Ensuring Privacy Preservation Access Control Mechanism in Cloud based on Identity based Derived Key

Suresha D
Research Scholar, VTURRC
Associate Professor, Dept. of CSE
Dr. AIT, Bangalore, Karnataka

Dr. K Karibasappa
Professor, Dept. of CSIT
Graphic Era Deemed To Be University
Dehradun, Uttarakhand

Dr. Shivamurthy
Associate Professor, VIAT,
Muddenahalli, Bangalore,
Karnataka, India

***Abstract*—Cloud computing is a dominant technology that involves massive amounts of data storage and access via the internet. Because there is a large amount of data stored in data centers, it is critical to implement appropriate access control mechanisms over data stored in a cloud. Today, there are numerous access control mechanisms available to provide confidentiality, privacy, and data origin authentication in a cloud environment. The available access control techniques may have a higher computational overhead and lack security concerns. In this paper, we designed and implemented a privacy-preserving access control in cloud computing using derived key identity-based encryption. The proposed method may reduce computational overhead while generating the key while also increasing the robustness of cryptographic keys. During the key generation process, the trusted key canter (TKC) is involved. The experimental results show that the proposed method reduces computational overhead and provides an easy way to implement an access control mechanism in a cloud environment.**

***Keywords*—Access control; cloud storage; confidentiality; data origin authentication; key derivation**

I. Introduction

The exponential growth of cloud services allows data owners and cloud users to store and access large amount of data in cloud data centers. When huge volume of data is stored and accessed, raises number of security concerns. One of the cloud's security concerns is providing reliable access control over cloud data. To improve data access control, a set of access control mechanisms has been implemented. Identity-based access control is a popular method for data owners to store data in the cloud. In this mechanism, the cloud user requests access to data from the cloud that has been uploaded by the specific data owner. The data owner receives the user's request and generates the public key using the cloud user's data credentials. The user's public key is used to encrypt the data, which is then shared with a cloud user. The trusted key center (TKC) is involved in the generation of the cloud user's private key using the TKC's secret value and the user's public key. The first major concern with identity-based encryption is the higher computational cost of obtaining the cloud user's public and private keys. The second major concern with this mechanism is that data origin authentication may fail. In this context, the proposed scheme is intended to reduce the limitations of identity-based encryption while also addressing key robustness effectively. The proposed scheme is purely based on Identity-Based Encryption (IBE). In IBE approach, the data owner encrypts the data using the public key which is derived from cloud user and the decryption at the cloud user is carried out using private key. The TKC is required whenever the private key is derived. The IBE which simplifies the management of public key certificates at the public key infrastructure (PKI) and this can be the alternative approach to public key encryption.

However, one main important drawback of IBE is that the computational overhead at key pair generation. The proposed work definitely overcome the limitations of Identity-Based Encryption. Instead of generating public-private key pair, pair of secret keys are generated for encryption/decryption as well as data origin authentication. The Trusted Key Center (TKC) is involved for secure communication between the data owner and the cloud user. The proposed approach can increase the comparability of data sharing, cloud user authentication and increases key robustness.

II. Literature Survey

Khalid Albulayhi et al. [1] conducted an analytical review on fine-grained access control mechanisms in cloud computing. They have analyzed existing access control methods such as traditional access control mechanisms (ACMs), encryption based ACMs and the modern ACMs and their limitations in the field of cloud computing to improve data access control in the cloud environment. Ajay Kumar Dubey et al. [2] proposed a attribute-based credit computation method to access the data in a withe help of access control mechanism form the cloud. Cloud services are essential and demanded in today's information technology for sharing and accessing data over the internet. The primary responsibility of the cloud service provider is to ensure the scalability, dependability, and security of cloud data. This paper emphasizes on the working of different types of access control mechanisms to control the aceesing of data. An algorithmic namely crowd review and attribute-based credit computation system has been proposed and the cloudsim tool has been used for simulating the results to demonstrate the credit value for the cloud environment.

Jialu Hao et al. [3] proposed a fine-grained attribute hiding policy for cloud-based internet of things (IoT). The attribute information is completely hidden using a randomizable technique and a fuzzy-based Bloom filter mechanism. A system has been proposed that achieves effective policy privacy preservation with the minimal storage and computation overhead. According to the experimental results, valuable attribute information is not disclosed to unauthorized parties. Shengmin Xu et al. [4] proposed fine-grained bilateral access control for secure cloud-fog computing. A new concept has been proposed that is cloud-fog computing which allow them to provide a variety of on-demand services over the network. The identification and retrieval of useful data from a large volume of cipher text data without the use of expensive decryption mechanisms remains a difficult problem. A system has been created that provides confidentiality as well as data origin identification by utilizing a new cryptographic technique known as matchmaking attribute-based encryption (MABE).

Sana Belguith et al. [5] proposed a privacy-preserving attribute-based framework for cloud access control. To distribute keys among cloud users, existing cryptographic techniques for access control impose a high computational cost on the data owner's site. A system has been created that combines attribute-based encryption and attribute-based signature mechanisms to allow the secure sharing of outsourced data via the public cloud. Mahender Kumar and Satish Chand [6] proposed a secure key issuing identity-based encryption in a cloud environment. Because of the high cost of storage, traditional public key cryptosystems are impractical in real-time environments. To overcome the limitations of traditional public key cryptography, the author has proposed a mechanism for generating the user's private key that eliminates the need for a key generation center (KGC) and to secure communication over a public channel, the ECC-based blind technique is used.

Shengmin Xu et al. [7] proposed fine-grained access control mechnism for dynamic groups in a cloud environment. Cloud computing is a new trend in which users can store data and access the data based on scalable on-demand services. At the same time, cloud computing introduces numerous security issues because cloud service providers (CSPs) doesn't share the same trusted domain for all the users. Yi Liu et al. [8] proposed a secure and fine-grained access control mechanism for storing electronic health care records in mobile cloud computing. To access the e-health care records in a controlled manner, the bilinear Diffie-hellman exponent assumption is used. The simulation results have been generated and the model proved to be suitable for mobile cloud computing.

Fangbo Cal et al. [9] conducted a survey on various access control techniques to protect information and system resources and to limit unauthorized users accessing information in the cloud environment. The paper provides the summary of the benefits and drawbacks of various access control mechanisms and suggested some future research directions. Shivanna K et al. [10] proposed a double encryption method for cloud computing privacy preservation. The author proposed a double encryption mechanism to improve the privacy of data stored on cloud, one for storing data and the other for accessing data from the cloud server.

Rashad Elhabob et al. [11] proposed identity-based encryption with an authorized equivalence test for cloud-aided IoT. A system has been proposed to collect the data using different sensors, encrypt the data and store the data on the cloud and the identity-based encryption with authorized equivalence test (IBE-AET) has been proposed to test the equivalence of two messages encrypted with the same or different identities and simulated the results and theoretical analysis has been done. Kwangsu Lee [12] proposed a revocable identity-based encryption method based on the subset difference method rather than the complete subtree (CS) method. A system has been proposed by combining identity-based and hierarchical identity-based encryption.

Hongbing Cheng et al. [13] proposed a method for accountable privacy preservation based on identity-based encryption. The system architecture for accountable privacy preservation, as well as a detailed security analysis, is presented. The system is designed to defend against various types of attacks, and the simulation results in increased efficiency. Chandrashekhar Meshram et al. [14] proposed an identity-based encryption technique based on fuzzy technique for data sharing in cloud environment. To reduce key disclosure, a technique has been implemented that protects against a chosen ciphertext attack and a chosen sub-tree attack and it has the proficiency to overcome the limitations of currently available methods in terms of security and public key length.

Hua Deng et al. [15] [16] [17] proposed Identity-Based Encryption for cloud data security with flexible data sharing among the users. They proposed two different kind of algorithms such as Identity-based Encryption (IBE) and Identity-Based broadcast encryption (IBBE). The main aim of this work is to tackle the critical issue of revocation of identity and propose a revocable IBE scheme in the server-aided setting. Nishat Farjana et al. [18] [19] [20] [21] secure identity-based data sharing scheme for social networks in cloud computing. The main aim of this scheme is to provide secure data sharing over authorized users. This approach is able to resist the keywords guessing attack as well as reduced computational cost.

Sharma et al. [22] [23] [24] proposed a blochchain based architecture using identity-based encryption. In today's world, blockchain and Internet of Things (IoT) are two dominant areas of information technology. These technologies are widely used in supply chain, logistic and automotive industry. They proposed a blockchain technology in order to improve the data sharing as well as security. The reliable sharing of health records can enhance the treatment process with a help of diagnosis accuracy, security and privacy. The architecture what they developed is purely based on Identity-Based Encryption (IBE) algorithm.

III. Design Consideration

The proposed scheme is envisioned to ensure flexible access control mechanism with reduced computational cost and storage overheads. The scheme has the following notations and their meanings as illustrated in Table I.

A. Design Goals

The proposed method can be summarized as the following design goals:

1) The design of the system is to construct two distinct secret keys from the single master key in which $SecKey^1$ is used to achieve data confidentiality and $SecKey^2$ is used to achieve data origin authentication.

TABLE I. NOTATIONS AND MEANINGS

Notations	Descriptions
$MasKey^1$	Master $key1$
$SecKey^1$	Secret $key1$
$SecKey^2$	Secret $key2$
$AuthTag^1$	Authentication $tag1$
$AuthTag^2$	Authentication $tag2$
TKC	Trusted Key Center
S_{tkc}	Secret value of TKC
X	Original data
Y	Encrypted data
E	Encryption
D	Decryption
HMAC	Hash message authentication code
SHA_1	Secure hash $algorithm1$
CSP	Cloud Service Provider

2) The TKC (Trusted Key Center) is used for semi key generation such that TKC has given an authority to generate the $SecKey^2$ for checking data origin authentication.
3) The system is designed to construct the secret keys using randomly selected characters from the master key with the random bits that increases the key robustness in such a way that adversary will not have a chance to capture the copy of the master key or even secret keys.
4) The $SecKey^1$ will be sent to the cloud user from the data owner in a secure channel and the $SecKey^2$ is generated by the TKC and the secrete key 2 will be sent to the cloud user if required by the cloud user to check the authenticity of the data owner.

IV. PROPOSED METHODOLOGY

From the point of view of the data owner, the following points are considered:

1) The data owner must register with the CSP before storing data in cloud storage.
2) The data owner derives the $MasKey^1$ using standard Blowfish algorithm.
3) Extract part1 and part2 of the $MasKey^1$ and send part2 to the trusted key center (TKC). The part1 of the $MasKey^1$ is retained its own.
4) Data owner computes $SecKey^1$ using SHA1 hashing technique and receives $SecKey^2$ from the TKC and encrypt the data using $SecKey^1$ and perform the data origin authentication using $SecKey^2$.

From the point of view of the TKC, the following points are considered:

1) If TKC accepts the credentials of the cloud user then send those credentials to the data owner.
2) The TKC receives the part2 of $MasKey^1$ from the data owner, derives the $SecKey^2$.
3) The part2 of the $MasKey^1$ is concatenated with a secret value Stkc known only by the TKC.
4) The $SecKey^2$ is derived using SHA_1 hashing function. The SHA_1 algorithm takes input as a concatenated result and produces the 160 bit hash code.
5) The TKC sends secret key2 to the cloud user through a secure channel if required by the cloud user to the authenticity of the data owner.

```
Algorithm 1   Deriving Master $key^1$ ($MasKey^1$) at the data owner site
Input:        Key generator algorithm (Blowfish)
Output:       Master $key^1$
1:            generate key using Blowfish algorithm
2:            initialize key with 128 bits
3:            initialize master $key^1$ ($MasKey^1$) $\leftarrow$ key
4:            convert master $key^1$ into a string
5:            record master $key^1$ for further processing
```

From the point of view of the cloud user, the following points are considered:

1) Whenever the cloud user requested for accessing the data from the CSP, cloud user identifies himself to the TKC by presenting appropriate credentials and request a $SecKey^1$ that corresponds to the data owner.
2) The cloud user receives the $SecKey^1$ from the data owner and decrypts the data.
3) The cloud user also receives the $SecKey^2$ from the TKC if required and verifies the data origin authentication.

In the following section, a set of algorithms have been proposed for accomplishing the privacy preservation using entity-based key derivation. The parameters used for configuring each of the algorithm are key generation, key derivation and Trusted Key Center (TKC) for information exchange, encryption/decryption and data origin authentication through signature verification. However, each cloud user receives two different secret keys in which one key is used for encryption/decryption and another key is used for data origin authentication. The proposed system is designed in such a way that two derived keys belongs to a specific user and those keys can not be breakable by the unauthorized users.

At the data owner's site, **Algorithm 1** is designed to generate master key^1 ($MasKey^1$). The standard Blowfish algorithm is used to generate the master key^1 of 128 bits. This master key is converted into a string of characters that is needed for subsequent processing.

The proposed work's main goal is to extract characters randomly from the $MasKey^1$. **Algorithm 2** is intended to split $MasKey^1$ into two parts, such as $part^1$ and $part^2$. $MasKey^1$ is initially made up of 128 bits, which are then converted into 16 characters, each of which stores 8 bits of data. In the first pass, extract 8 characters randomly from the master key and store it in $part^1$. Another set of 8 characters are extracted randomly and store it in $part^2$ in the second pass. The data owner keep $part1$ for generating the secret key^1 ($SecKey^1$) and send $part2$ to the TKC to generate secret $key2$ ($SecKey^2$).

Algorithm 3 TKC is used in the proposed work for semi-key generation that is TKC receives the data credentials from the cloud user such as user e-mail and contact information, and then send the credentials to the data owner. The data owner generates the $SecKey^1$ using the standard SHA_1 algorithm which concatenates the $part^1$ of the $MasKey^1$ and the user e-mail id and the output of the SHA_1 function is 160-bit hash code.

Algorithm 4 is used to derive secret key^2 ($SecKey^2$) at TKC (Trusted Key Center). The TKC is only involved in

```
Algorithm 2   Extract $part^1$, $part^2$ from $MasKey^1$ at data owner site
Input:   $MasKey^1$
Output:  $part^1$, $part^2$
1:  receives $MasKey^1$ Algorithm 1
2:  if $MasKey^1$ is string then
        compute length of the $MasKey^1$
3:      randomly extract half amount of the characters from
        $MasKey^1$ store extracted characters into $part^1$
4:      randomly extract half amount of characters from
        $MasKey^1$ store extracted characters into $part^2$
5:      $part^1$ is kept in its own and $part^2$ is sent
        to the TKC for further processing
```

```
Algorithm 3   Derive secret $key^1$ ($SecKey^1$) at the data owner site
Input:   $part^1$, email-ID of the cloud user
Output:  $SecKey^1$
1:  if TKC accepts the cloud user credentials then
2:      TKC sends user credentials to the data owner
3:      data owner computes $string^1 \leftarrow (part^1 \text{ —— } email_ID)$
4:      compute $SecKey^1 \leftarrow SHA_1(string^1)$
5:      initialize $SecKey^1 \leftarrow$ 160 bits
```

generating $SecKey^2$ using $part^2$ of $MasKey^1$ and secret value S_{tkc}. To generate $SecKey^2$, the TKC receives the $part^2$ of the master key from data owner and concatenates part2 with S_{tkc} value is know only to TKC. The $SecKey^2$ is computed using SHA_1 algorithm, which takes input as concatenated result and produces 160 bit hash code. The hash code of 160 bits is used as a $SecKey^2$ which is used for signature verification.

Algorithm 5 is designed for encrypting data at the data owner's site. The data owner encrypts the original data using $SecKey^1$ and generates encrypted data, which is then stored in the cloud.

To perform a reliable communication in cloud computing, the data owner who stored data in the cloud must be authenticated. **Algorithm 6** describes the construction of the authentication tag at the data owner's site.

Algorithm 7 is designed for decrypting encrypted data stored in the cloud environment at the cloud user site. To decrypt the data, the cloud user must obtain $SecKey^1$ from the data owner. The cloud user decrypts the data with the $SecKey^1$ obtained from the data owner.

Algorithm 8 describes signature verification via authentication tag at the cloud user site. Using $SecKey^2$, the cloud user can validate the data owner's authentication. In this case, the cloud user uses the HMAC algorithm to compute authentication tag^2 ($AuthTag^2$) of the original data. The HMAC algorithm takes original data and the $SecKey^2$ as input and produces the $AuthTag^2$. The data owner receives $AuthTag^1$ from CSP and compares it to $AuthTag^2$. If both authentication tags are successfully compared, the cloud user may confirm that data was sent by an authenticated party; otherwise, do not.

```
Algorithm 4   Derive secret $key^2$ ($SecKey^2$) at TKC
Input:   $part^2$, secret value of TKC ($S_{tkc}$)
Output:  $SecKey^2$
1:  receives $part^2$ of the $MasKey^1$ from the data owner
2:  compute $string^2 \leftarrow (part^2 \text{ —— } S_{tkc})$
3:  compute $SecKey^2 \leftarrow SHA_1(string^2)$
4:  initialize $SecKey^2 \leftarrow$ 160 bits
5:  establish $SecKey^2$ to data owner for
    signature computation
    establish $SecKey^2$ to the cloud user if required
```

```
Algorithm 5   Encryption (E) at the data owner site
Input:   Original data (X), $SecKey^1$
Output:  Encrypted data (Y)
1:  receives $SecKey^1$ from the TKC
2:  initialize $SecKey^1$ for encryption and decryption of data
3:  compute Y $\leftarrow$ E($SecKey^1$ (X))
4:  store encrypted data (Y) on CSP
```

```
Algorithm 6   Generate authentication tag at the data owner site
Input:   Original data (X), $SecKey^2$
Output:  Authentication $tag^1$ ($AuthTag^1$)
1:  receives $SecKey^2$ from the TKC
2:  initialize $SecKey^2$ for data origin authentication
3:  compute $AuthTag^1 \leftarrow$ HMAC(X, $SecKey^2$)
4:  store $AuthTag^1$ on CSP
```

```
Algorithm 7   Decryption at cloud user site
Input:   Encrypted data (Y), $SecKey^1$
Output:  Original data (X)
1:  receives $SecKey^1$ from the data owner via secure channel
2:  compute X $\leftarrow$ D ($SecKey^1$ (Y))
3:  utilize X
```

1) Implementation: The proposed scheme is implemented using a Java-based JSP web application. The scheme is being tested in a physical cloud environment such as Amazon Web Services. Device setup includes Tomcat 8.5 with Corretto 11 running on 64bit Amazon Linux 2/4.1.2.

V. Experimental Results

The proposed scheme is based on the identity-based encryption. The $MasKey^1$ is generated using standard blowfish algorithm and its size is 128 bits which is then converted into 16 characters. In the first pass, 8 characters are extracted randomly from the master key and store it in $part^1$ and in the second pass another set of 8 characters are extracted randomly and store it in $part^2$. The data owner keeps $part1$ for generating the secret key^1 ($SecKey^1$) and send $part2$ to the TKC to generate secret $key2$ ($SecKey^2$). The $SecKey^1$ is generated by concatenating $part1$ of the $MasKey^1$ and the unique attribute of the cloud user which was collected by the TKC and the concatenated result will be given as input to the SHA_1 algorithm and it produces 160 bit hash code. The hash code of 160 bits is used for encrypting data at data owner side. TKC generates the $SecKey^2$ using $part^2$ of $MasKey^1$ and secret value S_{tkc}. To generate $SecKey^2$, the TKC receives the $part^2$ of the master key from data owner and concatenates part2 with S_{tkc} value is known only to TKC. The $SecKey^2$ is computed using SHA_1 algorithm, which takes input as

```
Algorithm 8   Validate authentication tag at the cloud user site
Input:   Original data (X), $SecKey^2$
Output:  True or False
1:  receives $SecKey^2$ from the TKC if required
2:  compute Authentication $tag^2$ ($AuthTag^2$) such that
    $AuthTag^2 \leftarrow$ HMAC(X, $SecKey^2$)
3:  receives $AuthTag^1$ from CSP
4:  if $AuthTag^2$ == $AuthTag^1$ then
        return true ie authentication successful
5:  else
        return false ie authentication unsuccessful
    end if
```

concatenated result and produces 160 bit hash code. The hash code of 160 bits is used as a $SecKey^2$ which is used for signature verification. The computation cost for generating two secrete keys is reduced when compared to having two separate master keys is shown in Table II. The memory requirement also reduces when compared to the $MasKey^2$ is shown in the Table III.

The Fig. 1 depicts the time required to derive the $MasKey^2$ and the proposed method of key derivation. The graphical representation demonstrates that the proposed method has a low processing overhead when calculating the key.

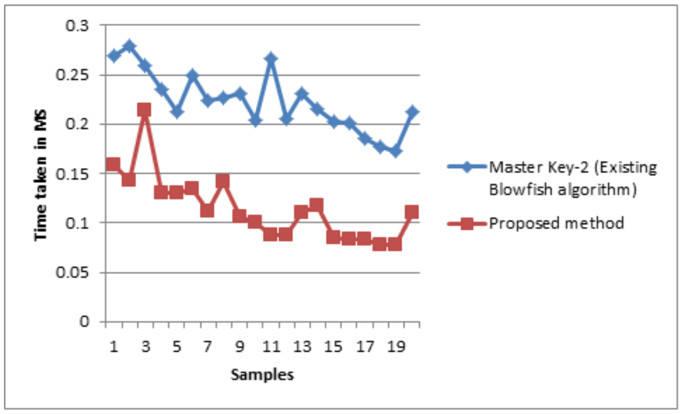

Fig. 1. Time Taken in ms (Master key^2 vs. Proposed Method)

The Fig. 2 depicts the time required to derive the $MasKey^1$ and the proposed method of key derivation. In this paper, the proposed scheme is compared to the existing solution's $MasKey^1$. The Fig. 2 shows that the computational cost of deriving the keys used in the proposed method is significantly lower than the existing key derivation technique.

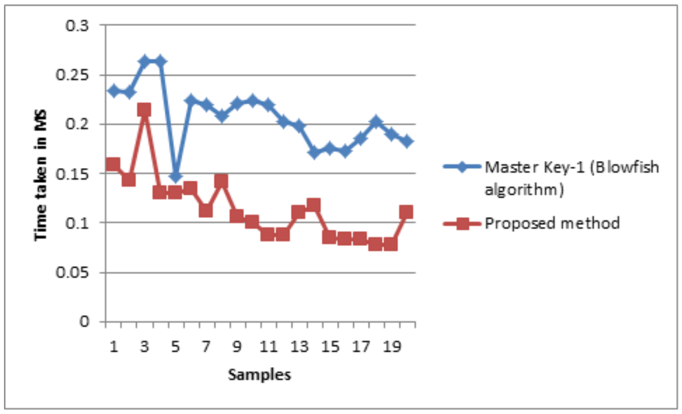

Fig. 2. Time Taken in ms (Master key^1 vs. Proposed Method).

Memory utilization is also a major concern during key generation in order to increase the efficiency of the cloud environment. The Table III compares the proposed scheme's memory utilization in megabytes to existing cryptographic keys such as $MasKey^1$ and $MasKey^2$. The Fig. 3 and 4 show a graphical representation of the memory utilization of the proposed method of key derivation and the $MasKey^1$. The proposed method's memory utilization of key derivation and $MasKey^1$ is nearly identical to the experimental data.

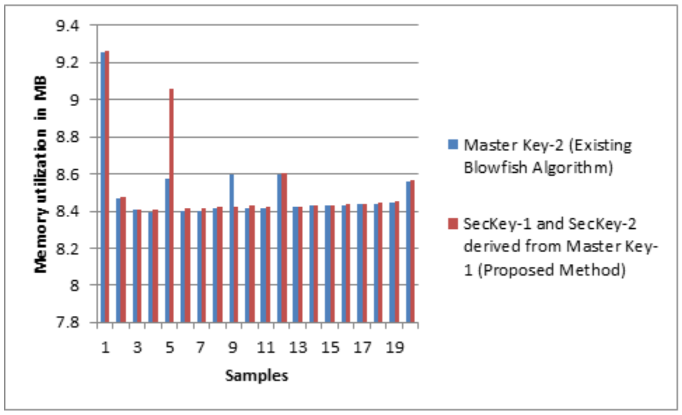

Fig. 3. Memory Utilization in MB (Master key^2 vs. Proposed Method).

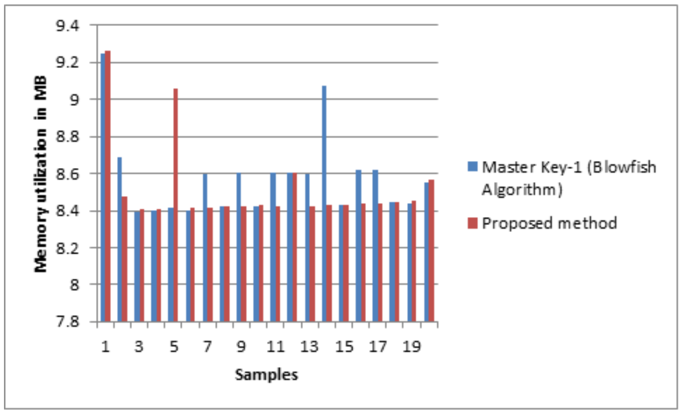

Fig. 4. Memory Utilization in MB (Master key^1 vs. Proposed Method).

VI. Experimental Analysis

The experimental analysis is made in this paper to tackle the following issues:

A. Key Derivation

In this paper, a system is proposed for deriving secret keys from a single master key, with the goal of limiting the number of master key generations. Jialu Hao et al. [3]proposed a cloud-based IoT device access control policy in which the author considered the system public key, master secret key, and an attribute set for generating the secret key. As the system public key and master key are considered for key generation, the computational overhead can easily increase. In this paper, we designed an algorithm that generates secret keys from a single master key in order to greatly reduce computational overhead.

B. Enhancement of Data Security

In the proposed system, we first computed a 128-bit master key using the standard Blowfish algorithm. The secret keys

TABLE II. TIME TAKEN (MS) FOR KEY DERIVATION (EXISTING VS. PROPOSED METHODOLOGY)

Sample	$MasKey^1$ (Blowfish Algorithm)	$MasKey^2$ (Existing: Blowfish Algorithm)	$SecKey^1$, $SecKey^2$ derived from $MasKey^1$ (Proposed methodology)
1	0.234	0.269	0.159
2	0.232	0.279	0.143
3	0.263	0.259	0.214
4	0.263	0.235	0.130
5	0.147	0.212	0.130
6	0.224	0.250	0.134
7	0.219	0.224	0.112
8	0.208	0.227	0.142
9	0.221	0.231	0.106
10	0.224	0.204	0.100
11	0.220	0.266	0.088
12	0.203	0.206	0.087
13	0.198	0.231	0.110
14	0.171	0.215	0.118
15	0.176	0.203	0.085
16	0.173	0.201	0.083
17	0.185	0.185	0.083
18	0.202	0.177	0.078
19	0.190	0.173	0.078
20	0.183	0.212	0.111
Avg.	**0.206**	**0.222**	**0.114**

TABLE III. MEMORY UTILIZATION IN MEGABYTES (EXISTING VS. PROPOSED METHODOLOGY)

Sample	$MasKey^1$ (Blowfish Algorithm)	$MasKey^2$ (Existing: Blowfish Algorithm)	$SecKey^1$, $SecKey^2$ derived from $MasKey^1$ (Proposed methodology)
1	9.247	9.255	9.259
2	8.685	8.466	8.475
3	8.395	8.404	8.405
4	8.399	8.396	8.407
5	8.412	8.575	9.057
6	8.402	8.398	8.413
7	8.594	8.403	8.415
8	8.425	8.414	8.425
9	8.602	8.597	8.424
10	8.420	8.416	8.431
11	8.604	8.414	8.423
12	8.602	8.599	8.601
13	8.600	8.424	8.422
14	9.072	8.432	8.432
15	8.434	8.431	8.430
16	8.620	8.433	8.441
17	8.623	8.436	8.440
18	8.445	8.440	8.443
19	8.441	8.447	8.450
20	8.553	8.558	8.568
Avg.	**8.578**	**8.496**	**8.518**

are derived from the master key by using randomly selected characters, the cloud user's data credential, and the trusted key center's secret value. During key generation, the SHA_1 hashing technique is used to generate a 160 bit hash code, which is then used as a secret key^1 and secret key^2. In this context, the secret key size is increased from 128 bits to 160 bits, which improves data security because increasing the key size automatically improves data security.

C. Semi Key Generation

The trusted key center (TKC) is involved in this paper for semi key generation in collaboration with the data owner. The TKC is only in charge of creating the secret key^2, which is only used for data origin authentication via signature verification. The TKC does not have complete control over the generation of secret key^1.

D. Key Robustness

The main goal of this paper is to improve the robustness of the derived keys. The system is designed in such a way that secret keys are generated using randomly selected characters and the cloud user's credential. In this context, the randomly selected characters and cloud user's credential should be required whenever the adversary attempts to crack the secret key. Even if the adversary captures the e master key, extracting the actual random characters for the next time is extremely difficult. Table IV shows a functional comparison of the proposed method with related research work.

VII. CONCLUSION

The proposed method has been put into practice in order to improve computational efficiency and data origin authentication. In order to increase the computational efficiency, secret keys are derived from the single master key so that processing overhead of key derivation is comparatively less compared with another master key. The trusted key center (TKC) is involved in the proposed method for secure communication between data owner and cloud user. The TKC play a major role for key generation and key establishment. The TKC validates the data credentials of the cloud user and the unique identity of the

TABLE IV. FUNCTIONAL COMPARISON

Parameters	Jialu Hao et al. [3]	Sana Belguith et al. [5]	Mahender Kumar et al. [6]	Rashad Elhabob et al. [11]	Proposed Method
Flexible Access Control	Yes	Yes	Yes	Yes	Yes
Privacy policy	Yes	Yes	Yes	Yes	Yes
Computation cost for key derivation	High	High	High	High	Low
Non-Repudiation	No	No	No	No	Yes
Storage overhead for key derivation	High	High	High	High	Low
Key robustness	Medium	Medium	Medium	Medium	High

cloud user is sent to the data owner to derive the secret key1 in order to provide the confidentiality. This leads to identity based key derivation that reduces the unauthorized access. It means that secret key1 generated for a specified user will not be used for another cloud user. At the same time the secret key2 has been generated by the TKC for data origin authentication. Since, the TKC is a trusted third party so that entire control for secret key2 generation is handed over to the TKC can able to make a mutual authentication between data owner and cloud user. The experimental results are compared to existing solutions, and the proposed scheme is capable of being deployed in a real-world cloud environment.

Compliance with Ethical Standards

The proposed methodology is designed under the guidance of eminent personalities and it is found that the proposed methodology is not available in the prior art search.

REFERENCES

[1] K. Albulayhi, A. Abuhussein, F. Alsubaei, and F. T. Sheldon, "Fine-grained access control in the era of cloud computing: An analytical review," in *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE, 2020, pp. 0748–0755.

[2] A. K. Dubey and V. Mishra, "Crowd review and attribute-based credit computation for an access control mechanism in cloud data centers," *International Journal of Computers and Applications*, pp. 1–8, 2020.

[3] J. Hao, C. Huang, J. Ni, H. Rong, M. Xian, and X. S. Shen, "Fine-grained data access control with attribute-hiding policy for cloud-based iot," *Computer Networks*, vol. 153, pp. 1–10, 2019.

[4] S. Xu, J. Ning, Y. Li, Y. Zhang, G. Xu, X. Huang, and R. Deng, "Match in my way: Fine-grained bilateral access control for secure cloud-fog computing," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[5] S. Belguith, N. Kaaniche, A. Jemai, M. Laurent, and R. Attia, "Pabac: a privacy preserving attribute based framework for fine grained access control in clouds," in *SECRYPT 2016: 13th International Conference on Security and Cryptography*, vol. 4. SciTePress, 2016, pp. 133–146.

[6] M. Kumar and S. Chand, "Eski-ibe: Efficient and secure key issuing identity-based encryption with cloud privacy centers," *Multimedia Tools and Applications*, vol. 78, no. 14, pp. 19 753–19 786, 2019.

[7] S. Xu, G. Yang, Y. Mu, and R. H. Deng, "Secure fine-grained access control and data sharing for dynamic groups in the cloud," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 8, pp. 2101–2113, 2018.

[8] Y. Liu, Y. Zhang, J. Ling, and Z. Liu, "Secure and fine-grained access control on e-healthcare records in mobile cloud computing," *Future Generation Computer Systems*, vol. 78, pp. 1020–1026, 2018.

[9] F. Cai, N. Zhu, J. He, P. Mu, W. Li, and Y. Yu, "Survey of access control models and technologies for cloud computing," *Cluster Computing*, vol. 22, no. 3, pp. 6111–6122, 2019.

[10] K. Shivanna, S. P. Deva, and M. Santoshkumar, "Privacy preservation in cloud computing with double encryption method," in *Computer Communication, Networking and Internet Security*. Springer, 2017, pp. 125–133.

[11] R. Elhabob, Y. Zhao, N. Eltayieb, A. M. Abdelgader, and H. Xiong, "Identity-based encryption with authorized equivalence test for cloud-assisted iot," *Cluster Computing*, vol. 23, no. 2, pp. 1085–1101, 2020.

[12] K. Lee, "A generic construction for revocable identity-based encryption with subset difference methods," *PloS one*, vol. 15, no. 9, p. e0239053, 2020.

[13] H. Cheng, C. Rong, M. Qian, and W. Wang, "Accountable privacy-preserving mechanism for cloud computing based on identity-based encryption," *IEEE Access*, vol. 6, pp. 37 869–37 882, 2018.

[14] C. Meshram, C.-C. Lee, S. G. Meshram, and M. K. Khan, "An identity-based encryption technique using subtree for fuzzy user data sharing under cloud computing environment," *Soft Computing*, vol. 23, no. 24, pp. 13 127–13 138, 2019.

[15] H. Deng, Z. Qin, Q. Wu, Z. Guan, R. H. Deng, Y. Wang, and Y. Zhou, "Identity-based encryption transformation for flexible sharing of encrypted data in public cloud," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3168–3180, 2020.

[16] H. Li, Y. Dai, and B. Yang, "Identity-based cryptography for cloud security," *Cryptology ePrint Archive*, 2011.

[17] J. Li, J. Li, X. Chen, C. Jia, and W. Lou, "Identity-based encryption with outsourced revocation in cloud computing," *Ieee Transactions on computers*, vol. 64, no. 2, pp. 425–437, 2013.

[18] N. Farjana, S. Roy, M. Mahi, J. Nayeen, and M. Whaiduzzaman, "An identity-based encryption scheme for data security in fog computing," in *Proceedings of International Joint Conference on computational intelligence*. Springer, 2020, pp. 215–226.

[19] Q. Huang, W. Yue, Y. He, and Y. Yang, "Secure identity-based data sharing and profile matching for mobile healthcare social networks in cloud computing," *IEEE Access*, vol. 6, pp. 36 584–36 594, 2018.

[20] 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, 2019.

[21] P. Mishra and V. Verma, "Study of identity-based encryption for cloud data security," in *Decision analytics applications in industry*. Springer, 2020, pp. 401–408.

[22] P. Sharma, N. R. Moparthi, S. Namasudra, V. Shanmuganathan, and C.-H. Hsu, "Blockchain-based iot architecture to secure healthcare system using identity-based encryption," *Expert Systems*, p. e12915, 2021.

[23] D. Unal, A. Al-Ali, F. O. Catak, and M. Hammoudeh, "A secure and efficient internet of things cloud encryption scheme with forensics investigation compatibility based on identity-based encryption," *Future Generation Computer Systems*, vol. 125, pp. 433–445, 2021.

[24] N. Vaanchig, Z. Qin, and B. Ragchaasuren, "Constructing secure-channel free identity-based encryption with equality test for vehicle-data sharing in cloud computing," *Transactions on Emerging Telecommunications Technologies*, p. e3896, 2020.