IJCSN - International Journal of Computer Science and Network, Volume 6, Issue 3, June 2017
ISSN (Online) : 2277-5420
www.IJCSN.org
Impact Factor: 1.5

Leakage - Resilient KASE for Group Data Sharing with Auxiliary Input via Cloud Storage

[1] Payal Bhagat; [2] Amar Buchade

[1] Master of Engineering, Dept. of Computer, Pune Institute of Computer Technology
Pune, Maharashtra 411043, India

[2] Assistant Professor, Dept. of Computer, Pune Institute of Computer Technology
Pune, Maharashtra 411043, India

Abstract - Nowadays cloud computing is very essential and important for secure storage of data due to its scalability. If the data owner wants to share his/her data with some authorized user and that user wants to access more than one document in such scenario, in existing system data owner sends separate key for each document to the user. To overcome drawback of this approach we designed a Key Aggregate Searchable Encryption (KASE) scheme in which data owner generates a single key for multiple number of documents and share the key with the user. Then the user uses that aggregate key to perform keyword searching over the group of any number of files through single trapdoor using aggregate key and decrypts the required documents. But still a number of side channel attacks are possible on that aggregate key to leak the master secret key. If the key gets leaked then the attacker will have access to all the documents. The system can be secured by reducing bit leakage so that an attacker cannot recover master secret key from aggregate key. In this paper, we propose a Leakage-Resilient KASE scheme in which numbers of leaked bits by the attacker are reduced. Evaluation results for the proposed system prove that our scheme is auxiliary input CPA secure based on the KASE scheme by reducing leakage by a ratio of 80% approximately.

Keywords - *Auxiliary Input, CPA Security Model, Key Aggregate, Leakage Resilient, Searchable Encryption.*

1. Introduction

Cloud storage has emerged as an exciting solution for providing ever-present, suitable, and on-demand accesses to huge amounts of data shared over the network. Sharing of data is an important point at issue in cloud storage, and method to share the encrypted data efficiently is a challenging issue in the cloud storage. There are two straightforward way to share the encrypted data in the cloud. The first method is that the owner of data can download the encrypted data, then decrypt them and send to authorized users. Though due to this way it loses the value of cloud, and it is also very inefficient. The second method is that the data owner can privately send the keys of the corresponding encrypted data to other users for sharing. But this way is still also very tricky due to a large number of encrypted data. Thus, the data owner may need to send many keys one time for sharing of data, which is not a very secure way. So, how can a data owner efficiently and securely send the decryption rights to other users is a challenging problem in cloud storage. However, the encryption of data makes it problematic for the users to search and then retrieve only the data matching given keywords. A standard solution is to implement a searchable encryption (SE) method in which the data owner is required to encrypt keywords and upload them to the cloud with encrypted data, so, for retrieving data matching a keyword.

Cui, Liu et al. [1] proposed a Key Aggregate Searchable Encryption (KASE) to solve the above problem. This technique keeps up the flexible decryption in such a way that any subset of cipher-texts can be decrypted by an aggregate secret key. The aggregate key is produced by the data owner's master secret key. In KASE, the cipher-texts are categorized into different classes, and the data owner can extract aggregate secret keys from different classes by using his master secret key. The aggregate key is compressed as one key for one cipher-text class, but it has the power of many such keys. So, now the data owner can only send one aggregate key to other users to share many documents in the cloud. However, many nodes/devices of cloud are mobile devices, and the aggregate key stored in it is easy to be leaked. Thus, how

to construct a leakage resilient KASE system is still a challenge problem.

In the practical world, side channel attacks grant the attackers to leak partial knowledge about secret key by noticing some physical properties of a cryptographic execution. Hence, the idea of leakage resilient cryptography has been proposed. In this paper, we propose a leakage resilient KASE for group data sharing with auxiliary input based on Cui, Liu and Wang's basic construction [1]. In our work, the system is secured by reducing bit leakage so that an attacker cannot recover master secret.

2. Literature Review

2.1 Key Aggregate Group Data Sharing via Cloud

Data sharing systems have received much more attention lately [3], [4], [5], [6] which is based on cloud storage. To be specific, Chu et al [6] considered many ways to reduce the data encryption key distribution numbers. The owner of data needs to distribute all keys which are used for encryption, to share several documents with the same user in a traditional approach which is almost infeasible. Taking this challenge with the aim to generate an aggregate key for the user to decrypt all the documents shared with the user, Chu et al [6] proposed a data sharing system using key-aggregate cryptosystem (KAC). This is a new cryptographic system, which allowed to give flexible decryption rights such that aggregate secret key can be used to decrypt any cipher-texts subset. The constant-size aggregate key is generated by the master secret key of the data owner. The power of many keys is aggregated as one key is compacted for one cipher-text class. Hence, to share many documents, only one aggregate key is needed to be sent by the data owner to the data requester in the cloud.

2.2 Multi-User Searchable Encryption

Multi-User Searchable Encryption (MUSE) technique was focused on some recent work [8], [9], [10], [11], [14], while to reach the objective these all adopted single key integrated with access control. In [7], [14], multi-user searchable encryption system are constructed to achieve coarse-grained access control by using broadcast encryption and the documents were shared by sending searchable encryption key to all users who can access it. Attribute Based Encryption (ABE) is applied in [8], [9], [10], [11], [12], [13] to achieve searching keyword using fine-grained access control. However, now in multi-user searchable encryption, control over which document can be accessed by which user is the main problem. As well as it does not consider how the number of shared keys and trapdoors can be reduced. Key aggregate searchable encryption (KASE) system can make MUSE more efficient and practical and also provide the answer for the keyword search and access control.

2.3 Multi-Key Searchable Encryption

When there is a scenario of multi-user application, we consider that when searching the keyword over documents the number of trapdoor will be same as the number of documents, Popa and Zeldovich el at [15] firstly proposed the multi-key searchable encryption (MKSE) system concept and in 2013 this system was put forward. Multi-key searchable encryption allows a user to search over documents encrypted with different keys by providing a trapdoor of single keyword to the server. This requirement looks very similar to the objective of KASE, but these two techniques are completely different from each other. The objective of KASE system is distributing the aggregate key to give the keyword search right to any user in a group data sharing system; however the aim of MKSE is to ensure that using single trapdoor keyword search can be performed on cloud server over different documents belonging to a user.

2.4 Public-Key Encryption with Keyword Search

The difficulty in public cloud system is searching over encrypted data through encryption key. Firstly introducing search query where the keyword were used for email gateway. Without knowing about the contents of data shared with specific keyword can be searched by the gateway and verifying satisfying document to rank documents accordingly. Public Key Encryption with Keyword Search system recognize all public key encrypted documents containing the same keyword given by data owner without decryption of data. Test over the gateway is executed to match encrypted keywords of sender and word of receiver choice, no more information is learned by the gateway. PEKS system implies Identity Based Encryption (IDE) scheme where owner encrypts data, such that user having required attributes can only decrypt the shared document. This system considered only single owner and user condition for performing keyword searches over multiple shared documents.

2.5 Leakage Security Model

Until now, some classic leakage models have been proposed, and there are primarily three leakage models. The first model is the bounded retrieval model [18], [19], [20], in this model the bit-length of the key must be significantly higher than the total number of bits leaked

over the lifetime of the system, and before the whole secret is leaked we hope that the attack is detected and stopped. The second model is the continual leakage model [21], [22], [16], where the secret key should be refreshed continually where there is no leakage during the update process and it assumes that leakage is bounded in term of a fraction of the secret key size between consecutive updates. The third model which is developed from the relative leakage model [17], is the auxiliary input model [23], [24], [25], [26], which permits any non-invertible function that no probabilistic polynomial time attacker can compute the actual pre-image with probability which is non-negligible. Zhiwei Wang and Lingyu Zhou el at [2] proposed a leakage resilient KAC with auxiliary input based on Chu et al.'s basic construction [6]. Where, the attacker still cannot recover any information of the master secret key, no matter how many bits are leaked from the aggregate key.

2.6 Leakage-Resilient KAC with Auxiliary Input

Chu et al. [6] proposed a key aggregate cryptosystem (KAC) to solve the problem of selectively sharing data by generating a single aggregate key. This is a new cryptographic system, which allowed giving flexible decryption rights such that aggregate secret key can be used to decrypt any cipher-texts subset. The constant-size aggregate key is generated by the master secret key of the data owner. In this system, different classes are created to classify the cipher-texts, and these different classes are used by a data owner to generate aggregate secret keys by using a master secret key. The power of many keys is aggregated as one key is compacted for one cipher-text class. Hence, to share many documents, only one aggregate key is needed to be sent by the data owner to the data requester in the cloud. But in the real world, attackers can learn partial information about the secret key by making many side channel attacks and observing cryptographic execution physical properties. Zhiwei Wang and Lingyu Zhou el at [2] proposed a leakage resilient key aggregate cryptosystem with auxiliary input, which is based on Chu et al.'s construction [6].

3. System Architecture

3.1 Functional Overview

The system architecture will give the explanation how the data owner share number of documents with single aggregate key and also show searching functionality over encrypted data, refer Figure 1. Our system consists of the following functionality

1. Setup (1λ, n) algorithm is run by the cloud provider to set up the scheme.
2. Keygen(pk, msk) algorithm is the run by the data owner to generate a random key pair.
3. Encrypt (pk, i) algorithm is run by the data owner to encrypt the i^{th} document and generate its ciphertext keywords.
4. Extract (msk, S) algorithm is run by the data owner to create an aggregate key for sharing a selected set of documents with other users.
5. Trapdoor (kagg, w) algorithm is run by the user who has the aggregate key with help of which he can search.
6. Adjust (params, i, S, Tr) algorithm is run by a cloud server to adjust the aggregate trapdoor to generate the right trapdoor for each different document.
7. Test (Tr_i, i) algorithm is run by the cloud server to perform a keyword search over an encrypted document.

3.2 Mathematical Representation of Functions

The functions of our system are described as mathematical formulation below –

1. Setup
   Input: $\{\lambda, n\}$
   Output: $\{\beta, PubK, H\}$
   Where,
   $\beta = (P, G, G1, (',',))$, is a bi-linear map
   P is order of G, $\forall 2\,\lambda \leq P \leq 2\,\lambda + 1$,
   $H: \{0, 1\}^* \rightarrow G$,
   $PubK = (g, g_1, g_{n\ldots}\, g_{2n}) \in G^{2n+1}$

2. Key Generation
   Input: P
   Output: Pk , Msk
   Where,
   Pk is public key,
   Msk is master secrete key,
   $Pk = g^{\gamma}$ and $Msk = \gamma$
   γ is random pick number, $\forall \gamma \in Z_p$

3. Encryption
   Input: i, Pk
   Output: Cw
   Where,
   i is index of doc,
   Pk is public key,
   Cw is keyword ciphertext,
   w is keyword.

4. Key Aggregation
   Input: msk, S
   Output: Kagg
   $Kagg = \Pi_{j \in S} g^{\gamma}_{n+1-j}$

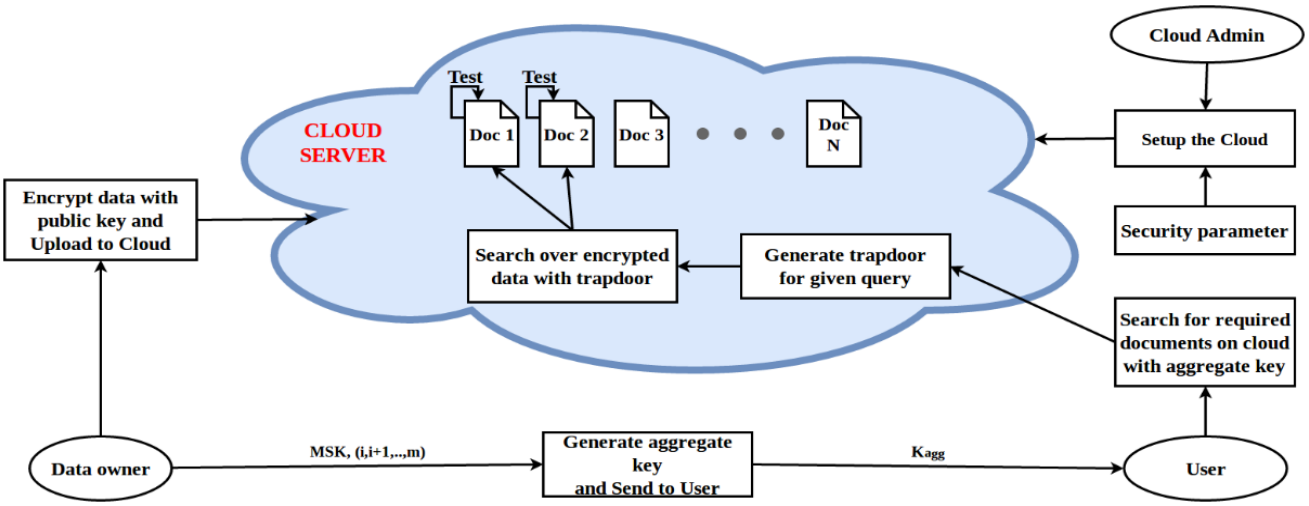

Fig. 1: System Architecture Overview

5. Trapdoor
   Input: Kagg , w
   Output: Tr
   Tr =Kagg . H (w)
   Where,
   w is keyword we want to be search,
   Tr is agg trapdoor for keyword w.

6. Adjust
   Input: params, i, S, Tr
   Output: Tri
   Where,
   Tri is trapdoor for doc index i $\in$ S.

7. Test
   Input: Tri, i, Δi
   Output: True or False

4. Proposed Methodology

4.1 Leakage Resilient System with Auxiliary Input

The Key Aggregate Searchable Encryption (KASE) system is prone to leakage of data. In practical, side channel attacks allow the attackers to learn partial information by observing physical properties of a cryptographic execution. To avoid leakage of information by applying chosen plaintext attack and obtaining bits of encrypted data, we propose a leakage resilient system using the auxiliary input model. Algorithm 1 shows the pseudo code of proposed system.

The Algorithm 1 describes the leakage resilient system. First, the setup of the system is done in which p, q and g are initialized. Second, the KeyGen is executed to generate public key Pk and master secret key MSK. Third, the auxiliary input is determined from the hash value of the public key of user and this is given to encryption function.

Fourth, the auxiliary input *a* is split into two equal lengths, the encryption of message/data M is done using the a1 as key. Then the values of r and s are determined which are used during decryption. Fifth, the auxiliary input k is determined and split into two equal lengths k1, k2 from which the encrypted data c is decrypted using the k1 as key.

Algorithm 1

Setup :
  $p \leftarrow$ a large prime number
  $q \leftarrow$ a large prime factor of p-1
  $g \leftarrow$ a integer with order q modulo p
KeyGen :
  $M_{sk} \leftarrow x$ is selected randomly from (1, 2,..., $q-1$)
  $P_k \leftarrow g^x \ mod \ p$
Auxiliary Input :
  $a \leftarrow H(P_k) \ mod \ p$
Encrypt :
  $a_1, a_2 \leftarrow$ split a into two equal length
  $c \leftarrow Encrypt(M, a_1)$
  $r \leftarrow H(M, a_2)$
  $s \leftarrow (r * P_k) \ mod \ p$
Decrypt :
  $k \leftarrow H(g^s * P_k) \ mod \ p$
  $k_1, k_2 \leftarrow$ split k into two equal length
  $M \leftarrow Decrypt(c, k_1)$

Algorithm 2

Handshake :
  $C \leftarrow F_e(A)$
Setup :
  $params \leftarrow Setup(\lambda)$
  $P_k, M_{sk} \leftarrow KeyGen(p, q)$
  $A \leftarrow P_k(C)$
Leakage :
  $D_1 \leftarrow Encrypt(M, P_{kC})$
  $D_2 \leftarrow Encrypt(M, k_{aggC})$
  $C \leftarrow Send(D_1, D_2)$
  $A \leftarrow Decrypt(D_1, M_{sk})$ and $Decrypt(D_2, M_{sk})$
Test :
  $C \leftarrow Send(M_1, M_2)$
  $E_M \leftarrow Encrypt(random(M_1, M_2), P_{kC})$
  $A \leftarrow Send(E_M)$
Verify :
  $A \leftarrow Verify(E_M, D_1, D_2)$

4.2 Proposed CPA Security Model

To ensure security against auxiliary input and chosen-plaintext attacks (CPA) we defined it as a game between the attacker A and the challenger C. Algorithm 2 shows the pseudo code of the proposed CPA security model.

In Algorithm 2, First, the attacker A sends a set of leakage functions Fe to the challenger C. Second, challenger runs params ← Setup (λ) and (Pk, Msk) ← keygen () to generate public and master secret key, then sends public key Pk to the attacker.

Third, attacker encrypts a data M using public key and aggregate key separately and sends it to challenger. Challenger reply to this by decrypting the data and sending back to the attacker.

Fourth, attacker submits two messages M1 and M2 with the same length to the challenger. From which challenger chooses a random message and returns it to attacker by encrypting by his public key.

Finally, attacker checks for leakage by matching the patterns of message and previous encryption.

5. Experimental Setup

5.1 Dataset

The key aggregate searchable encryption system accepts text files as an input. To evaluate the performance of our system we used text documents of sizes ranging from 50 KB to 500 KB. We used these files for testing of all the system modules.

5.2 Performance Metrics

We evaluated our proposed approach with respect to the following parameters of the system

- Cost of setup
- Cost of encrypt
- Cost of decrypt
- Cost of trapdoor
- Leakage ratio

We done the execution of each module to evaluate the time cost of the existing KASE system and proposed leakage-resilient KASE system, as our aim is to make our system leakage proof. To check whether our system have any effect on time cost after making system leakage proof using auxiliary input, we evaluated our system to compare with KASE scheme.

After the analysis, we found that our proposed system did not have a major effect on time of the system functions

such that, every time the cost of time for the proposed system is approximately near KASE system. The Fig 2, 3, 4 and 5 shows performance comparison with respect to cost of time for the existing system and our proposed system.

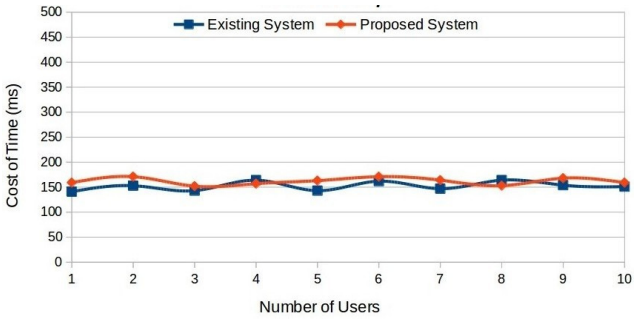

Fig. 2: Cost of Setup

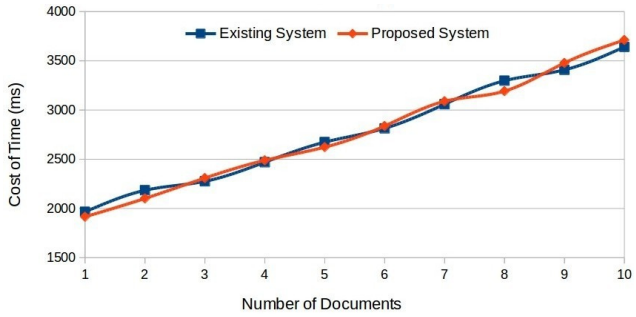

Fig. 3: Cost of Encryption

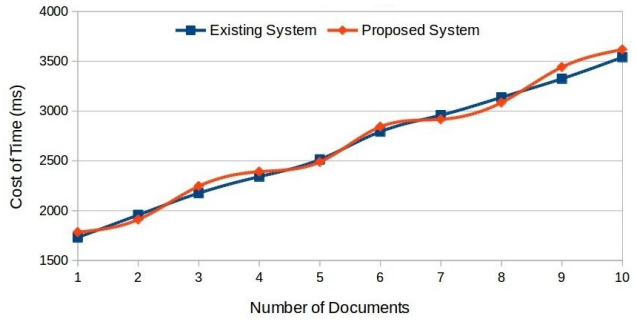

Fig. 4: Cost of Decryption

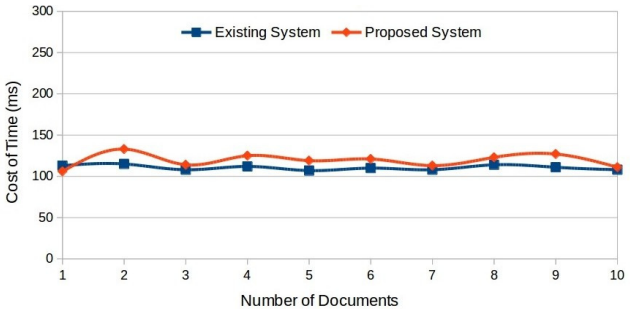

Fig. 5: Cost of Trapdoor

6. Data Tables and Results

6.1 System Leakage Comparison

To evaluate and compare the existing system with our proposed system with respect to leakage, we submitted messages manually to detect leakage. Table 1 shows sample of 5 messages, from which we can conclude that the leakage-resilient system have reduced leakage than existing KASE system.

Table 1: Leakage Testing Results

Msg ID	***Message***	**Existing System Leakage**	**Proposed System Leakage**
1	Cloud	79.73%	73.13%
2	Encrypt	80.58%	73.78%
3	Decrypt	79.37%	74.7%
4	Orange	77.17%	73.5%
5	Lab	94.25%	82.49%

6.2 Leakage Reduction Ratio

We executed our proposed system using multiple numbers of messages ranging from 5 to 50. Figure 6 show that a higher ratio of messages has reduced leakage as compared to message with unreduced leakage. About 80 % of messages were getting reduced leakages, which makes our system leakage-resilient.

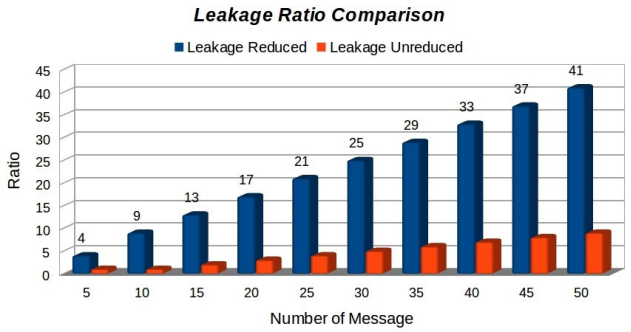

Fig. 6: Leakage Ratio

7. Conclusion

Key Aggregate Searchable Encryption (KASE) is very useful for the group data sharing in cloud storage, but can be prone to key leakage. In our proposed system we concentrate on making the leakage resilient KASE system, since when KASE system is used in cloud applications, it is prone to leakage by side-channel attacks. We propose a leakage resilient KASE system with auxiliary input, which is secured such that the number of bits leaked by the attacker using an aggregate key is reduced. We ensure that the attacker cannot recover any knowledge about master secret key. Evaluation and testing results show that our proposed leakage-resilient KASE don't have any impact on functionality of the system and reduces leakage by a ratio of approximately 80%.

References

[1] Baojiang Cui, Zheli Liu, and Lingyu Wang, "Key-Aggregate Searchable Encryption (KASE) for Group Data Sharing via Cloud Storage", IEEE Transaction on Computer, vol. 65, no 8, Aug 2016.

[2] Zhiwei Wang and Lingyu Zhou, "Leakage-Resilient Key-Aggregate Cryptosystem with Auxiliary Input", 25th International Conference on Computer Communication and Networks (ICCCN), 2016.

[3] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in Proc. IEEE Conf. Comput. Commun., 2010, pp. 534–542.

[4] R. Lu, X. Lin, X. Liang, and X. Shen, "Secure provenance: The essential of bread and butter of data forensics in cloud computing," in Proc. ACM Symp. Inf., Comput. Commun. Security, 2010, pp. 282–292.

[5] X. Liu, Y. Zhang, B. Wang, and J. Yan, "Mona: Secure multi-owner data sharing for dynamic groups in the cloud," IEEE Trans. Parallel Distrib. Syst., vol. 24, no. 6, pp. 1182–1191, Jun. 2013.

[6] C. K. Chu, S. Chow, W. G. Tzeng, J. Y. Zhou, and R. Deng, "Key-aggregate cryptosystem for scalable data sharing in cloud storage," IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 2, pp. 468–477, Feb. 2014.

[7] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: Improved definitions and efficient constructions," in Proc. 13th ACM Conf. Comput. Commun. Security, 2006, pp. 79–88.

[8] C. Dong, G. Russello, and N. Dulay, "Shared and searchable encrypted data for untrusted servers," J. Comput. Security, vol. 19, pp. 367–397, 2011.

[9] F. Zhao, T. Nishide, and K. Sakurai, "Multi-user keyword search scheme for secure data sharing with fine-grained access control," in Proc. Int. Conf. Inf. Security Cryptol., 2012, pp. 406–418.

[10] J. W. Li, J. Li, X. F. Chen, C. F. Jia, and Z. L. Liu, "Efficient keyword search over encrypted data with fine-grained access control in hybrid cloud," in Proc. 6th Int. Conf. Netw. Syst.Security, 2012, pp. 490–502.

[11] J. Li and K. Kim, "Hidden attribute-based signatures without anonymity revocation," Inf. Sci., vol. 180, no. 9, pp. 1681–1689, 2010.

[12] X. F. Chen, J. Li, X. Y. Huang, J. W. Li, and Y. Xiang, "Secure outsourced attribute-based signatures," IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 12, pp. 3285–3294, Dec. 2014.

[13] J. Li, X. F. Chen, M. Q. Li, J. W. Li, P. Lee, and W. Lou, "Secure deduplication with efficient and reliable convergent key management," IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 6, pp. 1615–1625, Jun. 2014.

[14] Z. L. Liu, Z. Wang, X. C. Cheng, and , C. F. Jia, K. Yuan, "Multi-user searchable encryption with coarser-grained access control in hybrid cloud," in Proc. 4th Int. Conf. Emerging Intell. Data Web Technol., 2013, pp. 249–255.

[15] R. A. Popa and N. Zeldovich, "Multi-key searchable encryption," Cryptol. ePrint Archive, Rep. 2013/508, 2013.

[16] Mingwu Zhang, Wei Shi, Chunzhi Wang, Zhenhua Chen, Yi Mu: Leakage-Resilient Attribute-Based Encryption with Fast Decryption: Models, Analysis and Constructions. ISPEC 2013, volume 7863 of LNCS, pages 75-90, 2013

[17] Akavia A, Goldwasser S and Vaikuntanathan V. Simultaneous hardcore bits and cryptography against memory attacks". TC-C'09, LNCS 5444, pp. 474-495, Berlin: Springer-Verlag, 2009.

[18] J. Alwen, Y. Dodis, M. Naor, G. Segev, S. Wallsh, and D. Wichs. Public-key encryption in the bounded-retrieval model. In EUROCRYPT, pages 113-134, 2010.

[19] J. Alwen, Y. Dodis, and D. Wichs. Leakage-resilient public-key cryptography in the bounded-retrieval model. In CRYPTO, pages 36-54, 2009.

[20] D. Di Crescenzo, R. J. Lipton, and S. Wallsh. Perfectly secure password protocols in the bounded retrieval model. In TCC, pages 225-244, 2006.

[21] Y. Dodis, K. Haralambiev, A. Lopez-Alt, and D. Wichs. Cryptography against continuous memory attacks. In FOCS, pages 511-520, 2010.

[22] A. Lewko, Y. Rouselakis, B. Waters. Achieving Leakage Resilience through Dual System Encryption. TCC 2011, LNCS 6597, pages: 70-88, 2011.

[23] Dodis, Y., Goldwasser, S., Kalai, Y.T., Peikert, C., Vaikuntanathan, V.: Public Key Encryption Schemes with Auxiliary Inputs. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 361C381. Springer, Heidelberg (2010).

[24] Tsz Hon Yuen, Sherman S. M. Chow, Ye Zhang, Siu-Ming Y-iu: Identity-Based Encryption Resilient to Continual Auxiliary Leakage. EUROCRYPT 2012: 117-134.

[25] Tsz Hon Yuen,Ye Zhang, Siu-ming Yiu, Joseph K. Liu: Identity-Based Encryption with Post-Challenge Auxiliary Inputs for Secure Cloud Applications and Sensor Networks, ESORICS 2014, Part I, LNCS 8712, pp. 130-147, 2014.

[26] Zhiwei Wang, Siu Ming Yiu: Attribute-Based Encryption Relient to Auxiliary Input, ProvSec 2015, LNCS 9451, pp. 371-390, 2015.

Payal Bhagat received Bachelor degree in Computer Science and Engineering from SIPNA College of Engineering, Amravati, Maharashtra, India, in 2015, and Currently doing Master of Engineering in Computer in Pune Institute of Computer Technology (P.I.C.T), Pune, Maharashtra, India.

Amar Buchade is a Assistant Professor in Pune Institute of Computer Technology (P.I.C.T), Pune, Maharashtra, India.