, pp. 45–56, Baltimore, Maryland, May 1995.

[60] National Institute of Standards and Technology, *P. 15 56 Standard, 1995. FIPS Pub 180-1*, National Institute of Standards and Technology, Gaithersburg, MD, USA, 1995.

[61] National Institute of Standards and Technology, *Secure Hash Standard. FIPS PUB 180-2*, National Institute of Standards and Technology, U.S. Department of Commerce, Gaithersburg, MD, USA, 2004.

[62] M. J. Dworkin, *SHA-3 Standard Permutation-Based Hash and Extendable-Output Functions*, DRAFT FIPS PUB 202, Gaithersburg, MD, USA, 2015.

[63] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche, "Sufficient conditions for sound tree and sequential hashing modes," *International Journal of Information Security*, vol. 13, no. 4, pp. 335–353, 2014.

[64] T. Bhatia and A. Verma, "Towards a secure incremental proxy re-encryption for e-healthcare data sharing in mobile cloud computing," *Wiley Online Library*, vol. 32, no. 5, p. e5520, 2020.

[65] I. Ullah, M. A. Khan, M. H. Alsharif, and R. Nordin, "An anonymous certificateless signcryption scheme for secure and efficient deployment of Internet of vehicles," *Sustainability*, vol. 13, p. 10891, 2021.

[66] A. N. Khan, M. L. Mat Kiah, S. U. Khan, and S. A. Madani, "Towards secure mobile cloud computing: a survey," *Future Generation Computer Systems*, vol. 29, no. 5, pp. 1278–1299, 2013.

[67] G. Harshit, A. Vahid Dastjerdi, K. G. Soumya, and B. Rajkumar, "The iFogSim toolkit for modeling and simulation of resource management techniques in internet of things, edge and fog computing environments," vol. 47, no. 9, pp. 1275–1296, 2017.

[68] GitHub, "GitHub - syncom/libg2hec: A genus 2 crypto C++ library," 2001, https://github.com/syncom/libg2hec.

[69] NTL, "A library for doing number theory," 2008, https://libntl.org/.

[70] M. I. GitHub miracl and S. D. K. Miracl Cryptographic, "Multiprecision Integer and Rational Arithmetic Cryptographic Library is a C software library that is widely regarded by developers as the gold standard open source SDK for elliptic curve cryptography (ECC)," 2021, https://github.com/miracl/MIRACL.

[71] PBC library - pairing-based cryptography - about, https://crypto.stanford.edu/pbc/, 2002.

[72] The GNU MP Bignum Library, https://gmplib.org/, 2010.