Journal of Information Hiding and Privacy Protection
DOI:10.32604/jihpp.2021.027834
Article

Tech Science Press

Hash Function Based Keyword Searchable Encryption Framework in Cloud Server Using MD5 and MECC

R. Lakshmana Kumar[1,*], R. Subramanian[2] and S. Karthik[1]

[1]Department of Computer Science and Engineering, SNS College of Technology, Coimbatore, Tamil Nadu, India
[2]Department of Electrical and Electronics Engineering, SNS College of Technology, Coimbatore, Tamil Nadu, India
[*]Corresponding Author: R. Lakshmana Kumar. Email: lakshmanakumar93@gmail.com
Received: 26 January 2022; Accepted: 02 March 2022

Abstract: Cloud Computing expands its usability to various fields that utilize data and store it in a common space that is required for computing and the purpose of analysis as like the IoT devices. These devices utilize the cloud for storing and retrieving data since the devices are not capable of storing processing data on its own. Cloud Computing provides various services to the users like the IaaS, PaaS and SaaS. The major drawback that is faced by cloud computing include the Utilization of Cloud services for the storage of data that could be accessed by all the users related to cloud. The use of Public Key Encryptions with keyword search (PEKS) provides security against the untrustworthy third-party search capability on publicly encryption keys without revealing the data's contents. But the Security concerns of PEKs arise when Inside Keywords Guessing attacks (IKGA), is identified in the system due to the untrusted server presume the keyword in trapdoor. This issue could be solved by using various algorithms like the Certificateless Hashed Public Key Authenticated Encryption with Keyword Search (CL-HPAEKS) which utilizes the Modified Elliptic Curve Cryptography (MECC) along with the Mutation Centred flower pollinations algorithm (CM-FPA) that is used in enhancing the performance of the algorithm using the Optimization in keys. The additional use of Message Digests 5 (MD5) hash function in the system enhances the security Level that is associated with the system. The system that is proposed achieves the security level performance of 96 percent and the effort consumed by the algorithm is less compared to the other encryption techniques.

Keywords: Certificateless Hashed Public Key Authenticated Encryption with Keyword Search (CL-HPAEKS); modified elliptic curve cryptography (MECC); digest 5(MD5); inside keywords guessing attacks (IKGA); public key encryptions with keyword search (PEKS)

1 Introduction

Considering Internet as a core, the Internets of Things (IoT) and various technologies that relay on the internet for storage of data that usually depends on the internal storage since the availability of the resources in internet for storing user data is very less and also cost effective. The use of Internet of Things (IoT) usually implants all elements that are available within the network by multiple sensing devices like sensors and actuators. It also proffers the smart network by amalgamating all technologies of smart identification, locating, tracking, and also monitoring [1]. With existing massive data volume available in IoT, the data storage alongside Data processing emerges now as the main issue. Cloud computing is observed as the best promising methodologies for rectifying this issue by storing the contents of the IoT

retrieved data and various data that are being collected by the resources that use internet for computing purposes [2]. Cloud computing is the use of the online storage and retrieval management to run practical services and applications that are related to day-to-day activities. It did not begin overnight; this might be tracked back to the times before computing systems had distant time-shared computational power and potential implementation. The use of virtualized programming foundation, such as network interfaces and various shared resources, are also used in cloud computing. The major advantage of the cloud includes the remote data usage facility in which the data could be accessed by the user anywhere only using the internet connection. Nowadays, Cloud technology is utilized widely by individuals and entrepreneurs for the reason that it allows managing individual's data conveniently and also at a very low cost [3]. Cloud users are privileged to hoard the data into the cloud and to access it from any location. This saves local storage capacity and enables easy access [4]. Nevertheless, Data storage in the Cloud sometimes brings some security issues. These issues arise due to the utilization of the Cloud storage by public devices that are connected to the database or the cloud storage. The services that are provided by the Cloud could be categorized as SaaS, PaaS, and IaaS.

To guard data privacy as well as combat undesired accesses on the cloud, Sensitive information of the users like individual health reports, tax forms, pictures, e-mail, and banking transactions, etc. [5–6] must be encrypted prior to outsourcing the data to cloud which improves confidentiality of the data. The encryption prevents data from being attacked by the intruders and illegal users who place attack on the sensitive information of the users. Nevertheless, data utilization, like keyword search, becomes another challenging issue as a consequence of data encryption since the encrypted data are not utilized by the keyword search options [7]. Though the recipient can retrieve encrypted data first from Cloud-server and decrypt it for perceiving actual information, this method requires a large amount of data transfer which could affect the user's storage capacity and raises the price owing to local computation [8].

To avoid such an issue, encryption schemes with searchable options, like single keyword and multiple-keyword search, ranked search, PEKS [9], etc., are developed [10]. The PEKS (public key encryption with keyword search) technique allows the user to search for encrypted terms without impacting the actual data's integrity. The type of searches that are included with the keyword search functions are the single keyword search, in which only one variant is being returned as a single word. When many variants are used for the single keyword, then the multiple keyword search might be used. The rank function might be used in protection of the document with owners and privacy simultaneously. The functionality of the PEKS is related to the flow that generally takes place in three steps that include the sender data is initially encrypted and next is the storage of that data of the sender in the remote server. The final step is the retrieval of data that is stored in the server using user's private key or decryption process.

Among all available encryption schemes, PEKS achieves enormous attention due to its hands-on applicability [11]. The PEKS values do not disclose anything about the text, but they do allow you to look for search terms. Our objective is to submit a brief private key to the mail server which will allow us to send encrypted messages. The servers will look for any communications that include the term, but will not learn anything else. To create the public or private secret key, it uses the KeyGen technique. The mail server feeds the supplied doorways into an algorithm to see if a particular email contains one of its user-specified keywords. It is developed with the added services together with traditional PEK [12]. In theory, Certificateless cryptography has been founded to combine the benefits of both the lack of key bond on PKI as well as the nonexistence of certificate supervision overheads in ID-centred cryptography [13].

Certificateless cryptography is a sort of public-key encryption that provides a combination of both classic PKI-based and connection public-key cryptographic. The keys are generated by the Key generation Centre (KGC) or the certificate authority in which the operations of the system are performed by the user with the private key generation. It offers protection even without requirement for a certificate authority to verify a digital signature, and it is vulnerable to damage from any private entity, particularly "trustable" third parties. Certificateless cryptography is a type of ID-based encryption that avoids the difficulty of secret exchange. Credentials are often produced by an authentication server or a key

generation centre (KGC), which is given complete autonomy and implicit confidence. PKI (public key infrastructure) is a fundamental of integrity in every company, allowing for encrypted and secure bonds between individuals, programs, and objects. Encrypted mail, message authentication, and SSL certificates are no longer the only use instances for PKI.

This approach ensures not just the confidentiality of the keyword, but also the fast retrieval of ciphertext by search term. [14]. It is actually a technique of applicable cryptographic with cloud storage. It protects the user's privacy data and then conserves search-ability of the server-end itself. It affords searchability and confidentiality, concurrently at the same time [15]. PKE schemes use exponentiation computations to do encryption in addition to decryption [16]. PEKS enables people to conduct keyword searches on data ciphertexts provided by standard-PKE.

Three categories of essentials were concerned with PEKS: i) a server, ii) a sender and iii) a receiver. Logically, a sender updates the file (encrypted) into a cloud server with an encrypted keywords' list [17]. For executing search queries, the receiver propels a generated keyword trapdoor wherein the searched keyword is encoded [18]. The receiver transmits the appropriate keyword that is implied as an entry to the cloud server, then after that examines the authenticated files for the existence of such a trapdoor [19]. At last, it will send the receiver the required encrypted data, which will contain the relevant searched keyword. Therefore, the PEKS reduces the maintenance costs of certificates. The purpose of searchable PEKS is to allow users to transmit a brief secret key to the web server, which the processor will be using to retrieve all mails comprising the keyword. The server immediately passes your recipient the appropriate emails.

Even though there are still some privacy issues, PEKS resolves the problem of searching over common encrypted data whereas some of PEKS have the hazard of Internal Keyword Guessing Attack (IKGA). IKGA is the problem in which the attacker may be aware of the email messages subject, exposing the users' information. The cloud server or another function within the cloud storage control normally launches such an assault. As a result, it is also known as the Inside Keyword Guessing Attack. According to the remedial measures, the KGA may function for two reasons. The attacker can first gain access to the trapdoor. Second, it is free to conduct the assessment. A malevolent adversary attempts keywords one after the other and verifies determines whether the ciphertext's relevant keyword corresponds to the nominee. The attack is feasible on account of the small space of realistic keywords [20]. Therefore, the cloud servers may change to inside KGA adversary who may recover sensitive data from available files [21]. In this specific paper, the Certificateless Hashed Public Key Authenticated Encryption with Keyword Search (CL-HPAEKS) is proposed for the protection of data privacy.

The CL-HPAEKS uses three algorithms or methods in implementing the system that is based on certificate less encryption with the hashed public key insertion function. The MECC (Modified Elliptic Curve Cryptography algorithm) is used along with the Mutation Centred Flower Pollination Algorithm (CF-FPA) and Message Digest 5 (MD5) algorithm. The MECC algorithm is used in this CL-HPAEKS to achieve the Protection that is achieved by using Multiple keys. The CM-FPA is used in Optimizing the keyword search that is present in the Cloud environment in which both the Local and Global Search could be achieved. The MD5 is used in Achieving Integrity in the messages or the data that is being stored in the cloud by the Utilization of Hash Functions. The CL-HPAEKS combines the use of the above algorithms in achieving the Hash based Encryption Search in the Cloud environment.

Data protection could be achieved using the Modified Elliptic Curve Cryptography method in which the key generation is carried out using the multiple factors that are utilized in the key generation. The MECC algorithm that is used in this work utilizes the CF-FPA which optimizes the search that is being performed as both the local and global search methodology. Another algorithm the MD5 is used in providing integrity to the data using the hash value that is being generated during the encryption process. Using the defined methods, the data integrity and security could be achieved in the cloud environment with improvement in privacy of the encrypted data that is stored in cloud.

The design of this proposal is structured into different sections. Section 2 investigates all allied tasks of the proposed work. Section 3 will brief the discussion about the method of the proposed solution. In Section 4, experimental outcomes are detailed, and the next Section 5 will list the conclusions.

2 Literature Review

The suggested length of a manuscript is 10 pages. Each page in excess of 15 will be charged an extra fee. In this section, the privacy concerns of the Encryption done under the PEKs are examined along with the discussion related to the Certificateless Hashed Public Key Authenticated Encryptions with Keyword Searches (CL-HPAEKS) scheme.

Public Key Encryptions with Keyword Search

The advancement in Computing brings in many benefits to society in storing the data that is required for computing and maintaining the balance between the data store and security that is provided on the data. Developments that are related to cloud computing depend on the introduction of new algorithms that are used in protection of data that is stored in Cloud. The data that is stored in the cloud must be encrypted to avoid the data from being intruded or modified by other users of the cloud since the storage of cloud is public in times. Data Search that is done on the Encrypted data brings in many issues related to the blocked data or the encrypted data that could not be retrieved by the search texts. Business organizations that have their data that is more sensitive and also the data need to be retrieved in time must be protected by other means since the protected data retrieval has the encryption done on data. The construct mechanism that is used in enabling the server in identifying the keyword searches is known as the Public Key Encryption with keyword Search.

The Public Key Encryption with Key Word Search (PKEKS) enables the user to search with the given keyword and hide the other details related to the stored document in the cloud storage. Both encryption and authentication are provided to the user's private key and the receiver's private key encryption using secret key of the sender. The PEK enables the search related to the trapdoor functions that is related to the keyword that is being searched in the system along with the cipher text and target keyword functionalities. The computations that are used by the bilinear search that is related to the system helps in providing bonding time restrictions and security enhancements in the system. The target word that is present in the search text that is being encrypted could be obtained by using the Public Key Encryption technique.

The scheme for prevention of Counter attacks that are happening in the system could be reduced by using the Publics-Key Authenticated Encryptions with Keywords Search (PAEKS). The encrypted keyword that is sent by the data holder must be authenticated and verified by the data owner (DO). The Concreate implementation of the PAEKS helps the system in constructing the Security enabling with the random Oracle design [22]. The dual server configurations could also be used in protection against the servers that do not support the authentication property of the system. This dual-server PAEKS (DPAEKS) could be used in along with the two non-colluding cloud servers that are responsible for the defend against IKGA attacks [23].

The use of Oracle random models could be avoided by using the SCF-PEKS to avoid the attack of ciphertext and preferred keyword attacks. The protection is given against the trapdoor utilization security and the IND-KGA security was also established in the system. By using the system resources, itself, the cloud security could be provided in the system [24]. Hence the above methods provide the supporting functionalities to avoid the trapdoor and the internal keyword guessing attacks in the system that is associated with the Public Key Encryption with Key word Search but these systems could not provide the efficiency needed for the system to avoid the attacks.

Certificateless Hashed Public Key Authenticated Encryption with Keyword Search

The Inside Keyword Guessing Attack that is caused due to the small keyword space could be overcome by using certificateless public key authenticated encryption with keyword search scheme.

The Certificateless Hashed Public Key Authenticated Encryption will secure the system against the IKGA attacks. The Data Organizer is responsible for the encryption that is done at the authentication side. The keyword encryption is not allowed for the third party encryptor other than the private key of the DO. Hence strong security would be provided against the IKGA attack that is possible in the two adversary types of the system in the IIOT usage of the system [25]. The encryption and authentication are carried out simultaneously for the selected keyword search the IKGA centred Diffies-Hellman model was used in the system as an oracle arbitrary model that is proved to resist the divergent attack types and is considered to be more secured than any other model types on the Certificateless cryptography with less loss in efficiency [26]. The use of deniably authentic encryption systems that are used along with the DAE methods are validated by using of messages and the encoded with the data holders. Hence the privacy protection and the efficiency that is done with the proposed dCLDAEKS system was considered to be worse than the other systems [27].

The use of bilinear pairing was done at the Certificate-less PKC with Authorized Equivalents inquiry (CL-PKC-AET) at which the PKE-ET and CL-PKE are blended together. The ability to test and design are provided by the CL-PKC-AET algorithm that could be matched with the Diffies-Hellman assumption [28]. Parallel keys that are associated with the certificate-less systems are responsible for the protection of IoT devices that store data in cloud. The use of key insulated Primal enables the system to reduce the key acquaintances. The parallel means that is done in the key sharing could be suitable for the IoT related functions in the system or the cloud environment [29]. The key environment could also be managed using the Certificateless Authentications Encryption (CLDAE) that is used in handling the keys effectively with the use of infrastructures-centred cryptosystems in which the tag-key encapsulations and data encapsulations are done. The CLDAE is more suitable for the location centred applications and it also includes the random oracles model [30].

The Certificateless Hashed Public Key Authenticated Encryption with Keyword Search provides the way in securing the data against the insider attacks and the trapdoor functions that are considered to be the backlog of the PEK algorithm. The CL-HPKAES utilizes the Message Digest 5 and the Modified Elliptic Curve Cryptography (MECC) in enhancing the security and privacy of data that is being stored in the cloud [31–34].

Modified Elliptic Curve Cryptography Algorithm

The Modified Elliptic Curve Cryptography Algorithm is used in protecting the sensitive data that is stored in the cloud environment. Data that are related to the sensitive devices like the IoT and business data might be prone to various types of attacks and this could be protected by using the ECC algorithms. The RSA algorithms that include the AES and DES provide security to the minimum amount since they are protected by the single key encryption. The ECC provides data security by using Multi key Encryption that is generated in a less time [35]. The ECC algorithm uses two keys, the public and the private keys that are used in encryption and decryption of the system based on the dependability. To eliminate the random choice of keys as used in the ECC, the Modified ECC uses the modified version in key selection and utilization [36]. Another problem with the ECC algorithm is that this algorithm uses a greater number of resources, memory capacity of the cloud and the encryption time is higher rather than its advantages like the minimal key sizes and Low consumption of bandwidth options [37].

The Modified ECC algorithm uses keys that are separately implemented for the users of the system and the administrators. The data to be accessed from the Cloud could be downloaded after the verification of the identity of the user. The key for decryption is generated separately ensuring the system privacy [38]. In some of the systems, the decryption process is carried out by the MECC whereas the encryption is carried out by some other process or algorithms [39].

The MECC algorithm that is applied in this work uses the Mutation Centred flower pollinations algorithm that depends on the Flower pollinations in which the pollen transfer from one plant to the other through pollinators or wind. The optimization process that is done depends on the search that is done at the biotic moment known as the Levy flight that is known as the distance between the large and the small

steps that is taken in the system [40]. The self-pollination that is done refers to the Local keyword search that is implemented within the system or the cloud environment. The global search and the Local search that are associated with the key word search in the cloud environment is done with the use of this Flower Pollination algorithms. The disadvantage faced by the Flower pollination algorithm is that it faces Low precession in search factors and the solution for complex or large search items are relatively low [41].

The Modified Elliptic Curve Algorithm uses the Mutation Centred flower pollinations algorithm to optimize the keyword search that is done in the Cloud environment.

Message Digest 5

Data security enhancement is done by the implementation of the Message Digest 5 (MD5) algorithm in the system. In order to increase the effectiveness of CL-HPKAES in terms of data security, the algorithm uses MD5. This is to protect the data with the implementation of hash function done at the data bits. The general rule of the MD5 is to take random data as inputs and produce the output as the defined set of data in the fixed length and size format, i.e., 32-bit form. It is also used in reviewing the correctness of the data and the originality of data present in the cloud environment. The input data is divided into 512 bits and division is made for the 16 sub-groups with 32-bit encryption [42].

Confidentiality of the data stored in cloud could be achieved using the MECC algorithm that generates elliptic curve in maintaining the data objectives. The MD5 utilizes the has value comparison techniques that are used by the system in characterizing the originality of the data. In simple terms it could be defined as the message that is stored in the cloud environment is protected by the hash vale and this value of data after the retrieval would be compared with the hash values must have the similar result. Hence data integrity could be implemented in the system [43]. Data compression could be obtained by using the MD5 algorithm since the information that is stored in the device could be compressed into a 128-bit format. The Utilization of non-linear algorithm protects the data from external intruders. The algorithm is free of cost and royalties and hence it could be used in large number of applications. The algorithm could also be termed as the irreversible transformation of data and hence the implementation of data integrity could be effectively done in the cloud environment. Table 1 illustrates the abbreviations and their corresponding full names included in this article.

Table 1: Abbreviations and their full name listed in literature review

Abbreviations	Full Name
PEKS	Public key Encryptions with Keyword Searches
IKGA	Inside Keyword Guessing attacks
CL-HPAEKS	Certificateless Hashed Public Key Authenticated Encryptions with Keyword Searches
ECC	Elliptic Curve Cryptography
MECC	Modified Elliptic Curve Cryptography
CM-FPA	Mutation Centred Flower Pollinations Algorithm
MD5	Message Digests 5
SL	Security Level
IoT	Internets of Things
DO	Data Owner
DR	Data Receiver
SCF-PEKS	Secure channel Free Public Key Encryptions with Keyword Searches
SCF-CKCA	Secure Channel Free Preferred Keyword and Ciphertext Attack
IIoT	Industrial Internets of Things
DDoS	Distributed Denial-Of-Service
RBM	Restricted Boltzmann Machine

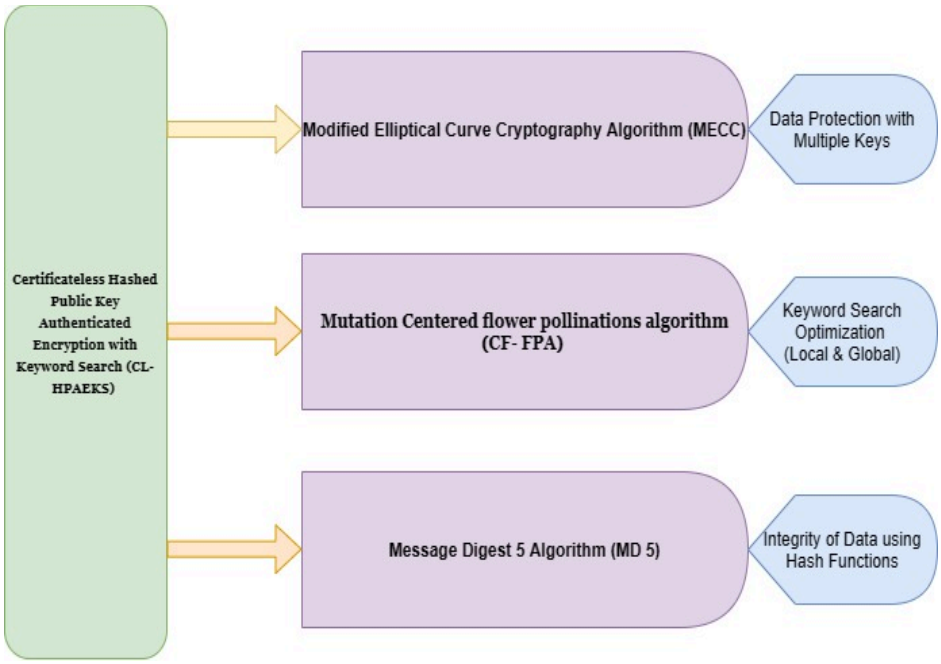
KG	Key Generation
KGC	Key Generations Centre
CS	Cloud Server
FPA	Flower Pollinations Algorithm

3 Proposed CL-HPAEKS Scheme for Secure and Efficient Authentication

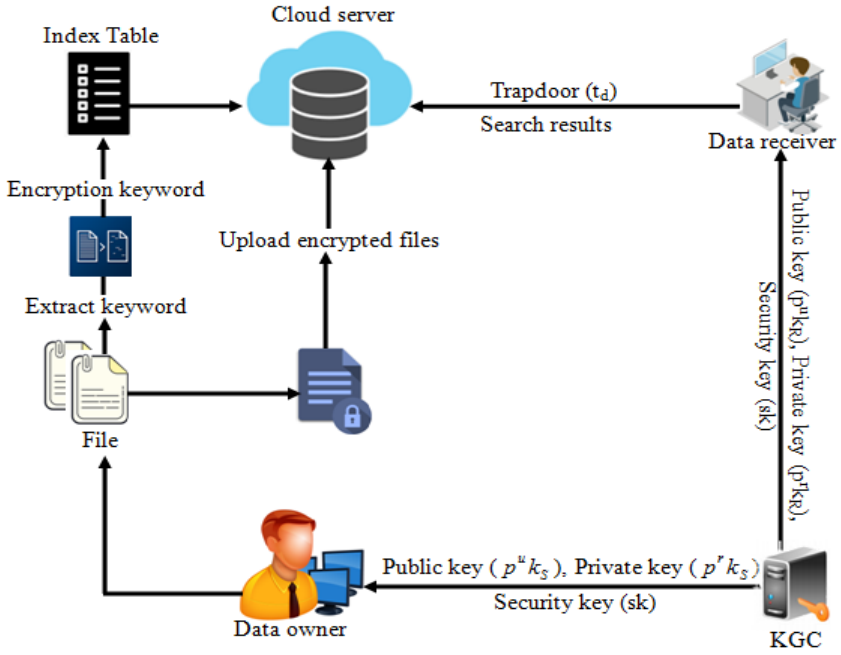
Searching a sensitive (encrypted) data effectively and also securely in cloud storage is a challenging issue. A DDoS attack is a big event that causes website infrastructure to become overwhelmed and unable to react to genuine customer request. If given sufficient time, attackers can locate and obtain access to such threats and vulnerabilities, as well as compromise authentication through APIs. Searchable-PEKS facilitate a storage server for retrieving the openly encrypted data devoid of exposing the original contents that render a secure storage method devoid of failure of data confidentiality as well as usability. However, most conventional PEKS cannot handle IKGA. Thus, opposing the IKGA is expected to be an indispensable property of all newfound PEKS. To tackle this IKGA issue, a CL-HPAEKS is proposed, which is secure in opposition to IKGA.

The proposed CL-HPAEKS encompasses '6' phases: i) Key Generation (KG), ii) setup, iii) index table, iv) encryption, v) trapdoor search, and vi) decryption phase. The KG (initial phase) is managed via the Key Generations Centre (KGC) that creates a private key as well as public-key intended for DO and Data Receivers (DR) by utilizing MECC's KG process. The public key will be used for encrypt and decrypt encrypted data and transform it to cypher text, while the recipient's private key is being used to decryption the cypher text and decrypt the letter. The data is stored hidden in secret key encryption, while one of the cryptographic switches is done in secret in public key encryption. MECC has a high storage capacity, however some nodes require a secret key to operate. It was noted each node would have its own set of queries. Both the GW-node and the login-node store the querying database of the enrolled user's mysterious data, so it is vulnerable to stolen-verifier attacks. In sensor systems, it relies on self-confirmed cryptosystems and use ECC to create paired entries.

In MECC, one more key (secret key) is created for enhancing the SL. The Crossover Mutation-centred Flower Pollination Algorithm (CM-FPA) optimizes these keys (generated). To develop a new generation, FPA normally uses a crossover operator, which involves pairing the individuals, and a mutation operator, which involves arbitrarily changing the individuals' components to enhance diversification. The flower pollination algorithm is a nature-inspired method for determining the best answer. Next, the MD5 converts these optimized keys into hashed ones. MD5 converts a variable-length communication into a fixed-length output by chunking the user input and padding it to make it manageable. Padding operates by appending a single bit, 1, towards the end of the message. After that, DO encrypt his/her file utilizing the optimized hashed key (i.e., dispatcher's private key, receiver's public key in addition to secret key), and then, upload the file to the CS in conjunction with the encrypted keyword. To keep our information better safe in the cloud, we employ the trapdoor approach to encode it with multiple random alternatives. In order to remember our files and any information, the same technique is followed to decrypt our information. The DR sends an enviable trapdoor encompassing keywords that he/she desires to pursuit for the CS. The server, subsequent to receiving a request, checks in case the keyword proportionate to the access is equivalent to the keyword of the ciphertext. If matched, the CS endeavours to decrypt that encrypted keyword utilizing the recipient's hashed private key, the sender's hashed public key in conjunction with a secret key. If keyword decryption is flawlessly made, then the CS proceeds to rebound the translated data in relation to the trapdoor. The receiver subsequent to that decrypts the file utilizing the optimized hashed key (receiver's private key, sender's public key as well as secret key). CL-HPAEKS's system model is exhibited in Fig. 1, and the workflow of hash function based keyword search encryption is also exhibited in Fig. 1.



(a)



(b)

Figure 1: (a): Workflow of hash function based keyword search encryption; (b): System model of CL-HPAEKS

3.1 Notations

Table 2 illustrates the notations included in this article.

Table 2: Notations

Notation	Depiction
DO	Data Owner
DR	Data Receiver
D	Data Owner's document
k_W	The Keywords
$p^r k_S$, $p^u k_S$	Private key in addition to public key of data owner
$p^r k_R$, $p^u k_R$	Private key as well as public key of data receiver
$\vec{k}_{Wi}$	Extracted keyword set of D
t_d	Trapdoor
$\hat{p}^r_{kS}$, $\hat{p}^u_{kS}$	Optimized hashed private key as well as optimized hashed private key of data owner
$\hat{p}^r_{kR}$, $\hat{p}^u_{kR}$	Optimized hashed private key in addition to optimized hashed private key of data receiver
$\hat{s}_k$	Optimized hashed secret key
I_s	Index table
$E(D)$	Encrypted document
$E(\vec{k}_{Wi})$	Encrypted keyword set

3.2 System Model of CL-HPAEKS

Fig. 1 exhibits the system model of CL-HPAEKS, which encompasses '4' entities, i.e., the KGC, DO, the DR, together with the CS, correspondingly. The task of every entity is elucidated below:

Key Generation Centre: This stands as a trusted 3rd party. The KGC produces a public shared key in addition to the private key of the user as per their identity that is given to it by the user, and after that, sending the keys to DO, CS, and also users. Underneath the password provided for each user upon enrolment, the Key Generation Centre could disseminate shared important information to every member of the group simultaneously, while sending the cypher text to every participant individually.

Data Owner (DO): The *DO* encrypts his/her documents D utilizing a customary encryption algorithm (i.e., (*ED*)). Additionally, *DO* extracts keywords k_W as of every document, and after that, encrypts these keywords utilizing CL-HPAEKS encompassing the receiver's publics key $p^u k_R$ in addition to the sender's privates key $p^r k_S$ (explicitly, CL-HPAEKS (k_W, $p^r k_S$, $p^u k_R$)). Finally, the encrypted data is amassed on the CS.

Data Receiver (DR): The DR search for keywords by means of sending these keywords' trapdoor to the cloud. Utilizing the receiver's privates key $p^r k_R$ along with the sender's publics key $p^u k_S$, the

trapdoor t_d is created (i.e., $t_d = Trapdoor(k_W, p^u k_s, p^r k_R)$). A trapdoor function is something that is simple to calculate in one direction but difficult to calculate in another without specific knowledge.

Cloud Server (CS): The *CS* encompasses vast storage space along with a sturdy computing power to manage as well as uphold the *DO*'s data. It is basically a semi-trusted entity. It returns the equivalent document when attaining a trapdoor as of the receiver.

3.3 Formal Definition of CL-HPAEKS

The proposed CL-HPAEKS includes the following 6 phases:

1. ***Setup ($\vec{k}_{Wi}$):*** This is the international parameter generation stage. It is largely focused to extract the keywords as of the chosen file by means of the sender.
2. ***KeyGen ($p^r k_S$, $p^r k_R$, $p^u k_S$, $p^u k_R$):*** This is a KG phase. This is handled by means of KCG. This algorithm is utilized to make a private, public key, in addition to the Secret key of the users.
3. ***Encryption (k_W , $p^r k_S$, $p^u k_R$):*** This is an encryption phase, wherein the encryption will be performed for the uploaded file along with its equivalent keywords.
4. ***Index Table ((k_W , $p^r k_S$, $p^u k_R$), k_W):*** This phase is chiefly utilized to uphold the encrypted big data storage location with the equivalent encrypted keyword.
5. ***Trapdoor ($t_d = Trapdoor(k_W, p^r k_R)$):*** This is a trapdoor generation phase. Here, the receivers enter the keyword for searching the encrypted big data over the CS. At that moment, the system endeavours to decrypt the encrypted keyword utilizing the requested user's private key.
6. ***Decryption ($p^u k_S$, $p^r k_R$):*** This is the decryption stage, wherein the receiver could decrypt the file utilizing the dispatcher's public key along with the receiver's private key.

The CL-HPAEKS follows aforementioned '6' phases to enhance data privacy, security and also to oppose IKGA. The procedure of every phase in CL-HPAEKS is, in brief, elucidated in the section below.

3.4 Function of Proposed CL-HPAEKS

The responsibilities of KG, setup, encryption, index table, trapdoor search, and decryption in CL-HPAAEKS is elucidated in the following parts.

3.4.1 Key Generation Phase

KG represents the process of producing keys on cryptography where key is employed for encrypting as well as decrypting any facts that is being encrypted or decrypted. The KGC takes care of this phase. When the user submits its information to the cloud for accessing the data as of the cloud, the KGC produces keys. The KGC creates a public as well as private key intended for the users. The user's public keys are the exclusive uniqueness details that are openly recognized. The proposed work utilizes the MECC key generation system for generating the handler's public key together with the private key. The KG procedure of MECC is elucidated in the following paragraphs.

Modified Elliptic Curve Cryptography (MECC)

The MECC in the proposed work is employed for KG. The prevailing ECC is more intricate and also harder to execute, which augments the probability of implementation errors, augments the encrypted message's size, therefore reducing the algorithm's security. The MECC is generated to overwhelm these cons. The prevailing ECC generates just dual keys: i) a public key ii) private key for encryption, whilst the proposed MECC produces secret key to ameliorate the system's security. This secret key is multiplied with the encryption formula as well as divided by way of the decoding formula. If its intricacy of

encryption in addition to decryption is augmented, then it is harder to detect the actual data. It automatically enhances the data's SL.

The MECC's mathematical equation is rendered as:

$$X^2 = z^3 + mz + n \quad (1)$$

Here, m and n are the integers. Consider a point P_c as base points on the curves. The public key (p_k^u) is created as follows:

$$p_k^u = p_k^r * P_c \quad (2)$$

Here, p_k^r signifies the private key (i.e.) randomly generated. Therefore, there is likelihood that the attacker strikes the data. A good private key will give an excellent encryption process. Thus, for selecting an apt private key, the optimization is implemented. In the projected work, the CM-FPA is exploited to choose the optimal key for encryption.

Optimized key selection using CM-FPA

The flower pollinations algorithm (FPA) is fundamentally a nature-stimulated meta-heuristics algorithm. It emulates the pollination activities of flowers. The proposed one uses the hybridized adaptation of FPA. The prevailing FPA confronts the drawback of lower precision of optimization in addition to slow convergence in the later step; additionally, it is simple to descend into a local optimum. To conquer these sorts of problems of FPA, the proposed work hybridizes the Crossover and Mutation (CM) operators to the FPA. This CM operator is utilized to pick the best-optimized key. Therefore, the proposed work is labeled as CM-FP). Regenerative and cross-pollination involving pollen-carrying pollinators were considered as global pollination processes, whilst abiotic and self-pollination are considered as local pollination processes. A switching possibility controls local pollination and global pollination. Local pollination can account for a large proportion of entire pollination efforts related to physiological closeness and other considerations.

To develop a CM-FPA, there are '4' rules, which are summarized as:

- **Rule 1:** Pollen-carrying pollinators may soar a greater distance than Lévy flights, indicating that biotic and also cross-pollination is indeed a common pollination technique.
- **Rule 2:** The biotic as well as self-pollination could be signified as local pollination.
- **Rule 3:** Flower reliability might be implied as an equal to a reproduction probability that is balanced to the likeness of involved '2' flowers.
- **Rule 4:** The process of switching in local pollinations as well as global pollinations is handled by aid of switch probabilities $p \in [0,1]$.

To devise the updating equations, the aforementioned rules are needed to have been transmuted into o appropriate upgrading calculations. The CM-FPA phases that are needed for the execution of the variety of an arbitrary-key meant for the safe algorithm are elucidated here.

This algorithm encompasses '2' main steps: a) global pollination b) local pollination.

- For both the global pollination step (Rule 1), flower constancy (Rule 3) is being expressed mathematically as

$$g_i^{t+1} = g_i^t + \xi L(\lambda)(f_s^* - g_i^t) \quad (3)$$

Wherein, g_i^t implies the pollen i at iteration t, f_s^* implies the current optimum result, ξ signifies the scaling aspect that controls the stage size, $L(\lambda)$ signifies the Lévy flights-centred step size, that signify the stability of pollination. Since insects could traverse across greater range with varying distance measures, a Levy flight would be used to effectively mimic this phenomenon. A Levy flight is a normal

distribution wherein the stride durations have a Levy probability, which is a heavy-tailed distribution function. When a walking in a region with more than one dimensional is established, the measures taken are in isotropic different directions. L > 0 is precisely drawn as just a Levy distribution.

$$L \sim \frac{\lambda\sigma(\lambda)\sin(\frac{\pi\lambda}{2})}{\pi}\frac{1}{s^{1+\lambda}} \qquad (s >> s_0 > 0) \tag{4}$$

Wherein, $\sigma(\lambda)$ implies the gamma functions along with this distribution is applicable for larger steps $s > 0$.

- Local pollination (Rule 2), as well as flower constancy (Rule 3), can indeed be summed up as

$$g_i^{t+1} = g_i^t + \mu(g_j^t + g_k^t) \tag{5}$$

Wherein g_j^t and g_k^t are pollen as of disparate flowers of the identical plant kind. If g_j^t and g_k^t originate as of the similar species or chosen as of the similar populace, it consistently turns out to be a local random walk if μ is drawn as of a uniform distribution in $[0,1]$.

- Flower pollination have been happened at every scale: i) local and ii) global. As a result, a switch probability (Rule 4) or else proximity probability may be effectively used to switch between the both.
- After that, fitness value is computed for the new best solution. On the off chance that the fitness of the new solution is superior compared to the old best solution which in turn it is swapped by means of new one. This fitness computation is done at every iteration t.
- Next, a CM function is implemented, with this, the algorithm is competent to pushing the pollen individuals in the directions of its “fittest-and-closest” neighbour. Crossover is done by utilizing ‘2’ crossover points (c_p, c_q).

$$c_p = \frac{\left|g_i^{t+1}\right|}{3} \tag{6}$$

$$c_q = c_p + \frac{\left|g_i^{t+1}\right|}{3} \tag{7}$$

- The mutation modifies a section of the best solution arbitrarily. In a mutation procedure, the worst solution is separated centred upon the fitness value and switched with a new arbitrary solution is produced. After that, fitness is computed for the new finest solution. Lastly, the best solution is updated centred upon the fitness value.

These steps are recurring until the termination condition. The proposed CM-FPA’s flowchart is displayed in Fig. 2.

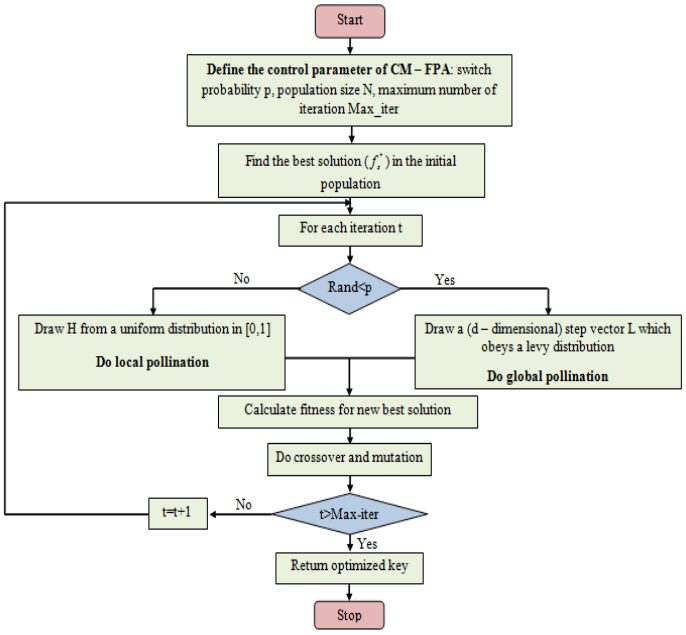


Figure 2: Flow chart of proposed CM-FPA

Like this, the optimized private key is chosen by utilizing CM-FPA. This key is employed in the MECC key generation procedure for creating an optimized public key. The KG procedure is rendered below.

Keygen ($\vec{p}_k^r$, $\vec{p}_k^u$, $\vec{s}_k$): Provided an optimized private key $\vec{p}_k^r$, public key together with the secret key is created as:

$$\vec{p}_k^u = \vec{p}_k^r * P_c \tag{8}$$

Wherein, $\vec{p}_k^u$ implies the optimized public key. Now, the secret key is created by means of summing the $\vec{p}_k^u$, $\vec{p}_k^r$ and also P_c that is written as:

$$\vec{s}_k = \sum(\vec{p}_k^u, \vec{p}_k^r, P_c) \tag{9}$$

Wherein, $\vec{s}_k$ implies the optimized secret key.

The public, private, along with secret key are produced by means of the KGC. Whenever a new user registers their information to the cloud, the KGC creates $\vec{p}_k^u$, $\vec{p}_k^r$ and $\vec{s}_k$. Regard that the sender's optimized private as well as the public key is as $\vec{p}_{k\,S}^r$ and $\vec{p}_{k\,S}^u$ correspondingly, and as well the receiver's optimized private key in addition to the public key is as $\vec{p}_{k\,R}^r$ and $\vec{p}_{k\,R}^u$ correspondingly. These optimized keys are produced with a higher-SL. Even if there is probability that the scheme undergoes the hazard of IKGA, it further enhances the SL by means of converting the sender as well as the receiver's optimized keys into the hash code. The MD5 is employed for hash code conversion.

MD5 Algorithm

The most prevalent method for verifying the authenticity of documents is MD5. Some implementations improve the MD5 algorithm by appending a salt value to the plaintext or repeating the hashing algorithm. In summarize, MD5 is used to verify the file's integrity, that might or might not

indicate a network issue. It is primarily used to verify the reliability of downloadable in order to make sure that no one else has interfered with them.

MD5 algorithm has been created through the chief motivation towards security in which it accepts any size inputs and also outputs a 128bit hash value. MD5 is rather fast compared to other editions of a message digest together with SHA-512. MD5 (Message Digest) and SHA512 (Secure Hash Algorithm) square measure the hash functions used everywhere. When compared to the performance of SHA1, MD5 is faster. SHA, on the other hand, is more secure than MD5. MD5 generates the hash code to the equivalent key via '3' steps: padding and append length, Buffer initialization, Process message on 16word bits. These '3' are elucidated as the following 3 steps.

Step 1: Padding bits and append length

Initially, the MD5 splits the input into 512 bits blocks each that is further split into sixteen blocks, each with 32 bits and generates the 128 bits message digest that is a compilation of '4' blocks where each hold 32 bits.

Step 2: Initialize MD buffer

A '4'-word buffer (A, B, C, D) is feeble to gauge the hashcode. They are initialized as:

Word A: 01 23 45 67

Word B: 89 *ab cd ef*

Word C: *fe dc ba* 98

Word D: 76 54 32 10

Step 3: Processing message in 16-word block

MD5 utilizes the supplementary functions that accept the '3' 32-bit numbers as input and generates 32bit as output. These utilize logical operators, such as OR, XOR, NOR.

$f(x, y, z) = (x\text{y})(not(\text{x})\text{z})$

$g(x, y, z) = (xz)\ or\ (y\ \text{not}\ (\text{z}))$

$h(x, y, z) = x\ \text{or}\ y\ \text{or}\ z$

$i(x, y, z) = y\ \text{xor y xor z}$

Using the '4' auxiliary element, the elements of a '4' buffers (A, B, C, D) have become blended mostly with words of the input (f, g, h, i). There seem to be four rounds, each involving sixteen basic core functions. Fig. 3 exhibits one MD5 operation.

Subsequently, the buffer A, B, C, D encompasses the MD5 productivity commencing with inferior bit A and also concluding with superior bit D. The output as of MD5 stands as a 128 bits message digests value. It is written as:

$$MDH_F : \vec{K}_i \rightarrow H(k) \tag{10}$$

Wherein, MDH_F implies the MD5 hash function, $\vec{K}_i$ implies the input optimized key ($\vec{p}^{\,r}_{kS}$, $\vec{p}^{\,u}_{kS}$ s, $\vec{p}^{\,u}_{kR}$, $\vec{p}^{\,r}_{kR}$ and $\vec{s}_k$), $H(k)$ implies the MD5's output and is written mathematically as:

$$H(k) = H_f(\vec{K}) \tag{11}$$

Wherein, $H(k)$ signifies the $\vec{K}$'s hash value. The assessment of hashcode is extremely hard for attackers to spot the original data. The attained hashed private key, public key, in addition to the secret

key are taken into the encryption phase to carry out encryption. However, prior to encryption, some essential setup process is done to ameliorate the system effectiveness.

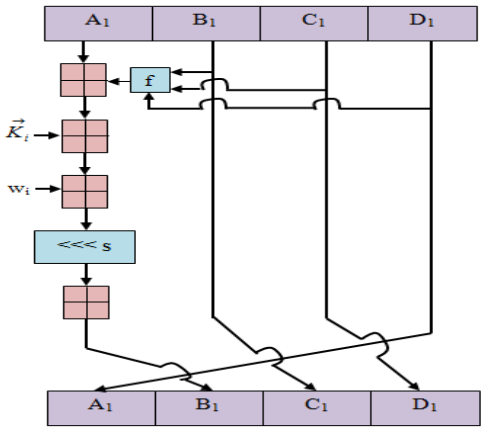


Figure 3: One MD5 iteration

Setup Phase

Here, as of the document (D), the system extracts the significant keywords. Keyword extraction is largely utilized to lessen the data searching time. This setup procedure is done prior to encrypting the file since retrieving the targeted encrypted data centred upon the keyword is not probable; in addition, the user authentication is executed by utilizing the keywords.

Setup ($\vec{k}_{Wi}$): Provided the document D, the extracted keywords are mathematically written as,

$$\vec{k}_{Wi} = \{k_{W1}, k_{W2}, k_{W3}, ..., k_{Wn}\} \tag{12}$$

Wherein, $\vec{k}_{Wi}$ implies the extracted keywords of D, k_{Wn} implies the n-number of keywords as of the file D.

3.4.2 Encryption Phase

Subsequent to extracting the significant keywords, *DO* encrypts the file by means of the sender's (*DO*) hashed private and secret key in addition to the receiver's hashed public key. It ameliorates the data's SL automatically. The proposed work, for data encryption, employed the MECC. This MECC is intricate enough for proscribing the attacker from deducing the plaintext as of the ciphertext as well as the encryption key. In a proposed encryption procedure, a secret key is employed to encrypt the data that is multiplied with the [encrypted] data; and in a Decryption Time (DT), it is split as of the data. The hash function does not utilise a key or anything; instead, it scrambles data using a computational formula. The shared secret technique of encryption, which affects a specific key to encrypt and decrypt data, is also known as symmetric encryption. The encryption procedure of the proposed MECC is shown below.

Keywordencrypt ($\vec{k}_{Wi}$, $\hat{p}^{r}_{kS}$, $\hat{p}^{u}_{kR}$, $\hat{s}_k$): Provided the extracted keyword set $\vec{k}_{Wi}$, a sender's hashed private key $\hat{p}^{r}_{kS}$, receiver's hashed public key $\hat{p}^{u}_{kR}$ and hashed secret key $\hat{s}_k$, it produces a keyword ciphertext. A monoalphabetic replacement is a type of keywords encryption. The character matchings of the cypher language to the regular script are determined by a keyword, which serves as the key. The encrypted data contains '2' ciphertexts that are mathematically signified as below:

$$E(\vec{k}_{Wi})_1 = (rd * P_c) * \hat{s}_k \tag{13}$$

$$E(\vec{k}_{Wi})_2 = \vec{k}_{Wi} + (rd * \hat{p}^r_{kS}\hat{p}^u_{kR}) * \hat{s}_k \quad (14)$$

$$E(\vec{k}_{Wi}) \rightarrow \{E(\vec{k}_{Wi})_1, E(\vec{k}_{Wi})_2\}$$

Documentencrypt (D , $\hat{p}^r_{kS}$, $\hat{p}^u_{kR}$, $\hat{s}_k$ ***):*** Provided the document D , a sender's hashed private key $\hat{p}^r_{kS}$, receiver's hashed public key $\hat{p}^u_{kR}$ as well as hashed secret key $\hat{s}_k$, it produces a document ciphertext.

$$E(D)_1 = (rd * P_c) * \hat{s}_k \quad (15)$$

$$E(D)_2 = D + (rd * \hat{p}^r_{kS}\hat{p}^u_{kR}) * \hat{s}_k \quad (16)$$

$$E(D) \rightarrow \{E(D)_1, E(D)_2\}$$

where, $E(D) \rightarrow$ Ciphertext of D , $E(\vec{k}_{Wi}) \rightarrow$ Ciphertext of $\vec{k}_{Wi}$, $rd \rightarrow$ Random number, which is in the gamut of 1 to n-1

Subsequent to attaining the encrypted data, the DO uploads the $E(D)$ to the CS.

3.4.3 Index Table Phase

The index table is often a database which always enhances the effectiveness of data retrieval functions. It is stated already that if any user needs to search any document as of a CS centred on the keyword, the server cannot return any data related to that searched keyword since the document is on the encrypted format. Thus, a DO constructs a searchable index table I_s to amass vital information regarding the document to ameliorate the searching efficiency.

Indextable $\left(I_s = \{\vec{k}_{Wi}, E(D)_L, E(\vec{k}_{Wi}), \hat{p}^u_{kS}, \hat{s}_k\ \}\right)$ **:** Given Ciphertext location $E(D)_L$, Keyword Ciphertext $E(\vec{k}_{Wi})$ and its corresponding extracted keyword set $\vec{k}_{Wi}$, *DO* 's hashed public key $\hat{p}^u_{kS}$ as well as hashed secret key $\hat{s}_k$, it produces an index table I_s .

This index table I_s sub-contracted to the cloud storage.

3.4.4 Trapdoor Generation Phase

The utmost significant function of the system is the creation of the trapdoor along with its security. Here, the *DR* transmits an enviable trapdoor for searching the encrypted data over a CS. The *DR* produces a trapdoor utilizing his/her hashed private key as well as keyword $\vec{k}_W$ and then, transmit it to the CS.

Trapdoor ($k_W, p^r k_R$ ***):*** Provided *DR* 's private key $p^r k_R$ as well as search keyword $\vec{k}_W$, it generated trapdoor t_d .

$$t_d = Trapdoor(k_W, p^r k_R) \quad (17)$$

Subsequent to sending the trapdoor to the CS, the CS checks that the keyword ($\vec{k}_W$) in matching to the trapdoor is equivalent to the keyword corresponds to the keyword ciphertext keyword ($\vec{k}_{Wi}$). If $\vec{k}_W$ =

$\vec{k}_{Wi}$, then the server endeavours to decrypt the encrypted keyword $E(\vec{k}_{Wi})$ utilizing the receiver's hashed private key as well as the sender's hashed public key. The keyword decryption is performed, which possibly means that the requested user is a legitimate user to access the file. If the keyword is flawlessly decrypted, then the server confirms that the receiver is an approved user. If not, the server redirects the registration. The decryption procedure is elucidated in the below section.

3.4.5 Decryption Phase

Here, the decryption is done. The authenticated user can recuperate the file as of the CS. The proposed method used the MECC for the decryption. The MECC utilizes the receiver's hashed private key, hashed secret key, along with senders hashed public key. Here, '2' decryption process: i) keyword decryption, ii) document decryption is done. The keyword decryption is executed for user authentication. The document decryption is executed only if the receiver is a certified user. The decryption procedure of MECC is rendered below.

Keyworddecrypt ***(*** $\{E(\vec{k}_{Wi})_1, E(\vec{k}_{Wi})_2\}$, $\hat{p}^r_{kR}$, $\hat{s}_k$, $\hat{p}^u_{kS}$ ***):*** Given keyword ciphertext1 $E(\vec{k}_{Wi})_1$, keyword ciphertext2 $E(\vec{k}_{Wi})_2$, DR 's hashed private key $\hat{p}^r_{kR}$, DO 's hashed public key $\hat{p}^u_{kS}$ and hashed secret key $\hat{s}_k$, it produces the decrypted keyword as follows,

$$\vec{k}_{Wi} = \left(\left(E(\vec{k}_{Wi})_2 - \hat{p}^r_{kR}\hat{p}^u_{kS}\right) * E(\vec{k}_{Wi})_1\right) \Big/ \hat{s}_k \tag{18}$$

Wherein, $\vec{k}_{Wi}$ implies the decrypted keyword set. In the case of an authorized user, the server returns the encrypted file $E(D)$, which is associated with the searched keyword. Then, the receiver downloads the encrypted files from the CS.

Subsequent to a successful retrieval of encrypted data, the user decrypts the files as follows introduction:

Documentdecrypt ***(*** $\{E(D)_1, E(D)_2\}$, $\hat{p}^r_{kR}$, $\hat{s}_k$, $\hat{p}^u_{kS}$ ***):*** Given keyword ciphertext1 $E(D)_1$, keyword ciphertext2 $E(D)_2$, DR 's hashed private key $\hat{p}^r_{kR}$, DO 's hashed public key $\hat{p}^u_{kS}$ as well as hashed secret key $\hat{s}_k$, it produces the decrypted keyword as follows:

$$D = \left(\left(E(D)_2 - \hat{p}^r_{kR}\hat{p}^u_{kS}\right) * E(D)_1\right) \Big/ \hat{s}_k \tag{19}$$

Wherein, D signifies the original document. This way the DR recovers the encrypted as of the CS. The receiver receives a document with a safe authentication scheme. The CL-HPAEKS scheme well resists the IKGA than other schemes by utilizing a hashing function.

The proposed method's performance is analyzed for disparate file sizes. The assessment of CL-HPAEKS is done, which is analyzed in the section below.

4 Results and Discussion

This sector estimates the CL-HPAEKS's efficiency and also contrasts it with the associated methods as of the features of computational expenditure, fitness, Encryption Time (ET), DT, key generation time (KGT), Memory Usage (MU) on encryption, decryption, as well as file upload along with download time. The time it takes to generate keys in cryptography is known as key generation time. To encryption and decryption whichever data has been encoded or decoded, a key is being used. The CL-HPAEKS is executed in the JAVA having the following machine configuration:

Processor: Intel i5/core i7

CPU Speed Rate: 3.20 GHz
OS: Windows 7
RAM: 4 GB

4.1 Performance Analysis

This section assesses the proposed schemes' work in CL-HPAEKS, like encryption schemes, optimization schemes, along with experimental outcomes.

4.1.1 Upload Time and Download Time of CL-HPAEKS

Obviously, upload time along with the download time of CL-HPAEKS is examined. The Fig. 4 exhibits the proposed technique's performance concerning upload and downloads time.

Upload Time: It is the time taken for uploading the encrypted file to the CS. It is mathematically expressed as:

$$T_{UD} = u'(t)_{end} - u'(t)_{start} \tag{20}$$

where, $u'(t)_{end} \rightarrow$ Uploading ending time

$u'(t)_{start} \rightarrow$ Uploading starting time

Download time: As of the CS, it is described as the time needed to retrieve the encrypted file, which is calculated as:

$$T_{DD} = d'(t)_{end} - d'(t)_{start} \tag{21}$$

where, $d'(t)_{end} \rightarrow$ Downloading ending time

$d'(t)_{start} \rightarrow$ Downloading starting time

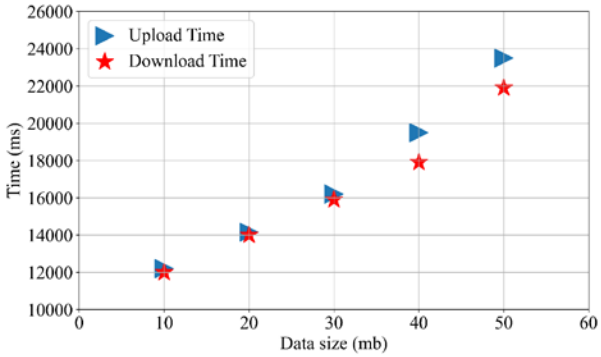


Figure 4: Performance analyses of proposed CL-HPAEKS regarding time of upload and time of download

Fig. 4 evinces the CL-HPAEKS's performance when it comes to file upload and download times. Unequal data sizes gamut [10,50] mb is engaged for estimation. The CL-HPAEKS utilize 12456 ms for uploading a 10mb files and also downloading it in 11887 ms. The CL-HPAEKS's upload time for 20, 30, 40, 50 mb data is 14235, 16275, 19486, 23434 ms as well as the download time of these data is 13886, 15534, 17778 along with 21464 ms correspondingly. The upload along with download time augments with the augmentation of the data size.

4.1.2 Performance Analysis of CM-FPA

The proposed CM-FPA's performance that is utilized in the KG was contrasted with the prevailing methods, says GA, FPA, CM-FPA. In the proposed work, the CM-FPA is employed to choose the best KG parameters that decrease the KGT and also augments the SL. The performance contrast of proposed CM-FPA with the existent GA along with FPA concerning fitness function as well as KGT is rendered in the table.

The fitness value along with KGT is taken for disparate iteration that ranges [5, 50]. On evaluating the table, it is clear that the proposed methods render superior performance to the prevailing GA along with FPA. Fitness's comparison graph is exhibited in Fig. 5.

The Fig. 5 calculates the proposed CM-FPA's performance with that of the existent methods, say GA along with FPA, concerning fitness value. The fitness value enhances system efficiency. For 5 iterations, the proposed CM-FPA's fitness value is 523; however, the prevailing GA along with FPA has 423 and 406 of fitness value, correspondingly. For 25 iterations, the GA along with FPA achieve 701 as well as 745 of fitness value, while the proposed one has a higher fitness value of 867. For lasting iterations, the proposed one's fitness value was high than the prevailing methods.

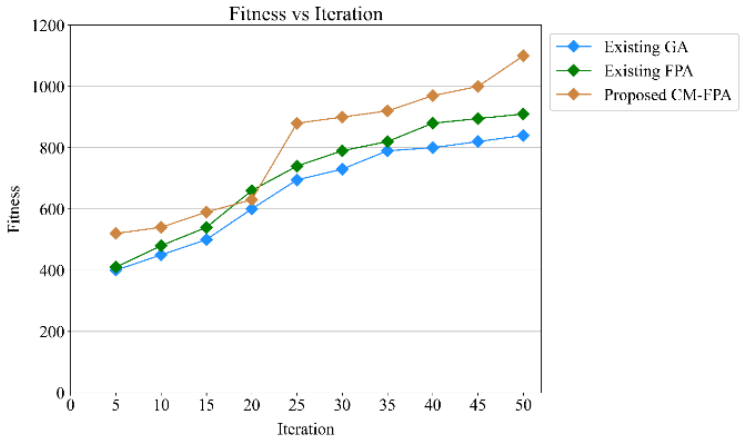


Figure 5: Fitness analysis of proposed CM-FPA algorithm with its prevailing algorithm

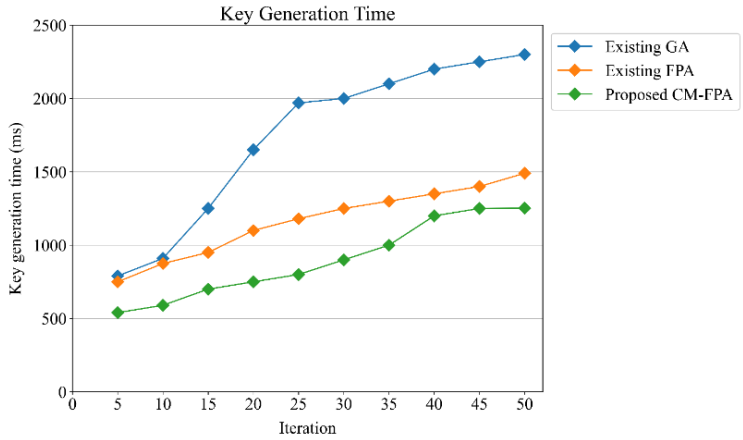


Figure 6: Performance analysis of proposed CM-FPA with existing techniques

Fig. 6 shows the proposed along with existing methods' performance concerning KGT. The key generation time (T_{KG}) can well be computed as

$$T_{KG} = KG(t)_{end} - KG(t)_{start} \tag{23}$$

where, $KG(t)_{end} \rightarrow$ Key generation termination time

$KG(t)_{start} \rightarrow$ Key generation initial time

When the KGT is lower, the system's speed may augment. The KGT is taken for disparate iterations. For every iteration, the proposed CM-FPA obtains lesser time for producing a key when contrasted with the prevailing methods, say, GA along with FPA. Therefore, it was established that the proposed CM-FPA was effective when contrasted to existent GA and FPA. The result is as shown in Fig. 7.

4.1.3 Performance Analysis of MECC

The performance comparison is done on the proposed MECC along with the existing RSA and ECC centred on ET, DT, and MU on encryption in addition to decryption. A public key and a private key are used in RSA. Everyone has access to the entire key, which will be used to secure communications. With significantly smaller data size, ECC can give the same cryptography effectiveness as an RSA-based solution.

(a) Encryption time and Decryption time

Encryption time: It is a total time taken to produce encrypted data as of the plaintext. The encryption time can well be calculated as,

$$T_{encrypt} = E'(t)_{end} - E'(t)_{start} \tag{24}$$

where, $E'(t)_{end} \rightarrow$ Encryption ending time

$E'(t)_{start} \rightarrow$ Encryption starting time

$D'(t)_{start} \rightarrow$ Decryption starting time

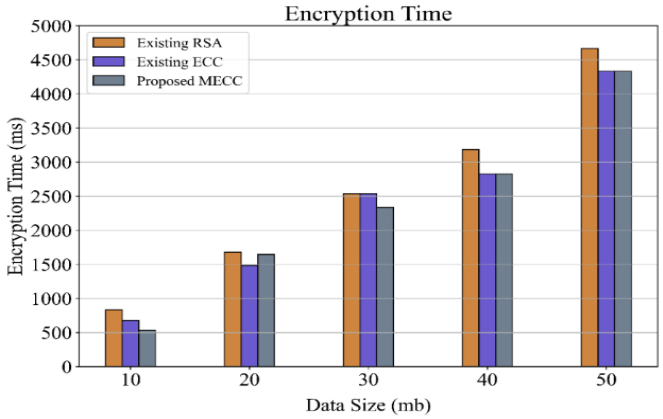
Decryption time: It is a total time taken to produce original data as of the encrypted data. The decryption time can well be calculated as,

$$T_{decrpt} = D'(t)_{end} - D'(t)_{start} \tag{25}$$

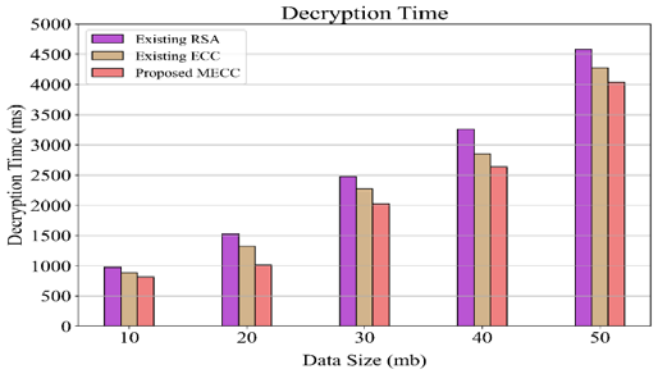
where, $D'(t)_{end} \rightarrow$ Decryption ending time

Table 3: Encryption and Decryption time comparison of the proposed and existing techniques

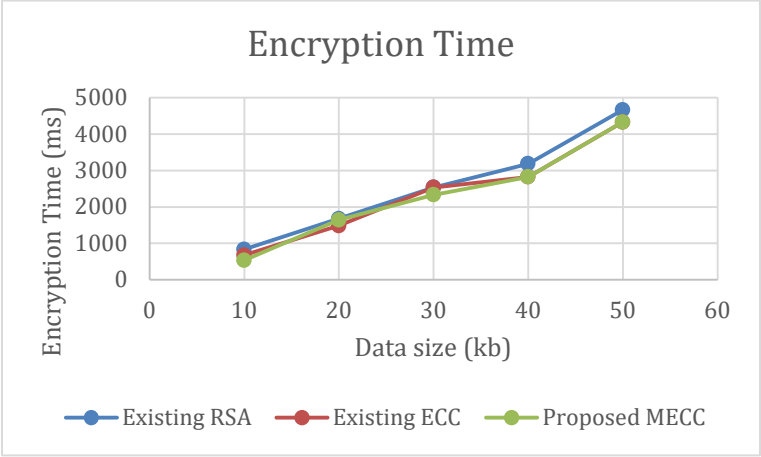
Data size (kb)	**Encryption Time (ms)**			**Decryption Time (ms)**		
	Existing RSA	*Existing ECC*	*Proposed MECC*	*Existing RSA*	*Existing ECC*	*Proposed MECC*
10	834	675	534	973	885	812
20	1678	1485	1643	1523	1322	1012
30	2534	2534	2334	2476	2272	2023
40	3186	2824	2824	3258	2846	2634
50	4666	4334	4334	4583	4273	4041


Encryption Time
Existing RSA
Existing ECC
Proposed MECC
Encryption Time (ms)
5000
4500
4000
3500
3000
2500
2000
1500
1000
500
0
10
20
30
40
50
Data Size (mb)

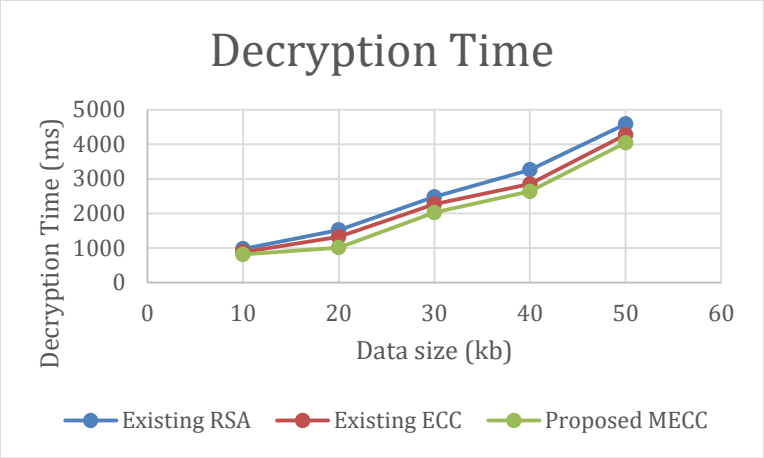
(a)


Decryption Time
Existing RSA
Existing ECC
Proposed MECC
Decryption Time (ms)
5000
4500
4000
3500
3000
2500
2000
1500
1000
500
0
10
20
30
40
50
Data Size (mb)

(b)


Encryption Time
Encryption Time (ms)
5000
4000
3000
2000
1000
0
0
10
20
30
40
50
60
Data size (kb)
Existing RSA
Existing ECC
Proposed MECC

(c)



(d)

Figure 7: Performance analysis of column chart format: (a) ET (b) DT; Performance analysis of line chart format: (c) ET (d) DT

(b) Security level

The proposed and existing techniques' SL is plotted in Fig. 8.

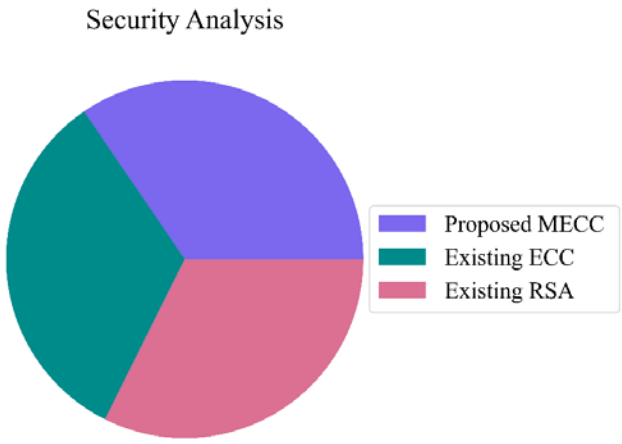


Figure 8: SL analysis of proposed MECC and existing RSA and ECC

Fig. 8 estimates the SL of MECC, ECC, along with RSA. The SL analysis measured the cryptographic primitive ability of the proposed cypher hash functions MECC with existing methods RSA and ECC in which MECC achieves an accuracy of 96% but ECC and RSA obtain only 92% and 90% accuracy. As compared with proposed method the performance of ECC and RSA is lower.

(c) Memory usage time on encryption and decryption

Here, the memory space that is taken by the CPU amid the encryption in addition to decryption process is examined. This MU is gauged in kilobytes. The memory usage on encryption and decryption is computed as,

$$(M_U)_{encrypt} = M_{total} - E'(M)_{remain} \quad (26)$$

$$(M_U)_{decrypt} = M_{total} - D'(M)_{remain} \quad (27)$$

Here, $M_{total} \rightarrow$ Total memory space

$E'(M)_{remain}$ and $D'(M)_{remain} \rightarrow$ Remaining memory subsequent to encryption and decryption correspondingly.

The performance comparison of the proposed together with prevailing techniques is exhibited in Fig. 9.

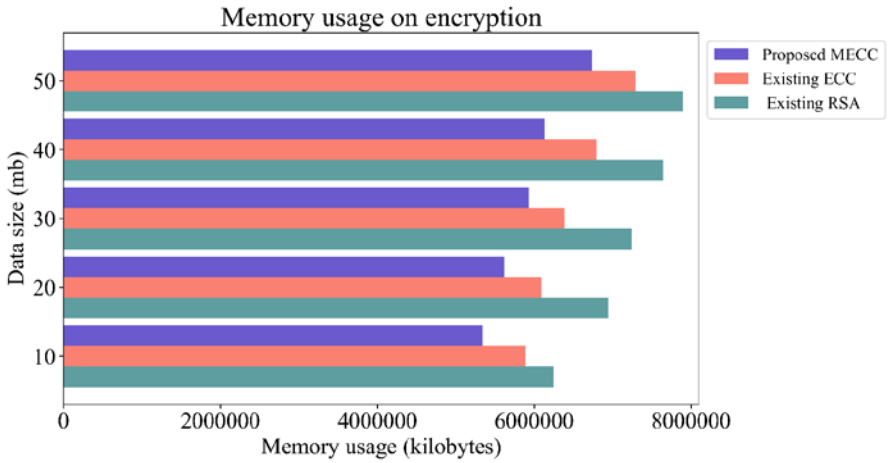


Figure 9: Analysis of MU of proposed MECC with existing ECC and RSA on encryption

The proposed work's turn with that of the prevailing encryption method concerning MU is evaluated in Fig. 9. Amid encrypting the 10 Mb file, the proposed MECC takes 5333976 kb of CPU memory. However, the prevailing RSA along with ECC takes 6245452 kb as well as 5888756 kb correspondingly. If the MU size is augmented, the system efficiency is reduced. For every other value, the proposed MECC used lesser memory contrasted to the prevailing method. Thus, it can well be said that the proposed MECC is more effective contrasted with ECC along with the RSA method.

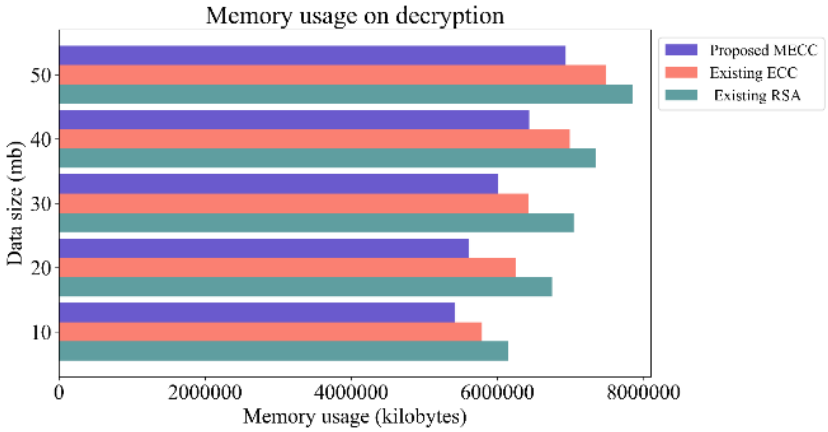


Figure 10: Memory usage analysis of proposed MECC with existing ECC and RSA on decryption

In Fig. 10, the proposed MECC is contrasted with the prevailing ECC along with RSA. The MU of proposed MECC utilizes 5421754 kb memory for 10mb data amid decryption, while, in the prevailing method (ECC along with RSA), the MU by the CPU is 5822345 kb along with 6134843 kb, which is high than the proposed one. For every data size, the proposed technique utilizes lesser memory size contrasted with the prevailing ECC along with RSA. The outcome obviously demonstrates that the proposed one utilize lesser memory space on decryption for all data size when contrasted with the prevailing one.

5 Conclusion

Here, a CL-HPAEKS system is proposed for the cloud database method. This system supports multiple DOs to exchange information with such a single user. The core goal of this document is to resourcefully guard the cloud storage structure as of the IKGA. The proposed CL-HPAEKS utilizes an MD5 for an effective user authentication system. In addition, a MECC was adopted to ameliorate the SL in opposition to IKGA. For performance examination, the proposed work is contrasted with the prevailing methods regarding several performance metrics. As of the performance examination, it was clear that the proposed CL-HPAEKS has the high-SL than the prevailing method. The proposed methods attain SL up to 96%, and the encryption time as well as decryption time of the proposed work is less than the prevailing method. The proposed CL-HPAEKS were, therefore, established to be extra effective and secure to oppose IKGA. The forthcoming plan is to focus on this specific context in the industrial internet of things, taking into account measurements to improve sensor energy efficiency and shorten keyword search times.

Ethical Statement: Hereby, we assure that this material is the authors' own original work, which has not been previously published elsewhere.

Data Availability Statement: My manuscript has no associate data.

Informed Consent Statement: I voluntarily agree to participate in this research journal.

Funding Statement: None.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] M. Ma, D. He, S. Fan and D. Feng, "Certificateless searchable public key encryption scheme secure against keyword guessing attacks for smart healthcare," *Journal of Information Security and Applications*, vol. 50, no. 50, 102429, 2020.

[2] R. Elhabob, Y. Zhao, A. Hassan and H. Xiong, "PKE-ET-HS: Public key encryption with equality test for heterogeneous systems in IoT," *Wireless Personal Communications*, vol. 113, pp. 313–335, 2020.

[3] D. N. Wu, Q. Q. Gan and X. M. Wang, "Verifiable public key encryption with keyword search based on homomorphic encryption in multi-user setting," *IEEE Access*, vol. 6, pp. 42445–42453, 2018.

[4] M. Ma, D. He, N. Kumar, K. K. Raymond Choo and J. Chen, "Certificateless searchable public key encryption scheme for industrial Internet of Things," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 2, pp. 759–767, 2017.

[5] N. Cao, C. Wang, M. Li, K. Ren and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 222–233, 2013.

[6] S. H. Seo, M. Nabeel, X. Ding and E. Bertino, "An efficient certificateless encryption for secure data sharing in public clouds," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 9, pp. 2107–2119, 2013.

[7] B. Wang, W. Song, W. Lou and Y. T. Hou, "Inverted index based multi-keyword public-key searchable encryption with strong privacy guarantee," in *2015 IEEE Conf. on Computer Communications (INFOCOM)*, pp. 2092–2100, 2015.

[8] P. Jiang, F. Guo and Y. Mu, "Efficient identity-based broadcast encryption with keyword search against insider attacks for database systems," *Theoretical Computer Science*, vol. 767, pp. 51–72, 2019.

[9] Y. Zhou and B. Yang, "Leakage-resilient CCA2-secure certificateless public-key encryption scheme without bilinear pairing," *Information Processing Letters*, vol. 130, pp. 16–24, 2018.

[10] Z. Xia, X. Wang, X. Sun and Q. Wang, "A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 340–352, 2015.

[11] L. Xu, J. Li, X. Chen, W. Li, S. Tang and H.T. Wu, "Tc-PEDCKS: Towards time controlled public key encryption with delegatable conjunctive keyword search for Internet of Things," *Journal of Network and Computer Applications*, vol. 128, pp. 11–20, 2019.

[12] E. Uwizeye, J. Wang, Z. Cheng and F. Li, "Certificateless public key encryption with conjunctive keyword search and its application to cloud-based reliable smart grid system," *Annals of Telecommunications*, vol. 74, no. 7–8, pp. 435–449, 2019.

[13] R. M. Daniel, E. B. Rajsingh and S. Silas, "An efficient eCK secure certificateless authenticated key agreement scheme with security against public key replacement attacks," *Journal of Information Security and Applications*, vol. 47, no. 39, pp. 156–172, 2019.

[14] T. Y. Wu, C. Meng, C. M. Chen, K. H. Wang and J. S. Pan, "On the security of a certificateless public key encryption with keyword search," in *Int. Conf. on Intelligent Information Hiding and Multimedia Signal Processing*, Springer, Cham, pp. 191–197, 2017.

[15] N. Pakniat, "Designated tester certificateless encryption with keyword search," *Journal of Information Security and Applications*, vol. 49, no. 2, pp. 102394, 2019.

[16] S. Sree Vivek, "Stateful certificateless public key encryption with application in public cloud," in *Int. Conf. for Information Technology and Communications*, Springer, Cham, pp. 130–149, 2015.

[17] Y. Lu and J. Li, "Constructing pairing-free certificateless public key encryption with keyword search," *Frontiers of Information Technology & Electronic Engineering*, vol. 20, no. 8, pp. 1049–1060, 2019.

[18] Q. Huang and H. Li, "An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks," *Information Sciences*, vol. 403, pp. 1–14, 2017.

[19] S. K. Hafizul Islam, M. S. Obaidat, V. Rajeev and R. Amin, "Design of a certificateless designated server based searchable public key encryption scheme," in *Int. Conf. on Mathematics and Computing*, Springer, Singapore, pp. 3–15, 2017.

[20] H. Li, Q. Huang, J. Shen, G. Yang and W. Susilo, "Designated-server identity-based authenticated encryption with keyword search for encrypted emails," *Information Sciences*, vol. 481, no. 13, pp. 330–343, 2019.

[21] W. Chen, L. Zhang, Q. Bo, Q. Wu and H. Zhang, "Certificateless one-way authenticated two-party key agreement protocol," in *5th Int. Conf. on Information Assurance and Security*, vol. 1, pp. 483–486, 2009.

[22] L. Wu, Y. Zhang, M. Ma, N. Kumar and D. He, "Certificateless searchable public key authenticated encryption with designated tester for cloud-assisted medical Internet of Things," *Annals of Telecommunications*, vol. 74, no. 7–8, pp. 423–434, 2019.

[23] R. Chen, Y. Mu, G. Yang, F. Guo and X. Wang, "Dual-server public-key encryption with keyword search for secure cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 4, pp. 789–798, 2015.

[24] L. Guo and W. C. Yau, "Efficient secure-channel free public key encryption with keyword search for EMRs in cloud storage," *Journal of Medical Systems*, vol. 39, no. 2, pp. 11, 2015.

[25] Y. Zhang, L. Wen, Y. Zhang, and C. Wang, "Designated server certificateless deniably authenticated encryption with keyword search," *IEEE Access*, vol. 7, pp, 146542–146551, 2019.

[26] D. He, M. Ma, S. Zeadally, N. Kumar and K. Liang, "Certificateless public key authenticated encryption with keyword search for industrial internet of things," IEEE Transactions on Industrial Informatics, vol. 14, no. 8, pp. 3618–3627, 2017.

[27] B. Qin, Y. Chen, Q. Huang, X. Liu, and D. Zheng, "Public-key authenticated encryption with keyword search revisited: Security model and constructions," *Information Sciences*, vol. 516, pp. 515–528, 2020.

[28] R. Elhabob, Y. Zhao, I. Sella and H. Xiong, "An efficient certificateless public key cryptography with authorized equality test in IIoT," Journal of Ambient Intelligence and Humanized Computing, vol. 11, no. 4, pp. 1–19, 2019.

[29] H. Xiong, Q. Mei and Y. Zhao, "Efficient and provably secure certificateless parallel key-insulated signature without pairing for IIoT environments," *IEEE Systems Journal*, vol. 14, no. 1, pp. 310–320, 2019.

[30] G. Chen, J. Zhao, Y. Jin, Q. Zhu, C. Jin *et al.,* "Certificateless deniable authenticated encryption for location-based privacy protection," *IEEE Access*, vol. 7, pp. 101704–101717, 2019.

[31] V. Dhanakoti and R. Nedunchezhian, "Streamlined alarms for intrusion recognition system," *International Journal of Intelligent Information Technologies*, vol. 11, no. 2, pp.40–54, 2015.

[32] A. Ponmalar and V. Dhanakoti. "Phishing attack in social network environment," *Journal of Advanced Research in Dynamical & Control Systems*, vol. 10, 2018.

[33] M. Mayuranathan, M. Murugan and V. Dhanakoti, "Best features based intrusion detection system by RBM model for detecting DDoS in cloud environment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 5, pp. 1–11, 2021.

[34] N. Deepa, P. Vijayakumar, B. S. Rawal and B. Balamurugan, "An extensive review and possible attack on the privacy preserving ranked multi-keyword search for multiple data owners in cloud computing," in *2017 IEEE Int. Conf. on Smart Cloud*, 2017.

[35] R. Qazi and I. A. Khan, "Data security in cloud computing using elliptic curve cryptography," *International Journal of Soft Computing and Engineering*, vol. 2, no. 3, 2012.

[36] S. Kota, V. N. Padmanabhuni, K. Budda and K. Sruthi, "Authentication and encryption using modified elliptic curve cryptography with particle swarm optimization and cuckoo search algorithm," *Journal of The Institution of Engineers (India): Series B*, vol. 99, no. 4, pp. 343–351, 2018.

[37] M. A. El Hafez Bakr, M. A. Mokhtar and A. El Sherbini Takieldeen, "Modified elliptic curve cryptography in wireless sensor networks security," in *2018 28th Int. Conf. on Computer Theory and Applications*, Alexandria, Egypt, 2018.

[38] M. Thangapandiyan, P. M. Anand and K. S. Sankaran, "Enhanced cloud security implementation using modified ECC algorithm," in *2018 Int. Conf. on Communication and Signal Processing*, pp. 1019–1022, 2018.

[39] D. R. Rani and G. Geethakumari, "Secure data transmission and detection of anti-forensic attacks in cloud environment using MECC and DLMNN," *Computer Communications*, vol. 150, pp. 799–810, 2020.

[40] M. Abdel-Basset and L. A. Shawky, "Flower pollination algorithm: A comprehensive review," *Artificial Intelligence Review*, vol. 52, no. 4, pp. 2533–2557, 2018.

[41] Y. Chen and D. Pi, "An innovative flower pollination algorithm for continuous optimization problem," *Applied Mathematical Modelling*, vol. 83, no. 6, pp. 237–265, 2020.

[42] V. Pandey and V. K. Mishra, "Architecture based on MD5 and MD5-512 bit applications," *International Journal of Computer Applications*, vol. 74, no. 9, pp. 29–33, 2013.

[43] P. Goyal, H. Makwana and N. Karankar, "MD5 and ECC encryption based framework for cloud computing services," in *2019 Third Int. Conf. on Inventive Systems and Control*, pp. 195–200, 2019.