



## Attribute-Based Encryption Approach for Storage, Sharing and Retrieval of Encrypted Data in the Cloud

**Mrs. M. Swaranalatha**

Computer Science & Engineering(CS)

(Head of the department)

Sphoorthy Engineering College.

College(JNTUH)

[hodcs@sphoorthyengg.ac.in@gmail.com](mailto:hodcs@sphoorthyengg.ac.in@gmail.com)

**Dr.Subba Rao Kolavennu**

Computer Science &Engineering

(JNTUH)

Sphoorthy Engineering College

(JNTUH)

[profrao99@gmail.com](mailto:profrao99@gmail.com)

**P. Shirisha**

Computer Science &

engineering(B.Tech, JNTUH)

Sphoorthy Engineering

College(JNTUH)

[siripaloju20@gmail.com](mailto:siripaloju20@gmail.com)

**G. Srija**

Computer Science & engineering

engineering(B. Tech, JNTUH)

Sphoorthy Engineering College

College(JNTUH)

[g.srija2233@gmail.com](mailto:g.srija2233@gmail.com)

**P. Aditya Lakshmi**

Computer Science & engineering

(B.Tech, JNTUH)

Sphoorthy Engineering College

(JNTUH)

[aihdya10@gmail.com](mailto:aihdya10@gmail.com)

**G. Nandini**

Computer Science &

(B.Tech, JNTUH)

Sphoorthy Engineering

(JNTUH)

[nandinishetty504@gmail.com](mailto:nandinishetty504@gmail.com)



## Abstract

One of the most cost-effective services in cloud computing is storage, used by businesses and individuals to outsource their massive data to untrusted servers. Efforts have studied problems around this application scenario in different fronts: efficiency, flexibility, reliability, and security. In this paper we address the security concerns of cloud storage under the scenario where users encrypt-then-outsource data, share their outsourced data with other users, and the service provider can be queried for searching and retrieval of encrypted data. As main distinctive, we propose a security approach for storage, sharing and retrieval of encrypted data in the cloud fully constructed on the basis of attribute-based encryption (ABE) thus enabling access control mechanisms over both the encrypted data and also for the information retrieval task through search access control. Compared to related works, our approach considers efficient encryption at three different levels: i) bulk encryption of data outsourced to the cloud, ii) keys management for access control over encrypted data by means of digital envelopes from attribute based encryption, and iii) novel construction for attribute based searchable encryption (ABSE). Our underlying ABE algorithms are carefully selected from the body of knowledge and novel constructions for ABSE are provided over the asymmetric setting (Type-III pairings) to support security levels of 128-bits or greater. Experimental results on benchmark data sets demonstrate the viability of our approach for practical realizations using Barreto-Naehrig curves.

## 1. Introduction

The high availability (access anytime, anywhere) and reliability of data at low cost are the main incentives for organizations and individuals to adopt cloud storage services. These services are in increasing demand due to the high amount of data generated by different sources (Internet of Things) and cloud enabled applications. However, data owners (DOs) outsourcing their data to untrusted servers in the cloud face the security concern that the cloud service provider (CSP) honestly stores the DO's patterns). 10) The encrypted documents  $Ek_1(D_0)$  found by CSP (satisfying the query) are sent back to DU. 11) If DO authorizes DU the access  $D_0$  in plaintext form, then DO shares  $k_1$  with DU

data and follows the agreed protocol but tries to learn as much as possible from the computations and interactions with the users (DU) that access the DO's data. This issue can be solved by providing DOs with a confidentiality security service for DOs to encrypt data before uploading it to the cloud. A straightforward encryption approach to prevent DO's data disclosure and to keep DO's data private from CSP or from any other entity, causes the provider cannot manipulate data, that is, loss of utility appears as the encrypted data cannot be used by the CSP for retrieval/searching purposes. Due to that inconvenience, DUs should download large volume of encrypted data, decrypt, and then search over the plaintext data (locally), re-encrypt and upload again its data to the cloud. Of course, so one approach incurs in huge communications and computations overhead and is completely inefficient. Searchable encryption (SE) [1] has been the most known approach to cope with the problem of searching over encrypted data stored in untrusted servers. SE is defined as the ability to identify and retrieve a set of objects from an encrypted collection that satisfy a query. In SE, the CSP executes DU's encrypted queries over encrypted data without decryption, so it does not learn anything about the data content, search criteria, nor search patterns. The most general method to implement SE in any of its existing families [2] is as follows: 1) DO creates cryptographic keys  $\{k_1, k_2, k_3\}$  with security strength  $\lambda$ ; 2) Given a set of documents  $D$ , DO extracts representative keywords  $W$  in  $D$ ; 3) DO encrypts  $D$  with a symmetric cipher  $E$  using key  $k_1$  ( $Ek_1(D)$ ); 4) DO creates a secure index from  $W$ , using key  $k_2$  ( $Ik_2(W)$ ); 5) DO uploads  $Ek_1(D)$  and  $Ik_2(W)$  to the cloud; 6) DO gives authorization to some DUs to search over its data, using  $k_3$ . 7) DU generates trapdoors for a given query word  $w_q$  in  $W$  using  $k_3$ ,  $Tk_3(w_q)$ ; 8) DU queries the CSP using  $Tk_3(w_q)$ , to retrieve  $D_0$  documents containing  $w_q$ . 9) CSP searches over encrypted data using  $Tk_3(w_q)$  and  $Ik_2(W)$  (CSP will not learn anything about data, search criteria nor query using  $Tk_3(w_q)$  at the time that R1-DO can execute  $Ek_1(D)$  efficiently to provide confidentiality over outsourced data to the cloud at the same time that enables fine-grained data access control and secure distribution of  $k_1$  for DUs, thus enabling secure data sharing.

R2 - DUs can query  $Ik_2(W)$  (via the CSP) by computing and secure fine-grained search control is enabled. R3 - DUs can ask the CSP to

and thus DU decrypts Ek1 (D 0 ) and get access to the documents set D 0 in clear. Generally, Ek1 (D) is not detailed in SE schemes nor any approach for key management for k1, which is crucial for practical SE since E is a symmetric cipher so the key for encryption and decryption must be the same. Keys {k2, k3} are relevant values in SE because k2 enables the creation of the secure index and k3 allows creating the trapdoors (encrypted queries) used for querying. The most studied and popular SE technique is Symmetric Searchable Encryption (SSE) [3], where k2 = k3 (sometimes k1 = k2). In public key encryption with Keyword search (PEKS) [4], {k2, k3} are related keys (based on public key encryption security assumptions), being k2 public and k3 private

## 2. Literature Survey

We have surveyed the existing projects and finally thought of making necessary modifications for getting the latest edition.

### Existing System:-

A straightforward encryption approach to prevent DO's data disclosure and to keep DO's data private from CSP or from any other entity, causes the provider cannot manipulate data, that is, loss of utility appears as the encrypted data cannot be

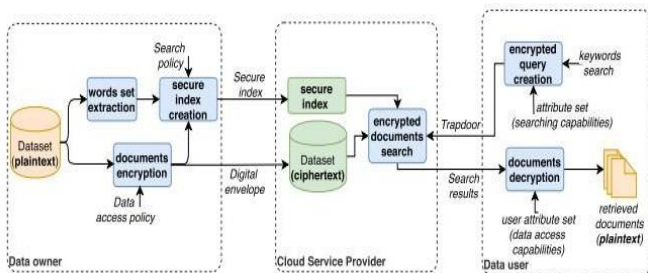


FIGURE 1. System model for document sharing and retrieval in the cloud.

used by the CSP for retrieval/searching purposes. Due to that inconvenience, DUs should download large volume of encrypted data, decrypt, and then search over the plaintext data (locally), re-encrypt and upload again its data to the cloud. Of course, so one approach incurs in huge communications and computations overhead and is completely inefficient. Searchable encryption (SE) has been the most known approach to cope with the problem of searching over encrypted data stored in untrusted servers. SE is defined as the ability to identify and retrieve a set of objects from an encrypted collection that satisfy a query. In SE, the CSP executes DU's encrypted queries over encrypted data without decryption, so it does not learn anything about the data content, search criteria, nor search patterns.

### Proposed System:-

We present a security approach for storing, sharing and retrieving of encrypted data in the cloud, fully constructed on the basis of attribute-based encryption (ABE). Our approach is well suited for a known cloud-based storage and sharing model, where DO uploads encrypted data to the cloud to

ensure confidentiality (by means of symmetric data encryption) and establishes access control mechanisms for data sharing using attribute based encryption; DU can selectively locate specific documents using an index-based structure and retrieve documents of interest in encrypted form, without revealing any information to the CSP and under a fine-grained search control. Our proposed approach aims at meeting the following four requirements to enable practical storage, sharing and retrieval of encrypted data in the cloud:

efficient algorithm to solve the discrete logarithm problem (DLP) in the groups  $G_1$ ,  $G_2$  and  $G_T$ . This estimation is reflected in recommended group size  $\lambda = \log_2 r$  given by the size in bits for  $q$  and  $k$  by several standards and international projects [34]. In this work, all the underlying constructions of ABE are for the Type-III pairing realized with updated groups size  $\lambda$  as

return the  $k$ -most relevant documents from the retrieval task results, ordered accordingly to their relevance to the query. R4 - Both R1 and R2 comply with recommended security levels 1 (i.e.  $\lambda \geq 128$  - bit). We called our approach FABECS (Fully Attribute-Based Encryption scheme for Cloud Storage, Sharing and Retrieval) which fulfills requirements R1-R4. FABECS includes a novel Cipher-text policy ABSE (CP-ABSE) construction to achieve R2 and R3 requirements. At the same time, FABECS reuses the settings of ABSE (pairings and curve parameters) for the set up of DET-ABE which provides cryptographically enforced fine-grained access controls needed to meet requirement R1

## 3. IMPLEMENTATION-

PAIRINGS AND SECURITY ASSUMPTIONS Pairings are

mathematical objects defined over groups and efficient computable pairings are used to construct several cryptographic algorithms and protocols such as ABE. 1) BILINEAR PAIRINGS A bilinear pairing [28] is defined as the mapping  $e : G_1 \times G_2 \rightarrow G_T$ , where  $G_1 = \langle g_1 \rangle$ ,  $G_2 = \langle g_2 \rangle$ , and  $G_T$  are cyclic groups of prime order  $r$ , with  $g_1$  and  $g_2$  the generators of  $G_1$  and  $G_2$  respectively. The pairing  $e$  must satisfy the properties of 1) Bilinearity:  $\forall g \in G_1, \forall \hat{g} \in G_2, \text{ and } a, b \in \mathbb{Z} * r, e(g^a, \hat{g}^b) = e(g, \hat{g})^{ab}$ . 2) Non-degeneracy:  $e(g_1, g_2) \neq 1$ .  $\mathbb{Z} * r$  is the set  $\{1, 2, \dots, r - 1\}$ . In practice,  $G_1$  and  $G_2$  are subgroups of a so-called friendly elliptic curve  $E$  defined over a finite field  $F_q$ , and  $G_T$  is the multiplicative group of extension field  $F_{q^k}$  with  $k$  referred to as the embedding degree of the elliptic curve, the smallest positive integer such that  $r$  divides to  $q^k - 1$ . If  $G_1 = G_2$  the pairing is symmetric (Type I), otherwise it is asymmetric. If the pairing is asymmetric and no efficient isomorphism is known between  $G_1$  and  $G_2$ , then the pairing



is said to be of Type-III. While a pairing-based cryptographic construction for a Type-III pairing is easy to transform to Type-I (by taking  $G_1 = G_2$ ), the opposite is not trivial. The main motivation to use Type-III pairing constructions of pairing-based cryptographic protocols such as ABE is because of performance and security [29]. Type-I pairings has shown serious security issues [30]. Several applications of Type-III pairing-based constructions are in practical use, for example for protecting privacy of transactions with the zk-SNARKs algorithm [31] and in several Blockchain projects [32]. A kind of friendly elliptic curve is the Barreto-Naehrig curve (BN curve) [33], which constructs Type-III pairings well suited for practical usage.

2) SECURITY ASSUMPTIONS Definition 1 (Discrete Logarithm (DL) Assumption): Let  $G$  be a cyclic group with  $\lambda$ -bit prime order  $r$ . Let  $g$  be a generator of  $G$ . Given  $g$  and  $g^a$ , the DL assumption is defined as: no probabilistic polynomial-time adversary  $A$  can compute  $a \in \mathbb{Z}^*_r$  with a non-negligible advantage  $\text{Adv}_{DL}^A(\lambda)$  in the security

recommended in [35]. As other pairing-based cryptographic schemes, the ABE constructions proposed in this work are based on the following security assumptions for the Type-III pairings. Definition 2 (Decisional Diffie-Hellman (DDH) Assumption): Let  $G$  be a cyclic group with  $\lambda$ -bit prime order  $r$ . Let  $g$  be a generator of  $G$ . Given the tuple  $e\{G_1, G_2, GT, g_1, g_2, r\}$ . The SXDH assumption holds if DDH holds in both  $G_1$  and  $G_2$ . Definition 4 (Decisional Bilinear Diffie-Hellman (DBDH) Type-III Assumption [36]): Let  $P_\lambda$  be the setting for a Type-III pairing consisting on the tuple  $\{G_1, G_2, GT, g_1, g_2, r\}$ . The DBDH assumption is defined as: no probabilistic polynomial-time adversary  $A$  can distinguish  $g_1, g_2, g^a, g^b, g^c, e(g_1, g_2)^{abc}$  from  $g_1, g_2, g^a, g^b, g^c, e(g_1, g_2)^z$  for random elements  $a, b, z \in \mathbb{Z}^*_r$  with a non-negligible advantage  $\text{Adv}_{DBDH}^A(\lambda)$ .

B. CP-ABE AND DET-ABE A Secret Sharing Scheme (SSS) in attribute-based encryption (e.g. CP-ABE) is crucial. In this context, an access structure  $A$  is defined as a non-empty subset of the power set  $P(U)$ , with  $U$  a set of attributes.  $SA$  is called an authorized set if  $SA \in A$ . An SSS involves a dealer that shares a secret  $s$  with a set of  $n$  parties defined by  $U$ . In CP-ABE,  $SAs$  are authorized sets of attributes to decrypt a ciphertext.  $A$  is said to be monotone, if given  $SA \in A$  and  $SA \subset SB$ , then  $SB \in A$ . That is, decryption privileges are not lost even if more attributes are acquired by decryptors. In the W11 ABE construction,  $A$  is an  $n \times 1$  matrix (for an access policy including  $n$  attributes) instead of the tree access structure used in the BSW07 construction, without loss of efficiency. In this work,

the linear SSS (LSSS) proposed by Liu and Cao in [27] is used, where the concept of Formatted Boolean Formula (FBF) over attributes is proposed to represent the access policy associated to the access structure  $A$ . CP-ABE (as other PKC schemes) relies on mathematical operations over large numbers (hundreds of bits). That is why CP-ABE is not used to encrypt data massively. For this purpose, symmetric ciphers  $E$  using the encryption key  $k_1$  (i.e.  $E_{k_1}(D)$ ) are faster and preferred. Digital envelopes [37] have been effective methods to encrypt large data with  $E$  and securely distribute  $k_1$  to intended decryptors. In [38], digital envelopes realized from attribute-based encryption (called DET-ABE) were proposed by using the BSW07 construction. Later, in [11] the construction was extended to the ABE construction based on W11. In DET-ABE, data  $D$  is encrypted with the Advanced Encryption Standard (AES) using a session key  $k_1$ , which is encrypted with CP-ABE with an access policy. Only those authorized entities with a valid set of attributes satisfying the ABE encryption policy could decrypt and recover the data encryption key  $k_1$  to then decrypt and recover  $D$ . DET-ABE can be formally defined by the four

parameter  $\lambda$ . The security level of pairing-friendly curves is estimated by the computational cost of the most polynomial-time algorithms [11]:

- 1)  $\text{DET-ABE.setup}(1\lambda) \rightarrow \{MK, PK\}$ : Creates  $MK$  (master private key) and  $PK$  (master public key), such that the length of both keys is compliant with the  $\lambda$  security strength. Initializes the attributes universe set  $U$ .
- 2)  $\text{DET-ABE.privateKeyGen}(MK, Su) \rightarrow SK_u$ : Creates the user private key  $SK_u$  by using  $MK$ , and given a set of attributes  $Su \subset U$ , specific for the user  $u$ .
- 3)  $\text{DET-ABE.encrypt}(D, PK, A) \rightarrow CTD$ : Encrypts data  $D$  using the AES cipher with a private session key  $k_1$ . Then, encrypts  $k_1$  with CP-ABE using  $PK$  and an access structure  $A$ , which corresponds to an access policy over attributes in  $U$ . The resulting encryption is the package  $CTD = \{AES_{k_1}(D), CP-ABE(k_1, A)\}$ .
- 4)  $\text{DET-ABE.decrypt}(CTD, SK_u) \rightarrow D$ : Decrypts  $CP-ABE(k_1, A)$  using decryptor's private key  $SK_u$  to recover  $k_1$ . Then, decrypts  $AES_{k_1}(D)$  using  $k_1$ . Decryption works only if the attribute set  $Su$  used to generate  $SK_u$  is in  $A$ . DET-ABE provides effective access control mechanisms and confidentiality over a large documents dataset. It has been successfully tested in other systems [39], [40], however, DET-ABE does not support searchable encryption (DET-ABE only meets requirement R1).

C. ATTRIBUTE-BASED SEARCHABLE ENCRYPTION (ABSE) In [9], authors proposed CP-ABSE, a realization of ABSE using the CP-ABE BSW07 construction as basis. As it is the case of other searchable encryption techniques, CP-ABSE constructs a secure index, that is, the index is encrypted so it does not support

traditional queries over plaintext keywords; the matching is cryptographically tested. The secure index maps encrypted keywords to encrypted documents, so the searching process does not leak information about the data being queried. CP-ABSE consists of the following five polynomial-time algorithms [9].

1) CP-ABSE.setup( $1\lambda$ )  $\rightarrow$  {dk1, dk2}: Creates two keys dk1 and dk2; dk1 is used to encrypt an index keyword given an access structure A and dk2 is used to generate user's private key given its attribute set SA in an attribute universe U. With private keys, users can create encrypted queries for data retrieval.


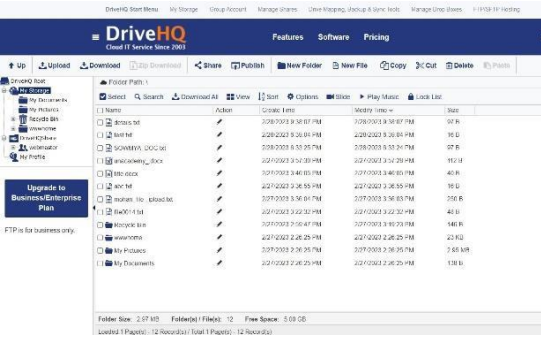
2) CP-ABSE.privateKeyGen(dk2, Su)  $\rightarrow$  SKu: Creates the user private key SKu using dk2 from a set of attributes Su  $\subset$  U, specific for the user u.

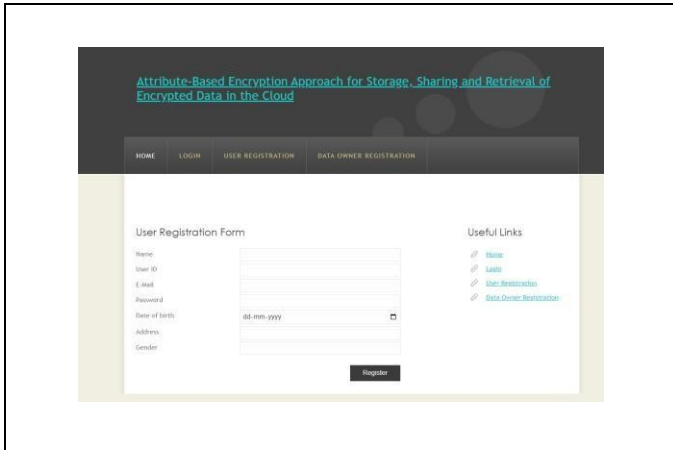
3) CP-ABSE.encInd(dk1,w,A)  $\rightarrow$  CTw: Creates a secure index by encrypting with CP-ABE each keyword w in the index. The result is the set of each encrypted keyword CTw = CP-ABE(dk1,w,A). 4) CP-ABSE.Trpdr(SKu, wq)  $\rightarrow$  Tu(wq): Generates an encrypted query from keyword wq, by using the user private key SKu. The encrypted query is the trapdoor Tu(wq).

5) CP-ABSE.search

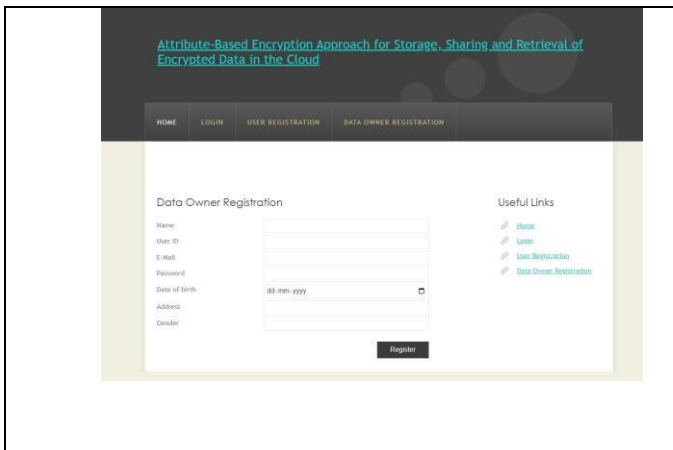
### 4. Results

We present a security approach for storing, sharing and retrieving of encrypted data in the cloud, fully constructed on the basis of attribute-based encryption (ABE).

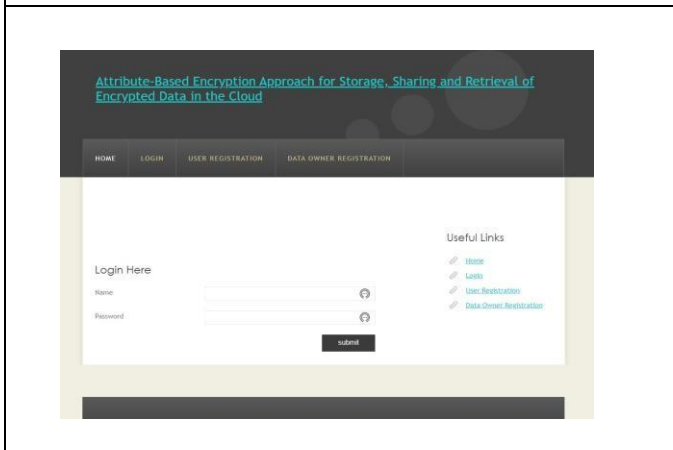
UI Design	Design Description (functions, operations...)
	<p>The above screen appears when we start execution process</p>
	<p>The above screen appears when Data owner wants to see the files Stored in DriveHQ(cloud)</p>



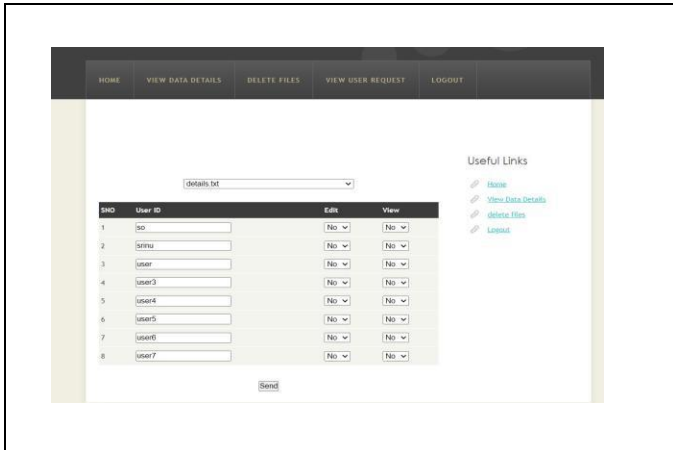
The above screen appears when user wants to register



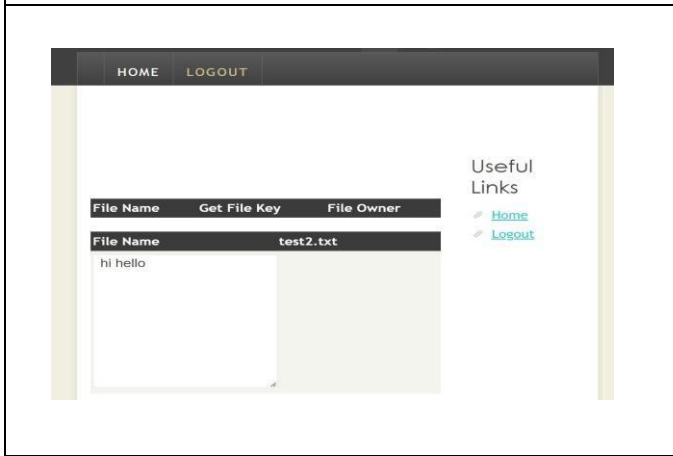
The above screen appears when Data owner wants to register



The above screen appears when Data owner or user wants to login



The above screen displays when user tries to select and give permission to files



The above screen appears to display the selected file



## 5. Conclusion-

We presented for the first time a secure scheme fully based on attribute-based encryption to ensure both, the confidentiality and access control over data outsourced (in encrypted form) by data owners to the cloud and the fine-grained search control for data users when retrieving encrypted data from the cloud; we called this scheme FABECS. Through a formal analysis and experimentation, we proved the correctness and efficacy of FABECS to be used for storing, sharing and retrieval of documents in a cloud based environment. Furthermore, we provided for the first time Type-III constructions for CP-ABSE and DET-ABE as main building blocks of FABECS. This setting allows using more efficient pairing-friendly curves to achieve recommended security levels, as minimum of 128-bits. These constructions were detailed and their efficacy proved by means of experimentation, over the LISA benchmark for the retrieval task. Further work is focused in the efficiency aspect, as the results presented in this paper did not consider acceleration strategies. For example, parallelization at several levels is possible, besides the scheme is friendly enough to be deployed using parallel patterns such as the manager-worker (for processing a group of attributes at a time) or data encryption (AES on GPUs). Also, as FABECS can be realized with other efficient pairing friendly curves, experimental evaluation could consider the Barreto-Lynn-Scott Curve (BLS) that is also being promoted to be used in practical applications.

## 7. References -

- [1] A. Bagherzandi, B. Hore, and S. Mehrotra, Search over Encrypted Data. Boston, MA, USA: Springer, 2011, pp. 1088–1093.
- [2] H. Pham, J. Woodworth, and M. A. Salehi, “Survey on secure search over encrypted data on the cloud,” *Concurrency Comput. Pract. Exper.*, vol. 31, p. 1–15, Apr. 2019.
- [3] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, “Searchable symmetric encryption: Improved definitions and efficient constructions,” in *Proc. 13th ACM Conf. Comput. Commun. Secur.*, New York, NY, USA, 2011, pp. 79–88, 2006.
- [4] M. Zeng, H.-F. Qian, J. Chen, and K. Zhang, “Forward secure public key encryption with keyword search for outsourced cloud storage,” *IEEE Trans. Cloud Comput.*, early access, Sep. 27, 2019, doi: 10.1109/TCC.2019.2944367.
- [5] S. Kamara, C. Papamanthou, and T. Roeder, “Cs2: A searchable cryptographic cloud storage system,” *Microsoft Res.*, Redmond, WA, USA, Tech. Rep. MSR-TR-2011-58, May 2011.
- [6] W. Song, B. Wang, Q. Wang, Z. Peng, W. Lou, and Y. Cui, “A privacy preserved full-text retrieval algorithm over encrypted data for cloud storage applications,” *J. Parallel Distrib. Comput.*, vol. 99, pp. 14–27, Jan. 2017.
- [7] A. G. Kumbhare, Y. Simmhan, and V. Prasanna, “Designing a secure storage repository for sharing scientific datasets using public clouds,” in *Proc. 2nd Int. workshop Data Intensive Comput. Clouds*, 2011, pp. 31–40.

[8] Z. Yang, J. Tang, and H. Liu, “Cloud information retrieval: Model description and scheme design,” *IEEE Access*, vol. 6, pp. 15420–15430, 2018.

H. Yin, J. Zhang, Y. Xiong, L. Ou, F. Li, S. Liao, and K. Li, “CP-ABSE: A ciphertext-policy attribute-based searchable encryption scheme,” *IEEE Access*, vol. 7, pp. 5682–5694, 2019.

[9] A. Sahai and B. Waters, “Fuzzy identity-based encryption,” in *Advances in Cryptology (Lecture Notes in Computer Science)*, vol. 3494, R. Cramer, Ed. Berlin, Germany: Springer-Verlag, 2005, pp. 457–473.

## 8. Appendix-

**ABSE** : attribute based searchable encryption

**ABE**: attribute-based encryption  
**CSP**: cloud service provider  
**DO** : data owners

**SE**: Searchable encryption

**CP-ABE**: cipher text-policy attribute-based encryption

**SSE**: Symmetric Searchable Encryption

**PEKS**: public key encryption with Keyword search

**FABECS**: Fully Attribute-Based Encryption scheme for CloudStorage, Sharing and Retrieval

**PKC** :public key cryptosystems