The Journal of Supercomputing (2023) 79:7377–7413
https://doi.org/10.1007/s11227-022-04964-9

Hybrid cryptographic approach to enhance the mode of key management system in cloud environment

Shahnawaz Ahmad[1] · **Shabana Mehfuz**[1] · **Javed Beg**[2]

Accepted: 16 November 2022 / Published online: 27 November 2022

Abstract

Cloud computing has gained great attention among the individual user and the organization. Transitioning to the cloud platform is not simple as it involves various cybersecurity and operational issues. Due to a large amount of data storage in the cloud, ensuring the security of the data outsourced in the cloud is highly important. A Key Management System (KMS) is a secure data protection method that is commonly used in e-healthcare information systems, information security (confidentiality, integrity, authenticity), large-scale organizations, architecture security, sensor security, identity management, access control (privacy preservation), identity proofing (authentication), and legal issues. Because this approach allows the secret key information to be exchanged safely, the security level may be guaranteed to be high. Using a random prime number, a master secret key, and a parameter value, one can generate a secure key that is difficult for hackers to break. Secure data transfer with exact and consistent authentication is the goal of this new approach. This research focuses on the development of secure secret key creation and the improvement in secure key sharing. To generate the key (in the form of a QR code), we have used an asymmetric Elliptic Curve Cryptography (ECC) method, whereas encryption and decryption of data have been done by a hybrid of Advanced Encryption Standard (AES) and ECC cryptography. The hybrid ECC-AES model was found to take less amount of time than the AES model and other existing models. Current algorithms have certain security issues, such as vulnerability to plaintext attacks, brute force attacks, side-channel attacks, and computational complexity. The proposed algorithm has been able to overcome the issue of key exchange that plagues AES, simpler than ECC and more reliable than AES. As a result, KMS has been designed to provide high levels of security for healthcare information. We have proposed a Hybrid Cryptographic Approach to enhance the Mode of Key Management System (HCA-KMS) in a Cloud Environment based on authenticated encryption with AES and ECC. The proposed algorithm has been compared to existing methods concerning confidentiality, integrity, time complexity, storage overhead, resource utilization, security, and log time to demonstrate its

Extended author information available on the last page of the article

efficacy. The proposed HCA-KMS has time complexity, encryption time (O (n)), and decryption time (O (logn)).

Keywords Key management system · Cloud computing · Cryptography · Cloud service provider · Key sharing

1 Introduction

In the cloud, security is of the utmost importance. Cryptography algorithms are used to protect the confidentiality of data. On a cloud storage system, data are stored and backed up in the cloud, and then made accessible to users over a network. The cloud service provider (CSP) is only responsible for maintaining, monitoring, and controlling the data once the data have been outsourced to the cloud. Using cloud services, you can access software and hardware that is hosted and maintained by a third party at a different location [1]. Online file storage, social networking, e-mail, and commercial applications are all examples of cloud services. Anywhere with an Internet connection, the user can utilize the cloud computing approach to access data and computer resources. Data storage, networks, computer processing power, and specific corporate and consumer applications are all part of the cloud computing service.

Most cloud service providers offer data storage, which allows users to access their data from any mobile device. As a result, there is a substantial chance that any device is able to access the contents stored there. Data files (such as business plans and other confidential documents) stored in the cloud that needs to be shared only with those who should have access are vulnerable to attacks entirely trusted when saved on cloud servers operated by the cloud providers [2]. When it comes to protecting data privacy, the simplest method of doing so is to encrypt the files and then transfer them to the cloud.

In terms of cloud security concerns, data security is the most important one to be taken care of. There are two types of cloud security: data at rest and data in transit [3]. When enterprises migrate data into the cloud, both types of security are required. A wide range of cloud data types exists, including user identity data, audit data, temporary runtime data, application data, and so on. Based on the nature of the data, the level of protection required varies [4]. Secure data, such as user records, necessitate high levels of protection. In most cases, a user's name and password are all that is required for privacy assurance. The user's level of security concerns varies depending on the type of data. Figure 1 depicts the data security architecture of the cloud system.

Issues with cloud data security include authentication, confidentiality, integrity, and scaling of keys (CSA 2009). Because of the multiple tenants, remote data storage, third-party cloud providers, and huge data sharing that cloud computing entails, data in the cloud must be kept private and secure. If you want to keep our personal information private, one must encrypt all of their data [5–12]. The distribution of keys and the upgrading of encryption and decryption

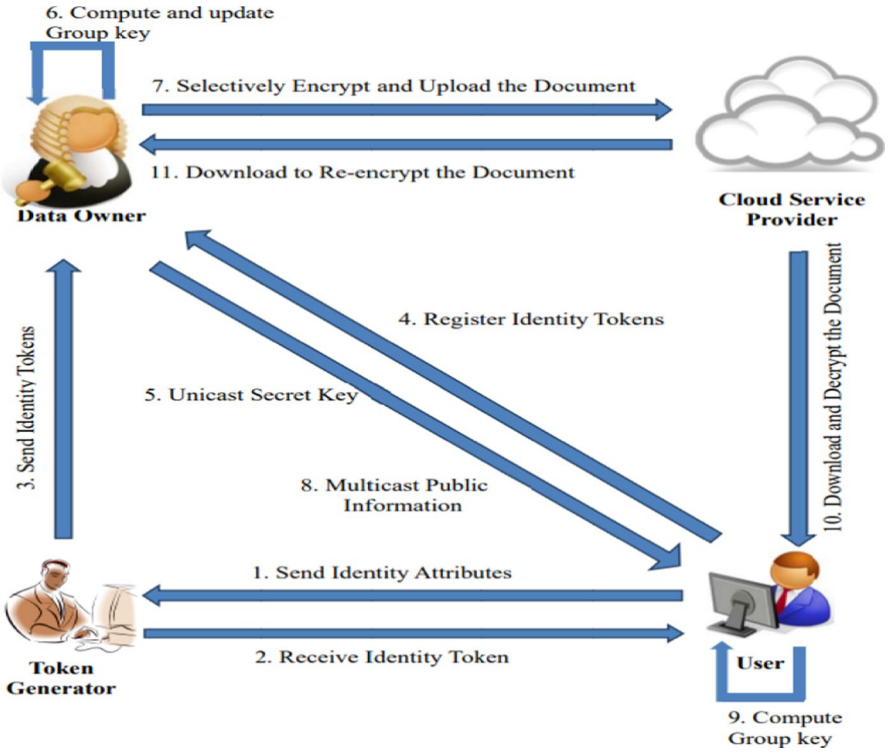


Fig. 1 Key management scheme in a Cloud environment

keys are the two most difficult aspects of encryption. There must be no tampering with the data at rest or in transit to ensure that it is secure. Data integrity must be checked both during transit and when stored in the cloud when users outsource their data. Cloud storage means that the data are more vulnerable to change by anyone who accesses it. Third-party audits and dynamic data are required to ensure the integrity of the system.

Although technology experts have explored a variety of methods for safeguarding data transfer, data encryption is the most widely used and effective method. The data are "scrambled" by encryption so that it cannot be read by someone who is not intended to read it. Data encryption entails converting information into a form that only licensed individuals with a decryption key will be able to read. Before encryption, data are referred to as plaintext, whereas after encryption, data are referred to as cipher text. Data encryption is done on purpose to protect confidential information while it is stored or transferred from one system to another. Encryption methods include Asymmetric Encryption and Symmetric Encryption. One encrypts the data with a symmetric key, while the other encrypts the data with an asymmetric key. In symmetric-key cryptography, the data are encoded and decoded using the same key, whereas, in asymmetric-key cryptography, the data are decoded and encoded using two distinct keys.

Both mechanisms have their advantages and disadvantages. The symmetric-key technique has the advantage of being rapid and easy to use. The asymmetric encryption method is a low-cost, high-efficiency strategy for securing data and uses a single safe key; therefore, it is less secure. Asymmetric algorithms, on the contrary, are sluggish and complicated, but they deliver an enhanced level of security. Two in-demand symmetric algorithms are Data Encryption Standard (DES) and Advanced Encryption Standard (AES) [13, 14]. Asymmetric-key cryptography, sometimes known as public-key cryptography, is a type of symmetric-key encryption, which encrypts and decrypts messages using unique keys. RSA, Elliptic Curve Cryptography (ECC)-based Elliptic Curve Digital Signature Algorithm (ECDSA) and the Elliptic Curve Digital Signature (ECDH) Algorithm are an examples of asymmetric encryption methods. ECC is also used in Diffie Hellman [15–18]. As a result, combining symmetric and asymmetric encryption algorithms can be one of the best potential solutions for encryption. As a result, hybrid (HCA-KMS) cryptography has been proposed in this work to ensure data security. Weak security protocols can result in the loss of major, confidential and sensitive data in different KMS applications. Data loss that cannot be recovered may occur in a worst-case scenario due to a security attack on the information. There are other consequences of weak security controls, such as high-risk network and busy server congestion, loss of user confidence, and high cost of recovery.

Data storage is the practice of making the records of an object, such as a corporation or organization, available for access and maintenance via a distributed network of cloud services that are linked to one another. The method of encryption is an essential component of the core data security strategy, and it plays an important part in the process of keeping secure contact across networks that are both consistent and scattered because the encryption method disordered the data to keep it protected by retaining "the secret." The operative of the communicator just needs a key to decrypt the data. The symmetric-key encoding method and the asymmetric-key encoding method are both used in verification procedures. During a symmetric-key encoding process, just one key is mandatory to execute both the encoding and decrypting of the data. When using asymmetric-key encryption, it is common to practice making use of two keys. Both the private key and public key are equally significant. The public key method is used for encoding the cipher text data. A second private key is obligatory to decode the information. The security of cloud computing is ensured by retaining a wide array of known and tested procedures. Cloud storing is a centralized database system tool that allows anyone to share information, substructure, and information through the Internet. Verification, anonymity, and integrity are just a few of the security concerns. Data encryption is commonly used to protect and secure data transmitted over the Internet. Several algorithms have been created to secure data and prevent hackers or offenders from exchanging information over the Internet. In this research, an HCA-KMS for the current authentication technique is proposed with an automatic key group utilizing a genetic procedure to improve the security of cloud services. According to the results, the proposed technique works well in terms of setup, encrypt, decrypting, key generation, extract, as well

as provides high levels of security for healthcare parameters i.e., log time, storage overhead, security with cloud users, and resource utilization rate, integrity, and confidentiality for files ranging in size from 100 to 1000 KB.

The main contribution of this paper is threefold:

- The AES has been modified to give better results with the help of a function (as mentioned in Sect. 3.2.1).
- The ECC has been used for decryption which incorporates the decoding function for the QR code (as mentioned in Sect. 3.2.1).
- HCA-KMS provides flexibility in terms of data file type being used for encryption and decryption before uploading the file to the cloud environment. The proposed algorithm's distinctive feature is that it allows us to encrypt any extension file into any other extension and then decrypt it to retrieve the original extension file. Every encrypted file also has a special QR code attached to it, making it impossible for a third party to decrypt the file without a QR code.

The rest of the paper is organized in the following manner. Section 2 presents a literature review of the existing work done regarding the key generation methods. Section 3 explains the existing methodology for secure data sharing and discusses the proposed methodology (HCA-KMS) for secure data sharing in a cloud environment; key assignment in the key aggregate cryptosystem which explains with the help of the private weighted sum aggregation process. Results and comparative analysis is presented in Sect. 4. Finally, Sect. 5 explains the results and is followed by the conclusion of the paper.

2 Literature review

In recent times, the arrangement of edge gadgets, from sensors and actuators to laptops and smartphones, has been rising every day globally. When such gadgets provide connectivity and a few kinds of smart applications, the complete system is termed as Internet of Things (IoT). A primary advantage of IoT is its capability to enhance original services using wide transmission along with that processing capacity and high data collection. The envisioned demand for high data collection capacity imposes IoT to rest on cloud computing (CC) for guaranteeing adequate data processing as well as required. CC [19] simplifies on-demand accessibility to a shared collection of capital, intending to allow huge storage or high processing capabilities [20]. However, CC cannot be regarded as a viable design for services with strict requirements concerning, for instance, latency for reliable e-health services or instant decision-making procedures in the industry [21]. Providing support to the security zone, authentication and key distribution are initial methods for providing secure integrity and transmission in a system.

Shanmuga Priya et al. [22] provided an improved method to the widely used information security model in the cloud. For user authentication, the proposed information safety model incorporates the group of OTPs using a Hash-based message verification code (HMAC). This study also contains a comparison of MDS5

and SHA algorithms for better performance of systems applications. This perfect is best for any of the coatings in it, and the authors do this by employing encryption methods that convert original material into a format that is incomprehensible to a third party. The proposed paradigm by Neela, K. L. et al. [23] is based on a decentralized architecture that is independent of any third-party scheme. The data security in this architecture can be improved by applying the cyclic shift transposition procedure. The authors have employed a rapid response code and a hash-based timestamp for safe data transmission and retrieval, preventing real-time assaults.

A Huffman code-based approach has been used to manage group keys as outlined in [24]. The TV channels can only be viewed by approved subscribers. The usage of Fast Fourier transforms and Euclidian techniques reduces the overall computing time. When users leave known operations, this method works well; when they leave unpredictable operations, it does not.

In [4], the authors have proposed a new way to distribute keys: via a proxy re-encryption mechanism. Even from the cloud provider, the data have been kept private by the writers. Data stored in the cloud are encrypted using a re-encryption key, which is then transmitted to the newly joined user by the cloud provider. The user gets the group key after applying decryption on receiving end. Assuming only a partially trusted server, the security level is relatively high. As a result, the revocation of the user's access to re-encryption keys and the production of new group and re-encryption keys is computationally costly.

In the proposed Certificateless Public Key Cryptography (CL PKC) [8], a third-party Key Generation Center (KGC) helps the user generate the key pair. The KGC does not have access to the user's private keys; therefore, they are safeguarded. CL-safety, PKC's on the other hand, is dependent on the KGC, and the secure channel is used to communicate.

As proposed in [25], a Cloud Key Management Client and a Cloud Key Management Server are employed in the Cloud Key Management Infrastructure (CKMI). Keys, operations on keys, and attributes stated on the keys are all part of a new key management system that has been proposed. If the CKMS fails, all of the data stored are destroyed without an appropriate backup or recovery strategy in place.

[26, 27] ECC is utilized for the encrypting and decrypting of data to deliver safe and effective services to a wide variety of users. To encrypt and decrypt data, the layered approach with two parts is utilized. The initial segment is made up of very small sections, which are used for lowering the size of the keys and adding extra bits to the process of encrypting the data. This makes it possible to gain access to information more quickly. PO through Pn is the elliptical curves that are utilized for the data encryption process in the first layer. In contrast, the elliptical curves that are utilized for the data encryption process in the second layer are divided into two groups. The processes of encrypting and decrypting the data both require all of these steps to be completed successfully. These two stages ensure that the data are secure. Data loss and security concerns have surfaced as a direct outcome of the approaches that came before them. ECC is used to mitigate the impact of these vulnerabilities by securing the data and preventing it from being compromised for unethical purposes.

Data security and augmentation of bigger datasets may be accomplished quickly and simply with this asymmetric cryptography technology, allowing for the provision of security services to be provided more quickly. ECC enables the two operations, i.e., accessing and securing data using cloud computing at the same time.

The strategies that were used by Chander Kant [28] to comprise the classification of data rendering are compassion and significance. This was followed using a diversity of cryptography methods, such as AES (which is a technique for symmetric cryptography), SHA-1 (which is a technique for hashing), and ECC (which is a method for elliptical bend cryptoanalysis) (an Asymmetric Cryptography technique). Most authors have used a single essential for both encoding and decoding from the dawn of time, leaving them vulnerable to a multitude of catastrophic attacks. This practice has been in place since the beginning of time. As a result, the hybrid technique requires the use of two distinct solutions for an individual encoding and decoding operation that is performed.

An ECC-based technique was proposed in a study that Manish Kumar et al. published [29]. This strategy was intended to improve the efficiency of DNA encoding. The RGB image is first encoded using DNA encoding, and then, it is encoded using an asymmetric encryption method that is based on the Diffie–Hellman key exchange. The photo is protected by encryption so that no one can make a copy of it using elliptic curve Diffie–Hellman (ECDHE). To evaluate the effectiveness of the suggested method, it is applied to a conventional sample collection of test images. In this study, key spaces, key sensitivity, and statistical analysis are all taken into consideration [30]. In the study, R. Balasubramanian et al. looked at the value that was obtained by multiplying two real-valued multiplicative purposes that had different inputs. This method determines the predictable number of primes in such a way that after a chance elliptical bend ended rationales have been abridged ended these primes. The resulting curve has N points, which is the same as the number of opinions on the random elliptical curve. A planning-enabled intermediary encryption solution was presented by Vijayakumar, V., et al. [31] to address the concerns regarding the level of security. Using this approach, a duly authorized agent will have restricted access to the documents for a period that has been set in advance. This strategy makes use of both a searchable encryption method and a technique known as proxy Re-encryption.

The AKM-IoV secure authenticated key management protocol was proposed by Wazid et al. [32] for use in loV placement that is associated with fog computing. After IoV transmitting entities have completed mutual authentication inside the AKM-IoV that has been configured, they create session keys for use in secure data transfers. Miao et al. [33] provide an outsourced Hybrid Keyword-Field Search on encoded data with effective Keys Management (HKFS-KM) technique by using a keyed hash tree and a suitable score function. This technique searches for keywords and fields simultaneously on encoded data. To address the vulnerabilities in the protocol's security that Shen et al. found, Park et al. [34] presented a key agreement mechanism for V2G in SIoT that is dynamic, protects users' privacy, and uses a minimal quantity of capital. The proposed protocol offers protection against many different categories of attacks, such as trace attacks and impersonation. It also guarantees

Table 1 A summary of the key management protocols (KMP) involved

KMP	Applied operation	Drawback
Choi et al. [35]	SHA-160, XOR, and ECC	SV attacks, Exposed to ESL, SSD/SSC, and forgery
Nikravan et al. [36]	SHA-160, BP, and XOR	Unprotected from UI and DI attacks
Choi et al. [37]	SHA-160, XOR, and ECC	SSD/SSC, SV attacks, UI, Unsafe against MITM, and forgery
Wazid et al. [38]	XOR, SHA-160, and ECC	Cannot withstand a DS assault
Jiang et al. [39]	SHA-160, ECC, and XOR	SSD/SSC, detection of unauthorized login, cannot defend against DoS, ESL attacks, and RP isn't available
Ali et al. [40]	XOR, SHA-160, and AES	MITM, mutual authentication is not possible, it's risky to pretend to be someone else, and SSD/SSC attacks
Jung et al. [41]	SHA-160, AES, and XOR	Vulnerable to User Interface, forgery attacks, SSD/SSC, PI, and PBC
Sadhukhan et al. [42]	ECC, SHA-160, and XOR	Unsafe in the presence of UA, does not obviate the need for a password-changing mechanism, replay, DoS attacks, and MITM
Moon et al. [43]	ECC, XOR, and SHA-160	SSD/SSC attacks, forgery, cannot withstand ESL, and UI
Challa et al. [44]	SHA-160, XOR, and ECC	DoS attacks, forgery, cannot withstand UI, and replay
Li et al. [45]	ECC, XOR, and SHA-160	UI attacks, Unsafe in the presence of PG, PI, replay, and Features such as RP and untraced ability are not affected
Mostafa et al. [46]	ECC, XOR, and SHA-160	It is not possible to give a revocation feature
Wu et al. [47]	ECC, XOR, and SHA-160	does not result in an effective PBC mechanism, DoS attack, and UI
Tanveer et al. [48]	ASCON, SHA-256, and XOR	does not result in an effective PBC mechanism
Li et al. [49]	XOR, SHA-160, and ECC	aspects of untraceability, Vulnerable to PG, RP is not rendered, and PI
Shuai et al. [50]	ECC, XOR, and SHA-160	Susceptible to UI, parallel session attacks, PG, UI, and SSD/SSC
Mahmood et al. [51]	SHA-160, ECC, and Pairing	ESL and impersonation attacks are a threat
Wazid et al. [52]	SHA-160, XOR	DI attacks, Susceptible to user interfaces, and IG
Jia et al. [53]	XOR, ECC, and SHA-160	MA is not attained, UI, replay attack, SV, and PFS
Lu et al. [54]	SHA-160, XOR	ESL and IG attacks are ineffective

Table 1 (continued)

KMP	Applied operation	Drawback
Chen et al. [55]	SHA-160, ECC, and XOR	Unsafe in the presence of SSD/SSC, Anonymity and MA characteristics are not rendered, PG, DoS attack, PI, and replay
Yang et al. [56]	SHA-160, ECC, and XOR	Doesn't protect devices from SV attacks and doesn't give device anonymity
Majid et al. [57]	XOR, SHA-160, AES	SSD/SSC, SV, ESL, and PFS assaults are all ineffective
Shin et al. [58]	SHA-256, XOR	De-Synchronization (DS) attack is possible, and there is a design weakness
Zhou et al. [59]	SHA-160, XOR	Replay, UI, SK disclosure assaults are all possible. And MA is not achieved

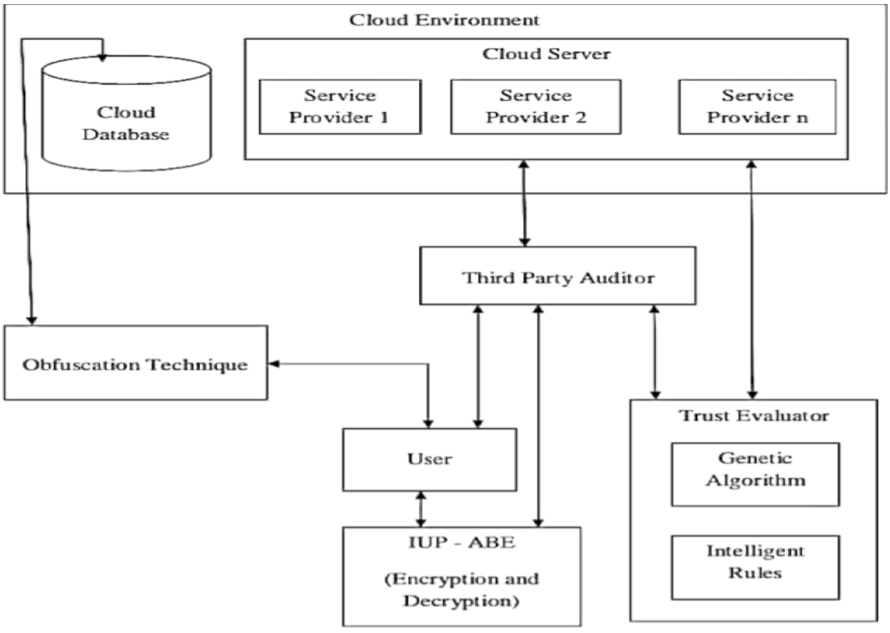


Fig. 2 Existing methodology for secure data sharing [24]

anonymity, session key safety against man-in-the-middle attacks, replay secure mutual authentication, offline password guessing, and perfect forward secrecy.

However, as shown in Table 1, the majority of KMP protocols are vulnerable to a variety of security threats such as User Anonymity (UA), Ephemeral Secret Leakage (ESL), Denial-of-Service (DoS), Privilege Insider (PI), Password/Biometric Change (PBC), User Impersonation (UI), Device Impersonation (DI), Man-In-The-Middle (MITM), Stolen Smart Device/Card (SSD/SSC), and Password Guessing are just a few examples (PG). We propose an HCA-KMS based on authenticated encryption with AES and Elliptic Curve Cryptography in this work (ECC).

Figure 2 illustrates how data integrity preservation and secure data sharing in the public cloud are accomplished in. The key generation in this proposed obfuscation approach is not regarded to be a separate process because the key is retrieved from the plain text that is being obfuscated [24]. The key for ciphering the plain text before this one is generated from the plain text before this one. To find the most essential terms in a text, it is vital to incorporate this language processing technology. Their primary goal is to avoid using a repeating key, such as the one used in the Vigenere cipher.

The full text was chunked into characters using a language processing technique, to avoid pressing the same key twice. In the case of plain text, each ten-word block is classified as a segment, and the full plain text is broken down into individual words [25, 60–63]. The first five words of a segment have been encrypted with the help of the five words. If we compare the lengths of characters, we may observe that they are not all created equal, as previously said.

In these instances, the approach of repeated keys developed by Vigenere has been employed.

Under the contributory key management strategy, each member of the group makes a small contribution to the establishment of the group's key. These fundamental management strategies can be implemented using both non-tree models and tree-based models, depending on the situation [64]. Tree-based models are frequently encountered in the literature on GKM. In the case of tree-based models, each leaf represents a user, and every leaf-to-root node represents an auxiliary key for that individual. The design of the structure includes a hierarchy among its constituent pieces, which is inherent in its construction. By arranging keys logically, a simple rekeying procedure can be achieved.

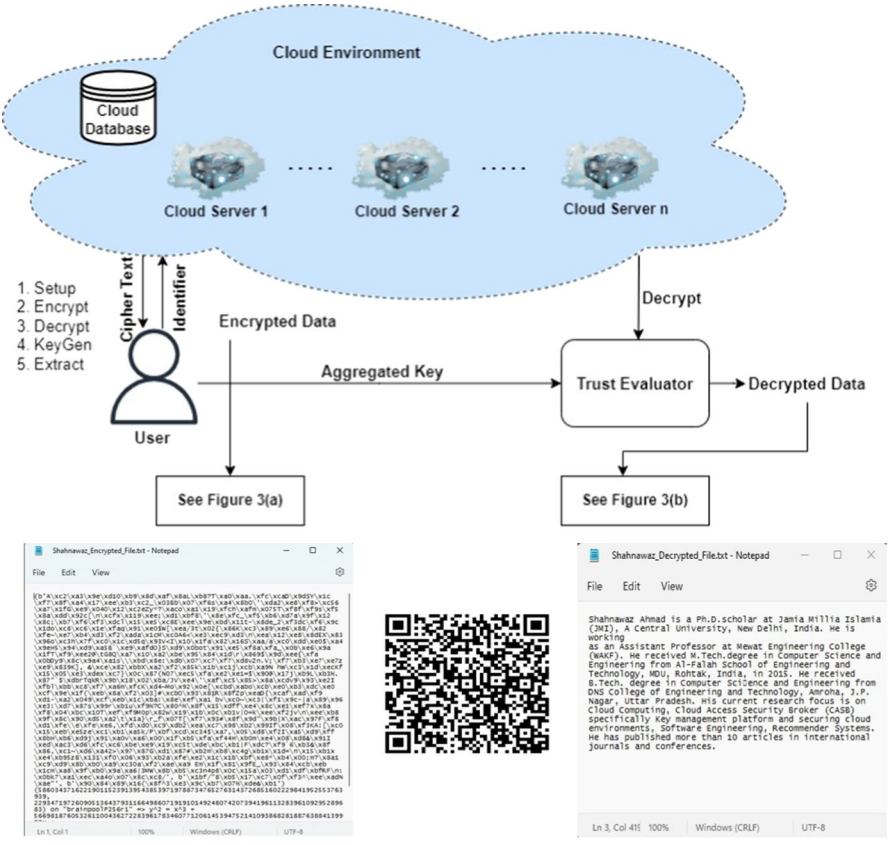
When a person joins or departs an organization, a new key must be supplied to all of the organization's members. In GKM, this is accomplished by the distribution of new keys. According to academic literature, hierarchical tree structures are used to manage and distribute keys to keep track of who has what [65]. A technique for key management is given that makes use of a compressed binary tree to address the task at hand. Because of its capacity to address the core problem of key management scalability, the proposed Cryptographically Transferred General Key Management (CTGKD) technique is ideal for cloud systems with a large number of users. Communication, processing, and key distribution costs are all decreased as a result of the CTGKD technique's implementation. This is a win–win situation. GC is originally employed to construct a tree-based group in which it serves as the root node. Those who join the group are automatically inserted as the left child if their user id begins with zero or one, respectively, and as the right child otherwise. It is necessary to address the user's departure from the group by removing him or her from the group and finishing the key updating operation.

3 Proposal for enhancement of KMS

Using the Internet and a remote server, cloud computing has exploded in popularity as a means of storing and accessing data. Physical resources are not owned by clients, but rather are leased from a third party. The most common methods for managing users and keys are user setup, key development, expiration, and destruction. Businesses are concerned about data security because of the amount of sensitive information they send over the Internet [66–71]. Any system that deals with cryptographic keys can be said to be using this term. It entails the creation, distribution, processing, and use, as well as the destruction and replacement of keys during crypto shredding. Cryptographic protocols, key servers, client procedures, and other related protocols are all included in this. Users or systems may share keys at the user or system level. In contrast to key preparation, which relates to the management of keys in the cipher's internal functioning, this is not the case here. The safety of a cryptosystem depends on its ability to effectively manage its keys. Rather than relying solely on computation, cryptography encompasses features of social technology such as system processes, user training, organizational and departmental interactions, and so on.

Protecting sensitive and private information from unauthorized access is part of security. To ensure the safety of the data, it must be kept confidential, protected from unauthorized access, and limited in integrity so that only authorized individuals to have access to it when it is needed (availability). The discovery of key cryptography in the field of cryptography appears to be ground-breaking. Key cryptography can be used for both encryption and secrecy [72]. Public key cryptography, such as RSA, is one such method.

New and improved public-key cryptography key management systems have been proposed. This method encrypts the data using the user's key and then uploads it to the cloud, making it secure. To make the KMS (Secure Key Management System) highly secure, the encryption (E) and decryption (D) processes rely on key creation. In contrast with standard key management systems, KMS cryptanalysis employs a time-consuming method of analyzing keys.



(a) Encrypted Data using HCA **(b)** Decrypted Data using HCA

Fig. 3 HCA-KMS a proposed methodology for secure data sharing in a cloud environment **a**: encrypted data using HCA, **b**: decrypted data using HCA

Figure 3 depicts the HCA-KMS, the proposed methodology for secure data sharing in a cloud environment. In the proposed methodology, the user first encrypts the file using the proposed HCA-KMS. After the successful encryption of the file, a QR code is generated and sent to the registered email address of the existing user. Then, the encrypted file is uploaded to the cloud for the client and a link of the same is shared with the client along with the generated QR code in the.png extension which acts as the KeyGen for the decryption of the file. An HCA algorithm-based encryption process is shown in Fig. 3a in which a file (of any format) has been uploaded using a proposed algorithm to be encrypted. A QR code is generated at the time of the decryption process in the form of KeyGen by which a file of any format is successfully decrypted using the proposed HCA algorithm (as shown in Fig. 3b). From the perspective of implementation, a block diagram of secure data sharing in a cloud environment is depicted in Fig. 4. Figure 5 shows the encryption process using the HCA algorithm while the decryption process is shown in Fig. 5. Before encrypting the file, double-check the file's format to validate the experiment's findings. The file can be opened with ease before encryption, but it cannot be opened after encryption.

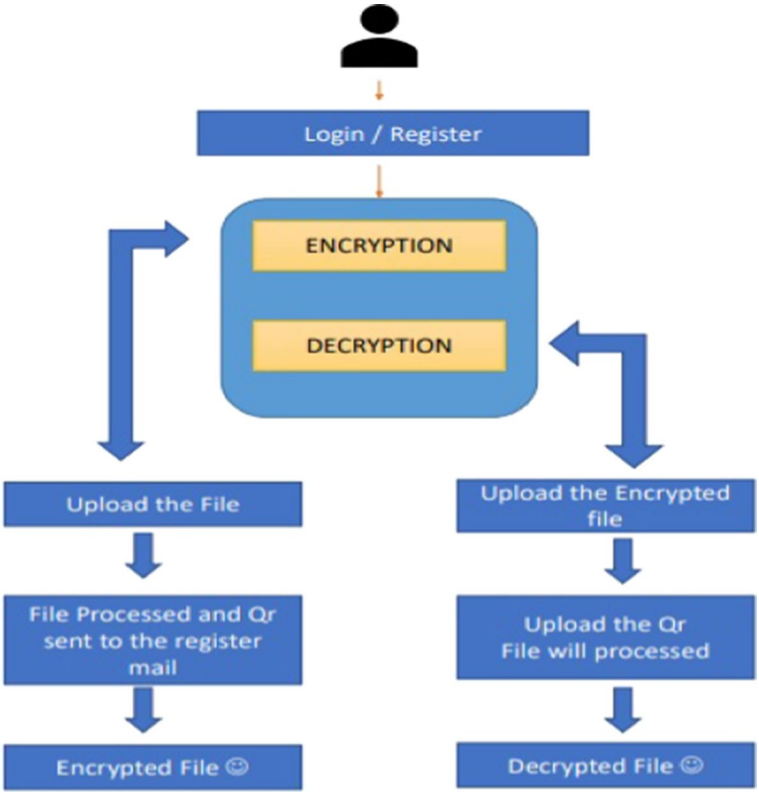


Fig. 4 Block diagram of proposed secure data sharing in a cloud environment

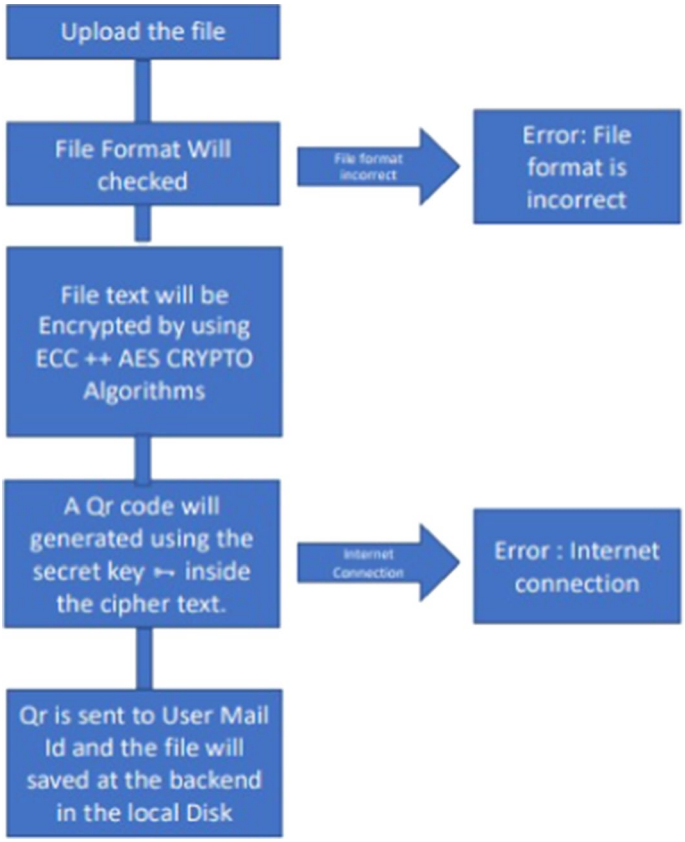


Fig. 5 HCA algorithm-based encryption process

Similarly, utilizing the proposed algorithm to decrypt the encrypting file will yield the identical original file (Fig. 6).

3.1 Phase of proposed HCA-KMS

Different phases (i.e., setup, encrypt, keygen, decrypt, and extract) of the proposed HCA-KMS are incorporated in Fig. 3. A description of the same is given below:

a. *Setup (N, 1^n)*: The data owner or cloud server (Cloud Server 1, Cloud Server 2, …. Cloud Server n) runs this algorithm first. It outputs the public parameters when given a security parameter 1^n and the number of files N.
b. *Encrypt (Pub_k, i, w, Select file(), Encrypt_file(), Encrypting(), Qr_Manage())*: The data owner is in charge of this algorithm. It generates the keyword ciphertext (Ci) for each file index (i) and keyword (w).
c. *KeyGen*: The data owner uses this procedure to generate one pair ($public_{key}$, $Master_{key}$), which consists of a public key and a master key.

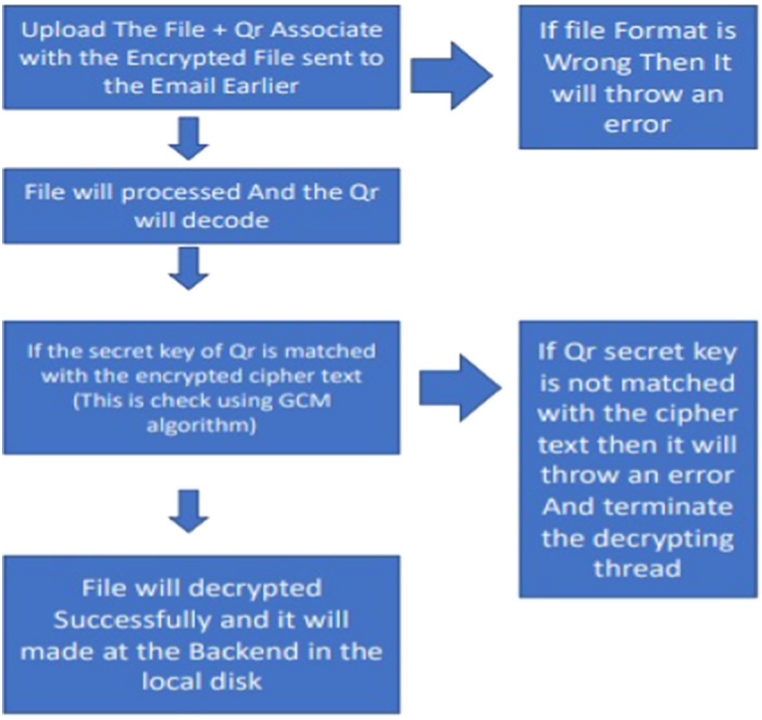


Fig. 6 HCA algorithm-based decryption process

d. *Decrypt* ($Priv_{Key}$, *Choose file(), Select _ qr(), Decrypting(), decrypt_text()*): Trust evaluator received the decrypted file from the cloud environment with the help of a QR code.
e. *Extract* (Pub_k, $Master_{Key}$, $Files_S$): The data owner is in charge of this algorithm. It generates the aggregate key KS from the data owner's pub_{key} $master_{key}$ and a set S of file indices. The data owner then sends the authorized user the aggregate key KS and the set S.

Steps a, b, and c are contemplated with the help of enhanced AES which incorporates a function for QR code generation (DesfQr_manage (string) referred from a subsection of 3.2.1). The steps d and e have been carried out with the help of an enhanced ECC algorithm which incorporates the function for decoding the QR code.

3.2 Encryption and decryption mode of HCA-KMS

A cryptographic key management system, commonly known as the Key Management Method (KMS), is an automated system for creating, transmitting, and preserving cryptographic key management schemes (CKMS). Everything from key creation to key exchange to consumer processing is handled securely in this system.

In a KMS, not only are the back-end functions of generating and delivering keys contained, but so are the user's capabilities for injecting, storing, and altering device keys. Scalability, protection, dependability, diversity, and governance are just a few of the issues that IT departments must deal with while trying to manage and preserve their encryption keys. Whether you're securing your home or your information, a key is a vital component of any security strategy. The encryption key can be used to protect sensitive information from being accessed by other users. If the application has a large number of users, the key management system must generate and distribute a unique key for each user [73]. To effectively manage a key, a variety of strategies and approaches can be used to achieve success. Basic principles of cloud governance have been broadly established in recent years.

For private weighted sum aggregation using secret weights and secret information, this paper addresses the problem of aggregator programs that attempt to compute the weighted sum of local data for a collection of agents. Based on the literature review, we analyzed an existing method for private weighted sum aggregation and then offer a unified strategy. Hopefully, you have found this paper helpful in choosing the best solution for your knowledge dissemination and privacy needs. In Fig. 7, the private weighted sum aggregation process is shown.

Cryptographic keys are generated, stored, and exchanged through key management. Ensure that this is done safely to avoid assaults like the man-in-the-middle attack [66]. Symmetric and asymmetric encryption is orthogonal to key management [74]. It is anticipated that cryptographic key management is implemented and integrated, which means that the keys are assumed to be safely kept, retrieved, and used.

Because encryption prevents unwanted access to the original information and prevents it from being seen by others, it provides security. The encryption algorithm relies on computational complexity, making it nearly impossible to decipher. As a countermeasure to specific dangers, encryption and key management techniques are not a panacea for data security, but they can be considered acceptable in some instances anyhow. Encryption can assist both the consumer and the service provider keep their data safe in the cloud. Aside from that, encryption is frequently mandated by law or may be useful to comply with it. Although encryption does not prevent data from being lost or compromised in and of itself, the concept of encryption makes it considerably more difficult for anyone other than the intended recipient to make use of the data in any manner.

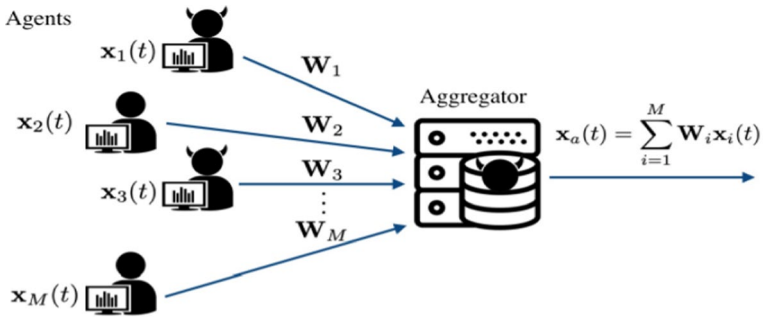


Fig. 7 Private weighted sum aggregation process

Key management issues must also be taken into account when contemplating the encryption process in order to ensure that the keys utilized are not compromised. Mobile devices used to access the cloud often go missing, putting the security of any keys stored on them at risk. This is especially true if keys are saved on those devices. When dealing with key management issues, it is necessary to evaluate the keys' whole life cycle, from the time they are created until they are no longer needed to protect the data they protect. Key management and associated concerns will be covered in this paper. Cryptosystems use a process called "key management" to regulate all aspects of keys, from creation to destruction. It has several safeguards and protocols in place to ensure that only authorized personnel have access to the keys at all times. The most difficult component of cryptography, however, is "key management." Key management policies must be in place to ensure that data encryption operations are safe and secure [75].

A new type of public-key encryption called KMS is introduced. Messages are encrypted in KMS not just with a public key, but also with a ciphertext identity known as a class. The key owner is the sole holder of a master secret key, which is what it is called. For different classes, it can be used to extract secret keys. Class-specific aggregate keys match extracted keys that may be secret keys for the class in question. Large numbers of such keys are at its disposal. However, the decryption power for any subset of ciphertext classes is multiplied by the power of multiple such keys. A key assignment is a key aggregate cryptosystem as shown in Fig. 8.

The following steps are considered for executing the KMS Encryption.

Step 1: In setup, the data owner sets the public system parameter that is accessible to all users.

Step 2: KeyGen is used to create a public and a master secret key pair.

Step 3: Emails are protected by Encrypts security measures.

Step 4: The plaintext communication that is to be encrypted is referred to as ciphertext.

Step 5: This decryption key is generated by extracting a set of ciphertext classes and the owner's master secret.

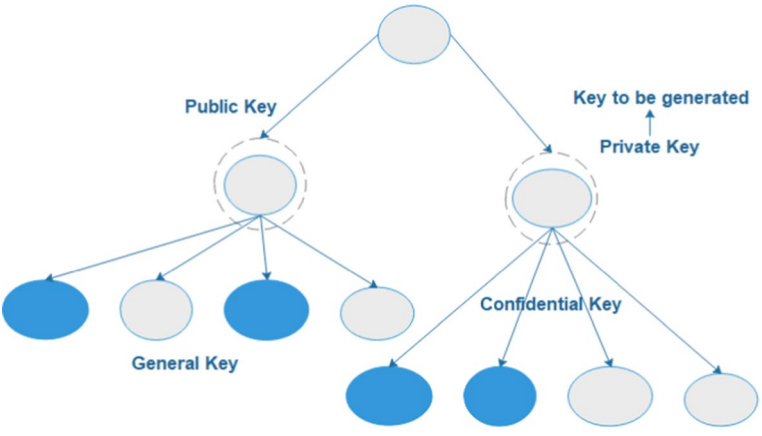


Fig. 8 Key assignment is a key aggregate cryptosystem

Step 6: A secure email or secure channel is used to distribute the generated keys to delegates.

Step 7: It is possible to decrypt any ciphertext with an aggregate key, provided the ciphertexts class is included in the aggregate key.

Step 8: Aggregation of ciphertext is calculated using the mentioned key management system algorithm.

Step 9: Authenticate the authentic user using the aggregation of signature.

Step 10: After getting the authenticated user, verify the user by using the verification process.

Step 11: The algorithm terminates or stops, once all the steps are successfully executed.

Following is the algorithm for the proposed HCA-KMS strategies:

The proposed algorithm takes up the following:

- The pseudo-code generates a class encryption, which the user can use to encrypt data using an encryption method. The proposed algorithm is also known as a digest algorithm, which digests the data input from the user and generates key-based data content. A key (qr code) is used to generate the decryption and it is received from the key generation and management service. Users are not required to keep the key on them. They can securely delete this key.
- The upload query will appear after the successful login process of the user. The user can upload the data in the form of files (of different extensions) by selecting the yes option. By selecting the No option, the user will exit the uploading phase.

3.2.1 Pseudocode of data encryption

Following is the algorithm for the proposed HCA-KMS strategies:

Key Management System Algorithm

1: Start
2: Inputs: Key length, Initial seed
3: Outputs: Public Key (pu), Private key (pr)
4: Generate 2 different Prime Numbers say 'P' and 'Q'.
5: Calculate the modulus
N = P * Q /* Modulus (N) */
6: Calculate the public key
Pu ← [E, N] /* public key (encryption exponent)*/
7: Calculate the private key
Pr ← [E, N] /* private key (encryption exponent) */
8: Aggregation of Cipher Text
$\hat{c} = \prod_{i=1}^{n-1} g^{m_i+nr_i} = g^{\sum_{i=1}^{n-1} m_i+mr_i} = g^{\sum_{i=1}^{n-1} m_i}$ /* Cipher Text Aggregation */
9: Aggregation of Signature
$\hat{\sigma_i} = \sum_{i=1}^{n-1} \sigma_i$ /* Authentication by Signature*/
10: Verification
$e(g_1, \hat{\sigma}) = \prod_{i=1}^{n-1} e(v_i, h_i)$ /* Verification process verified */
11: Stop

The following is the pseudocode for steps a, b, and c in the proposed HCA-KMS provided in 3.1 sub-section.

Class Encryption:
Methods:

Select file().
Encrypt_file().
Encrypting().
Qr_Manage().

Working of File Selection:

1. Set file path = select file()
2. Try:

   a. encrypt_file

   Except:

   b. Display Error Of internet or File failure
3. Encrypting()
4. Qr_manage()

Def Select _file()

1. Try:

   a. Open tkinter file dialog box and choose the path file
   b. Set path = chosen file

   Except:
   Show error

Def Select _file()

1. Try:

   a. Save_path = Open tkinter file dialog box and get the location for saving the encrypting file.
   b. Thread = Encrypting()
   c. Start Thread

2. Except:

   a. Display Error On gui

Def Encrypting()

1. File = Open(path)
2. Set content = Read file text
3. Set encrypt_message = encrypt_text(content)
4. Set secret key = Secret_key(encrypt_message[4])
5. Try:
   a. Qr_manage(secret ley)
   b. File = open(save path)
   c. Write content to file
   d. Display Done message on GUI
6. Except:

Display Error Message

Desf Qr_manage(string)

1. Qr = Make_Qr(string)
2. Save Qr with Security
3. Set mail = user-id
4. Try:
   a. Send message(mail,subject,message,Qr)
   b. Os.remove(Qr)
   c. Return
5. Except:

Os.remove(Qr)

Display error

3.2.2 Pseudocode of data decryption

The following is the pseudocode for steps a, b, and c in the proposed HCA-KMS provided in 3.1 sub-section.

Class Decryption:
Methods:

1. Choose file()
2. Select _ QR()
3. Decrypting()

4. decrypt_text()

Working of Choose File for Decryption:

1. File = Choose_file()
2. Qr_path = select_qr()
3. Try:

Decrypting ()

Except:
Display Error

Def Choose file ()

1. Set File_path = open tkinter file dialog box to take the path of the encrypted file

Def select_Qr()

1. Qr_path = open tkinter box to take the Qr image path associated with the encrypted file

Def Decrypting ()

1. Thread = Thread(decrypt_text)
2. Thread.start()

Def decrypt_text()

1. If file_path = = None

Display Error
Display Choose file

2. Else if Qr_path = = None

Display Error

3. Else:

   a. Try:
   b. File = Open(file_path)
   c. Content = file.read()
   d. Encode content to bytes using ast module
   e. Content = ast.literal(content)
   f. Decode the Qr_ code
   g. Dec = decode(Qr_path)
   h. Decrypted text = decrypt message(content,Dec)
   i. File = open tkinter box and ask for saving the file
   j. Save the decrypted text into the file

4. Except:

   a. Display error

Display "Unable to decrypt the file"

3.3 Experiments and analysis of the proposed scheme

The experimental results are presented in this section, along with an analysis of the findings. On operating systems (7, 10, 11), Linux, and Mac with Intel i7 pre-processors, minimum processor i3/Rygen 3, and with 30 GB storage and 3 GB < RAM, the proposed technique was implemented in Python version (3.9 and above). The modules of Python used in the implementation process are ast, os, JSON, random, string, threading, kivy, tkinter, data time, QRcode, pyzbar, and PIL.

3.3.1 High levels of security for healthcare information

Patient information like confidentiality, integrity, resource utility, security with respect to cloud users, and log-time comparison is currently the most critical challenges in the healthcare sector for establishing excellent patient care standards. We tested it on a set of healthcare users joining and leaving dynamically and confirmed it by applying it to virtual servers, demonstrating that the healthcare network system's rekeying overhead, threats, and computing cost are decreased when compared to existing schemes. Table 2 depicts the difference in time required to securely communicate information about varying numbers of healthcare users. The relationship is therefore expressed as follows: Number of cloud users (N) Time taken (T) (i.e., $N = pT$, 0 k1), where p is a smoothing constant and $p = N/T$.

Table 3 contains user identity information such as first name (Fn1, Fn2,..F10), last name (Ln1, Ln2,…..LnN) e-mail address, user name, and password. When the number of users exceeds the amount of Umax, the server split is shown in Table 3. Local hosts are used to identify the servers. With Umax = 20 for each server, Table 3

Table 2 Analysis of proposed HCA-KMS healthcare information without key server approach

No. of cloud users (N)	Data of healthcare (D) in KB	Time (T) in Seconds	Constant smoothing ($p = N/T$)
1	10	9	0.11
2	15	10	0.2
3	20	12	0.25
4	25	14	0.28
5	20	40	0.125
6	15	35	0.171
7	20	80	0.0875
8	10	140	0.5714
9	40	160	0.05625
10	60	180	0.05555

Table 3 Healthcare user details along with master key and virtual server

First name	Last name	Email address	User name	Password	Server (virtual)	Master key
Fn1	Ln1	abc@xyz.com	abc	abc@123	Localhost:8080	23,412,387
Fn2	Ln2	def@xyz.com	def	def@123	Localhost:8080	86,723,516
Fn3	Ln3	ghi@xyz.com	ghi	ghi@123	Localhost:8180	98,124,657
Fn4	Ln4	jkl@xyz.com	jkl	jkl@123	Localhost:8180	14,236,475
Fn5	Ln5	mno@xyz.com	mno	mno@123	Localhost:8080	32,653,185
Fn6	Ln6	pqr@xyz.com	pqr	pqr@123	Localhost:8080	41,632,498
Fn7	Ln7	stu@xyz.com	stu	stu@123	Localhost:8080	13,427,584
Fn8	Ln8	vwx@xyz.com	vwx	vwz@123	Localhost:8280	51,273,465
Fn9	Ln9	yza@xyz.com	yza	yza@123	Localhost:8280	76,136,542
Fn10	Ln10	bcd@xyz.com	bcd	bcd@123	Localhost:8280	86,543,613

is obtained. As a result, when the 21st user joins a server's group, the healthcare user is immediately assigned to the next server. Users 1 to 20 are on Server 1 with the local host 8080, and new users are routed to the next server with the local host 8280. Similarly, when users leave the group on a whim, the servers rebalance their workload.

The data shown in Table 4 show that when the number of healthcare users grows, the time it takes for them to communicate with one another decreases from seconds to milliseconds.

For patient enrolment in each semantic group, the completeness of sample electronic patient data is computed and summarized in Table 5.

Login verification and encrypted transmission ensure the security of healthcare data. The login file can be used to link a specific person to a given transaction, hence increasing accountability. Furthermore, if the number of users and group size (N groups) grow rapidly, the key server is separated into many virtual key servers for quick access to healthcare data and to improve security in sensitive group

Table 4 Analysis of proposed HCA-KMS healthcare information with key server approach

Secure group communication has a computational cost (in milliseconds)						
Healthcare data (D) in bytes	Users group size					
	5 User	10 User	15 User	20 User	25 User	30 User
5	30	56	70	83	103	189
10	42	67	124	173	184	280
20	139	237	272	315	395	450
30	226	335	433	445	470	555
200	423	651	705	820	978	1095
400	561	814	898	925	1045	1198
500	651	887	555	1165	2015	1456
800	793	954	1005	1845	3055	2300
1000	842	1006	1255	2230	3095	3235
1500	973	1130	1735	2895	4095	3298
2000	1065	1250	2075	3505	5025	4518
2500	1115	1678	2188	4290	6035	5198
3000	1285	2035	2498	5110	6845	6189
3500	2557	2754	2945	7245	9925	1360
4000	4129	6984	7088	9985	1278	4578

communication. The proposed HCA-KMS mitigates the risks to the greatest extent practicable. As a result, the proposed scheme can be organized into a powerful solution that allows for fast, safe access to any type of healthcare network application, such as Telemedicine and Healthcare Informatics. For key generation, encryption, and decryption of healthcare materials, the proposed approach employs a master key encryption algorithm. Furthermore, if the number of users grows rapidly, the key server is split into numerous virtual key servers to provide faster access to healthcare data and to improve the security of the group-organized healthcare network system. The results show that rekeying overhead is lowered while communication time (reduced from seconds to milliseconds) is reduced securely with less time complexity and communication overhead. Surgical treatments are increasingly being performed remotely by physicians. Disturbing this situation could put the patient having the treatment in danger. As a result, the proposed system is compatible with telemedicine and e-healthcare information system correctness and has dependability in providing secure and fast access to patient data.

4 Results and comparative analysis

The results of the experiments are then compared to those of other studies to make sure they are valid. The present approach is matched with a certain parameter's base to arrive at an estimate of research performance. A large amount of security is provided for healthcare information through the proposed enhanced version of

Table 5 Evaluation of the completeness of a sample of electronic patient records after network access

Patient sample	Patient characteristics that can be documented in a fraction of the time (X) (ms)	The average percentage of patients that have data documented (Y)	Patient data completeness ($Z=X*Y$)
Patient age	0.9	0.92	0.828
Ethical concern	0.08	0.48	0.0384
Patient characteristic	0.38	0.69	0.2622
Therapy or surgery	0.72	0.48	0.3456
Literacy	0.36	0.19	0.0684
Pharmaceutical medication	0.39	0.2	0.078
Sex	1.2	0.98	1.176
Health care treatment	0.59	0.46	0.2714
Health status	0.65	0.75	0.4875
Diagnostic laboratory results	0.59	0.39	0.2301
Disease and symptom	0.87	0.69	0.6003
Diagnostic laboratory test	0.59	0.37	0.2183
Pregnancy-related activity	0.32	0.5	0.16
Addictive behavior	0.68	0.7	0.476
Disease stage	0.33	0.51	0.1683
Lifestyle option	0.83	0.81	0.6723
Allergy-related problems	0.29	0.69	0.2001
Condition of organ	0.072	0.8	0.0576

KMS. Log time, storage overhead, security of cloud users and resource utilization, integrity, and confidentiality are all factors that are examined when determining performance.

Data sharing is a common use of the key-aggregate cryptosystem. When the delegation is expected to be efficient and adaptable, the key aggregation attribute comes in handy. Using these techniques, a content provider can securely and selectively distribute her material to approved users while maintaining complete control over the ciphertext expansion.

To judge a node that has only interacted with it three times, for example. It may, however, have 100% confidentiality in a goal node with which it has interacted 100 times. We offer a statistical trustworthiness metric. Higher levels of confidence result in higher levels of direct trust; lower levels of confidence need a computation of trust values based on both direct and indirect trust values. The error level influences a node's direct trust calculation for another node. The error between the nodes can be calculated using Eq. 1, where x and y are the variables of γ and c and β are the constant values for p and U.

$$\gamma_{x,y} = \frac{\int_{dt_{x,y+\varepsilon}}^{dt_{x,y-\varepsilon}} (1-p)^{\beta-1} \mathrm{d}p}{\int_0^1 \cup^{\propto-1} (1-\cup)^{\beta-1} \mathrm{d}\cup} \tag{1}$$

4.1 Confidentiality

Figure 9 depicts the results of comparing the methods in practice with the proposed method in terms of secrecy. The *X*-axis shows the delegation ratio, while the Y-axis shows the confidentiality ratio. The delegation ratio refers to the proportion of delegated cipher text classes to the total number of classes. Dual encryption with an identifier is popular because of the methods proposed in an earlier technique. KMS,

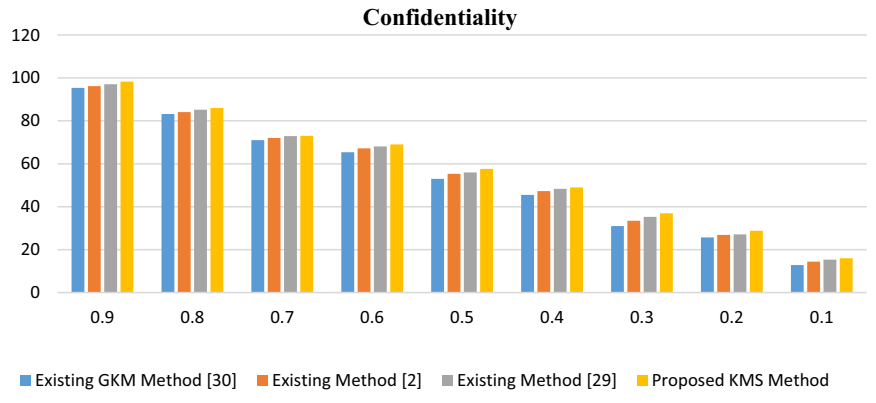


Fig. 9 Plot for confidentiality comparison for the existing and proposed method

Table 6 Comparison table for the confidentiality

Delegation ratio (%)	GKM 1 (%) [67]	GKM 2 (%) [2]	GKM 3 (%) [76]	Proposed KMS method
0.9	95.4	96.2	97.1	98.3
0.8	83.2	84.1	85.2	86
0.7	71	72	72.9	73
0.6	65.4	67.2	68.1	69
0.5	53	55.3	56	57.6
0.4	45.5	47.25	48.32	49
0.3	31	33.45	35.25	36.9
0.2	25.7	26.85	27.12	28.8
0.1	12.8	14.4	15.36	16

a new method for safely transferring keys, does just that. Results from real-world applications show that the KMS methodology delivers maximal confidentiality when compared to GKM methods. Table 6 shows the comparison between the existing GKM methods with the proposed KMS method. The delegation ratio can also be calculated using the below-given equation.

$$\text{Delegation Ratio } (\%) = \text{Discretion}/(1 - \text{Constraints}) \tag{2}$$

Discretion means revealing confidential information and constraints means a limitation.

Table 7 Comparison table for the integrity

Delegation ratio (%)	GKM 1 (%) [67]	GKM 2 (%) [76]	GKM 3 (%) [2]	Proposed KMS method
0.9	94.4	96.35	97.1	98.3
0.8	82.2	84.1	85.7	86
0.7	73	75	75.5	76
0.6	64.4	66.2	67	68
0.5	51	54.3	55.6	57.6
0.4	44.5	46.25	47.3	48
0.3	35	36.95	37.7	38.9
0.2	26.7	28.25	29	29.8
0.1	18.8	19.9	20.8	21

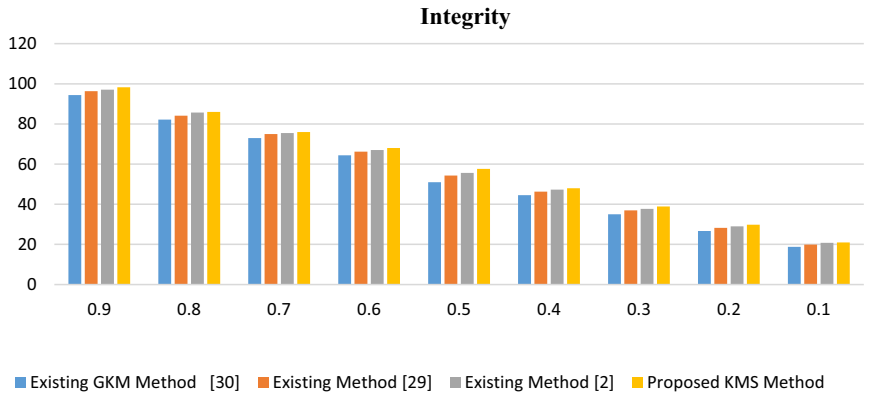


Fig. 10 Plot for integrity comparison for existing and proposed method

4.2 Integrity

Using Table 7, we compare the proposed methods' integrity values to those of the existing methods. A delegation ratio of 0.9 results in 94.4 percent integrity, according to KMS. Figure 10 depicts the comparison of the current approach's integrity results with those of the new KMS technique. The *X*-axis shows the ratio of delegated work to total work, and the Y-axis shows secrecy. The delegation ratio refers to the proportion of delegated cipher text classes to the total number of classes. Dual encryption with an identifier is a popular method that has been around for a while. KMS, the technique under consideration, makes use of secret key management. According to practical results, the new KMS approach achieves its maximum integrity in comparison to existing methods.

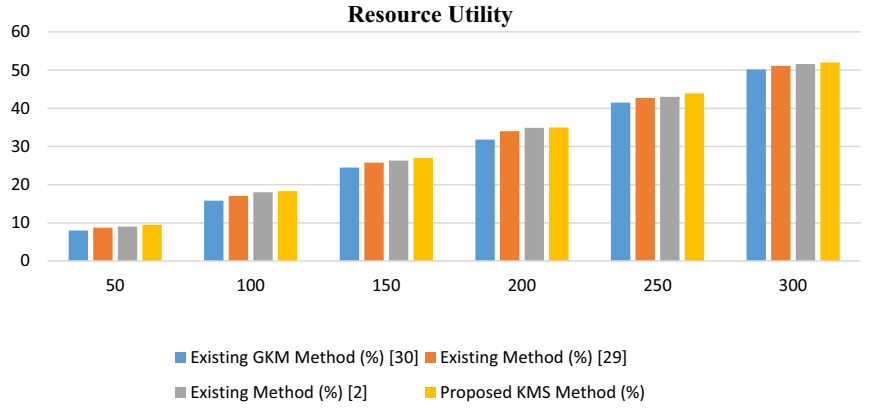


Fig. 11 Plot for resource utility comparison for existing and proposed method

Table 8 Comparison table for the resource utility in terms of task scheduling

No of tasks	GKM 1 (%) [67]	GKM 2 (%) [76]	GKM 3(%) [2]	Proposed KMS Method (%)
50	8	8.75	9	9.5
100	15.8	17.05	18	18.3
150	24.5	25.75	26.3	27
200	31.8	34	34.9	35
250	41.5	42.7	43	43.9
300	50.2	51.1	51.6	52

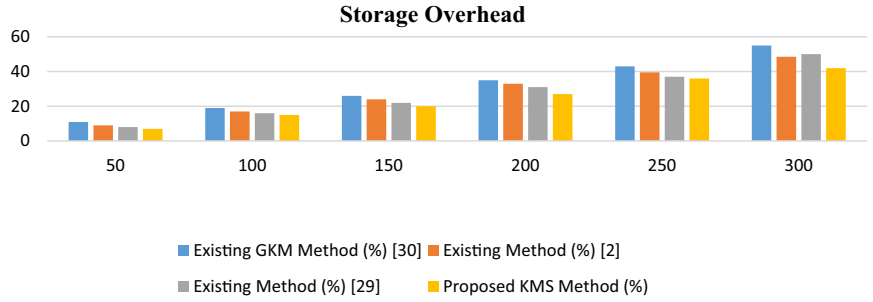


Fig. 12 Plot for storage overhead comparison for existing and proposed method

4.3 Resource utility

A CC environment's resource consumption rate might fall into a variety of different ranges. The resource utilization rate is the ratio of the total amount of assigned resources to the total amount of accessible resources. An important performance metric is directly influenced by the profit or loss of the cloud provider. Accordingly, this part compares the suggested KMS with the present approach to determine which yields the highest rate of parameter resource usage, as shown in Fig. 11. 50–300 additional users are added as a result of an increase in the rate of resource usage. Lower-demand tasks are prioritized over higher-demand tasks, so more high-capacity resources are available to those users. Using Table 8, we have compared the proposed methods' resource utility values to those of the existing methods in terms of task scheduling.

4.4 Storage overhead

KMS authentication is lightweight in terms of storage because just the actual files and logs are required to keep track of user information. Further, JAR acts as a compressor, and input files are converted into XML format of the files that it handles. Repeated logging can be seen when a large number of entities are constantly

Table 9 Comparison table for the storage overhead in terms of users

No of users	GKM 1 (%) [67]	GKM 2 (%) [2]	GKM 3 (%) [76]	Proposed KMS method (%)
50	11	9	8	7
100	19	17	16	15
150	26	24	22	20
200	35	33	31	27
250	43	39.5	37	36
300	55	48.5	50	42

accessing the data. Figure 12 depicts the findings of the second experiment, which focuses on the storage overhead involved in creating a log file. The proposed framework has a storage overhead of 7 percent for 50 cloud users, whereas the existing framework has a storage overhead of 11 percent for 50 cloud users. Table 9 shows the comparison (between the existing with the proposed method) for the storage overhead in terms of users. Memory requirements for GKM 1 $\{11+19+26+35+43+55\}=189$ bits, GKM 2 $\{9+17+24+33+39.5+48.5\}=171$ bits, GKM 3 $\{8+16+22+31+37+50\}=164$ bits, and proposed KMS method $\{7+15+20+27+36+42\}=147$ bits. It is obvious from Fig. 12 that the proposed KMS method requires less storage overhead as compared to GKM 1, GKM 2, and GKM 3. Storage overhead can be calculated by equation number 3:

$$\sum_{i=50}^{n=300} \text{GKM} \tag{3}$$

where n is the number of cloud users.

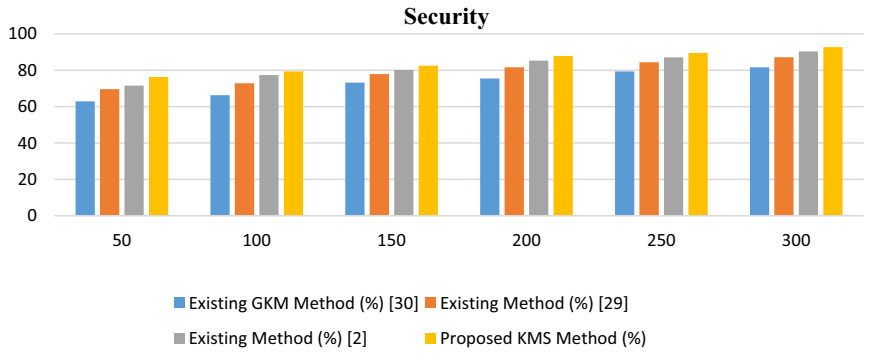


Fig. 13 Plot for security comparison for existing and proposed method

Table 10 Comparison table for the security levels in terms of users

No of users	GKM 1 (%) [67]	GKM 2 (%) [76]	GKM 3 (%) [2]	Proposed KMS method (%)
50	62.89	69.6	71.53	76.31
100	66.32	72.85	77.32	79.38
150	73.18	77.87	80.2	82.56
200	75.45	81.655	85.321	87.86
250	79.32	84.42	87.12	89.52
300	81.61	87.16	90.32	92.71

4.5 Security with respect to cloud users

The third round of testing was conducted to establish the level of security provided by the aforementioned methods of authentication, and the results are depicted in Fig. 13. The overall security grade of the proposed framework is 87.86 percent based on 200 cloud users' feedback. To calculate log file size, the following formula (see Eq. 4) has been used: Security data sent and received have a different size in terms of log file size and thus are aware of this while calculating log file size. Using Table 10, we have compared the proposed methods' security values to those of the existing methods in terms of users. Using Eq. 4, we can calculate the log file size of cloud users.

$$\sum_{i=50}^{n=300} (\text{log file size}) \tag{4}$$

The amount of data transferred from data owners to cloud users is expected to increase. You will not have any leaks if you are concerned about security during transmission. This research found that the difference in log file size between the data transmitted by the data owner and the data transmitted by the cloud user is a good indicator of data security. These log files are generated by the log harmonizer. If the size of the log file is changed, the level of security in that transmission will be reduced. When the size of the log file is altered, it is required to validate the integrity of the data included within the log file. Based on the findings, it can be stated that the KMS framework is exceptionally safe while also being incredibly space-efficient when compared to alternate approaches to the problem.

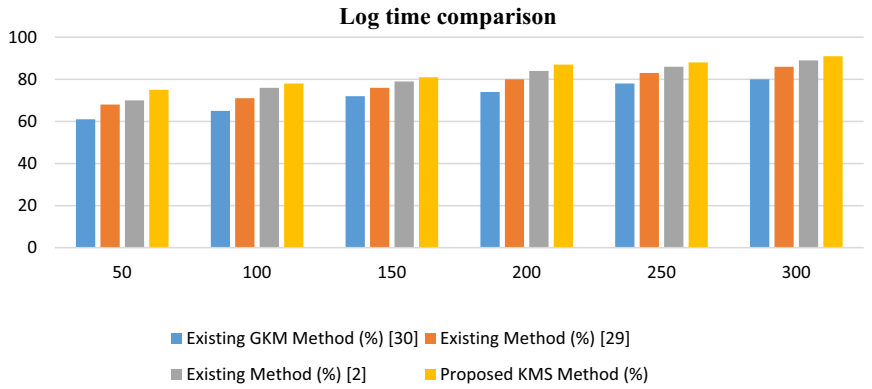


Fig. 14 Plot for log time comparison for existing and proposed method

Table 11 Comparison table for the log time comparison in terms of users

No of users	GKM 1 (%) [67]	GKM 2 (%) [76]	GKM 3 (%) [2]	Proposed KMS method (%)
50	61	68	70	75
100	65	71	76	78
150	72	76	79	81
200	74	80	84	87
250	78	83	86	88
300	80	86	89	91

4.6 Log time comparison

The first set of tests looks at how long it takes to produce a log file while entities are continuously accessing the data and documenting everything they see. Log file creation time grows linearly with log file size, as illustrated in Fig. 14. Specifically, it takes roughly 1.9 s to create a 100 KB file and about 6.21 s to create a 1000 KB file. The experiment maintains the basis for determining the length of time specified between dumps in an uncomplicated manner, even when variables like space restrictions or network traffic are taken into account. Using Table 11, we compare the suggested methods' log time comparison values to those of the existing methods in terms of users.

5 Conclusion

We have proposed an HCA-KMS based on authenticated encryption with AES and Elliptic Curve Cryptography in this work (ECC). To ensure the safety of CC data storage, the proposed Centralized Cloud Information Accountability (CCIA), i.e., an

HCA system with a KMS, is explained in this article. To maintain the highest level of security, the private key is securely exchanged. By selecting one random prime number for each parameter value and master secret key, secure key generation has been completed. Secure data transmission makes precise and reliable authentication feasible. Attribute-based encryption provides safe and dependable access control restrictions. Research methods may be fully implemented in CloudSim to produce better results than with the proposed methods. Using this key management system, the data owner cannot only inspect their material but also establish strong back-end security if necessary. The data owner provides the Cloud server with the data, the list of users, and the parameters needed to generate JAR files. Key management, encryption, and decryption are all handled by the Cloud Server, which is a trusted third party. Using the private key generated by the KMS, the Cloud server encrypts the data. Cloud computing attacks can be avoided with the suggested KMS architecture because of its ability to ensure lightweight and powerful accountability. We have been working on this project intending to enhance public cloud storage solutions' privacy and security. We believe we have made some progress toward ensuring the privacy and security of public cloud users, data owners, and service providers. This research project could be expanded by comparing multicast communication with other types of cryptographic algorithms and determining the best cryptosystem for secure healthcare communication. There is expansion of single-bit encryption to multi-bit encryption for the first construction and lowering of the number of trapdoors in the multi-owner's scenario for the second construction. Furthermore, we will work to make the above schemes leakage-resistant in the future. A multiprocessing system for better performance and low latency may be used during the implementation process.

Acknowledgements The authors acknowledge the financial support received own, for their support and encouragement in carrying out his college work. The authors also would like to acknowledge the administration of Jamia Millia Islamia, which the authors represent.

Authors' contributions Conceptualization was performed by SA and SM; data curation by SA and SM; formal analysis by SM and JB; investigation by SM and JB; methodology by SA and SM.; resources by SA and JB; supervision by SM; validation by SM and JB; visualization by SM; writing—original draft by SA; writing—review and editing—by SM and JB.

Funding Not applicable.

Declarations

Conflict of interests The authors declare that they have no competing interests.

Consent for publication All authors have read and agreed to the published version of the manuscript.

References

1. Hu X et al (2021) STYX: A hierarchical key management system for elastic content delivery networks on public clouds. IEEE Trans Depend Secure Comput 18(2):843–857. https://doi.org/10.1109/TDSC.2019.2918278

2. Zhang Y, Xu C, Ni J, Li H, Shen XS (2021) Blockchain-assisted public-key encryption with keyword search against keyword guessing attacks for cloud storage. IEEE Trans Cloud Comput 9(4):1335–1348. https://doi.org/10.1109/TCC.2019.2923222
3. Ahmad S, Mehfuz S, Beg J (2022) Cloud security framework and key management services collectively for implementing DLP and IRM. Mater Today Proc 62:4828–4836. https://doi.org/10.1016/j.matpr.2022.03.420
4. Froelicher D, Troncoso-Pastoriza JR, Pyrgelis A, Sav S, Sousa JS, Bossuat J-P, Hubaux J-P (2021) Scalable privacy-preserving distributed learning. Proc Privacy Enhancing Technol 2021(2):323–347
5. Celiktas B, Celikbilek I, Ozdemir E (2021) A higher-level security scheme for key access on cloud computing. IEEE Access 9:107347–107359. https://doi.org/10.1109/ACCESS.2021.3101048
6. Schulze Darup M, Alexandru AB, Quevedo DE, Pappas GJ (2021) Encrypted control for networked systems-an illustrative introduction and current challenges. IEEE Control Syst 41(3):58–78
7. Goswami PS, Chakraborty T (2020) Design of a quantum one-way trapdoor function. In: Mandal JK, Bhattacharya D (eds) emerging technology in modelling and graphics. Springer, Singapore, pp 547–555
8. Alexandru AB, Gatsis K, Shoukry Y, Seshia SA, Tabuada P, Pappas GJ (2020) Cloud-based quadratic optimization with partially homomorphic encryption. IEEE Trans Automat Control 66(5):2357–2364
9. Zhang S, Han S, Zheng B, Han K, Pang E (2020) Group key management protocol for file sharing on cloud storage. IEEE Access 8:123614–123622. https://doi.org/10.1109/ACCESS.2019.2963782
10. Zhang Z, Zeng P, Pan B, Choo K-KR (2020) Large-universe attribute-based encryption with public traceability for cloud storage. IEEE Internet Things J 7(10):10314–10323. https://doi.org/10.1109/JIOT.2020.2986303
11. Alexandru AB, Tsiamis A, Pappas GJ (2020) Towards private data-driven control. In: Proceedings of the 59th conference on decision and control (CDC), pp. 5449– 5456. IEEE
12. Berberich J, Köhler J, Muller MA, Allgower F (2020) Data-driven model predictive control with stability and robustness guarantees. IEEE Trans Automat Control 66(4):1702–1717
13. Singh G, Supriya, (2013) A study of encryption algorithms (RSA, DES, 3DES, and AES) for information security. Int J Comput Appl 67(19):33–38
14. Burr W (2003) Selecting the advanced encryption standard. IEEE Secure Priv 1(2):43–52
15. Frunza M, Asachi GH (2007) Improved RSA encryption algorithm for increased security of wireless networks. In: ISSCS International Symposium, vol. 2
16. Kodali R, Sarma N (2013) Energy efficient ECC encryption using ECDH. Emerging research in electronics, computer science and technology Lecture Notes in Electrical Engineering, vol 248. Springer, New Delhi, pp 471–478
17. Johnson D, Menezes A, Vanstone S (2001) The elliptic curve digital signature algorithm (ECDSA). Int J Inf Secure 1(1):36–63
18. Balitanas M (2009) WiFi-protected access-pre-shared key hybrid algorithm. Int J Adv Sci Technol 12
19. Subramaniam N, Jeyaraj A (2018) Recent security challenges in cloud computing. Comput Electrical Eng 71:28–42
20. Chenthara S, Ahmed K, Wang H, Whittaker F (2019) Security and privacy-preserving challenges of e-health solutions in cloud computing. IEEE Access 7:74361–74382
21. Wazid M, Das AK, Vasilacos AV (2018) Authenticated key management protocol for cloud-assisted body area sensor networks. J Netw Comput Appl 123:112–126
22. ShanmugaPriya S, Valamathi A, Yuvaj D (2019) The personal authentication service and security enhancement for optimal strong password. Concurr Comput Practice Exp 31:e5009
23. Neela KL, Kavita V (2018) Enhancement of data confidentiality and secure data transaction in cloud environment. Clust Comput 21(1):115–124
24. Blatt M, Gusev A, Polyakov Y, Rohloff K, Vaikuntanathan V (2020) Optimized homomorphic encryption solution for secure genome-wide association studies. BMC Med Genomics 13(7):1–13
25. van Waarde HJ, De Persis C, Camlibel MK, Tesi P (2020) Willems' fundamental lemma for state-space systems and its extension to multiple datasets. IEEE Control Syst Lett 4(3):602–607
26. Chen Y, Liu H, Wang B, Sonompil B, Ping Y, Zhang Z (2021) A threshold hybrid encryption method for integrity audit without a trusted center. J Cloud Comput 10:3
27. Shridharan S, Arokiassamy A (2017) Effective secure data storage in cloud by using ECC algorithm, Middle-East. J Sci Res 25:117–127

28. Goyal V, Kant C (2018) An effective hybrid encryption algorithm for ensuring cloud data security. Big data analytics. Springer, Singapore, pp 195–210
29. Kumar M, Iqbal A, Kumar P (2016) A new RGB image encryption algorithm based on DNA encoding and elliptic curve Diffie-Hellman cryptography. Signal Process 125:187–202
30. Amalarethinam DIG, Leena HM (2018) Asymmetric addition chaining cryptographic algorithm (ACCA) for data security in the cloud. Advances in big data and cloud computing. Springer, Singapore, pp 331–340
31. Askazadeh A (2016) "A novel metaheuristic method for solving constrained engineering optimization problems", crow search algorithm. Comput Struct 169:1–12
32. Wazid M, Bagga P, Das AK, Shetty S, Rodrigues JJ, Park Y (2019) AKM-IoV: authenticated key management protocol in fog computing-based internet of vehicles deployment. IEEE Internet Things J 6(5):8804–8817
33. Miao Y, Liu X, Deng RH, Wu H, Li J, Wu D (2018) Hybrid keyword field search with efficient key management for the industrial internet of things. IEEE Trans Ind Inf 15(6):3206–3217
34. Park K, Park Y, Das AK, Yu S, Lee J, Park, (2019A) dynamic privacy-preserving key management protocol for V2G in social internet of things. IEEE Access 7:76812–76832
35. Choi Y, Lee D, Kim J, Jung J, Nam J, Won D (2014) Security enhanced user authentication protocol for wireless sensor networks using elliptic curves cryptography. Sensors 14(6):10081–10106
36. Nikravan M, Reza A (2020) A multi-factor user authentication and key agreement protocol based on bilinear pairing for the Internet of Things. Wireless Pers Commun 111(1):463–494
37. Choi Y, Lee Y, Won D (2016) Security improvement on biometric-based authentication scheme for wireless sensor networks using fuzzy extraction. Int J Distrib Sens Netw 12(1):8572410
38. Wazid M, Das AK, Kumar N, Alazab M, (2020) Designing authenticated key management scheme in the 6G-enabled network in a box deployed for industrial applications. IEEE Trans Ind Inf, 1–1
39. Jiang Q, Ma J, Wei F, Tian Y, Shen J, Yang Y (2016) An untraceable temporal-credential-based two-factor authentication scheme using ECC for wireless sensor networks. J Netw Comput Appl 76:37–48
40. Ali Z, Hussain S, Rehman RHU, Munshi A, Liaqat M, Kumar N, Chaudhry SA (2020) ITSSAKA-MS: an improved three-factor symmetric-key based secure AKA scheme for multi-server environments. IEEE Access 8:107993–108003
41. Jung J, Kim J, Choi Y, Won D (2016) An anonymous user authentication and key agreement scheme based on a symmetric cryptosystem in wireless sensor networks. Sensors 16(8):1299
42. Sadhukhan D, Ray S, Biswas G, Khan M, Dasgupta M (2020) A lightweight remote user authentication scheme for IoT communication using elliptic curve cryptography". J Supercomput 77(2):114–1151
43. Moon J, Lee D, Lee Y, Won D (2017) Improving biometric-based authentication schemes with smart card revocation/reissue for wireless sensor networks. Sensors 17(5):940
44. Challa S, Das AK, Gope P, Kumar N, Wu F, Vasilakos AV (2020) Design and analysis of authenticated key agreement scheme in cloud-assisted cyber-physical systems. Futur Gener Comput Syst 108:1267–1286
45. Li X, Niu J, Bhuiyan MZA, Wu F, Karuppiah M, Kumari S (2017) A robust ECC-based provable secure authentication protocol with privacy-preserving for Industrial Internet of Things. IEEE Trans Ind Inf 14(8):3599–3609
46. Moghadam MF, Nikooghadam M, Al Jabban MAB, Alishahi M, Mortazavi L, Mohajerzadeh A (2020) An efficient authentication and key agreement scheme based on each for wireless sensor network. IEEE Access 8:73182–73192
47. Wu F, Xu L, Kumari S, Li X (2017) A privacy-preserving and provable user authentication scheme for wireless sensor networks based on Internet of Things security. J Ambient Intell Humaniz Comput 8(1):101–116
48. Tanveer M,.Abbas G, Abbas ZH, (2020) LAS-6LE: a lightweight authentication scheme for 6LoWPAN environments, In: 2020 14th international conference on open-source systems and technologies (ICOSST), pp. 1–6.
49. Li X, Niu J, Kumari S, Wu F, Sangaiah AK, Choo K-KR (2018) A three-factor anonymous authentication scheme for wireless sensor networks in the Internet of Things environments. J Netw Comput Appl 103:194–204
50. Shuai M, Yu N, Wang H, Xiong L (2019) Anonymous authentication scheme for the smart home environment with provable security. Comput Secur 86:132–146

51. Mahmood K, Li X, Chaudhry SA, Naqvi H, Kumari S, Sangaiah AK, Rodrigues JJ (2018) Pairing based anonymous and secure key agreement protocol for smart grid edge computing infrastructure. Futur Gener Comput Syst 88:491–500
52. Wazid M, Das AK, Kumar N, Vasilakos AV, Rodrigues JJ (2018) Design and analysis of secure lightweight remote user authentication and key agreement scheme on Internet of Drones deployment. IEEE Internet Things J 6(2):3572–3584
53. Jia X, He D, Li L, Choo K-KR (2018) Signature-based three-factor authenticated key exchange for Internet of Things applications. Multimed Tools Appl 77(14):18355–18382
54. Lu Y, Xu G, Li L, Yang Y (2019) Anonymous three-factor authenticated key agreement for wireless sensor networks. Wireless Netw 25(4):1461–1475
55. Chen Y, Lopez L, Martinez J-F, Castillejo P (2018) A lightweight privacy protection user authentication and key agreement scheme tailored for the Internet of Things environment: Lightpriauth. J Sens 2018:1–16
56. Yang Z, Lai J, Sun Y, Zhou J (2019) A novel authenticated key agreement protocol with a dynamic credential for WSNs. ACM Trans Sens Netw (TOSN) 15(2):1–27
57. Alotaibi M (2018) An enhanced symmetric cryptosystem and biometric-based anonymous user authentication and session key establishment scheme for WSN. IEEE Access 6:70072–70087
58. Shin S, Kwon T (2019) A lightweight three-factor authentication and key agreement scheme in wireless sensor networks for smart homes. Sensors 19(9):2012
59. Zhou L, Li X, Yeh K-H, Su C, Chiu W (2019) Lightweight IoT based authentication scheme in cloud computing circumstance. Futur Gener Comput Syst 91:244–251
60. Hadjicostis CN, Dominguez-Garcia AD (2020) Privacy-preserving distributed averaging via homomorphically encrypted ratio consensus. IEEE Trans Automat Control 65(9):3887–3894
61. Murguia C, Farokhi F, Shames I (2020) Secure and private implementation of dynamic controllers using semihomomorphic encryption. IEEE Trans Autom Control 65(9):3950–3957
62. van Waarde HJ, Eising J, Trentelman HL, Camlibel MK (2020) Data informativity: a new perspective on data-driven analysis and control. IEEE Trans Automat Control 65(111):4753–4768
63. Ye Y, Chen H, Xiao M, Skoglund M, Poor HV (2020) Privacy-preserving incremental ADMM for decentralized consensus optimization. IEEE Trans Signal Process 68:5842–5854
64. Alexandru AB, Pappas GJ (2020) Secure multi-party computation for cloud-based control. Privacy in dynamical systems. Springer, Singapore, pp 179–207
65. Mallik A (2019) Man-in-the-middle-attack: understanding in simple words. Cyberspace: Jurnal Pendidikan Teknologi Informasi 2(2):109–134
66. Song C et al (2019) Hierarchical edge cloud enabling network slicing for 5G optical fronthaul. J Optic Commun Netw 11(4):B60–B70. https://doi.org/10.1364/JOCN.11.000B60
67. Yao Y, Zhai Z, Liu J, Li Z (2019) Lattice-based key-aggregate (searchable) encryption in cloud storage. IEEE Access 7:164544–164555. https://doi.org/10.1109/ACCESS.2019.2952163
68. Wang S, Pei R, Zhang Y (2019) EIDM: a ethereum-based cloud user identity management protocol. IEEE Access 7:115281–115291. https://doi.org/10.1109/ACCESS.2019.2933989
69. Miao Y et al (2019) Hybrid keyword-field search with efficient key management for industrial internet of things. IEEE Trans Industr Inf 15(6):3206–3217. https://doi.org/10.1109/TII.2018.2877146
70. Ma M, Shi G, Li F (2019) Privacy-oriented blockchain-based distributed key management architecture for hierarchical access control in the IoT scenario. IEEE Access 7:34045–34059. https://doi.org/10.1109/ACCESS.2019.2904042
71. Wang F, Xu L, Gao W (2018) Comments on "SCLPV: secure certificateless public verification for cloud-based cyber-physical-social systems against malicious auditors." IEEE Trans Comput Social Syst 5(3):854–857. https://doi.org/10.1109/TCSS.2018.2858805
72. Xu Q, Tan C, Fan Z, Zhu W, Xiao Y, Cheng F (2018) Secure multi-authority data access control scheme in cloud storage system based on attribute-based signcryption. IEEE Access 6:34051–34074. https://doi.org/10.1109/ACCESS.2018.2844829

73. de Ree M, Mantas G, Rodriguez J, Otung IE (2022) DECENT: decentralized and efficient key management to secure communication in dense and dynamic environments. IEEE Trans Intell Transp Syst. https://doi.org/10.1109/TITS.2022.3160068
74. Chen X, Ding J, Lu Z (2022) A decentralized trust management system for intelligent transportation environments. IEEE Trans Intell Transp Syst 23(1):558–571. https://doi.org/10.1109/TITS.2020.3013279
75. Tanveer M, Khan AU, Kumar N, Hassan MM (2022) RAMP-IoD: a Robust authenticated key management protocol for the internet of drones. IEEE Internet of Things Journal 9(2):1339–1353. https://doi.org/10.1109/JIOT.2021.3084946
76. Upadhyay D, Zaman M, Joshi R, Sampalli S (2022) An efficient key management and multi-layered security framework for SCADA systems. IEEE Trans Netw Serv Manag 19(1):642–660. https://doi.org/10.1109/TNSM.2021.3104531

Authors and Affiliations

Shahnawaz Ahmad[1] · Shabana Mehfuz[1] · Javed Beg[2]

✉ Shabana Mehfuz
smehfuz@jmi.ac.in

[1] Department of Electrical Engineering, Jamia Millia Islamia (A Central University), New Delhi 110025, India

[2] Oracle, Noida, Uttar Pradesh, India