Journal of Information Technology Management

Print ISSN: 2008-5893

Online ISSN: 2423-5059

Enhanced Blockchain-based Key Generation using Butterfly Optimization Algorithm for Efficient Data Sharing in Cloud Computing

P. Anbumani *

*Corresponding Author, Department of Computer Science and Engineering, Karpagam Academy of Higher Education, Tamilnadu, India. E-mail: anbuanc@gmail.com

R. Dhanapal

Department of Computer Science and Engineering, Karpagam Academy of Higher Education, Tamilnadu, India. E-mail: dhanapal.r@kahedu.edu.in

Abstract

Cloud Computing, employed in various applications and services, refers to using computational resources as a service depending on customer needs via the Internet. The computing paradigm is built on data outsourcing to third-party-controlled data centers. Despite the significant developments in Cloud Services and Applications, various security vulnerabilities remain. This research proposes the EBBKG Model for Efficient Data Sharing in Cloud. For secure data sharing in the cloud, the approach combines BBKG with ABS. The method offers good data management that efficiently specifies the subsequent processing processes. The paradigm imposes encrypted access control, along with specific enhanced access capabilities. Secondly, the user's privacy may be adequately protected with a secure authentication paradigm that employs ABS to safeguard the user's private data. The key is optimized using BOA to enhance security and cloud providers and limit dangerous user threats using these implementations. Criteria like security, time complexity, and accountability govern the suggested method's effectiveness.

Keywords: Cloud Computing; Efficient Data Sharing; Blockchain; Key Generation; Butterfly Optimization Algorithm.

Journal of Information Technology Management, 2023, Vol. 15, Special Issue, pp. 21-42
Published by University of Tehran, Faculty of Management
doi: https://doi.org/ 10.22059/jitm.2023.91561
Article Type: Research Paper

Received: December 22, 2022
Received in revised form: January 15, 2023
Accepted: February 20, 2023
Published online: March 15, 2023

Introduction

Nowadays, cloud storage and cloud computing have become prominent subjects. Both are transforming the way we live and, in some instances, dramatically increasing manufacturing effectiveness. We choose to keep various sorts of data in cloud servers at the moment owing to smaller capacity and the need for easy access, which is also a viable alternative for enterprises and organizations to reduce the expense of establishing and sustaining equipment when data is kept physically. The cloud server offers people and companies an open and accessible storage platform but also raises security concerns. A cloud system, for example, might be attacked by both cloud providers and the wrong users. Securing the data stored in the cloud in these instances is critical. Various solutions developed in the preceding systems were solely designed to address the security concerns of a single data owner. On the other hand, multiple data owners wish to safely publish their findings in a group setting in specific applications. As a result, a protocol for safe group data sharing in the case of cloud computing is required (Xu et al., 2019).

A key agreement protocol is employed to produce a shared conference key for several participants to maintain the security of their subsequent interactions. It may also be utilized in cloud computing to promote safe and efficient data exchange. A key agreement protocol describes a cryptographic mechanism that allows two or more people to agree on a key without influencing the outcome. For example, a secure key agreement mechanism guarantees that the attacker cannot access the produced key through malicious assaults like eavesdropping. As a result, the key agreement protocol is extensively applicable in interactive communication contexts having high-security needs (Lin et al., 2021).

One of the significant appealing advantages of cloud storage is the ability to share data between numerous users. As a result, it is also vital to guarantee that the integrity of cloud-based shared data is proper. Owing to the alteration implemented by these two separate users, distinct blocks ended up signing by different individuals. A public verifier must then pick the correct public key for every block to appropriately audit the integrity linked with the complete contents. Due to the distinctive coupling between a public and identity key through digital certificates underneath PKI, this public verifier will eventually understand the identity associated with the signer on every block. Failure to maintain identity privacy on shared data under public auditing will disclose sensitive information to the public verifier. It is vital to maintain identity privacy to secure this secret information (Huang et al., 2019).

The paper's contributions are.

● To propose the EBBKG model for efficient data sharing in the cloud by combining BBKG with ABS.

● To impose the encrypted access control from the data owner's perspective and specific enhanced access capabilities.

● Adequately protect the user's privacy with a secure authentication paradigm that employs ABS to safeguard the user's private data.

● To optimize the key using BOA to enhance security and compare the proposed model with existing methods amidst various factors to describe its superiority.

Literature Review

Shen et al., (2017) introduced a unique block design-oriented key agreement protocol that endorsed many participants and may dynamically increase the count of participants. We give generic formulas for creating the shared conference key IC for many participants based on the suggested group data-sharing paradigm. It's worth noting that the proposed protocol's computing cost grows linearly with the count of participants, but the complexity was considerably decreased. Tao et al., (2019) for CECS, they've introduced a novel secure data search and sharing strategy. The approach enhanced the traditional secure CECS system in the two aspects below.

On the other hand, the current system necessitates edge servers' usage. Secondly, it employs encryption to make data searches safer, more effective, and more adaptable. Concerning security, the method assures cloud data secrecy, safe data searching and sharing, and eliminates failure. Concerning performance, the outcomes suggested that outsourcing most cryptographic operations to edge servers greatly saved consumers' computational expenditures. Compared to the previous secure CECS technique, the method minimizes the processing and communication costs for constructing a search trapdoor.

Liu et al., (2012) Mona has been suggested that any cloud user could securely exchange data with others by utilizing dynamic broadcast encryption and group signature methods. In the meantime, the system's encryption calculation cost and storage overhead are unaffected by the count of banned users. Additionally, we use strong proofs to examine the method's security and verify its effectiveness in tests. Wang et al., (2014) suggested a new privacy-preserving approach that allows for public auditing of cloud-based shared data. We use ring signatures specifically to generate the verification information required to verify the validity of shared data. The identity linked with the signer on every block was maintained and hidden from public verifiers using the approach, allowing them to rapidly validate integrity without having to get the complete file. Furthermore, rather than validating each auditing work individually, the method may conduct several auditing tasks simultaneously.

Han et al., (2019) suggested an SSGK to prevent unwanted access to shared data and communication. In contrast to previous efforts, SSGK used a group key. Lin et al., (2021)

explored the security of the group signature technique, pointing out that it did not deliver the secrecy they promised and provided an attack in response. Venkatesan et al., (2017) developed an LDSS for mobile cloud computing, that used in CP-ABE. LDSS offloads a considerable chunk of the CP-computationally ABE's costly access control tree processing to external proxy servers. Wei et al., (2016) presented a concept known as RS-IBE, which could offer cipher text security by combining user revocation with cipher text updating functionality. We also provide a concrete RS-IBE architecture and demonstrate its protection under the security framework. The comparisons show that the suggested RS-IBE method offers functional and economic benefits, making it practicable for a practical data-sharing platform. Furthermore, we presented the proposed system's outcomes to illustrate its viability.

Xu et al., (2019) presented a revolutionary patient health information-sharing method that protects patient privacy while allowing HSPs to search and access PHI files securely. With multi-keyword search and keyword range search, we apply the searchable encryption approach. We employ a form of message authentication code and bloom filter to categorize PHI files, remove false data, and ensure the integrity of search outcomes. Simulations on synthetic and real-world data demonstrate the system's viability and effectiveness, while security analysis shows its privacy-preserving qualities. Huang et al., (2019) addressed conditional dissemination and multi-owner secure data group sharing strategy that presented owner priority, full permit, and majority permit to address the privacy conflicts generated by various access regulations. According to the security analysis and testing findings, the approach was practicable and effective for sharing safe data with many cloud computing owners.

Different Phases of Data Confidentiality

The different phases of data confidentiality are elaborated as follows,

Setup phase

In the setup phase, the adjacent edge server creates a data-sharing secret key L as well as a data-search secret key L_T for every device.

Data Uploading Phase

When a device (designated by J) wishes to save gathered data G to a cloud server, device J transmits data G, extracted keywords X, and the list of authorized users V to a nearby edge server (designated by B). Edge B obtains from the Key Generation Center (KGC) the private key SL_J of device J as well as the public keys $\{PL_v \mid v \in V\}$ of the authorized users. After that, it encrypts data G having secret key L, creates searchable symmetric key ciphertexts using a private key L_T and keywords X, and encrypts secret keys L having public keys

{PL_v |v∈ V}. Furthermore, it signs data G with the private key SL_J to verify data integrity and upload the ciphertexts and signatures created above to the cloud server.

Data Search phase

An authorized user (signified by v, in which v∈ V) selects a keyword X as his search request and sends it to a nearby edge server (represented by C) to share the expected data of the device J through the cloud server. Edge C obtains edge B's secret key L_T, creates a search trapdoor using a keyword X and the secret key L_T, and transmits it to the cloud server. The cloud server looks for ciphertexts that match and sends them to edge C. Edge C obtains the public key PL_J of device J and the private key SL_v of user v from the KGC, decrypts the matching data G, confirms the integrity of data G using the public key PL_J of device J, and further, deliver data G to user v.

Data Sharing phase

A user provides a request to edge C to share the entire device J's data. Edge C retrieves the ciphertexts, decrypts accurate data, validates data integrity, and ultimately delivers complete data to the user. The data confidentiality for maintaining security is shown in Figure 1.

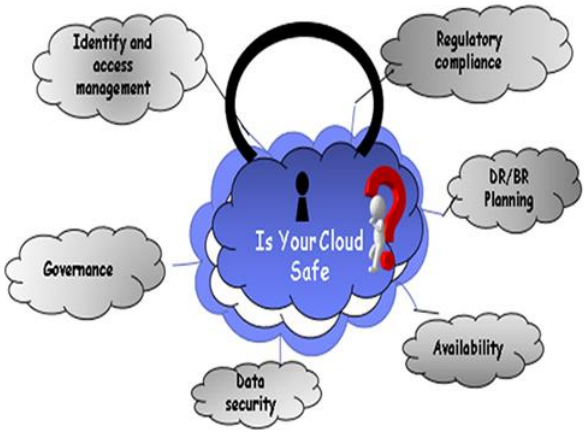


Figure 1. Data Confidentiality for maintaining Security in Cloud Computing

Enhanced Blockchain-based Key Generation for Efficient Data Sharing in Cloud

As depicted in Figure 2, the recommended architecture process is as below. Data owners request essential creation to produce global parameters, as well as public and master keys, to enroll in the introduced architecture. The blockchain network carries out the critical creation function, making a public key and a master key for the data owner. The user submits a registration request to the data owner, including attributes such as user ID, email address, department, phone number, DOB, address, and so on. The data owner saves user information in the introduced architecture and uses the BOA method to construct access restrictions for its attribute list. Utilizing the BOA technique, the data owner produces a secret key. The data

owner creates the user keys and maps them to one another, employing the produced keys, which are kept in the developed architecture. The data owner encrypts and decrypts the data, and this shared secret key is used for both. A processor with 16-bit memory addresses can directly access 64 KB of byte-addressable memory

The data owner uses the suggested architectural capability to upload the data to the cloud storage. The File's Meta information, such as Merkle root, file hash, and mapping details of file and user/owner, are generated by the data owner and stored in the blockchain. The data owner encrypts and stores the data in cloud storage using the created secret key. The data is stored on cloud storage servers, optimizing the resource to reduce overhead and execution time while holding it. The user transmits the access request and the File ID to the appropriate data owner. These are retrieved from the blockchain by the data owner. After reviewing the access information, the user's access may be allowed or refused. The encrypted data is fetched from the cloud storage and sent to the specified user. The user decrypts the encrypted data received using the secret key. The data owner retrieves the file and user information from the blockchain, does the integrity verification, and then delivers the output to the user.

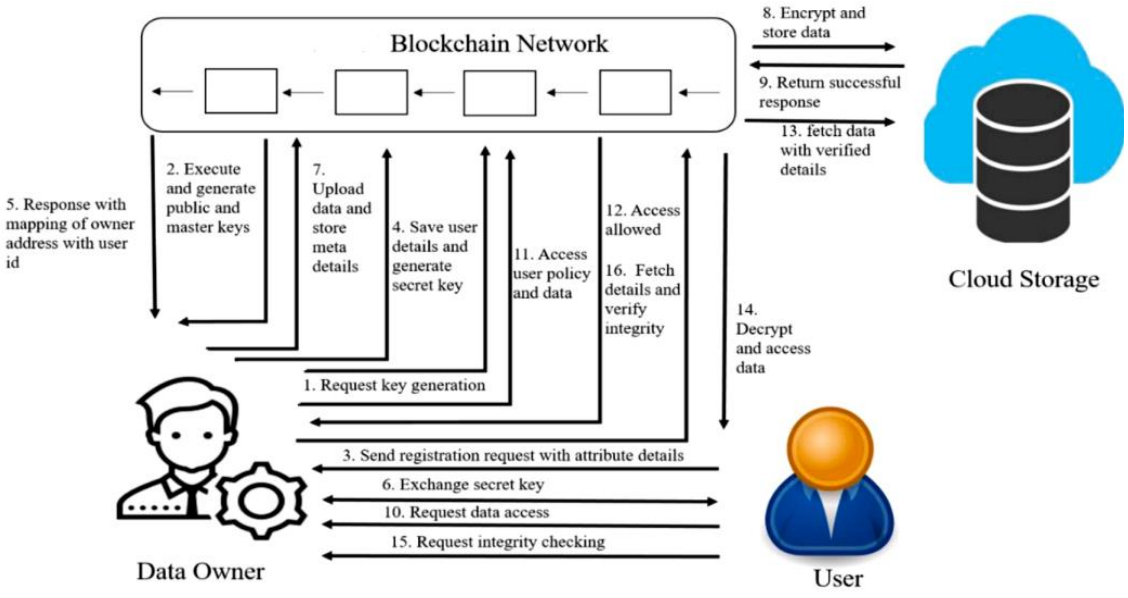


Figure 2. Overall system model of blockchain-based key generation

Enhanced Blockchain-based Key Generation Process

Here, the key is optimized by BOA; thus, the security gets enhanced to a greater level. Users can also send registration requests to data owners using attributes such as email id, user id, contact information, department, etc. The user attribute list is denoted by the letters $\{B_1, B_2, B_3, \cdots, B_o\}$. The data owner inputs global parameters $GQ(q, H, h, f, H_U, I)$ and the master key ML, assigns attribute policy to the user and creates a secret key SLand a public key $User_{id} = \{B_1 \varepsilon B_o\}$. The owner runs Algorithm 1 of the developed architecture to finish the

critical creation process. The following is the mathematical expression for the necessary generation mechanism:

- Global parameters$GQ(q, H, h, f, H_U, I)$.
- Every user attribute list is shown as$B_j = \{B_1, B_2, B_3, \cdots, B_o\} = \{1,2,3, \cdots, o\}$
- Next parameter $params = [q, h, f, h_2, i, Z = f(h, h_2)^{\alpha}, U_1, U_2, U_3, \cdots, U_o]$
- Data owner public key is shown by $PL = \{H, h, h^b, h^c, f(h, h)^c, I\}$
- Data owners' master keys are shown by $ML_j = \{\alpha_1, \alpha_2, \cdots, \alpha_o\}$

The secret key of every user withBattributes is produced as $\mathrm{SL_B} = \left[\mathrm{B}, \mathrm{e_1} = \mathrm{h^s}, \mathrm{e_2} = \mathrm{h_2^{\alpha} i^s}, \mathrm{e_j} = \mathrm{U_j^s}, \forall \mathrm{j} \in \mathrm{B}\right]$

Algorithm 1: Key generation(q, H, h, f, H_U, I)
Input: Data Owner Set, $DP = \{DP_1, DP_2, DP_3, \cdots, DP_j\}; 1 \leq j \leq o$
User Set, $v = \{v_1, v_2, v_3, \cdots, v_k\}$in which$DP_j \varepsilon \{v_1, v_2, v_3, \cdots, v_k\}$

$$v_k \leftarrow Attribute_{list}(B_1, B_2, \cdots, B_k)$$

Output: ML,SL
for$(j = 1; j \leq o, j++)$
{
Produce master keys forDP_j: $ML_j \leftarrow Key_{generation}(q, H, h, f, H_U)$
}
for$(j = 1; j \leq o, j++)$
{
for$(k = 1; k \leq o, k++)$
{
Produce secret key$SL_{jk}$for$v_{jk} \leftarrow Key_{generation}(q, H, h, f, H_U, ML_j, B_k)$
}
}

Encryption Process

The data is encrypted by the owner using the BOA algorithm. The encryption algorithm creates ciphertext CU using the global parameters GQ, file G, and the secret critical SL as input. Before the owner uploads the content to the cloud, it is split into many shards and Merkle_root is generated, with this Meta data being stored in the blockchain network. The user information, including the application binary interface owner public key, is expressed as a JSON object, and the file G is uploaded using the procedures below.

To compute the Merkle_root of the file, the owner divides it into many shards first. On the blockchain network, the computed Merkle_root storage takes place. The owner then encrypts the file G using the public secret key using the encryption technique. On the cloud server, {User_id,G_id,CU}, the user id User_id (public key) is saved with the encoding output, as well as G_id. Given the user's public key and identifiers, the owner exchanges common key SL. Next, using a shared key, a systematic encryption procedure is used. The blockchain is used to store this data. The encryption procedure is based on the equation as follows.

$$Encrypt(params[q, h, f, h_2, i, Z = f(h, h_2)^{\alpha}, U_1, U_2, \cdots, U_o]) \quad (1)$$

$$CU = [X, D_1, D_2, D_3], \text{in which } D_1 = G \cdot Z^T, D_2 = h^t, D_3 = \left(i \prod_{k=1}^{l} U_{jk}\right)^t, t\overleftarrow{S}\mathbb{Z}_q \quad (2)$$

Decryption Process

The global parameters GQ, encrypted text CU, and a shared common key SL are required by the decryption algorithm. The owner gets the User_id and G_id from the blockchain structure and then verifies to view if the requested file is present in the blockchain network. The decryption procedure is aborted if the requested file does not present. However, utilizing the global parameters, producing a shared key, and ciphertext, the owner, permits a user to follow the decryption process and receive the plain text. Decryption makes plain text if the attribute group meets the access policy criterion; else, it fails. The following is the decryption procedure of the chosen attribute-oriented method, including correctness confirmation, after acquiring the CU and SL of the parameters needed by the decryption algorithm:

- $Decrypt\left(params, SL_B = \left[B, e_1, e_2, e_j, \forall j \varepsilon B\right]\right), CU = [X, D_1, D_2, D_3]$ if $\{j_1, j_2, \cdots, j_l\} \neg \subset B$, decryption fails

  If $\{j_1, j_2, \cdots, j_l\} \subset B$, then $e = e_2 \prod_{k=1}^{l} e_{jk}$ and $G = \frac{D_1 \cdot f(e_1, D_3)}{f(e, D_2)}$ (3)

- $Correctness(SL_B, CU)$

$$SL_B = \left[B, e_1 = h_s, e_2 = h_2^{\alpha} i^s, e_j = U_j^s, \forall j \in B\right] \quad (4)$$

$$CU = \left[X, D_1 = G \cdot Z^t, D_2 = h^t, D_3 = \left(i \prod_{k=1}^{l} U_{jk}\right)^t\right], \text{in which } Z = f(h, h_2)^{\alpha} \quad (5)$$

$$\frac{\boldsymbol{D_1 \cdot f(e_1, D_3)}}{\boldsymbol{f(e, D_2)}} = \frac{\boldsymbol{G \cdot f(h, h_2)^{\alpha t} \cdot f\left(h^s, \left(i \prod_{k=1}^{l} U_{jk}\right)^t\right)}}{\boldsymbol{f\left(h_2^{\alpha} i^s \prod_{k=1}^{l} U_{jk'}^s, h_t\right)}} \quad (6)$$

Butterfly Optimization Algorithm (BOA)

Arora et al., (2019) solves global optimization problems by mimicking butterflies' food-seeking and mating behavior. The structure is mainly inspired by butterflies' foraging

approach, which involves using their sense of smell to locate nectar or a mating partner. The complete notion of detecting and processing the modality is built around three key terms: sensory modality d, input intensity J, and power exponent b. The amplitude of the physical/actual stimulus is referred to as J. Linear response, regular expression, and response compression are all possible with the parameter b. When J rises, the scent g rises faster than J.

The fluctuation of J and the derivation of g are at the heart of the natural phenomena of butterflies. The encoded goal function is related to the J of a butterfly for convenience. Nevertheless, g is subjective, meaning that the remaining butterflies should be able to detect it. d is employed to distinguish scent from various sensory modalities. As the butterfly having fewer J gets closer to the butterfly with greater J, g grows faster than J. As a result, we should permit g to fluctuate with an absorption degree determined by the power exponent b. The scent in BOA is framed as a function of physical stimulation intensity as below:

$$g = dJ^b \tag{7}$$

Here, g shows the perceived magnitude of the scent, i.e., how various butterflies experience the smellies, d shows the sensory modality, J performances the stimulus intensity, and b shows the modality-dependent power exponent, which factors for the changing degree of absorption. BOA is divided into three phases: initialization, iteration, and finalization. The algorithm specifies the goal function and solution space during the initialization step. The parameters employed in BOA have their values selected as well. After determining the variables, the program creates an initial population of butterflies for optimization. The method performs a count of iterations in the second step of the process, known as the iteration step. The complete butterflies in the solution space migrate to innovative places in every iteration, and their fitness values are then assessed. The algorithm begins by calculating the finished butterflies' fitness values at various points in the solution space.

$$w_h^{s+1} = w_h^s + (q^2 \times f* - w_h^s) \times e_h \tag{8}$$

The solution vector w_hlinked with the hth butterfly at iteration count s is given by w_h^s. The best solution identified within the complete solutions in the present iteration is denoted by f*. The hth butterfly's fragrance is indicated by e_h, while q shows a random number in the range [0, 1]. The phase of local search can be described as follows:

$$w_h^{s+1} = w_h^s + \left(q^2 \times w_i^s - w_j^s\right) \times e_h \tag{9}$$

w_i^s and w_j^s shows the i^{th} and j^{th} butterflies from the solution space, respectively. Equation (9) is now a local random walk if w_i^s and w_j^s correspond to a specific swarm, and q shows a random count in the range [0, 1]. The pseudo-code of BOA is shown in Algorithm 2.

Algorithm 2: Butterfly Optimization Algorithm

Start
Objective function $e(w), w = (w_1, w_2, \cdots, w_{dim}() \, dim =)$ dimensions
Initial population generation of m butterflies $w_h = (h = 1,2, \cdots, m)$
Stimulus intensity I_h at w_h is shown by $e(w_h)$
Describe switch probability o, power exponent b and sensor modality d
While the end condition is not reached, do
For all the butterflies ce do

$$g = dJ^b$$

End for
Get best ce
For all the butterflies, ce do
Random number generation q in [0,1]
if $q < o$ then

$$w_h^{s+1} = w_h^s + (q^2 \times f * -w_h^s) \times e_h$$

else

$$w_h^{s+1} = w_h^s + \left(q^2 \times w_i^s - w_j^s\right) \times e_h$$

End if
End for
update b value
End while
Output optimal solution
Stop

Attribute-based Signature and Data Access for Cloud Computing

Attribute-Based Signature

Consider the universe of characteristics $V = \{1,2, \cdots, o\}$. Next, for every characteristic $j \in V$, pick a count u_j uniformly at random from $\mathbb{Z}_q$, and select t uniformly at random from $\mathbb{Z}_q$. The public master parameters PLare $U_1 = h^{u_1}, \cdots, U_{|V|} = h^{u_{|V|}}, X = \tilde{h}^x$, while the master private key MLis: $u_1, \cdots, u_{|V|}, x$.

Key Generation (Γ, ML): If and only if $\Gamma(\gamma) = 1$, the method generates a private key that allows the user to sign a message using a group of qualities γ. Select a polynomial r_y for every node y (along with the leaves) in the tree U, beginning with the root node s and working your way down. Place the degree e_y of the polynomial r_y of every node y in this tree to one less than the threshold value l_y of considered node, i.e. $e_y = l_y - 1$. To entirely describe the polynomial r q, put $r_s(0) = x$ and $e_s$additional points of the polynomial r_s at random for the root node s. Place $r_y(0) = r_{\text{parent(y)}}(\text{index(y)})$ and pick e_y other points at random to fully describe r_y for any remaining node y. After the polynomials have been determined, we offer the user the appropriate secret value for every leaf node y as $E_y = \tilde{h}^{\frac{r_y(0)}{u_j}}$, in which $j =$

$\mathrm{att}(y)$ and $y \in z$. The private key E is made up of the personal values listed above. Secondly, the algorithm generates the critical parts: $Z = h^z$.

Hence, its public key is $pl =< h, \tilde{h}, X, U_{j_1}, \cdots, U_{j_v}, Z >$, and its private key is $sl =< h, \tilde{h}, E_{leaf_1}, \cdots, E_{leaf_v}, z >$.

Signing: $\sigma = (B, C, D, E) = \left(h^s, \tilde{h}^{\frac{1}{z+s}}, \tilde{h}^{\frac{1}{n+s}}, \tilde{h}^{xs}\right)$ with s picked from $\mathbb{Z}_o$, and $FE_j = \tilde{h}^{\frac{sr_y(0)}{u_j}}$, then output signature $T = \left(\sigma = (B, C, D, E), \{FE_j\}_{j \in \gamma}\right)$.

Verifying: Accept if and only if: on input $\left(h, Z = h^z, \tilde{h}\right)$, a message n, and an intended signature T: (1) $f(h^z B, C) = f\left(h, \tilde{h}\right)$ (2) $f(Bh^n, D) = f\left(h, \tilde{h}\right)$ and (3) $f(h, E) = G_y$.

Data Access from Cloud: When a data user requests access to files from the cloud, the user receives the CD. The method of retrieving data from the cloud may be divided into two steps:

Data reading from Cloud: The data gets encrypted data from the cloud and performs the decryption procedure.

Writing data onto the cloud: The user should meet the access policy specified during the initial setup phase to write data to an existing file. After the user identification and signature verification processes have been completed successfully, the writing operations can be processed.

Results and Discussion

Experimental Setup

Utilizing the CloudSim Tool, the suggested EBBKG-BOA mode is analyzed and tested depending on parameters like the model's security rate, efficacy, PDR packet loss, transmission latency, communication overhead, storage overhead, and time complexities. The findings are compared to traditional methods like CP-ABE, IBE, AASM and KP-ABE to demonstrate the technique's usefulness. The assessments are done with time complexity and computational complexity-oriented analysis depending on the time parameters. Furthermore, the average count of time windows per cloud user is used to analyze the impact of the time window on the entire method's performance.

Communication Overhead Analysis

The findings for calculating communication overhead within models are shown in Figure 3. In a cloud paradigm, communication among the user, cloud owner, and CSP must be as low as possible. The suggested method accomplishes minimum communication with a greater

security rate than the comparison methods, as shown in the image. The proposed approach has a 24.9% lower communication overhead compared to the other models. The simulation results of communication overhead analysis is listed in Table 1.

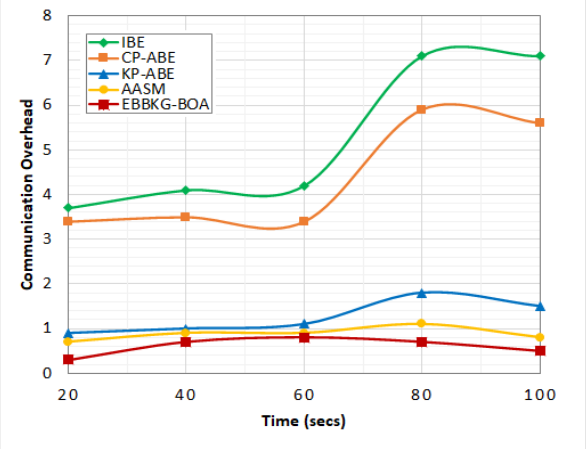


Figure 3. Comparison of communication overhead analysis for various methods

Table 1. Communication overhead analysis

Methods	Simulation Time				
	20secs	40secs	60secs	80secs	100secs
IBE (Bentajer et al., 2019)	3.7	4.1	4.2	7.1	7.1
CP-ABE (Ma et al., 2021)	3.4	3.5	3.4	5.9	5.6
KP-ABE (Touati & Challal, 2016)	0.9	1.0	1.1	1.8	1.5
AASM (Anbumani, & Dhanapal, 2022)	0.7	0.9	0.9	1.1	0.8
Proposed EBBKG-BOA	0.3	0.7	0.8	0.7	0.5

Packet Delivery Ratio Analysis

The PDR, which signifies smooth communication within nodes, must be more prominent for an effective communication rate. Figure 4 depicts the associated findings, which demonstrate that the suggested method has a greater rate of PDR than existing methods CP-ABE, IBE, AASM and KP-ABE, as well as superior security. The PDR values of various methods are represented in Table 2. Furthermore, in the event of PDR, the suggested method achieves a higher rate of PDR because the encryption paradigm is successfully handled for safeguarding the communication, with an average speed of PDR of 86.9%.

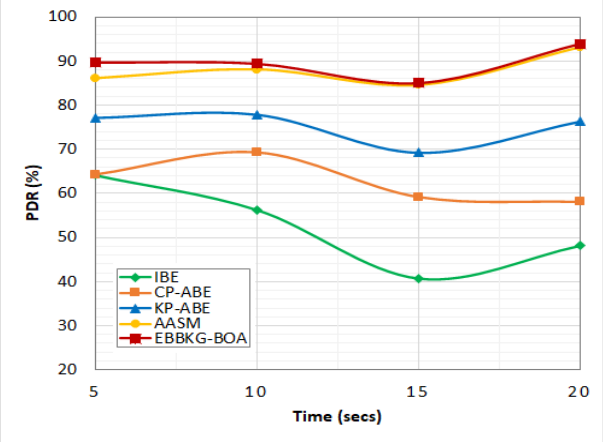


Figure 4: Comparison of PDR analysis for different methods

Table 2. Packet delivery ratio analysis

Methods	Simulation Time			
	5secs	10secs	15secs	20secs
IBE (Bentajer et al., 2019)	64.1 %	56.2 %	40.6 %	48.1 %
CP-ABE (Ma et al., 2021)	64.3 %	69.3 %	59.2 %	58.1 %
KP-ABE (Touati & Challal, 2016)	77.1 %	77.8 %	69.2 %	76.3 %
AASM (Anbumani, & Dhanapal, 2022)	86.1 %	88.1 %	84.6 %	93.1 %
Proposed EBBKG-BOA	89.7 %	89.4 %	85.1 %	93.9 %

Transmission Delay Analysis

Transmission Delay is another critical parameter for evaluating the method's efficacy, and the related findings are shown in Figure 5. The suggested EBBKG-BOA model has a low latency and efficiently uses the system model having security-oriented incorporations from signature-oriented approaches. As a result, tabulated in Table 3, the likelihood of assaults is efficiently minimized, and the delay rate is underrated. The transmission delay is calculated in this case as a function of packet size.

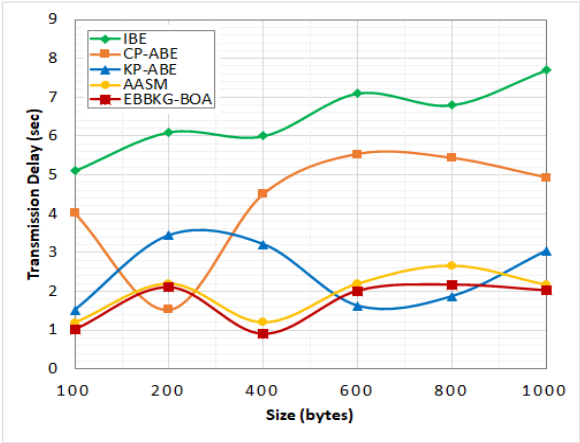


Figure 5. Transmission delay analysis with different methods

Table 3. Transmission delay analysis

Methods	Packet Size					
	100 bytes	200 bytes	400 bytes	600 bytes	800 bytes	1000 bytes
IBE (Bentajer et al., 2019)	5.10	6.1	6.0	7.1	6.8	7.7
CP-ABE (Ma et al., 2021)	4.01	1.53	4.50	5.53	5.43	4.93
KP-ABE (Touati & Challal, 2016)	1.52	3.44	3.21	1.63	1.87	3.04
AASM (Anbumani, & Dhanapal, 2022)	1.19	2.19	1.2	2.19	2.65	2.16
Proposed EBBKG-BOA	1.01	2.10	0.9	2.01	2.17	2.02

Packet Drop Analysis

Packet Drop describes an essential statistic for evaluating the method's efficacy, and the relevant outcomes are shown in Figure 6. The suggested EBBKG-BOA method provides a low loss rate by combining a system method with security-oriented integrations from signature-oriented approaches. From the simulated results listed in Table 4, the likelihood of an assault is substantially minimized. The drop rate is calculated here using simulation time.

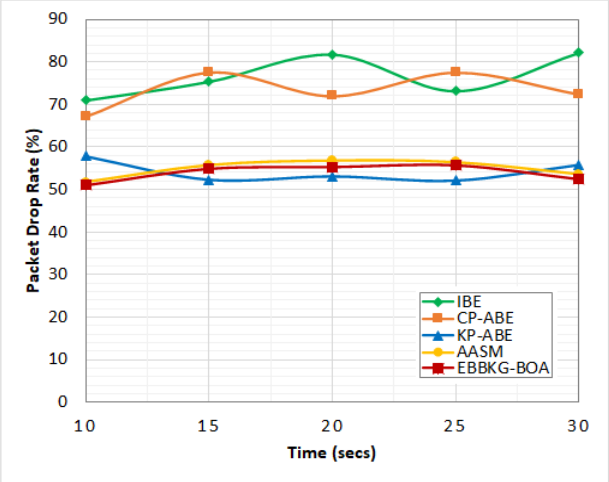


Figure 6. Comparison of packet drop analysis for different techniques

Table 4. Packet drop analysis

Methods	Time				
	10secs	15secs	20secs	25secs	30secs
IBE (Bentajer et al., 2019)	70.9 %	75.3 %	81.7 %	73.1 %	82.1 %
CP-ABE (Ma et al., 2021)	67.2 %	77.5 %	72.0 %	77.5 %	72.4 %
KP-ABE (Touati & Challal, 2016)	57.8 %	52.3 %	53.1 %	52.1 %	55.8 %
AASM (Anbumani, & Dhanapal, 2022)	51.7 %	55.8 %	56.9 %	56.5 %	53.6 %
Proposed EBBKG-BOA	51.1 %	54.9 %	55.3 %	55.7 %	52.5 %

Overall Processing Time Analysis

The findings of the time efficiency tests are also evaluated, and the outcomes are shown in Figure 7. Optimizing the time efficiency on the cloud, determined by the file size kept on the cloud, is critical. The total processing time for protecting data to be exported using an attribute-oriented security architecture is given in Table 5. As seen in the accompanying Figure, the method took the least time compared to the other works.

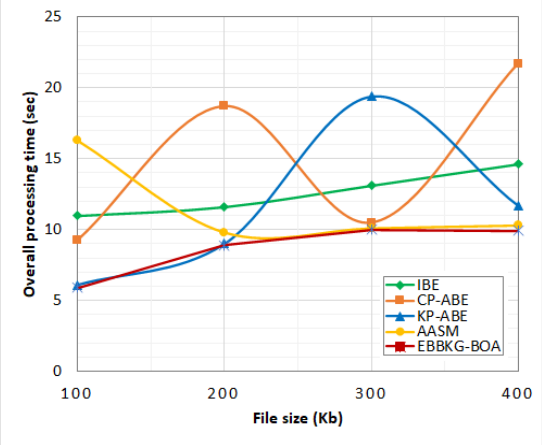


Figure 7. Comparison of overall processing time analysis for various methods

Table 5. Overall processing time analysis

Methods	File Size			
	Size=100Kb	Size=200Kb	Size=300Kb	Size=400Kb
IBE (Bentajer et al., 2019)	10.98	11.6	13.1	14.6
CP-ABE (Ma et al., 2021)	9.29	18.7	10.5	21.7
KP-ABE (Touati & Challal, 2016)	6.1	9.0	19.4	11.7
AASM (Anbumani, & Dhanapal, 2022)	16.3	9.8	10.1	10.3
Proposed EBBKG-BOA	5.9	8.9	10.0	9.9

Security Rate Analysis

The security factor-oriented evaluations are performed and presented in Figure 8. When the assaults are provided in the communication, the suggested method's security rate is efficiently assessed and delivers a lower security rate. The security rate analysis of various methods is represented in Table 6.

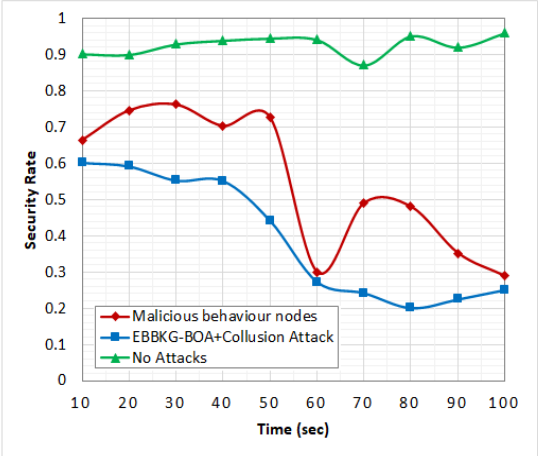


Figure 8: Simulation of security rate analysis with different methods

Table 6: Security rate analysis

Methods	Time									
	10sec	20sec	30sec	40sec	50sec	60sec	70sec	80sec	90sec	100sec
Malicious behavior nodes	0.664	0.746	0.762	0.703	0.726	0.3	0.491	0.480	0.352	0.289
EBBKG-BOA+Collision Attack	0.602	0.592	0.553	0.552	0.442	0.273	0.242	0.201	0.225	0.251
No Attacks	0.902	0.90	0.929	0.939	0.945	0.941	0.871	0.951	0.920	0.960

Attack Analysis with the proposed method

Figure 9 depicts the outcomes based on the considerations that compromised node attacks, and collusion attacks are present. The graphs show that the suggested method efficiently assesses the security rate, which is lower when an assault occurs. As a result, users and cloud providers are notified, and data transfer among the organizations is protected. The attack analysis of various methods is represented in Table 7.

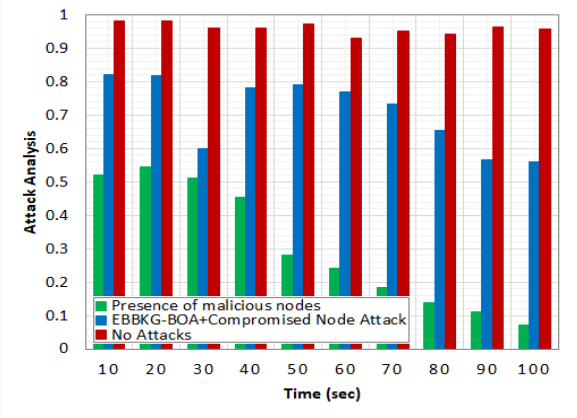


Figure 9. Comparison of attack analysis with different methods

Table 7. Attack analysis

Methods	Time									
	10sec	20sec	30sec	40sec	50sec	60sec	70sec	80sec	90sec	100sec
Presence of malicious nodes	0.52	0.544	0.512	0.453	0.280	0.24	0.183	0.139	0.112	0.073
EBBKG-BOA+ Compromised Node Attack	0.821	0.817	0.6	0.781	0.791	0.770	0.732	0.653	0.567	0.561
No Attacks	0.982	0.982	0.961	0.960	0.972	0.931	0.951	0.941	0.962	0.956

Storage Cost Analysis

The method's cost efficiency is assessed, and the findings are shown in Figure 10. In every way, the suggested approach is less expensive than the alternatives. The storage effectiveness of methods is calculated using communication network-oriented parameters and displayed against the number of files in a graph. As mentioned in Table 8, the suggested process requires less storage than the remaining efforts.

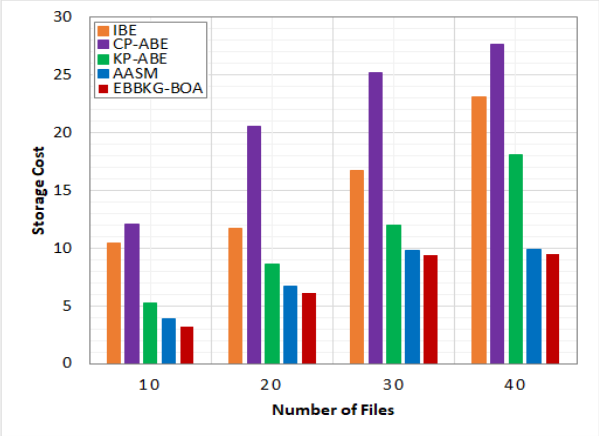


Figure 10. Comparison of storage cost analysis with different techniques

Table 8. Storage cost analysis

Methods	Number of Files			
	File=10	File=20	File=30	File=40
IBE (Bentajer et al., 2019)	10.4	11.7	16.7	23.1
CP-ABE (Ma et al., 2021)	12.1	20.5	25.2	27.6
KP-ABE (Touati & Challal, 2016)	5.2	8.6	12.0	18.1
AASM (Anbumani, & Dhanapal, 2022)	3.9	6.7	9.8	9.9
Proposed EBBKG-BOA	3.2	6.1	9.4	9.5

Key Generation Time Analysis

The security method's critical generation time is efficiently estimated to assess the method's effectiveness, and the outcomes are shown in Figure 11. The key generation time values of various methods are represented in Table 9.

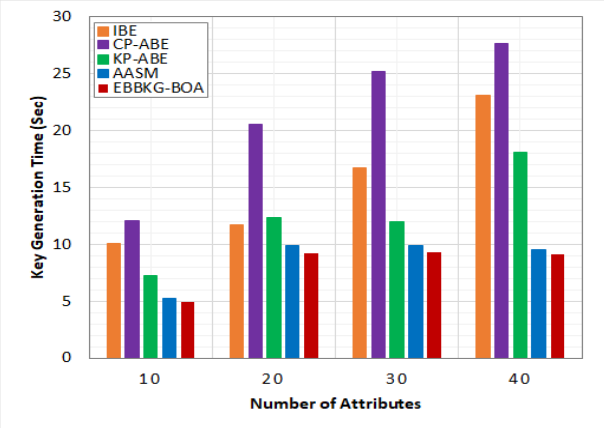


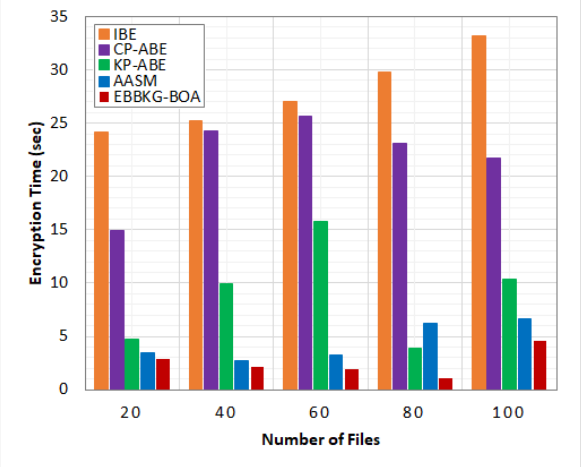
Figure 11. Comparison of key generation time analysis with different methods

Table 9. Key generation time analysis

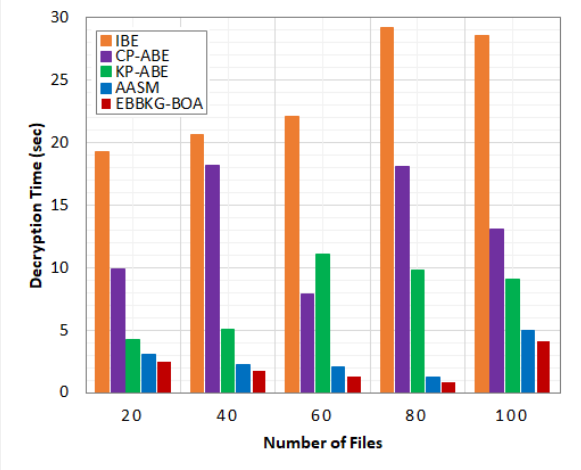
Methods	Attribute count in every key			
	10 attributes	20 attributes	30 attributes	40 attributes
IBE (Bentajer et al., 2019)	10.1	11.7	16.7	23.0
CP-ABE (Ma et al., 2021)	12.1	20.5	25.2	27.6
KP-ABE (Touati & Challal, 2016)	7.2	12.3	12.0	18.1
AASM (Anbumani, & Dhanapal, 2022)	5.2	9.9	9.9	9.5
Proposed EBBKG-BOA	4.9	9.2	9.3	9.1

Encryption and Decryption Time Analysis

The average time for encrypting and decryption operations is also calculated and displayed in Figure 12. The encryption procedure for the suggested method takes an average of 4.22 seconds, and the decryption process takes an average of 2.7 seconds. The findings demonstrated in Table 10 shows that the presented method takes the least time compared to other methods.



Encryption time



Decryption time

Figure 12. Comparison analysis of encryption and decryption time

Table 10. Encryption and decryption time analysis

Methods	Number of Files									
	20 files		40 files		60 files		80 files		100 files	
	Encryption time	Decryption time	Encryption time	Decryption time	Encryption time	Decryption time	Encryption time	Decryption time	Encryption time	Decryption time
IBE	24.2	19.3	25.2	20.6	27.0	22.1	29.8	29.2	33.2	28.5
CP-ABE	14.9	9.9	24.3	18.2	25.6	7.9	23.1	18.1	21.7	13.1
KP-ABE	4.7	4.2	9.9	5.1	15.8	11.1	3.9	9.8	10.4	9.1
AASM	3.5	3.1	2.7	2.2	3.2	2.1	6.2	1.2	6.7	5.0
EBBKG-BOA	2.9	2.5	2.1	1.7	1.9	1.3	1.1	0.8	4.6	4.1

Cloud Service Provider Analysis

The method's efficacy is successfully estimated based on the elements described above, and the findings are shown in Figure 13. As a result of the successful coupling of attribute-oriented encryption and signature operations, the suggested method is more efficient than prior efforts. User authentication is primarily used in the process to protect outsourced data as well as user privacy. The cloud service provider analysis of various methods is shown in Table 11.

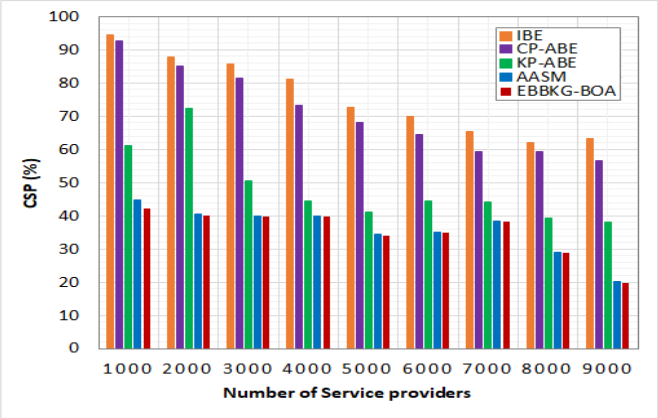


Figure 13. Comparison of cloud service provider analysis with different methods

Table 11. Cloud service provider analysis

Methods	Number of Service Providers								
	1000	2000	3000	4000	5000	6000	7000	8000	9000
IBE (Bentajer et al., 2019)	94.40	87.80	85.64	81.32	72.57	69.81	65.49	62.19	63.33
CP-ABE (Ma et al., 2021)	92.8	85.04	81.62	73.29	68.01	64.47	59.42	59.43	56.55
KP-ABE (Touati & Challal, 2016)	61.17	72.39	50.43	44.43	41.01	44.32	44.13	39.45	38.13
AASM (Anbumani, & Dhanapal, 2022)	44.91	40.53	40.00	40.00	34.53	35.06	38.37	29.07	20.2
Proposed EBBKG-BOA	42.29	40.12	39.82	39.81	34.12	34.83	38.14	28.91	19.6

Conclusion

For effective data sharing in the cloud, this study presented the EBBKG model. The technique combined the BBKG with ABS for safe data exchange in the cloud. Moreover, the method effectively managed data by defining the upcoming processing steps. The paradigm imposed encrypted access control and some increased access capabilities from the data owner's perspective. Second, with a secure authentication paradigm that leveraged ABS to preserve the user's private data, the user's privacy might be adequately maintained. The pattern secures users and cloud providers using these implementations and minimizes severe user threats. The efficacy of the given strategy was determined by factors such as security, time complexity, and accountability.

Conflict of interest

The authors declare no potential conflict of interest regarding the publication of this work. In addition, the ethical issues including plagiarism, informed consent, misconduct, data fabrication and, or falsification, double publication and, or submission, and redundancy have been completely witnessed by the authors.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article

References

Anbumani, P., & Dhanapal, R. (2022). AASM: Attribute based Advanced Security Model for Reliable Data Sharing in Cloud Computing.

Arora, S., & Singh, S. (2019). Butterfly optimization algorithm: a novel approach for global optimization. Soft Computing, 23(3), 715-734.

Bentajer, A., Hedabou, M., Abouelmehdi, K., Igarramen, Z., & El Fezazi, S. (2019). An IBE-based design for assured deletion in cloud storage. Cryptologia, 43(3), 254-265.

Han, S., Han, K., & Zhang, S. (2019). A data sharing protocol to minimize security and privacy risks of cloud storage in big data era. IEEE Access, 7, 60290-60298.

Huang, Q., Yang, Y., Yue, W., & He, Y. (2019). Secure Data Group Sharing and Conditional Dissemination with Multi-Owner in Cloud Computing. IEEE Transactions on Cloud Computing, 9(4), 1607-1618.

Lin, X. J., Sun, L., & Qu, H. (2021). Cryptanalysis of an Anonymous and Traceable Group Data Sharing in Cloud Computing. IEEE Transactions on Information Forensics and Security, 16, 2773-2775.

Liu, X., Zhang, Y., Wang, B., & Yan, J. (2012). Mona: Secure multi-owner data sharing for dynamic groups in the cloud. IEEE transactions on parallel and distributed systems, 24(6), 1182-1191.

Ma, J., Wang, M., Xiong, J., & Hu, Y. (2021). Cp-abe-based secure and verifiable data deletion in cloud. Security and Communication Networks, 2021.

Shen, J., Zhou, T., He, D., Zhang, Y., Sun, X., & Xiang, Y. (2017). Block design-based key agreement for group data sharing in cloud computing. IEEE Transactions on Dependable and Secure Computing, 16(6), 996-1010.

Tao, Y., Xu, P., & Jin, H. (2019). Secure data sharing and search for cloud-edge-collaborative storage. IEEE Access, 8, 15963-15972.

Touati, L., & Challal, Y. (2016, May). Collaborative kp-abe for cloud-based internet of things applications. In 2016 IEEE International Conference on Communications (ICC) (pp. 1-7). IEEE.

Venkatesan, C., Balamurugan, D., Thamaraimanalan, T., & Ramkumar, M. (2022, March). Efficient Machine Learning Technique for Tumor Classification Based on Gene Expression Data. In 2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS) (Vol. 1, pp. 1982-1986). IEEE.

Wang, B., Li, B., & Li, H. (2014). Oruta: Privacy-preserving public auditing for shared data in the cloud. IEEE transactions on cloud computing, 2(1), 43-56.

Wei, J., Liu, W., & Hu, X. (2016). Secure data sharing in cloud computing using revocable-storage identity-based encryption. IEEE Transactions on Cloud Computing, 6(4), 1136-1148.

Xu, C., Wang, N., Zhu, L., Sharif, K., & Zhang, C. (2019). Achieving searchable and privacy-preserving data sharing for cloud-assisted E-healthcare system. IEEE Internet of Things Journal, 6(5), 8345-8356.

Bibliographic information of this paper for citing:

Anbumani, P. & Dhanapal, R. (2023). Enhanced Blockchain-based Key Generation using Butterfly Optimization Algorithm for Efficient Data Sharing in Cloud Computing. *Journal of Information Technology Management*, 15 (Special Issue), 21-42. https://doi.org/ 10.22059/jitm.2023.91561