An Effective Cloud Data Security Framework for Real-Time Cloud Computing Environments Based on Dynamic Keyword Search

J V S Arundathi [1] , Dr.K V V Satyanarayana[2]

[1]Research Scholar, Department of Computer Science and Engineering, Koneru lakshmaiah Education Foundation (KLEF), Greenfields, Vaddeswaram, Guntur Dist., Andhra Pradesh, India.

[2]Professor, Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation(KLEF) , Greenfields, Vaddeswaram, Guntur Dist., Andhra Pradesh, India

.

Abstract: - Most of the existing secure keyword search models are infeasible and not applicable for massive sets due to their high computational and memory requirements for data processing and dynamic security parameter initialization along with integrity size, which leads to an exponentially huge computational search space. It is more difficult to keep the data in the public/ private cloud server safe. Particularly, we are considering the unstructured data formats which are extremely hard to keep data structure and avoid error. It plans to develop and implement a hybrid dynamic keyword-based cloud data security framework for large cloud databases. This framework combines a hybrid dynamic hash-based keyword search and encryption and decryption model into a unique concept especially provided for the cloud environment based on a unique non-linear chaotic hash algorithm and a hybrid multi-user-based encryption and decryption model to accomplish a dynamic cloud data and service access and protection via a static keyword-based cloud data security framework for Big-Data structure and tasks. Experimental results show that this model presents faster runtime for keyword search or decryption/encryption operation compared to conventional models. Therefore, the proposed model will pave the way for those data security experts working with the cloud data storage system.

Keywords*:* keyword search, multi-document security, cloud document storage.

1. Introduction

Cloud computing is the most widely used cloud service. People and organizations use cloud computing through the use of virtual machines, the data handling of which do not have any relation with the hardware and software details of the client system. Thus, the hardware and software requirements of client systems are easily configurable and one can integrate easily into any cloud storage systems. One can also use tools like Hadoop for extra computing power. Scalability and flexibility are the reasons why the smart move when it comes to IT infrastructure is to utilise the services provided by cloud computing, which allows customers to interconnect and stored material on demand, paying for whatever it is that they require according to use. The end user no longer controls the data at data centres they keep at remote sites, instead relying on a third-party service provider to do it for them. For all its benefits, cloud computing carries with it some unique security challenges, such as those pertaining to the personal sensitive data outsourcing and storage. Providing adequate protection mechanisms for sensitive data on the cloud is of utmost importance. Privacy-preserving techniques rely on encrypting algorithms to safely share sensitive information. The use of homomorphic encryption algorithms is essential to mitigate potentially harmful consequences resulting from the unauthorised access to processed ciphertexts. Ensuring the confidentiality of the user’s data remains the principal goal of any credible encryption scheme. Homomorphic encryption methods allow users to compute on encrypted data while maintaining the integrity of the data and preserving confidentiality.

While homomorphic encryption holds promising opportunities, its application in crunching complex data tasks is technically complex. From a technical perspective, homomorphic encryption algorithms run on single-user computers. However, the algorithms behind fully homomorphic encryption are complex, extremely computation intense and slow. Thus, while few basic computations can be carried out, it is currently impractical to execute any computationally intensive tasks in real applications. With cloud computing, customers do not store the data locally, but on a remote cloud of servers. These cloud servers are responsible for serving the customer's needs. However, if the data gets stored in the cloud, the security and privacy of the stored information becomes a great concern, and therefore it is crucial to come up with a flexible and effective data access control system and a secure encryption mechanism for sharing corporate data among different cloud servers. Attributewise encryption systems have been recognised to be an effective solution to achieve flexibility and good security for data sharing in the dynamic environment of cloud systems, especially for large cloud datasets that need flexible yet robust data security with different access strategies. Objectives of the proposed models would be to cater security (encrypted data in the cloud), scalability and strong (efficient as well as hard) data security with novel attribute-based encryption for a very large data that would be stored on the cloud security.Likewise, the scalability, efficiency, as well as usability of integrity-based cloud security to the authenticated customers that belong to both private as well as public domains would also be ensured. The proposed models must be designed in such a way that these models should grip large data scalability in the key management, ways of communication with cloud information computing, cloud data storage process complexity terms. Two types of attribute-based encryption tech-niques, KP-ABE and CP-ABE, are shown in the preceding two details. The KP-ABE enters the multiple attributes that need to match the muti-user sets that compose multiple access structures. The CP-ABE enters the muti-access structure that needs to match the muti-user sets and is a special kind of user set. Such a CP-ABE is often adopted for applications needed for multiple access control. As these two tech-niques both fit the process of keyword search and can be implemented, the customers can choose the one that fits bests their needs. Due to the fact that cloud computing services are demand-based and that resource rearrangement is readily available, cloud usage can diminish users' overhead work in maintaining multiple information systems. For using the cloud, however, users have to outsource their objects to distributed heterogeneous entities. A protocol is designed to show the ownership of the images in the cloud so that no third party can change images put in the cloud by the owner. This authentication scheme restricts data flow and does not allow images to be transferred or modified without the owner's permission. The medical images stored in the cloud in this scheme can help to manage all the knowledge and computing resources needed by each health system, physician and individual. It reduces unnecessary use of local storage. Integrity of the nodes and communication data should be verified in Cloud networks. However, the verification algorithms concerning integrity are useless in dynamic Cloud networks, especially for large amounts of information. A variety of cryptographic integrity features are proposed to assure the validity of communication information in Cloud networks. Authentication models are caried out to authenticate each cloud client when cloud data is communicated in Cloud networks. Traditional authentication models used in cloud computing are time consuming and not suitable for large cloud networks that have limits on the size of data, variable-length integrity value, and hash value generation in dynamic environment of real-time cloud computing.

2. Objectives

Pakniat [8] suggests an evolutionary multi-objective solution for the cloud resource allocation problems. Thus, monitoring and resource discovery techniques form part of the resource allocation system. Finally, they also provide an input for resource optimisation algorithms because it is necessary to know the context in terms of services and resources that require optimisation[9-12]. The approach is designed to address data encryption and boundary maintenance and proofing of data, and it is all handled seamlessly with a view management scheme.This technique offers some benefits. It preserves entity privacy, ensures the accessibility of data, and allows secure sharing of data. Each sensor manages its own view with a boundary that is not violated. The ACO-IBE is a four-stage algorithmic technique given in pseudocode. The configuration stage is responsible for generation of the master key by the receiver. The receiver is authenticated with features such as SSN, which are passed on to the Private Key Generator (PKG) as an input to the KeyGen algorithm.The KeyGen algorithm generates the private key of receiver. The encryption step refers to a scenario in which sender knows the email address of the recipient.

These address features are used to encode the message in an efficient manner. For decrypting the encrypted signal, the receiver uses the same private key generated by the PKG. As the sender, in a CB-IBE, will not need to perform any data with the KDC, generating the public key, it will only need to know who the receiver is. This information is provided by its identifier. We show how the weaknesses of the described cloud-based IBE approaches can be overcome through the use of a slight extension of the ABE scheme proposed above. Consider the multi-user IBE policies associated with the identifiers of the n sensors in the traditional ABE system. In an ABE with attribute system (our model above), Æ the ciphertext is generated by Tuple Λe, the attribute, ΦE is the access structure (a policy), and M is the message. A general ABE system is composed of four main algorithms: Setup, KeyGen, Encryption, Decryption. Once the sever is not secured, external and internal threats will exploit the vulnerabilities of the cloud which will result in confidentiality issues, integrity and the availability of information. It is possible that some untrusted service providers will not disclose weaknesses in the system, as they do not want to be blamed for any incident that caused harm to the users. Sometimes, the cloud storage is expanded by removing least accessed data.Some sensors and organisations save their private information in clouds and due to attack such private information can be revealed that includes the business organisation and sensor information. The ant colony optimisation method has also been suggested as a new approach to find the point of attack by modelling the pheromone activity on the entire certificat sequence that indicates the confidence of the node. The main limitation of this method is an increase of nodes to improve the scalability of the network. It can be extended to detect which chains are made up of certificates Sybil identity nodes. In 2004, Zhange proposed a cryptographic schema to protect information in cloud path planning in a restricted cloud network, known as CP-ABE model, which is related to ciphertext , which is closely related to the attributes-based-access control system, sets concealed secret keys or associated attributes, based on multidimensional attributes, set no specific privilege information, access according to different attributes; Modifying and screening the fine ability for multidimensional access structure, there is a certain property, access rights are achieved when properties meet specific access strategy, compared with the early KP-ABE model, the operation of CP-ABE is upside-down, CP-ABE has many comparative advantages on the other hand, it has relatively complex application, which can be widely used, It can effectively overcome the main defects of the KP-ABE approach that cannot regulate who decrypt the ciphertext. Thus, it can be applied to more real-world cases. However, CP-ABE faces a limitation itself. Because of its lack of constraint, it can only be used in some business applications, not working in other real-world applications. Moreover, it is less applicable in business scenarios due to low efficiency and rigidity, which bring adverse effect to the operation of the system. For example, the decryption process has the following conditions:

(1) Single set properties, users select single Multi-user or set properties from that set.

(2) During the phase of the decrypt, a collection of multiparty user sets of policy such as the authorization to decrypt, can withstand any collusion of users without any successful decrypting operation.

(3) The decrypting operations of the m users in the user set with degree k of set degrade with k, with k≥1.

In recent years, a large number of researches have been addressed to this problem. Experts have developed an efficient decentralization approach based on the job duplication and load balancing idea in CP-ASBE (Cipher Text Policy Multi-user Set Based Encryption) system, and then divide the job are sub-clouds and distributed them is used. Skewness strategy is used in this sense that based on the computing the diversity of resource utilization, if the variance of the resource loading is too greater, too large, some servers may be overloaded; but too small, it will cause excessive load balance; It is used to distribute according to the optimal resource resource scheduling.Moreover, the CP-ASBE have following advantages itself improve the system overload condition in the next two ways: predict the next day loading and migration (migration of VM) to effectively avoid overloading . In a survey conducted in 2016, a new security scheme for path routing was proposed, efficiently protecting against wormhole-free attacks on clouds but only small-scale, mainly defending against DoS attacks. A cryptographic protection scheme for malicious attacks during data communication was also expressed. As data the volume and number of types enlarges, the exponentially increase in loading capacity and runtime make it hard to detect malicious attacks[14]. Govindarajalu applied this new technique – a form of trusted attacks detection even in cloud environments of limited trust [15] Ongoing attacks in these cloud nodes are difficult to detect.

Zhang et al designed a cloud network based on encryption methods of authentication of a service oralicy [16].It is an usual thing in the further development of service orality communication networks for user authentication. But the long computing time is the god of their formerly mentioned fine-grained access control strategy. That's woefully a deficiency for further study. Computing time appreciably should be reduced in future research. Ma et al. pointed out single-threaded hashing that handling this timecost is very tedious, so they presented a new hashing scheme with the parallelism support [19]. It works on multi-core processors for reducing the required time. It was also made as a way to validate and analyse the proposed work. Speed-time graph also shows that finegrained access problem is solved, which is higher as compared to conventional approach. Similar work in multi-privileged one-to-many communications might be focused on, in the coming future.A fast handover authentication model that provides integrity protection in static cloud networks, proposed by Zhang et al. [14] uses proxy signatures to ensure integrity on one's behalf; but does not ensure integrity when a client on one end of a communication accelerates their request to another. Their proposed system can automatically carry out mutual authentication on integrity in static cloud networks and detect attacks launched from clients in static cloud. However, it cannot be used for verifying integrity processes. In one of his later inventions, called GIDAC [16], Govindarajalu came up with a brand new concept specially designed to identify and isolate attacks in low-trusted cloud environments, where detection of attacks happening in cloud nodes proves to be hard. In this model, all cloud nodes are authenticated during cloud network initialisation or bootstrapping.they proposed a semi-anonymous privilege control formulation for privacy in all above access control cases In the proposed model, by combining a trustworthiness verification approach that the cloud owner runs on his trusted nodes and a form of integrity-verification (similar to the approach taken in [30]), where the identity of the intruder will be recorded but hidden, they were able to use the privacy-preserving access control model to trace attacks.They presented a fully protected and elegant solution for anonymity and identity, called AnonyControl-F, based on the trusted computing model TPM algorithm. On a similar note, prior works handle only the privacy of GID, and fail to extend protection from other attack scenarios seen in the multi-attribute based approach. This multi-key translation puzzle is a complete solution to the problem. Further extensions such as identity/attribute obfuscations are also feasible. Zhang et al., proposed a modelling of cloud network's authentication architecture, based on the encryption methods of service-oriented communication [16]. It is a common form of user authentication in service-oriented communication networks and in the extension field, but the current fine-grained strategy of their access control leads to a very long computing time, which is a drawback in further study. In the future research, the computing time should be shortened substantially. Ma et al. [19] eliminated the need for single-threaded and time-consuming hashing and devised a parallelizable hashing scheme to lower the required time in practical cases. Their new proposed hashing scheme works in a multi-core processor. They claim their technique provides significantly higher security, speed, and efficiency. Another drawback of the CP-ABE model is that the data owner is completely unaware of the initial users who wanted to use the data. This results in loss of the fine-grained access. Using our proposed method, the access of the ORIGINAL users of the data can be obtained. So, user access would be allowed if the access of the original multi-user is the same as the access of the user itself. The validation and the analysis of the proposed work demonstrate that the fine-grained access problem is solved and is more efficient than the other existing approaches, leading to further work that might take a similar approach when it comes to multi-privileged one-to-many communications.A new modified version of CP-ABE approach using a searchable framework . Their technique uses partially invisible access structure along with Multi user revocation. The authors are using DBDH and DL assumption for making security of the method secure. Also can support as lazy proxy re-encryption. Multi-valued Multi-user helps in hiding users information through the access structure. Extremely it increases flexibility of this technique. In future the researcher can include anonymous decentralized multi-authority scheme to make it fully improved . On providing more efforts security , generality , efficiency and performance can be increased significantly[24].

3. Methods

Multi-user based encryption is another best scalable approach for cloud data security of multiple attributes and policies. This is also called ABE. Every legitimate cloud user is allotted a number of final permitted Multi-user sets, policies and a privacy key in this encryption and decryption process. A large number of several attribute-based encryption systems have been proposed and could not implemented in the literature for keyword based cloud data security. Since the size of the multi-users and Multi-user key setup phase are large, the size of the Multi-users is exponentially restricted to security parameters. Regarding the data set size PK , the data size is exponentially limited to the security parameters. This is a similar data size as the adverse concept of sub-policies in the KP-ABE.The basic concept of CP-ABE is almost the same as KP-ABE inverse process. This versatile method can be used for many other extensions as a fundamental unit. Therefore, the user can be permitted to select single attribute or mulitiple attributes from that set.

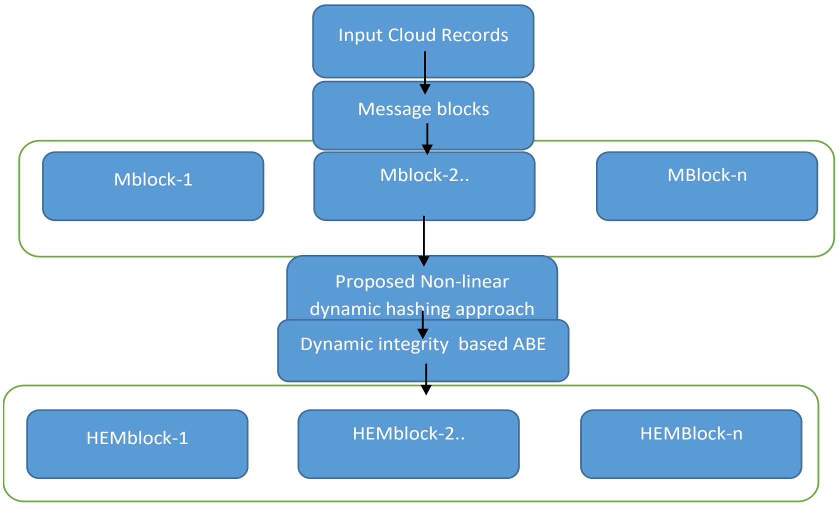

Figure 1: Proposed Model

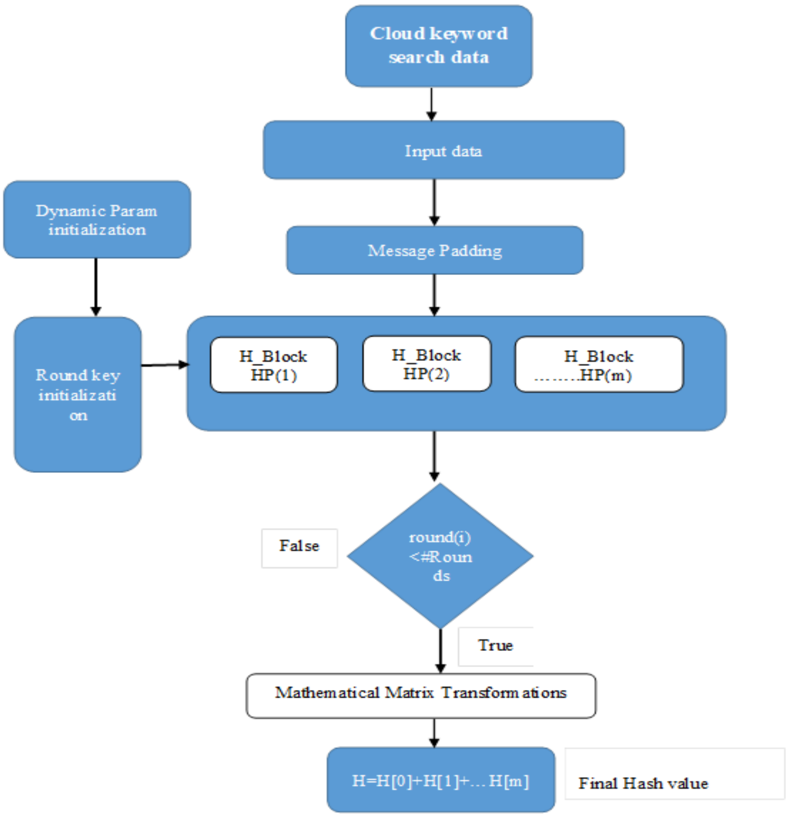

Figure 2: Proposed Mathematical matrix transformation process

would be represented by hierarchical structure. Meanwhile, role based access control techniques assign the user access control permissions and roles based on the business function of the organization. The role is the mapping between user and access permisson.The model introduces a cloud data security new approach using an innovative integrity based framework with hash mechanism.In general, cloud-based block hash security system, standard integrity algorithm: MD5, SHA, Whirlpool is employed to verity data integrity on the cloud environment. This new framework use a composite non-linear dynamic integrity algorithm for improving the hash bit change rate during hash construction.Figure 1 displays the model structure, which each keywords research dataset transaction is process within the hash framework to secure the blocks. The security block compute Hash and encrypt the input transaction. Finally, the encyrpt data of the transaction would be stored to the cloud storage. The storage function is used to perfom integrity verification. The integrity value computed in this model would be used to the data verfication in each keywords search.

Figure 2 shows the basic structure of the cloud-based hash framework. When the input data and cloud storage capacity have increased, the key risk of data security problems and threats in the remote cloud storage will be rising. It partly due to inadequate measures security. The internal attack is the unlawful access by own employees of sensitive customer data (including both encrypted and unencrypted) is a security threat.The model emphasizes confidentiality as a critical issue in cloud storage services, noting the threat posed by brute force attacks. To combat such threats, various cryptographic algorithms have been developed. For instance, the AES encryption algorithm is designed to thwart brute force attacks and addresses associated key management challenges, including key generation, distribution, storage, and regeneration.The model further supports multithreading, which allows simultaneous evaluation and upload of file chunks to different storages. This section introduces a hybrid integrity checking approach that conducts a series of non-linear mathematical transformations on input data to compute unique hash values for both encoding and decoding processes, as depicted in Figure 2.

1: Initialization of healthcare records as H_records and cloud server identifiers as Cloud_IDs[]. For each operation in the Cloud_IDs[]:

```
  Do
  If (H_records == "EHR")
  then
  Combined_Data = Cloud_IDs[i] + H_records;
  Done
```

2: Divide the collected data Combined_Data into k segments.

3: If the input data exceeds the segment size, pad the message with 0000001.

4: Divide each of the k segments into smaller sub-segments of 32 bits. The sub-segments are utilized to execute a series of non-linear transformations:

```
  Sub_Segments SS[] = Segments[Segment_Size/32];
  For each sub-segment in SS[]:
  Do
  Perform nonlinearT(SS[i]);
  Done
```

5: Execute a non-linear process named nonlinearT:

```
   For each byte in Combined_Data[](array of data in sub-segment partition):
   Execute segment operation
```

Process hash block

In the proposed chaotic piecewise non-linear chaotic function (CPLNCF), several distinct randomization parameters are utilized to derive a unique chaotic value for the permutation matrix formation. The extended piecewise non-linear chaotic function can be described as follows:

$$\begin{aligned}&\mathrm{CPLNCF}\ (v) = c_1 \cdot \frac{p(v+1)}{256} + c_2 \cdot |\sin\ (p(v+1))|^2 \\ &+c_3 \cdot |\sin\ ((v+1))| \cdot |\cos\ (p(v+1))|^2 \\ &+c_4 \cdot |\sin\ (v+1)| \cdot |\sin\ (p(v+1))|^3 \\ &+c_5 \cdot |\sin\ (p(v+1))| + c_6 \cdot (1 - \mathrm{CPLNCF}\ (v-1)) + c_7 \cdot (1 - \mathrm{CPLNCF}\ (v - 1\ {}^3\mathrm{D}^3 \delta \mathrm{ne}\end{aligned}$$

$$R_{total} = \sum \ \mathrm{CPLNCF}(v) \cdot \mathrm{Eigen}(M \cdot N)$$

$$H[index] = R_a \oplus R_b \oplus R_{total}$$

Here $r_1 \ldots r_7$ represent random numbers from the range (0,1).

In this non-linear chaotic function, M and N symbolize the dynamic permutation matrices. These matrices are generated using the CPLNCF function.

For each byte in $P[index]$:

$$R_a = \mathrm{SK}^{\mathrm{T}} \cdot [\mathrm{N} \cdot \mathrm{MaxEigen}\ (SK)]$$

$$R_b = \left(\frac{[M \cdot \mathrm{SumofSquares}\ (SK) \cdot \det\ (SK)]}{(\sum\ SK[index])}\right)$$

In the data processing approach, the input data is first mapped to a byte array associating the medical user ID and its corresponding record. This is further followed by mapping each multi-user data entry of transactions list. Now, input data is split up into k blocks, each block is a string of 8-bits. The size of the input data if exceeds the block size then padding is performed to add the extra data as the final part of input data. Once all the operations completed, each block is divided up into sub blocks and each sub block is 32 bits long. These sub blocks are further involved into mathematical transformations to calculate the hash value after completing process all the sub-blocks sub block hash value are cascaded to generate the final hash value for the given input data.

Multi-Authority ABE Techniques

The Multi-authority ABE (Attribute-Based Encryption) methodology is tailored for real-time cloud computing environments. This strategy involves several entities collectively managing the distribution of user attributes. It operates with numerous multi-user authorities, collectively referred to as K, and a singular central authority. Each multi-user is linked to a specific value, dk. The MA-ABE technique unfolds in several stages, with Phase 1 focusing on utilizing each user's attributes and access policies to derive the master and public keys essential for securing cloud data within a hash framework. During this phase, policies based on randomized hash keys are established for generating these keys.

1. **Define Geometric Distribution**:

$$\kappa(\xi) = \xi(1-\xi)^{\rho},\ \rho = 0,1,2, \ldots$$

2. **Define Uniform Distribution**:

$$\theta(\varphi) = \frac{\varphi}{\delta_1 - \delta_2} \text{ for } \delta_1 \leq \varphi \leq \delta_2$$

3. **Declare Group Elements**:

Suppose $\zeta_\rho, \Gamma_1, \Gamma_2$ are elements of a multi-user access control based cyclic group.

4. **Compute Bilinar Map with Geometric Distribution**:

$$\alpha = \text{bilinear\_map}(\zeta_{\rho}, \mu_{\kappa(\xi)});$$

5. **Multiple Published Key Using Max Function**:

$$\text{Mult\_PubK}(\gamma) = \text{bilinear\_map}(\Gamma_1, \max\{\mu_{\theta(\varphi)}, \mu_{\kappa(\xi)}\});$$

6. **Multiple Published Key with a Transformation**:

$$\text{Mult\_PubK}(\gamma_{\pi}) = \text{bilinear\_map}(\Gamma_2, \sigma_{\kappa(\xi)});$$

7. **Multiple Master Key**:

$$\text{Multi\_MasK}(\beta) = \text{bilinear\_map}(\Gamma_2, \sigma_{\theta(\varphi)});$$

8. **Composite Key Construction**:

$$\text{Multi\_MasK}(\gamma - \alpha) = \text{bilinear\_map}(\text{Mult\_PubK}(\gamma_{\pi}), \alpha^{\zeta_{\rho}});$$

9. **Hierarchical Key Derivation**:

$$\text{Mult\_PubK}(\eta) = \text{bilinear\_map}((\text{Multi\_MasK}(\beta))^{\zeta_{\tau}},\text{Mult\_PubK}(\gamma));$$

10. **Key Reduction and Combination**:

$$\text{Mult\_PubK}(\gamma - \alpha) = \text{bilinear\_map}(\text{Multi\_MasK}(\gamma - \alpha),\text{Mult\_PubK}(\gamma));$$

According to Fig. 2, a multiusers initialisation parameters taken from attribute list and integrity constraints policies are involved in this stage which refers to create cipher text in the hash framework. Referring to Figure 2; CP-ABE encodes data based of policy list according to CP-ABE Access tree policy.

According to Fig. 2, a multiusers initialisation parameters taken from attribute list and integrity constraints policies are involved in this stage which refers to create cipher text in the hash framework. Referring to Figure 2; CP-ABE encodes data based of policy list according to CP-ABE Access tree policy.

Multi-user Cipher text is given as MCT={ c1, c2, Multi-user Atree T,{ Pubk.h.(Zn) }}

Let the multi-user access policy be Γ_P, the multi-user public key be Γ_“Publickey” , and the multi-user master key be Γ_“MasterKey” . Then the two secret ciphertext parameters are:

$$\text{GeoDist}(\xi) = \xi(1-\xi)^{\phi},\ \phi = 0,1,2,\ldots$$
$$\text{UniDist}(\mu) = \frac{\mu}{\delta_1(1-\delta_2)} \text{ for } \delta_1 \leq \mu \leq \delta_2$$
$$c = H_{4096}(\text{GeoDist}(\alpha)^{*}\Gamma_{\text{Ai}}); \Gamma_{\text{i}} = 1,2,3 \ldots \Gamma_{\text{n}}$$
$$c_2 = H_{4096}(\text{UniDist}(\beta)^{*}\Gamma_{\text{Pi}}); \Gamma_{\text{i}} = 1,2 \ldots \Gamma_{\text{n}}$$

Here, α and β are elements of the cyclic group and are relatively prime to the multiplicative group.

The multi-user ciphertext is given as MCT = {c_1, c_2, Multi-user Atree T, {Pubk.h.(Zn)}}.

The decryption phase, takes cipher text, secret key, Access tree, and policies as input and delivers decrypted data.

4. Results

All the experiments are executed in a real-time java environment with the cloud platform. In this work, an experimental transaction dataset is captured in a real-time cloud server environment. Transaction keyword search information is also available on a real-time cloud platform. The experiment is implemented in a block hash framework. We used real-time Amazon AWS server and keyword search transaction data. The inputs to the hash tables are 1000 medical transactions. However in medical transactions, there are two entities are deal with doctor and patient influences in this evaluation, and it can be easily modifiable in our experiment. Two user accounts

(Doctor and Pediatrician) are used in our experiment to access the public data in cloud. The hash bit change measures the changing impact on integrity bits . The input data is changes from 100 to 175 with an increase of 35 units. The different block hash functions are evaluated in random. The traditional hash functions such as SHA, MD5, Whirlpool and parallel chaotic hash are considered in Fig. 1.

Table 1: Encryption algorithm for the large textual cloud file.

```
PI                                                                                              S
€a6767874c74a5c9e591622a8a04445d4853e3ecae93136faa4c4f0bcc422f1179e41d049e2c5f276109407f5b8
6ee64f1f5e9cc5aba4b1e9d884c3ef7e99e4ddE                    €gI_µ=†®Ìbª…t’-ÉžiP]=h¶[¶X&á§Ö„ÿ
        ‡T'^åÌZÜ!;u|oN“Ô#)7Nþ6æÁ‡ÿý’úÕ},/Å                  "æ|’—»ïf„(d¡Iƒ×¨¶Ñ\e%ŸbIÊ;p•  Üø(
§…µz…SØ`æ»'ÀAE
€a6767874c74a5c9e591622a8a04445d4853e3ecae93136faa4c4f0bcc422f1179e41d049e2c5f276109407f5b8
6ee64f1f5e9cc5aba4b1e9d884c3ef7e99e4dd[      E      €kcÁB’U)Œ"n…PSÌÖh©Ð(½+0ož£/~YÙ:×‰ºÔÙ„-
/f@#íé¢Åñ‹â•  äØ1[3%½¢à‘h Â§åÍë¹T-0¹U§ò————————————————————————————————————
:•  c/$Ë4ÚÑÜ)(Úqx…[¤ëi"¥æ,,W°Àb¡E   €D,ZDL,!$íúñ†ë`1Ìàà
éváÇy•  ÑŽ¯xë[§‹ìá——————————————————————z-
3KòqØàë¢îP°Ðxz}È®&'isX…™iù¬«i¦ZËì˜[“÷hÆ•}")“{Å"a¢Ìy&,CÍ¿Â@¾Úí®O?qHìë          @•  ™———
Ô`R¨z“€½í©8DÞ½¡–3©¬£ÐÏ‘v±n8ÞbßØºò°É3æf±ÉŒ}Ÿ±ô© 6—Tk 7A‘¯d7¼¹óNáµÚH¨——————————
"ÕdÆu".                                                              uC•  ‡>D€`EÊÏ%e.€Üüÿû‹¥
#ïÖ…¨•  tñX(’ÁY`Ö>«íÙvÓwå¶áDxšïÒVX¡•  Ç%áž“½'övFd)×éo"uhØš°i-

‘»16ùY4‰.ùîc—¥————————————————————————————————————————————————————
Ñ,dÓ0Ä>I        í|÷ð®GG"@–á*±0u——————————————————————————————————————————————
ÌHãU·p¢šƒJ·g¨“¯ö———————————————————————ùÏD#µ
múšUè?ê@ÅT¹RëMe˜Iç

çð]$¤m3'2o»-•  Ì"˜øKV+wŠÎ2%4————————————————————————ÙÐ,8é°ŽŠØÂQé•  µÍð92ùª<¶t
E:lñ•  áñ09o-/åªY—Ä·Ø-ôwÔÚq~ÛD}Eµ†?

0€.•  ô^:ŠÀ$œÈì÷¤T¤~0NjÑ0'^žŸÃd³I%î"¯ÂÊ™Ù²HGKD²Ðd-¥=1G’Ô
C¶7 þTV± $e!j¬à+ªæGª^ß©sõ¹%4é=¨×Í*pùŠ@út*Ù   A3ìÔ7Ä¢³Íãð0
Û‰BGr‡Ö)è¯&æ•  $´¶šqÊpÍ¡ºŽlÊd…âwÑß¸Uï õè^sîÕÓ—½t\\ìV(\•  Ùy6^q&Ñè=¦µó%Æ®"ìå

È•  b

€X{ÙI@Ú^>ÆÄ•  ½•  ¬

Ä:‹————————————————————————U<¡ocÒ'æô›d³„
>+r¢i%Ò úÙ1,Q› ÓÇä‹Šçãº J*Wc~’jh)£•  veÃ•  ð×Å~Ç8_§n——————————————————————————
”]¡o…»¹°÷1Gí-,‡&Ýz<Ÿ$3‰-+Ý§¶UÃq=Áû™S¨71åEløAÝ/7¨®‡\,KKvEÇ6ƒ‡›ÂŽÏüŸ‹¤+tHšâº
v¾è¥–Un/Û`øòØ@ñ³⁄4^tÈ>“)ItPr   ,ë‹ÏD•  ïÓÑZÝÑO}-¤j•

[®ÅOysÙ3¼!Š¯æ`á½‡b——————————————————————————————————————————————————
³ý@õ¤qÐG•  ¯¡d

kq±¶5îEA–÷n3•  6+gŸãÇXÉì¦%JG ŽÚbXÑb1nk`:Qél•  &á±éš•  †_ (š©ÁÄ2C«Y†˜.º,á-YM0Îo

iøË{&ëF5A-
À=zíPÜx°Ìüòfç¶å{š-È4‡†óŠ‡9x*`2Ö²YÑ¤l8å²oaÜWœÏAï†ƒ¤ß•  ?Í„³ãÔZ,ÿåÃª§"›kfj^¸ÇòFàIûÒÈãŸ–
Óµ‘º‹ýxý{5Ëâõöàx"ta+ù³O«
~ßâÒœ±œEO´Í^¼rôI»hŒ,½æã£7 Í:F>Ð««ñAžŒ»ÔcyÒÚ¹¶c&zùS.~ÂT§äÚS4ò¦(“oòÈ%•  •  „^2——————————
é®¤cQƒWuï@á]‹³/DAŒdm-ÃÑA°öZƒ_¤â-W[¢ï¶4òìdTGS•  x"ŽÖÃÁ¡’aYò×€5)#¨î(Ó!×r§²Ü4-Mbkg*”8´«ÔÅ
¤4À‰Ëê1ßù^¿«·Æä›xO=#åÝázZ
åEÁ –DHÜï~¹"Sr»¤:•  ——————————————————————á2¶¤%ÛE°^KíöT&òPà¼á±¶zZ;•  '—
;þ}:ïåÑ£Tv!/Ké°9ò(»í¾é–gþ¢Q³dÆŸVªÀÔö(´˜eÅµ3Ï•^
;¡qêÜ´ŽH°¢3û\È%ÇŽýº”å KË-“¢• ³Ë
@ò6hé¨@0…Ã{G¿{èUéHÓ$³'½ñAÄþ
Ð!³tycd*|·üæºa• P_ñÏË,´ŠH£sYÎÙ¬(ÿ¨•

ì¢ædžçp• dX¢Ñ"%bZ¶|»%³õŽ9àËPÓò"1ùÏ• LLð?ô¦• gn™UÜ-ø                    \-ÂüXŒ5ý=%‘
ñ{FáŠü¤4jÛí²„øZwð®0©"_Ò-=ú…ŽtGBCž• äfÛIäh)
›30pT?]µ:-P†¶û-#|Nü§¾&ú¬Jªêÿ°†»9wm• L-fé‹Í/7¶î

!Òjó-Ë¡ÅŠ6”]<AµóqÆèe=ž)ÄCGú£¹›       Ö4à³ï]Ð‹¡ž“js#Z¤ê¼ŸÒi:
½[_.N^ÇIR1Ò"ü¸4a IbmI
‹jþ¬×ølC0ï6+ö"9ÝN`§ýksîÌ+œè,úšLç‡¼iŽ,AÈ                                  ®Ð¬$Øl'=s
6lStÚ-Ò„X:vnÅÂI u0-• êføBôí
            ÿXø$©½‘·ÕXÓ!„¨òÔ»¨ïnMß×nÝ¼+†”ûã¿‰ÃsÆLÙÍ¨?tTíA3ºNÕ#õc®`ë+PJZë           v¦d'H8€µ±ã
ÚG1ªê’™0Áñ¥ªS“$®• «zm—YæLÀÉ`x°^Å¸- I-!ƒ-ú-7Cå;

õEÉ8èàÉìê{!E#Ã"-Àÿl+|áë,Lßý• •®®‰9õVûÀIVáI×½r¿ÉK)Äírðô÷óQåÓ¤÷°[|'Ááí¡˜@çï¿ÙcAÛùe+dq09À
wírŠuîAC”¥Ç$¥Â0! û‘6 aíwÙ¬ª<à• Ëfò“M• • "ßaì\—û˜í_l¿s…2M

’H-á-å½öS!ëÿF*Ûª§¶Fãœšàlùt L]k÷i• AßßO?qyÚK¸üwFE·?Yý9˜çfJÑ¾9Ùûó8•Fªh

¤Áâ›V‹¹£f¹

”Öí• mggeX5ÍåÛ5 ,“Ý
```

Table 2: Master key generated in the proposed encryption algorithm

```
™ŠÃà• !‘Õ)c2
V›¡™µ{â¸&‹ê\Í¯ŽTƒ%h(¦ˆÙ€ã
5¾y/à þ =—ÄÇ/¬¡Jm.Làn@ÇñÊU…ûnE  €F€§¸xÚ• ÍòòáBCiíÃÆ#-œI^‡à0”àwá}×üô¹÷uuüÎ• »VÊç¹ñš¥ª=-u
©hÁwe&ç(_Ô·€®²6

Ác†§)üô3å÷›1·Ã‡%ÑÑpñØF¯;• • s¶>Ã•Eª -à2•¥· áo |(-Ã«±Ç©          ?[ë¡k————————————
~8±½´1ÃL™ösÕR• •
åtößÉ".‡m²¬2                                                                          üÓéw¿
[¸¤£cÈ†|å6672¬DŒÝq¯ý~——————————————————v¿ÖãN½¥ËßE¿Ìç§X}&w£"MÛIU‘6ÆT¤òt•
ËE• EÀÐÖæð.8·°&ÈíÿÉE™¡NüR =    ¸œ3?zøâ¿˜X`ÙKJýÁÆrRÈŒ‡ï"R?7þ”    †Oõ1qçU\E“pìù¸"ö    „Hˆz²4-
AXjáþ*ìüD¾É«žnàH;$• ¸t——@˜½Ä-
gB• |5$oÈf™Ë§Y2ûÉà“ë0š…“pû%I=¤ü¯¼ôˆÎ%ñQ/¬OÅ‡€øÂ5‰ÛÏåÉ‹¸KaQ¦Cát…(w°Géâ—• ¨"®µ·
           ûÏãÃSU§äL        ’îR²à¥³oI°ÉÚW²—Õñ¸¿M1à¤QÉñ›• ´2• úQQ~œMqÐ—ÌžÃ
]\†4Èêò2ºÖg)Mw¯¸j[Á——————————————————————————————————
ÎSÎ²ýPñ´‹žoI‡%¸*ê>Î…g¯”• dE‡J’?«è™¦ëxõówÒlñ–5ð¥ˆîõÏ_O1¨1dT¹g·öæ—
T®/ÝúPsØÆú³Ob¡ï)þë!1æÂV·¹0ˆµö¡\DÖÕQ,CkOt’±Tv;¬I0• ƒÆWÁ]€-iôPo„ÏÆT,•mL>

!‘;Ýåæ7
```

Table 2 describes the master key value of the input data, which is determined in encryption process . The master key is generated in the setup phase of the proposed encryption model using the integrity value , policies list and attributes list .

Table 3: Public Key:

```
„Ññ¨vßd¦O¬FG¶)Ÿ,ó¬‘¥ÕAû¤È2¾CX‘ä¿Gdlaý£óÔx+ìµ‡CÒg8Nù¨\k&³G8*û¹O¥MÙHžê¶êßzàs½dx
ƒÍœŽÂ• ëüEð’J8lIq ·¥™‰È¾4#P£\Rf¡koêÊ[]…B×• “ÐJŸG

o¦CÈ©ËÀ³W¬I
÷=¿‡#ú8Ž[¾èÒë‡B;ê¬®>£ë• ë …JN«J’OüØ• u;ÍJ³ÃàÐðŠ'Çv• îÚJËqn0K
```

```
.a_øÂOßÙˆÕµë©‹wæì¾¨[ò»HËýçGü¼˜L´±UÁ5l9Qa-4EŠâ!LiŒp‹6?LW–�'ÚT'q7µ„w`I¡
[¼6ÔÎ>ß'Òjc`$ªµó˜I.€KaÖ�EUÏ
˜¦-žC—ÞJâØ_

yåyéþV• ãÔúÕ8G†¢UuÓ‡ð¿²£%‰áÜ©à_• %$™_ÀpHˆ%¬£ø#MkqåD&˜X‹z!Ð,YIC0C
ƒ{Ãpô[ÛtÜ;^Z";Ar/ì;^žiYÎ]S¾• &BÏ‹%Ñ�pŠÊñÁ¾%õÍº‰tšl=Í          Ô-¢ÔCËvèZö1)dAõg|

¤ß„L3H%4¦Ä'¢Îv'ÒÇ±&,wuþÝq@È• Ì`ìÆ±¢ÂdU½×¿Ëk}"÷Ýv",»ì#X%*Ðs4SDÎi¨        õJIÙ_Û]à}
,9¦8†$róH[hs————————————————————åß&Ï¶• $-hÀ÷u:«,š¼ÞY¤I¥
```

Table 4: A sample of encrypted data for keyword search in the block hash framework.

```
'X9§Ëü!ª¥˜uKÁ————————————————————U‹ž¤Cš"˜~IÍ#{Ö¯£4º{¡• +Õ.·4P:AÆ5• *vš&l¹ýstQ)'Th8?
ý/—!5h`¹U¥á'øºèJkoòz,3ZÛXg

Š†®H|èü$ëAà£E`Þ…)j¶zm>E   €Uæ°Ìdâÿ´Ï ¯êCB55¯yd}Ó‹›
Þ+¦_ùÒâðÃ·Ž€¼Iƒ/H3óÇ¦Çýže,OP • ~šoâõÜÌö}%‡âñª          Òò·¬5]A¼å„\•• Ô¾õöñnŸ@oÁ©|üFCû÷b-
Azï• 1
9\¶————————————————————sPcpeRMi4• L¯Äzï
Ú63T¿Õ¤T¹0{F¡<à˜cùƒ/lÉ•AVm{ö~ø————————————————————\_                    n-‰ßYglñ³
í‰-&¥W6Š`sSQÂ,%ï–kÎÄ                                          q5#¶o&S!yæ°:"('/zó/»ØÊÁ¯!¤• >e;Bðó—
zËS]&Z¼¾®·bÚÒ6ŽN• mç¿Ìðº‹É`€F+!2²Wy€…ÞI®5T¨÷èÝ5"Š————————————————————
.ŽH}ŒŽð• sˆÈMÌ|ŸÌÝ³3œx                 ¯$3nüñ'F"ƒÔ¬Š              ˜Â…I:%Í-tÑ?ùMEÛ«¾>ÈœÃ—lkÑ0ù"-
• ûha¢âÍêáž&7iÔüálM,g(  q«5¸¿U‰¬• <$4
Á¦ÓX7————————————————————¦|©úyæQ…÷…üA·Óÿ¿óiG#£É?Òf' ¸iÃRÁœš
Èè3R7æÞâpÝ—ÃnØTšÉÝ Â½‹Ä÷Ä˜û
A¼°œõ\;SÏÛ®Ø:•r¿d-,°.¦Ìz&ôî Z¨ê• XÆMFa‰³êâT
qRtóÄ'`Ž�F HOûÎ3[Iÿ       ˜-7Êç-œÝ‡I>'|`/UŸæ-©æYÊÊÐg|6ZVÇl6Éý˜ÈÙ-½-Qò/• )Ä1SÉnÙ(B;œ-
Ù©ë*::áµðY_"jÉº• Ò~¸=Ø• ÖûdèÄÚ4Ÿjw-›c˜)&YË‡O—[Ñó*ØSrÂ`Ý$\*{å¦Ò¡9,g-
ŠçáT{0/¡uã'©J>©ŸÝí! ª3£-^¤!.y°¼w<ÊÊû�ìK¢*ûªTèÎ-
X˜ùîw• ™òÀD=~™Õ®oîÓûU§)wÍ\ÒB)ÖðÆœX»öZ`[ìC-• • .j<EG§„ê?Vn4|0íS¾•
• ñÉ/‰Ê"yhqnáCò•Á„Ó»¢$Ëý• ðáqv776éV• á
:[|á^õ¿|o¾HŽ·ˆ»¿i[<*UÚÛ]ûA

————————————————————]ùIò¾´#«• èvKKò%@BªIlµj³Z2Q£™Ó˜<.`~u]ÝåÌ:W

,¹H¤À• âè}áÄ%»),-Ÿé§b»gMÿ=ÈÝ_)8ß>rBâú¡>;{m                                          ?w'=6'Ð-
ü„v*ü;Whç`¸z÷‡ƒßþÆò,Z&Öš•„ÇŠË´^ä¼Þ¥pÒ€l9ð�ì.ËzüR½µË`%]ã• ½¦î«ÌÛÄF|7²;eNã‹
          K ±ì;tß1• ª• `œ|Ú˜/ûŒZ&âD
DŽá-ik!,uïÇh` \ŒùÄž:Ô¡ÓÊNv†AÓÜÍ÷þÚK<&˜²eñÙÓ+J• w¶d¤S-‹4šÎ¿VÛÃ…GAQù-Xaòå@R• i–
zõRtÑ>‹Öw [çªÇíþUZÃ–kæÚ:U™-\ˆZÝ'Ô¶ÈNæ,h~Ç¼˜×uÊ/Ûû————————————————————
ª2æ¾�Öjë

• ¯°L1$`™ÂTk™• _Ô+®"\°ù^ Ó]• '…þva‡a¤©â-*————————————————————
\>Æ:¥ÒNJÚ3• PÞÄøU¸1éçûT³ÕÑ'¨™E,ÓÉÉËë_âmÃë————————————————————
HX_©ÿp,±Ú7ëÎ• Ñ
µÏn%ë^-?ï————————————————————ûÌ,ô q• |%«fieg35ÄÉã————————————————————
1—2#ó¦7ë*EÛw,w1 –• �áÆm.

s—lN'ªtè¯¼-e_Î´îÓ×µN±ºà¿hCÀŒ?63¸š ºášà½‡W„ZÐñ1²îžœ€ÉoÝ Á¤/MÏð2ëS¥ ÿøƒ•IÃ¹ñ¯kDË1_ÄRÃ+

….$Ì#gŠ¿sc¦è¡ò   ×å·&ƒyðÊ¥         ˜nêF      Æ¿äiÖ>à
```

Table 5: Results for the integrity value of a single user in block-wise processing.

```
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x60e80756
r = (sum_bytes c_[j]) = 421
r = R_mn+ (r % S) = 3
0x4b6c4693
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xab8541c5
r = (sum_bytes c_[j]) = 566
r = R_mn+ (r % S) = 3
0x6f80fb63
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xe4378290
r = (sum_bytes c_[j]) = 557
r = R_mn+ (r % S) = 4
0x5cf4b563
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x8af406df
r = (sum_bytes c_[j]) = 611
r = R_mn+ (r % S) = 3
0x2b6d404f
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xa1994698
r = (sum_bytes c_[j]) = 536
r = R_mn+ (r % S) = 3
0x1d93d29c
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xbc0a9404
r = (sum_bytes c_[j]) = 350
r = R_mn+ (r % S) = 2
0x2cdc890b
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x90d61d07
```

```
r = (sum_bytes c_[j]) = 394

r = R_mn+ (r % S) = 6

 Final hash integers

Proces block: [0x32, 0x39, 0x30, 0x2c, 0x20, 0x32, 0x39, 0x30, 0x80, 0x01, x00, 0x08, 0x00, x00, 0x08, 0x00,
x00, 0x08, 0x00, x00, 0x08, 0x00, x00, 0x08]

Partitinp block into sub block p_[]: [0x3239302c, 0x20323930]

Initializes c0 = (last 4 bytes of h ) = 0x00000000

Initialize r = RMAX = 6

Realizing iteration 0

0x18ba54ce

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x2a8364e2

r = (sum_bytes c_[j]) = 499

r = R_mn+ (r % S) = 6

0x35bee613

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x3f0fbbc1

r = (sum_bytes c_[j]) = 458

r = R_mn+ (r % S) = 5

0xe8b47dae

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x57bac66f

r = (sum_bytes c_[j]) = 582

r = R_mn+ (r % S) = 4

0x9a18b768

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xcda27107

r = (sum_bytes c_[j]) = 487

r = R_mn+ (r % S) = 4

0xea36d3dd
```

```
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x2794a2da
r = (sum_bytes c_[j]) = 567
r = R_mn+ (r % S) = 4
0x0000b052
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x27941288
r = (sum_bytes c_[j]) = 341
r = R_mn+ (r % S) = 3
0x2823d94e
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x0fb7cbc6
r = (sum_bytes c_[j]) = 599
r = R_mn+ (r % S) = 6
0x649f80b2
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x6b284b74
r = (sum_bytes c_[j]) = 338
r = R_mn+ (r % S) = 5
0xf5d74b63
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x9eff0017
r = (sum_bytes c_[j]) = 436
r = R_mn+ (r % S) = 3
0xe56f24ba
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x7b9024ad
r = (sum_bytes c_[j]) = 476
r = R_mn+ (r % S) = 3
0x48defb95
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x334edf38
r = (sum_bytes c_[j]) = 408
```

```
r = R_mn+ (r % S) = 5

0x8ac1546d

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xb98f8b55

r = (sum_bytes c_[j]) = 552

r = R_mn+ (r % S) = 4

0xca64d90d

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x73eb5258

r = (sum_bytes c_[j]) = 520

r = R_mn+ (r % S) = 2

0xeb36542d

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x98dd0675

r = (sum_bytes c_[j]) = 496

r = R_mn+ (r % S) = 3

0xe692aace

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x7e4facbb

r = (sum_bytes c_[j]) = 564

r = R_mn+ (r % S) = 6

0xedcd780c

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x9382d4b7

r = (sum_bytes c_[j]) = 672

r = R_mn+ (r % S) = 4

0xdaee69fd

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x496cbd4a

r = (sum_bytes c_[j]) = 444

r = R_mn+ (r % S) = 6

0xe2659955
```

```
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xab09241f

r = (sum_bytes c_[j]) = 247

r = R_mn+ (r % S) = 4

0x713790f2

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xda3eb4ed

r = (sum_bytes c_[j]) = 697

r = R_mn+ (r % S) = 4

0x779c529e

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xada2e673

r = (sum_bytes c_[j]) = 680

r = R_mn+ (r % S) = 2

0x95d13e5f

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x3873d82c

r = (sum_bytes c_[j]) = 431

r = R_mn+ (r % S) = 3

0x02474a8f

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x3a3492a3

r = (sum_bytes c_[j]) = 419

r = R_mn+ (r % S) = 6

0x09293bd6

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x331da975

r = (sum_bytes c_[j]) = 366

r = R_mn+ (r % S) = 3

0x63fb6288

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x50e6cbfd

r = (sum_bytes c_[j]) = 766
```

```
r = R_mn+ (r % S) = 3

0x686748c7

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x3881833a

r = (sum_bytes c_[j]) = 374

r = R_mn+ (r % S) = 6

0x150fff39

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x2d8e7c03

r = (sum_bytes c_[j]) = 314

r = R_mn+ (r % S) = 6

0x82b56cd0

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xaf3b10d3

r = (sum_bytes c_[j]) = 461

r = R_mn+ (r % S) = 3

0x5e4dace1

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xf176bc32

r = (sum_bytes c_[j]) = 597

r = R_mn+ (r % S) = 4

0xafa30474

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x5ed5b846

r = (sum_bytes c_[j]) = 561

r = R_mn+ (r % S) = 3

0x38902de2

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x664595a4

r = (sum_bytes c_[j]) = 484

r = R_mn+ (r % S) = 6

0x8d46f4d4
```

```
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xeb036170

r = (sum_bytes c_[j]) = 447

r = R_mn+ (r % S) = 4

0xc8642307

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x23674277

r = (sum_bytes c_[j]) = 323

r = R_mn+ (r % S) = 5

0xa4962048

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x87f1623f

r = (sum_bytes c_[j]) = 537

r = R_mn+ (r % S) = 4

0xbbbb641f

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x3c4a0628

r = (sum_bytes c_[j]) = 180

r = R_mn+ (r % S) = 2

0xa2a32a6b

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x9ee92c43

r = (sum_bytes c_[j]) = 502

r = R_mn+ (r % S) = 4

0x3684c2f9

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xa86deeb2

r = (sum_bytes c_[j]) = 693

r = R_mn+ (r % S) = 5

0x6b88adb0

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xc3e54302

r = (sum_bytes c_[j]) = 493
```

```
r = R_mn+ (r % S) = 5

0xa07e1b1d

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x639b5817

r = (sum_bytes c_[j]) = 365

r = R_mn+ (r % S) = 2

0x52103ff6

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x318b67e1

r = (sum_bytes c_[j]) = 516

r = R_mn+ (r % S) = 3

0xbc1dc386

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x8d96a46f

r = (sum_bytes c_[j]) = 566

r = R_mn+ (r % S) = 3

0xdd5f6d19

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x50c9c976

r = (sum_bytes c_[j]) = 600

r = R_mn+ (r % S) = 2

0xcccf6ac0

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x9c06a3be

r = (sum_bytes c_[j]) = 515

r = R_mn+ (r % S) = 2

0x35e11154

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xa9e7b2ea

r = (sum_bytes c_[j]) = 812

r = R_mn+ (r % S) = 4

0x9b6659d9
```

```
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x3281eb3b

r = (sum_bytes c_[j]) = 473

r = R_mn+ (r % S) = 5

0xded76ef9

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xec5685c2

r = (sum_bytes c_[j]) = 649

r = R_mn+ (r % S) = 6

0x0c87a7a7

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xe0d1226d

r = (sum_bytes c_[j]) = 576

r = R_mn+ (r % S) = 3

0x180048f8

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xf8d16a95

r = (sum_bytes c_[j]) = 712

r = R_mn+ (r % S) = 4

0xdac9b84b

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x2218d2d6

r = (sum_bytes c_[j]) = 482

r = R_mn+ (r % S) = 4

0x34dfe4ff

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x16c73629

r = (sum_bytes c_[j]) = 316

r = R_mn+ (r % S) = 3

0xfd278746

r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xebe0b167

r = (sum_bytes c_[j]) = 739
```

```
r = R_mn+ (r % S) = 6
0xd8416cc2
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x33a1dda5
r = (sum_bytes c_[j]) = 598
r = R_mn+ (r % S) = 5
0xb174d96b
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x82d504c6
r = (sum_bytes c_[j]) = 545
r = R_mn+ (r % S) = 2
0x154dfe20
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x9798fae6
r = (sum_bytes c_[j]) = 783
r = R_mn+ (r % S) = 5
0x5b3e40ed
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0xcca6ba03
r = (sum_bytes c_[j]) = 559
r = R_mn+ (r % S) = 6
0xcac5a609
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x06631c0a
r = (sum_bytes c_[j]) = 143
r = R_mn+ (r % S) = 5
0x13af5cf0
r_[i] = pt_[i] ^ r_[i-1] ^ b = 0x15cc40f2
r = (sum_bytes c_[j]) = 531
r = R_mn+ (r % S) = 3
```

Table 5 illustrates the sample integrity value within the hash framework, focusing on a single hash record. Table 5 details the exact process by which integrity values are calculated for one hash record according to the hash framework. It clearly illustrates the intermediate inputs, intermediate calculations, and the final integrity value for the hash record. It serves as a helpful visual representation to learn how integrity values are calculated according to the hash framework.

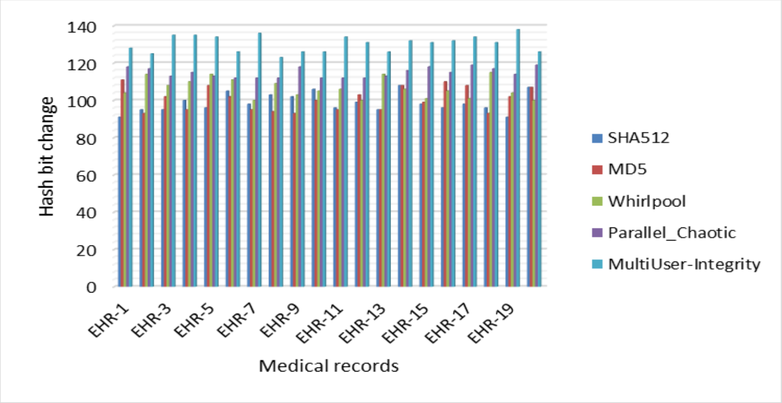

Figure 3: Comparison of performance for the proposed integrity verification model vs existing hashing approaches for attribute reconstructability, with a fixed hash length of 2048 bits.

This comparison assesses the contribution of the integrity-verification model that we have proposed and the verified hashing techniques that already exist, with respect to reconstructability, i.e. how much of the original attributes can be recovered or reconstructed by an adversary that tries to regain these attributes when a fixed hash length of 2048 bits is considered.

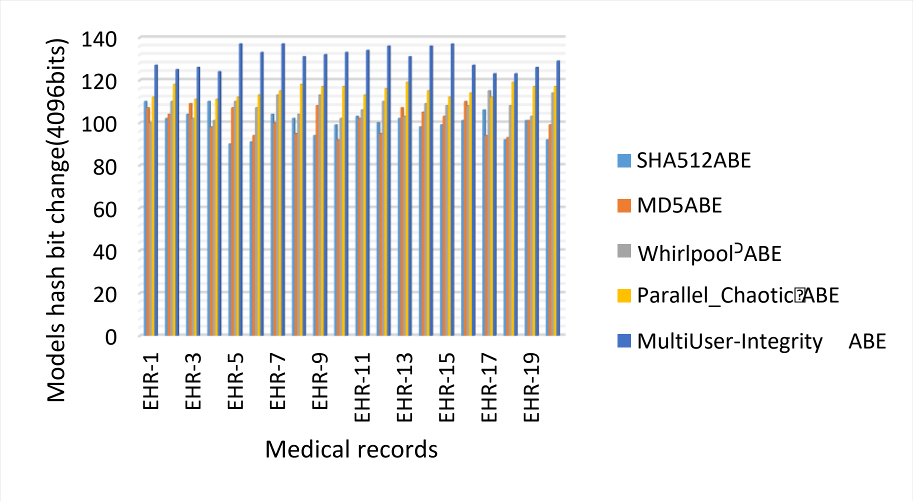

Figure 4 illustrates the performance comparison between the proposed integrity-based encryption model and existing hash approaches for variable size attributes and keyword search data, using a hash size of 4096 bits.

Figure 4 Show size attributes and keyword match data for the proposed integrity-based encryption model, the signature-hash, and HMAC for variable size attributes, under 4096 bits hash size. We show how well the proposed model performs comparing to existing hash approaches in encryption, integrity checking, and keyword searching for the variable size attributes. The Figure help us understand the differences among these approaches with clear illustrations.

Table 1: presents a performance analysis comparing the proposed integrity model with conventional approaches for keyword search data, using a hash size of 2048 bits.

Transactions	SHA512	MD5	Whirlpool	Parallel_Chaotic	MultiUser-Integrity
KeywordSearchFile-11	5800	5617	4964	4867	4116
KeywordSearchFile-12	5514	5245	5826	5410	4118
KeywordSearchFile-13	4862	5110	5578	4944	4357
KeywordSearchFile-14	4969	5381	5108	4945	4474
KeywordSearchFile-15	5353	5028	5168	4985	4055
KeywordSearchFile-16	4864	5031	4904	5304	4405
KeywordSearchFile-17	5510	5102	5209	5317	4430
KeywordSearchFile-18	5683	5400	5363	4923	4548
KeywordSearchFile-19	5480	4905	5845	5792	4521
KeywordSearchFile-20	5513	5625	5463	5794	4646

The above table, further analyses comparatively the speed and performance of the proposed integrity model and conventional approaches of handling keyword search data. This is illustrated through a comparative analysis of the two models using a hash function of 2048 bits.

The table is used to find the encryption and decryption runtime, and search efficiency with different attributes.

Table 2: Performance comparison of the proposed integrity model and existing approaches for the keyword search data with Hash size 4096 bits.

Transactions	SHA512	MD5	Whirlpool	Parallel_Chaotic	MultiUser-Integrity
KeywordSearchFile-1	5782	4849	5537	5348	4641
KeywordSearchFile-2	4861	5617	5762	5448	4281
KeywordSearchFile-3	5641	5031	5565	5567	4421
KeywordSearchFile-4	5149	4985	4935	5833	4414
KeywordSearchFile-5	5341	5406	5337	5380	4612
KeywordSearchFile-6	5747	4926	5470	5772	4615
KeywordSearchFile-7	5537	5320	4853	5457	4366
KeywordSearchFile-8	5601	5726	5042	5373	4631
KeywordSearchFile-9	5329	4877	5346	5390	4521
KeywordSearchFile-10	4928	5590	5599	5155	4321

The above table presents a detailed comparison of the performance of the proposed integrity model and the traditional solution in dealing with keyword search data specifically. It is done by hash value of 4096 bits. This table include various parameters such as encryption and decryption speed, search efficiency, security in terms of runtime(ms)

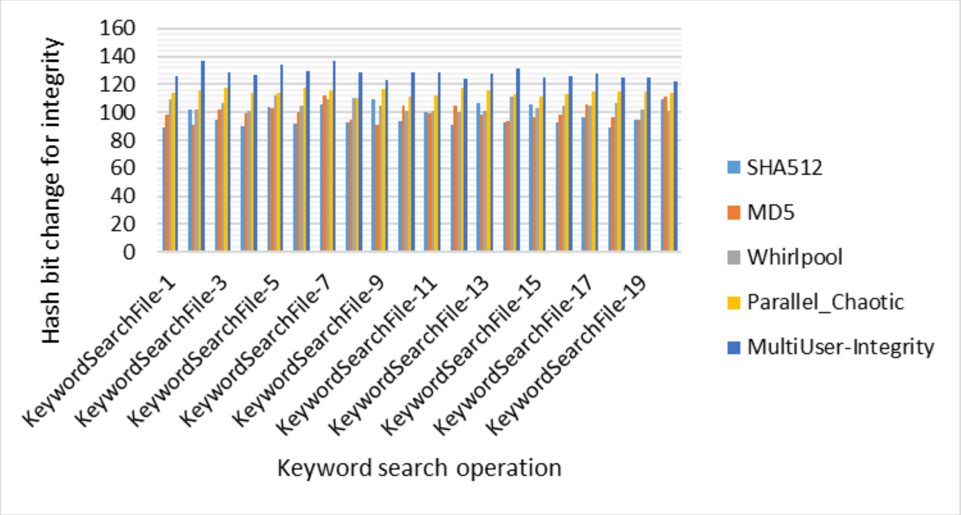

Figure 5 displays the analysis of proposed model and conventional methods performance in different hash bit length changes, keyword search data and attribute size variations.

Figure 5 analyses in detail the performance of the proposed integrity model and three conventional approaches to address a given task, when faced with three different conditions or scenarios. The three conditions were (1) given attribute sizes, (2) given changes to hash bit lengths used, and (3) given keyword search data.

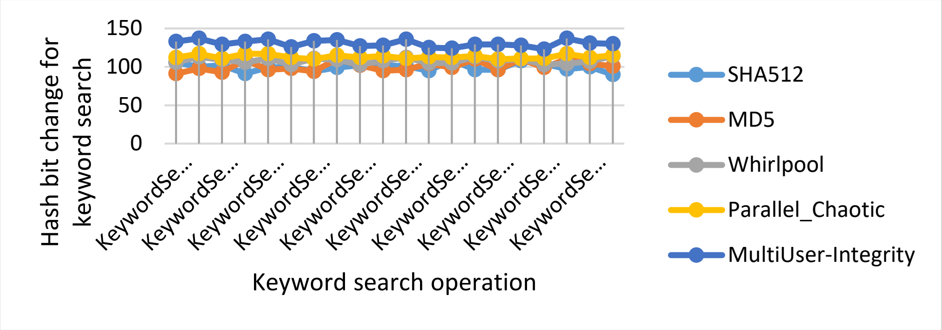

Figure 6 compares the proposed integrity-based multi-user average encoding and decoding model with traditional encoding models in terms of runtime analysis for keyword search data.

5. Discussion

In this work, a hybrid keyword search-based cloud data security framework is devised and implemented for the real-time cloud computing environment. In the keyword data search, a hybrid integrity-induced keyword search method is designed to increase the efficiency of the concurrent integrity and encryption operation in the large database. The reason why all the traditional keyword search model is considered independent from integrity checks during the keyword search the document applying is because all the large amounts of data searching operations cause high memory and run time occupancy in the computing environment. In this work, a hybrid integrity model combined with an encryption model is proposed to achieve the high efficiency of the run time of both the integrity and encryption model in the keyword searching operation. The final result indicates that the keyword search optimization model that is proposed has better efficiency than the existing models. That is, its run time and hash bit change are better than the existing model.

References

[1] H. Yin, Z. Qin, J. Zhang, H. Deng, F. Li, and K. Li, “A fine-grained authorized keyword secure search scheme with efficient search permission update in cloud computing,” Journal of Parallel and Distributed Computing, vol. 135, pp. 56–69, Jan. 2020, doi: 10.1016/j.jpdc.2019.09.011.

[2] H. He, J. Zhang, P. Li, Y. Jin, and T. Zhang, “A lightweight secure conjunctive keyword search scheme in hybrid cloud,” Future Generation Computer Systems, vol. 93, pp. 727–736, Apr. 2019, doi: 10.1016/j.future.2018.09.026.

[3] P. Prajapati and P. Shah, “A Review on Secure Data Deduplication: Cloud Storage Security Issue,” Journal of King Saud University - Computer and Information Sciences, Nov. 2020, doi: 10.1016/j.jksuci.2020.10.021.

[4] F. Yin, R. Lu, Y. Zheng, J. Shao, X. Yang, and X. Tang, “Achieve efficient position-heap-based privacy-preserving substring-of-keyword query over cloud,” Computers & Security, vol. 110, p. 102432, Nov. 2021, doi: 10.1016/j.cose.2021.102432.

[5] J. Cui, H. Zhou, H. Zhong, and Y. Xu, “AKSER: Attribute-based keyword search with efficient revocation in cloud computing,” Information Sciences, vol. 423, pp. 343–352, Jan. 2018, doi: 10.1016/j.ins.2017.09.029.

[6] A. Cuzzocrea, C. K. Leung, B. H. Wodi, S. Sourav, and E. Fadda, “An Effective and Efficient Technique for Supporting Privacy-Preserving Keyword-Based Search over Encrypted Data in Clouds,” Procedia Computer Science, vol. 177, pp. 509–515, Jan. 2020, doi: 10.1016/j.procs.2020.10.070.

[7] M. R. Senouci, I. Benkhaddra, A. Senouci, and F. Li, “An efficient and secure certificateless searchable encryption scheme against keyword guessing attacks,” Journal of Systems Architecture, vol. 119, p. 102271, Oct. 2021, doi: 10.1016/j.sysarc.2021.102271.

[8] N. Pakniat, D. Shiraly, and Z. Eslami, “Certificateless authenticated encryption with keyword search: Enhanced security model and a concrete construction for industrial IoT,” Journal of Information Security and Applications, vol. 53, p. 102525, Aug. 2020, doi: 10.1016/j.jisa.2020.102525.

[9] Q. Xu, Q. Zhang, B. Yu, N. Shi, C. Wang, and W. He, “Decentralized and Expressive Data Publish-subscribe Scheme in Cloud based on Attribute-based Keyword Search,” Journal of Systems Architecture, p. 102274, Sep. 2021, doi: 10.1016/j.sysarc.2021.102274.

[10] Q. Xu, C. Tan, W. Zhu, Y. Xiao, Z. Fan, and F. Cheng, “Decentralized attribute-based conjunctive keyword search scheme with online/offline encryption and outsource decryption for cloud computing,” Future Generation Computer Systems, vol. 97, pp. 306–326, Aug. 2019, doi: 10.1016/j.future.2019.02.067.

[11] P.-W. Chi and M.-H. Wang, “Deniable search of encrypted cloud-storage data,” Journal of Information Security and Applications, vol. 58, p. 102806, May 2021, doi: 10.1016/j.jisa.2021.102806.

[12] M. Hozhabr, P. Asghari, and H. H. S. Javadi, “Dynamic secure multi-keyword ranked search over encrypted cloud data,” Journal of Information Security and Applications, vol. 61, p. 102902, Sep. 2021, doi: 10.1016/j.jisa.2021.102902.

[13] H. Zhong, Z. Li, J. Cui, Y. Sun, and L. Liu, “Efficient dynamic multi-keyword fuzzy search over encrypted cloud data,” Journal of Network and Computer Applications, vol. 149, p. 102469, Jan. 2020, doi: 10.1016/j.jnca.2019.102469.

[14] Q. Zhang, G. Wang, W. Tang, K. Alinani, Q. Liu, and X. Li, “Efficient personalized search over encrypted data for mobile edge-assisted cloud storage,” Computer Communications, vol. 176, pp. 81–90, Aug. 2021, doi: 10.1016/j.comcom.2021.05.009.

[15] M. Govindarajalu and S. R. Suresh, “Intelligent secure phrase search of encrypted data in cloud based IoT,” Materials Today: Proceedings, Jan. 2021, doi: 10.1016/j.matpr.2020.11.612.

[16] X. Zhang, Y. Tang, H. Wang, C. Xu, Y. Miao, and H. Cheng, “Lattice-based proxy-oriented identity-based encryption with keyword search for cloud storage,” Information Sciences, vol. 494, pp. 193–207, Aug. 2019, doi: 10.1016/j.ins.2019.04.051.

[17] A. S. AlAhmad, H. Kahtan, Y. I. Alzoubi, O. Ali, and A. Jaradat, “Mobile cloud computing models security issues: A systematic review,” Journal of Network and Computer Applications, vol. 190, p. 103152, Sep. 2021, doi: 10.1016/j.jnca.2021.103152.

[18] M. Li, G. Wang, S. Liu, and J. Yu, “Multi-keyword Fuzzy Search over Encrypted Cloud Storage Data,” Procedia Computer Science, vol. 187, pp. 365–370, Jan. 2021, doi: 10.1016/j.procs.2021.04.075.

[19] M. Ma, S. Fan, and D. Feng, “Multi-user certificateless public key encryption with conjunctive keyword search for cloud-based telemedicine,” Journal of Information Security and Applications, vol. 55, p. 102652, Dec. 2020, doi: 10.1016/j.jisa.2020.102652.

[20] J. Cui, H. Zhou, Y. Xu, and H. Zhong, “OOABKS: Online/offline attribute-based encryption for keyword search in mobile cloud,” Information Sciences, vol. 489, pp. 63–77, Jul. 2019, doi: 10.1016/j.ins.2019.03.043.

[21] 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, Apr. 2020, doi: 10.1016/j.ins.2019.12.063.

[22] H. Yin, Z. Qin, J. Zhang, L. Ou, F. Li, and K. Li, “Secure conjunctive multi-keyword ranked search over encrypted cloud data for multiple data owners,” Future Generation Computer Systems, vol. 100, pp. 689–700, Nov. 2019, doi: 10.1016/j.future.2019.05.001.

[23] L. Cheng and F. Meng, “Security analysis of Pan et al.’s ‘Public-key authenticated encryption with keyword search achieving both multi-ciphertext and multi-trapdoor indistinguishability,’” Journal of Systems Architecture, vol. 119, p. 102248, Oct. 2021, doi: 10.1016/j.sysarc.2021.102248.

[24] M. Miao, Y. Wang, J. Wang, and X. Huang, “Verifiable database supporting keyword searches with forward security,” Computer Standards & Interfaces, vol. 77, p. 103491, Aug. 2021, doi: 10.1016/j.csi.2020.103491.

[25] Y. Liang, Y. Li, Q. Cao, and F. Ren, “VPAMS: Verifiable and practical attribute-based multi-keyword search over encrypted cloud data,” Journal of Systems Architecture, vol. 108, p. 101741, Sep. 2020, doi: 10.1016/j.sysarc.2020.101741.