RESEARCH ARTICLE OPEN ACCESS

Preserving Privacy and Deduplication on Cloud with Attribute-Based Encryption and AES

[1]DR. D. JAYA KUMARI, [2]Y. RAMYA YESU SAI KRISHNAVENI

[1] Professor and HOD, Dept. of CSE, Sri Vasavi Engineering College, Pedatadepalli, Tadepalligudem, West Godavari ,

[2] MCA, Sri Vasavi Engineering College, Pedatadepalli, Tadepalligudem, West Godavari,

Abstract:

"Cloud Computing" is a general term for the delivery of hosted services over the internet. Cloud computing is moving increasingly to a destination with no return: the consolidation as an essential tool for the future existence of the internet world.Attribute-based encryption (ABE) has been widely used in cloud computing where a data provider outsources his/her encrypted data to a cloud service provider, and can share the data with users possessing specific credentials (or attributes). However, the standard ABE system does not support secure deduplication, which is crucial for eliminating duplicate copies of identical data in order to save storage space and network bandwidth. In this paper, we present an attribute-based storage system with secure deduplication in a hybrid cloud setting, where a private cloud is responsible for duplicate detection and a public cloud manages the storage. Compared with the prior data deduplication systems, our system has two advantages. Firstly, it can be used to confidentially share data with users by specifying access policies rather than sharing decryption keys. Secondly, it achieves the standard notion of semantic security for data confidentiality while existing systems only achieve it by defining a weaker security notion. In addition, we put forth a methodology to modify a ciphertext over one access policy into ciphertexts of the same plaintext but under other access policies without revealing the underlying plaintext. But generation of ciphertext is a heavy computation in Attribute-Based Encryption (ABE) for large files. To improve the system's performance we are using the Symmetric Encryption algorithm, such as AES. The procedure of Encryption is performed by the data owner himself/herself first chooses a random number K as the symmetric key and encrypts the plaintext message M using K with the symmetric encryption algorithm. The encrypted data can be denoted as E_K (M). Then the owner encrypts the symmetric key K using CP-ABE under the access policy defined by him/her.

Keywords — **ABE, Storage, Deduplication, Advanced Encryption Standard.**

I.INTRODUCTION

The term Cloud refers to a network or internet. In other words, we can say that Cloud is something, which is present at remote location. Cloud can provide services over network, i.e., on public networks or on private networks like WAN, LAN or VPN. Applications such as e-mail, web conferencing, customer relationship management (CRM), all run in cloud. Cloud computing greatly facilitates data providers who want to outsource their data to the cloud without disclosing their sensitive data to external parties and would like users with certain credentials to be able to access the data. This requires data to be stored in encrypted forms with access control policies such that no one except users with attributes (or credentials) of specific forms can decrypt the encrypted data.We consider the following scenario in the design of an attribute-based storage system supporting secure deduplication of encrypted data in the cloud, in which the cloud will not store a file more than once even though it may receive multiple copies of the same file encrypted under different access policies. A data provider, Bob, intends to upload a file M to the cloud, and share M with users having certain credentials. In order to do so, Bob encrypts M under an access policy A over a set of attributes, and uploads the corresponding ciphertext to the cloud, such that only users whose sets of attributes satisfying the access policy can decrypt the ciphertext. Later, another data provider, Alice, uploads a ciphertext for the same underlying file M but ascribed to a different access policy A0. Since the file is uploaded in an encrypted form, the cloud is not able to discern that the plaintext corresponding to Alice's ciphertext is the same as that corresponding to Bob's, and will store M twice. Obviously, such duplicated storage wastes storage space and communication bandwidth.

1.1 Our Contributions

In this paper, we present an attribute-based storage system which employs ciphertext-policy attribute-based encryption (CP-ABE) and supports secure deduplication. Our main contributions can be summarized as follows:

• Firstly, the system is the first that achieves the standard notion of semantic security for data confidentiality in attribute-based deduplication systems by resorting to the hybrid cloud architecture [1].

• Secondly, we put forth a methodology to modify a ciphertext over one access policy into ciphertexts of the same plaintext but under any other access policies without revealing the underlying plaintext. This technique might be of independent interest in addition to the application in the proposed storage system.

• Thirdly, we propose an approach based on two cryptographic primitives, including a zero-knowledge proof of knowledge [2] and a commitment scheme [3], to achieve data consistency in the system.

In a typical storage system with secure deduplication (e.g., [4]), to store a file in the cloud, a data provider generates a tag and a ciphertext. The data provider uploads the tag and the ciphertext to the cloud. Upon receiving an outsourcing request from a data provider for uploading a ciphertext and an associated tag, the cloud runs a so-called equality checking algorithm, which checks if the tag in the incoming request is identical to any tags in the storage system. If there is a match, then the underlying plaintext of this incoming ciphertext has already been stored and the new ciphertext is discarded. It is apparent that such a system with a tag appended to the ciphertext does not provide the standard notion of semantic security for data confidentiality [5], because if the plaintexts can be predicated from their tags, an adversary can always make a correct guess by computing the tag of a plaintext and then testing it against the tag in the challenge phase in the semantic security game. To circumvent this obstacle, we bring in our system a hybrid cloud architecture [1], which consists of a private cloud responsible for tag checking and ciphertext regeneration and a public cloud storing the ciphertexts. However, endowing such a tag checking ability to the private cloud is not sufficient to achieve deduplication in the attribute-based storage system which employs CP-ABE for data encryption. In the proposed attributed-based system, the same file could be encrypted to different ciphertexts associated with different access policies, storing only one ciphertext of the file means that users whose attributes satisfy the access policy of a discarded ciphertext (but not that of the stored ciphertext) will be denied to access the data that they are entitled to. To overcome this problem, we equip the private cloud with another capability named ciphertext regeneration. For a ciphertext c of a plaintext M with access policy A, the private cloud will be provided with a trapdoor key which is generated along with the ciphertext C by a data provider. The private cloud can use the trapdoor key to convert the ciphertext c with access policy A to a new ciphertext C with another access policy A1 without knowing the underlying message M. Thus, if two data providers happen to upload two ciphertexts corresponding to the same file but under different access policies A and A1, the private cloud can regenerate a ciphertext for the same underlying file with an access policy A U A1 using the corresponding trapdoor key and then store the new ciphertext instead of the old one in the public cloud.

Another key challenge in secure deduplication is to make it secure against duplicate faking attacks [6] in which a legally generated message is unnoticeably replaced by a fake one. In such an attack, a malicious user may intercept an outsourcing request and tamper with the ciphertext, and then sending the modified ciphertext but the original tag to the cloud. Later, an honest data provider wants to upload a ciphertext for an identical file. The cloud spots that the tags of the two ciphertexts match each other, and thus might discard the ciphertext from the honest data provider and keeps the maliciously modified ciphertext. When a user downloads the ciphertext, a tampered message M1 rather than the correct M will be returned, which violates data integrity. In order to address this problem, we require the data provider to produce a proof of consistency reflecting that the tag and the ciphertext are legitimately generated. Our approach of producing such a proof makes use of the randomness reuse technique in the generation of the tag and the ciphertext with an additional zero-knowledge proof of knowledge (PoK) [2] on the shared random coin in the tag and the ciphertext. Therefore, it is impossible for an adversary to perform duplicate faking attacks unless the adversary casually obtains the content of the plaintext hidden in the ciphertext.A straightforward way to achieve this is to save the tagsand the ciphertexts in pairs in the public cloud, but if thetag and the corresponding ciphertext are both known to thepublic cloud, then as we mentioned before, it is impossibleto obtain semantic security. To achieve the standard securitynotation for data confidentiality [5], we ask a data providerto generate a label, in addition to the prior tag and ciphertext, using a commitment scheme [3]. This label is bound toteciphertext and tag using the aforementioned PoKsystembut reveals no information about the underlying plaintextto the public cloud and users who are not entitled with thedecryption privilege, and will be outsourced to the publiccloud with the ciphertext instead of the tag, so that evenif an adversary who is aware of the data that an

honestdata provider may upload, the duplicate faking attacks canbe detected by users who download and decrypt the data.Note that because the label is stored by the private andpublic clouds, the tampering behavior to the label in thepublic cloud will be immediately detected by the privatecloud. Therefore, a user having decryption privilege to theciphertext can always check the correctness of the plaintextvia the label since the tag and the label must be tied to thesame plaintext in terms of the proof.

1.2 Related Work

Attribute-Based Encryption: Sahai and Waters [7] introduced the notion of attribute-based encryption (ABE), and then Goyal et al. [8] formulated key-policy ABE (KP-ABE) and ciphertext-policy ABE (CP-ABE) as two complimentary forms of ABE. The first KP-ABE construction given in [8] realized the monotonic access structures, the first KP-ABE system supporting the expression of non-monotone formulas was presented in [9] to enable more viable access policies, and the first large class KP-ABE system was presented by in the standard model in [10]. Nevertheless, we believe that KP-ABE is less flexible than CP-ABE because the access policy is determined once the user's attribute private key is issued. Bethencourt, Sahai and Waters [11] proposed the first CP-ABE construction, but it is secure under the generic group model. Cheung and Newport [12] presented a CPABE scheme that is proved to be secure under the standard model, but it only supports the AND access structures. A CP-ABE system under more advanced access structures is proposed by Goyal et al based on the number theoretic assumption. In order to overcome the limitation that the size of the attribute space is polynomially bounded in the security parameter and the attributes are fixed ahead, Rouselakis and Waters [13] built a large universe CP-ABE system under the prime-order group. In this paper, the Rouselakis-Waters system is taken as the underlying scheme for the concrete construction.

Secure Deduplication. With the goal of saving storage space for cloud storage services, Douceur et al proposed the first solution for balancing confidentiality and efficiency in performing deduplication called convergent encryption, where a message is encrypted under a message-derived key so that identical plaintexts are encrypted to the same ciphertexts. In this case, if two users upload the same file, the cloud server can discern the equal ciphertexts and store only one copy of them. Implementations and variants of convergent encryption were deployed in [14]. In order to formalize the precise security definition for convergent encryption, Bellare, Keelveedhi and Ristenpart [6] introduced a cryptographic primitive named messagelocked encryption, and detailed several definitions to capture various security requirements. Abadi et al. [15] then strengthened the security definition in [6] by considering the plaintext distributions depending on the public parameters of the schemes. This model was later extended by Bellare and Keelveedhi by providing privacy for messages that are both correlated and dependent on the public system parameters. Since message-locked encryption cannotresist to brute-force attacks where files falling into a known set will be recovered, an architecture that provides secure deduplicated storage resisting brute-force attacks was put forward by Keelveedhi, Bellare and Ristenpart and realized in a system called server-aided encryption for deduplicated storage. In this paper, a similar technique to that in [15] is used to achieve secure deduplication with regard to the private cloud in the concrete construction.

Message-locked encryption and secure deduplication: Motivated by the problem of avoiding duplication in storage systems, Bellare, Keelveedhi, and Ristenpart have recently put forward the notion of Message-Locked Encryption (MLE) schemes which subsumes convergent encryption and its variants. Such schemes do not rely on permanent secret keys, but rather encrypt messages using keys derived from the messages themselves. We strengthen the notions of security proposed by Bellare et al. by considering plaintext distributions that may depend on the public parameters of the schemes. We refer to such inputs as lock-dependent messages. We construct two schemes that satisfy our new notions of security for message-locked encryption with lock-dependent messages. Our main construction deviates from the approach of Bellare et al. by avoiding the use of ciphertext components derived deterministically from the messages. We design a fully randomized scheme that supports an equality-testing algorithm defined on the ciphertexts. Our second construction has a deterministic ciphertext component that enables more efficient equality testing. Security for lock-dependent messages still holds under computational assumptions on the message distributions produced by the attacker. In both of our schemes the overhead in the length of the ciphertext is only additive and independent of the message length.

II. LITERATURE REVIEW

Cloud Computing refers manipulating, configuring, and accessing the applications through online. It offers online data storage, infrastructure and application. Cloud Computing is both a combination of software and hardware based computing resources delivered as a network service. In this should be registered with this application. If he wants to access his activities the authority has to provide permission to

him. The data provider will upload the files with encrypted data. The user will access all files to download [4].

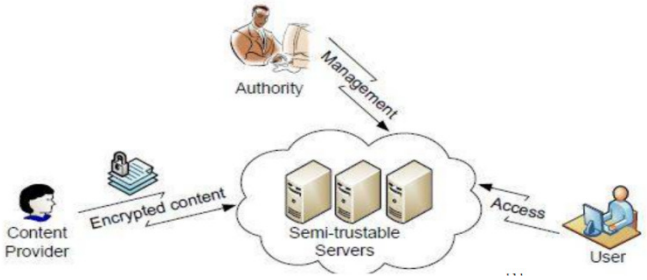


Fig.1.User access to download the files if valid

In this encryption the user has public key information about unique identification. In this scenario the program allows to anyone to generate public key. With this public key the user can access his activities, but every group of less than k participants cannot obtain any information about the secret [5].

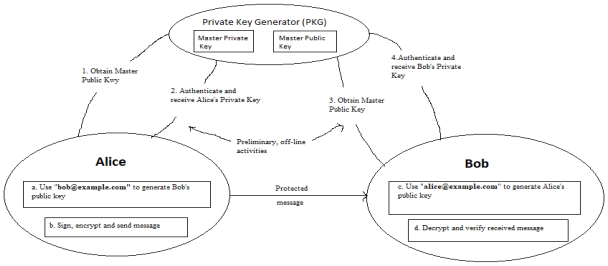


Fig.2. Key Working process between Alice and Bob

First solution to provide secure de-duplication + compromise resilience

- Can be deployed transparently over existing systems
- Implementations over Drop box, Google Drive
- Nominal performance overhead over plaintext de-duplication
- Storage savings match plaintext de-duplication [8].

The client communicates with the Trusted Cloud over a low bandwidth, secure channel. The two clouds are connected with an insecure, high bandwidth channel. The Commodity Cloud further provides un-trusted storage [12].

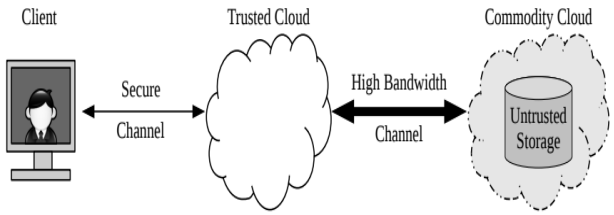


Fig.3.Information passed through cloud

From the above fig initially the user will login to cloud and creates a profile, he can upload files into database. The file will send to the data owner for giving the ownership for the user. It is checked utilizing the MD5 (Message-Digest) calculation. MD5 calculation creates the hash work. This MD5 algorithm generates digital signature guaranteeing consistency (integrity) of data [24].

III. EXISTING SYSTEM

In existing system, a data provider Bob intends to upload a file M to the cloud, and share M (file data) with users having certain credentials. In order to do so, Bob encrypts M under an access policy A over a set of attributes, and uploads the corresponding ciphertext to the cloud, such that only users whose sets of attributes satisfying the access policy can decrypt the ciphertext. Later, another data provider Alice uploads a ciphertext for the same underlying file M but ascribed to a different access policy A0. Since the file is uploaded in an encrypted form, thecloud is not able to discern that the plaintext corresponding to Alice's ciphertext is the same as that corresponding to Bob's, and will store M twice. Obviously, such duplicated storage wastes storage space and communication bandwidth. We present an attribute-based storage system which employs ciphertext-policy attribute-based encryption (CP-ABE) and supports secure de-duplication. In the proposed attributed-based system, the same file could be encrypted to different cipher texts associated with different access policies, storing only one ciphertext of the file means that users whose attributes satisfy the access policy of a discarded ciphertext (but not that of the stored ciphertext) will be denied to access the data that they are entitled to. To overcome this problem, we equip the private cloud with another capability named ciphertext regeneration. For a ciphertext c of a plaintext M with access policy A, the private cloud will be provided with a trapdoor key which is generated along with the ciphertext c by a data provider. The private cloud can use the trapdoor key to convert the ciphertext c with access policy A to a new ciphertext C with another access policy A0 without knowing the underlying message M. Thus, if two data providers happen to upload two cipher texts corresponding to the same file but under different access policies A and A0, the private cloud can regenerate a ciphertext for the same underlying file with an access policy A UA0 using the corresponding trapdoor key and then store the new ciphertext instead of the old one in the public cloud.

Limitations:

Loss of Space: When duplicate file is stored then wastes the memory. These will loss of memory space.

Increase the Network Bandwidth: While increasing the memory space it should increase the network bandwidth.

IV. PROPOSED WORK

In this system for improving the system's performance we are using the Symmetric Encryption algorithm, such as AES. The procedure of Encryption is performed by the data owner user first chooses a random number K as the symmetric key and encrypts the plaintext message M using K with the symmetric

encryption algorithm. The encrypted data can be denoted as E_K (M). Then the owner encrypts the symmetric key K using CP-ABE under the access policy defined by the user. The below system architecture and formal definition of ciphertext-policy attribute-based storage system supporting secure de-duplication (CP-ABE) and advanced encryption algorithm (AES), in which four entities are involved: data providers, attribute authority (AA), cloud and users.In this section, we describe the system architecture and the formal definition of ciphertext-policy attribute-based storage system supporting secure deduplication.

4.1 System Architecture

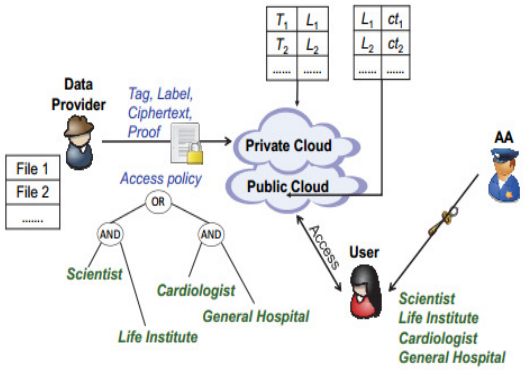


Fig.4. System architecture of attribute-based storage with secure deduplication

The architecture of our attribute-based storage system with secure deduplication is shown in Fig.4 in which four entities are involved: data providers attribute authority (AA), cloud and users. A data provider wants to outsource his/her data to the cloud and share it with users possessing certain credentials. The AA issues every user a decryption key associated with his/her set of attributes. The cloud consists of a public cloud which is in charge of data storage and a private cloud which performs certain computation such as tag checking. When sending a file storage request, each data provider firstly creates a tag T and a label L associated with the data, and then encrypt the data under an access structure over a set of attributes. Also, each data provider generates a proof pf on the relationship of the tag T, the label L and the encrypted message ct3, but this proof will not be stored anywhere in the cloud and is only used during the checking phase for any newly generated storage request. After receiving a storage request, the private cloud first checks the validity of the proof pf, and then tests the equality of the new tag T with existing tags in the system. If there is no match for this new tag T, the private cloud adds the tag T and the label L to a tag-label list, and forwards the label and the encrypted data, (L, ct) to the public cloud for storage. Otherwise, let ct1 be the ciphertext whose tag matches the new tag and L0 be the label associated with ct1, and then the private cloud executes as follows.

- If the access policies in ct and ct0 are not mutually contained, the private cloud runs the ciphertext regeneration algorithm to yield a new ciphertext for the same underlying plaintext file and associated with an access structure which is the union of the two access structures, and forwards the original label and the resulting ciphertext to the public cloud.
- At the user side, each user can download an item, and decrypt the ciphertext with the attribute-based private key generated by the AA if this user's attribute set satisfies the access structure. Each user checks the correctness of the decrypted message using the label, and accepts the message if it is consistent with the label.

4.2 Advanced Encryption Standard (AES)

The Advanced Encryption Standard (AES) is an encryption algorithm for securing sensitive data. It has been adopted by the United States government as anAdvanced Encryption Standard, a standard algorithmused to encrypt and decrypt sensitive information. AESis a symmetric block cipher with a block size of 128bits. It allows for three different key lengths which canbe 128 bits, 192 bits, or 256 bits; referred to as AES-128, AES-192, and AES-256, respectively. The numberof rounds in the encryption process for AES-128 is 10,for AES-192 it is 12, and for AES-256 it is 14.

The major loop of AES executes the functions given below:

Functions of Advanced Encryption Standard

- SubBytes()
- Shift Rows()
- MixColumns()
- AddRoundKey()

AES makes use of 10, 12 and 14 rounds. The plain text is transformed into cipher text after repeated transformation rounds in AES. This makes the data secure on the cloud.

AES-128, AES-192, AES-256 (128, 192 and 256 are bits) process the data block in, respectively, 10, 12, or 14 rounds. The transformations are predefined. All the rounds are similar except the last one where the transformation is missed. The rounds operate on two 128 bits i.e., state and round key. Each round from 1 to 10 or 12 or 14 uses a different round key.

- The data block is processed as follows:
- The AES encryption routine begins by copying the 16-byte input array into a 4×4 byte matrix named State.
- Input data block also known as state is XORed with the first 128-bits of the cipher key.

- Then the resulting State is serially passed through 10/12/14 rounds.
- The result of the last round is encrypted data.

The process of AES encryption algorithm using 128-bit key, is diagrammatically represented in figure 5.

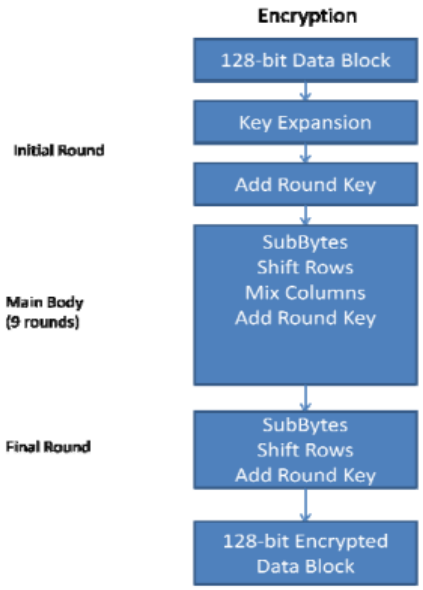


Fig. 5 Process of AES

Algorithm Process:

a) Key Expansion: Using the key schedule of Rijndael, round keys are derived from the cipher key

b) Initial Round – AddRoundKey: Then using bitwise XOR each byte of the state is combined with the round key.

c) Rounds

i) SubBytes: This is a non-linear substitution step where each byte is swapped with another according to a lookup table.

ii) ShiftRows: In this transposition step each row of the state is shifted cyclically a certain number of steps.

iii) MixColumns: A mixing operation which operates on the columns of the state, combining the four bytes in each column.

iv)AddRoundKey

d) Final Round (no Mix Columns)

v) SubBytes

vi) Shift Rows

vii) AddRoundKey

In encryption AES is more suitable. ABE is considered to be more expensive. So the data is not directly encrypted using ABE. Generally symmetric key is used for encrypting bulk of the data and asymmetric key like ABE is suitable for encrypting short key value. First data is encrypted using AES with 128 bits keys and the AES keys are again encrypted/ decrypted using ABE and are sent together with ciphertext.The more popular and widely adopted symmetric encryption algorithm likely to be encountered nowadays is the Advanced Encryption Standard (AES). It is found at least six times faster than triple DES.A replacement for DES was needed as its key size was too small. With increasing computing power, it was considered vulnerable against exhaustive key search attack. Triple DES was designed to overcome this drawback but it was found slow.

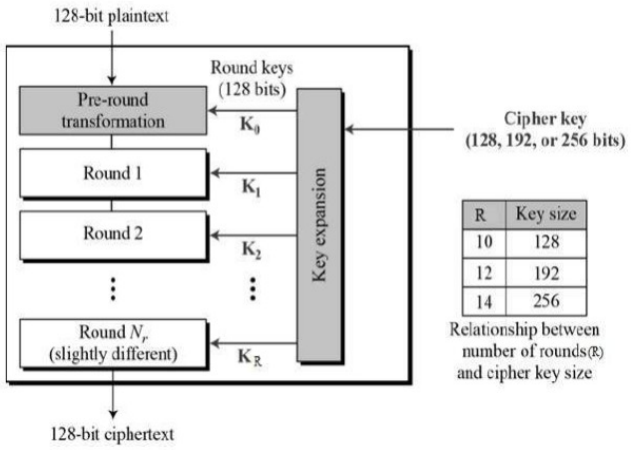
The features of AES are as follows:

- Symmetric key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- Stronger and faster than Triple-DES
- Provide full specification and design details
- Software implementable in C and Java

Operation of AES

AES is an iterative rather than Feistel cipher. It is based on 'substitution–permutation network'. It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix. Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.

The schematic of AES structure is given in the following illustration –

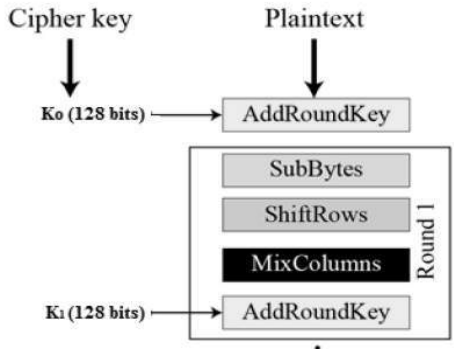


R	Key size
10	128
12	192
14	256

Relationship between number of rounds(R) and cipher key size

Encryption Process

Here, we restrict to description of a typical round of AES encryption. Each round comprise of four sub-processes. The first round process is depicted below –



Byte Substitution (SubBytes)

The 16 input bytes are substituted by looking up a fixed table (S-box) given in design. The result is in a matrix of four rows and four columns.

Shiftrows

Each of the four rows of the matrix is shifted to the left. Any entries that 'fall off' are re-inserted on the right side of row. Shift is carried out as follows –

- First row is not shifted.
- Second row is shifted one (byte) position to the left.
- Third row is shifted two positions to the left.
- Fourth row is shifted three positions to the left.
- The result is a new matrix consisting of the same 16 bytes but shifted with respect to each other.

MixColumns

Each column of four bytes is now transformed using a special mathematical function. This function takes as input the four bytes of one column and outputs four completely new bytes, which replace the original column. The result is another new matrix consisting of 16 new bytes. It should be noted that this step is not performed in the last round.

Addroundkey

The 16 bytes of the matrix are now considered as 128 bits and are XORed to the 128 bits of the round key. If this is the last round then the output is the ciphertext. Otherwise, the resulting 128 bits are interpreted as 16 bytes and we begin another similar round.

Decryption Process

The process of decryption of an AES ciphertext is similar to the encryption process in the reverse order. Each round consists of the four processes conducted in the reverse order –

- Add round key
- Mix columns
- Shift rows
- Byte substitution

Since sub-processes in each round are in reverse manner, unlike for a Feistel Cipher, the encryption and decryption algorithm needs to be separately implemented, although they are very closely related.

4.3 System Implementation

Our ciphertext-policy attribute-based storage system with secure deduplication consists of the following algorithms:

Setup → (pars, msk):

Taking the security parameter λ as the input, this setup algorithm outputs the public parameter pars and the master private key msk for the system. This algorithm is run by the AA.

KeyGen(pars, msk, A) → skA

Taking the public parameter pars, the master private key msk and an attribute set A as the input; this attribute-based private key generation algorithm generates an attributebased private key skA for the attribute set A. This algorithm is run by the AA.

Encrypt (pars, M, A) → (skT, CT)

The procedure of Encryption is performed by the data provider himself/herself. To improve the system's performance, the owner first chooses a random number k as the symmetric key and encrypts the plaintext message M using κ with the symmetric encryption algorithm. The encrypted data can be denoted as E_K (M). Then the owner encrypts the symmetric key k using CP-ABE under the access policy (A) defined by himself/herself. Taking the public parameter pars, a message M and an access structure A over the universe of attributes as the input, this encryption algorithm outputs a trapdoor key skT and a tuple CT = (T, L, ct, pf), where T and L are the tag and the label associated with M respectively, ct is the ciphertext which includes the encryption of M as well as the access structure A, and pf is a proof on the relationship of tag T, label L and ciphertext ct.

Validity-Test (pars, CT) →1/0

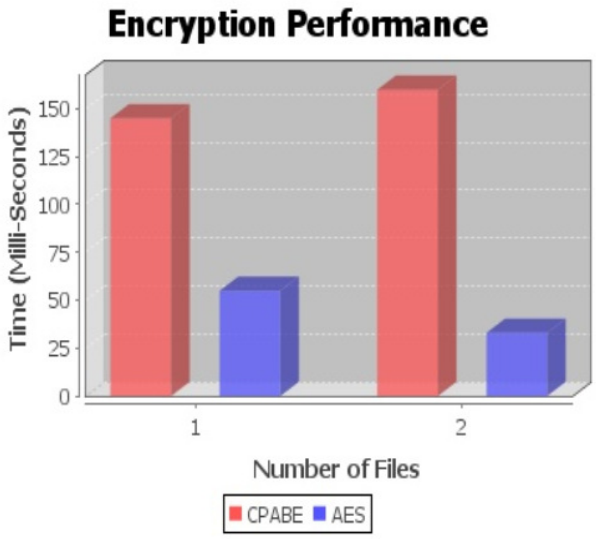
Taking the public parameter pars and a tuple CT as the input, thisvalidity testing algorithm parses CT as (T, L, ct, pf),and outputs 1 if pf is a valid proof for (T, L, ct) or 0otherwise.This algorithm is run by the private cloud.

Equality-Test (pars, (T1, L1, ct1), (T2, L2, ct2)) →1/0
Taking the public parameter pars and two tuples (T1, L1, ct1) and (T2, L2, ct2) as the input, this equality testing algorithm outputs 1 if both (T1, L1, ct1), (T2, L2, ct2) are generated from the same underlying message or 0 otherwise. This algorithm is run by the private cloud.

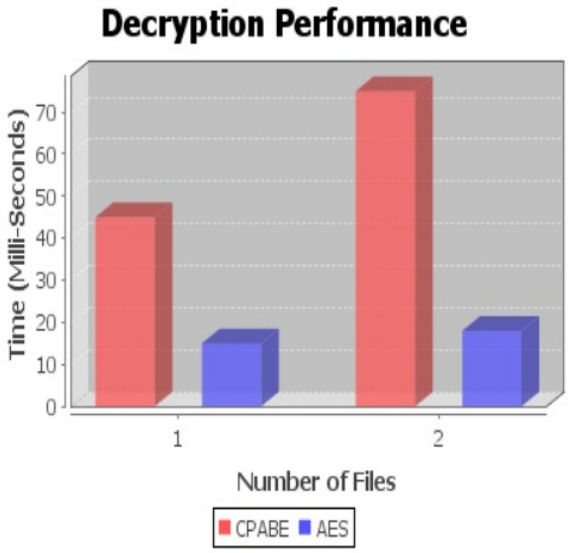
Re-encrypt (pars, skT, (L, ct), A0) →(L, ct0)
Taking the public parameter pars, the trapdoor key skT, a tag and ciphertext pair (L, ct) and an access structure A0 as the input, this re-encryption algorithm outputs a new ciphertext ct0 associated with A0 sharing the same label L of the ciphertext ct0.This algorithm is run by the private cloud.

Decrypt (pars, (L; ct), A, skA) →M
Taking the public parameter pars, a label and ciphertext pair (L; ct) and an attribute-based private key skA associated to an attribute set A as the input, this decryption algorithm outputs either the message M when the private key skA satisfies the access structure of the ciphertext (ct) and the label L is consistent with M (to be defined later), or false indicating the failure of the decryption. This algorithm is run by the user.

4.4 System Results



(a) Encryption between CP-ABE &AES



(b) Decryption between CP-ABE & AES

V. ATTRIBUTE-BASED STORAGE WITH SECURE DEDUPLICATION

In this section, we describe a concrete construction of anattribute-based storage system supporting secure deduplication, analyze its security.

5.1 Construction

Onthe basis of the large universe CP-ABE scheme proposedin [22], below we present an attribute-based storage systemwith secure deduplication.

• **Setup.** This algorithm takes the security parameter λ as the input. It randomly chooses a group G of a prime order p with a generator g, and a bilinear pairing e^: G × G → G1. Then, it randomly chooses collision resistant hash functions f0: G1 →Zp, f1: M→Z p, F: G1→K, H: G5→Zp. Also, it randomly chooses α€Zp∗, u, h, v, w €G. Thepublic parameter is pars = (f0, f1, F, H, g, u, h, w, v, e^ (g; g)α), and the master private key is msk =gα.

• **KeyGen.** This algorithm takes the public parameter pars, the master private key msk and a set A = {A1… A|a|}of attributes as the input, it randomly chooses r, r1… r|A| €Zp∗, and computes

sk1 = gαwr; sk2 = gr;
¥i € A sk2(i) = gri; sk1(i) = (uAih)riv−r.

It outputs the attribute-based private key skA = (sk1,{sk1(i)}i €A, sk2, {sk2(i)}i € A) associated with a set ofattributes A.

• **Encrypt.** This algorithm takes the public parameter pars, a message M 2 M and an LSSS access structure (M, ρ) where ρ

is a function which associatesthe rows of M to attributes as the input. Let M bean l × n matrix. It randomly chooses a vector v = (µ5, y2… y_n) €Zpn, of which the values will beused to share the encryption exponent µ. For i =1… l, it calculates v_i = v· Mi, where Mi is thevector corresponding to the i^{th} row of the matrixM. In addition, it randomly chooses β€ G_1, z_1… z_l€ Zp, and computes

U = gf(M)µ, L = gf1(M)hf0(β),

E = SE:Enc(F(β), M)

B = gµ; C = β · e^(g; g)αµ,

¥i € [1, l] Ci = wvivzi, Di = gzi, Ei = (uρ(i)h)−zi,

PoK{(M; β): U = Bf(M) ^ L = gf1(M)hf0(β)}

It outputs a trapdoor key skT = wµ, and a tuple oftag, label, ciphertext and proof CT = (T , L, ct, pf)where T = (U, B), ct = (M, ρ), E, B, C, {(Ci,Di, Ei)}i¥[1,l], and pf is a zero-knowledge proof ofknowledge (PoK) for the equality of µ in U, B andf(M) in U, L without leaking the values of µ, M andβ. Here PoK is a zero-knowledge proof composed of (U, B, L, θ1, θ2) and can be computed as follows. Itrandomly chooses d1, d2 €Zp∗, and computes

R1 = Bd1, R2 = gd1hd2, c = H (U, B, L, R1, R2),θ1 = d1 − c · f1(M), θ2 = d2 − c · f0(β).

• **Validity-Test.** This algorithm takes the public parameter pars and a ciphertext CT as the input. To test thevalidity of the ciphertext, it computes R1 = U cBθ1, R2 = Lcgθ1hθ2.

If c = H (U, B, L, R1, R2), it accepts CT, and storesL, ((M, ρ), E, B, C, {Ci, Di, Ei} i€[1, l])) to thepublic cloud. Otherwise, it rejects CT.

• **Equality-Test.** This algorithm takes the public parameter pars and two tags (U1; B1) and (U2; B2) of theoutsourced data as input. It outputs 1 if e^ (U1; B2) =e^ (U2; B1). Otherwise, it outputs 0.

• **Re-encrypt.** This algorithm takes the public parameter pars, a trapdoor key skT , a ciphertext (M, ρ),E, B, C, f(Ci, Di, Ei)g with a label L and anLSSS access structure (M0, ρ0) where the function ρ0associates the rows of M0 to attributes as the input.Let M0 be an l0 ×n0 matrix. It randomly chooses v¯ = (¯ µ, y¯ 2… y¯ n') € Zpn'.it outputs the new ciphertext as

B'= B · gµ¯ , L' = L, E' = E, C' = C · e^ (g, g)αµ¯ ,

Ci' = wM' i'−v' vzi'; Di' = gzi', Ei' = (uρ'(i')h) −zi'.

It is straightforward to see that the distribution of L', ((M', ρ'), E', B', C', fCi', Di', Ei'gi'€ [1,l']) isconsistent with that outputted by the encryption algorithm Encrypt(pars, M, (M', ρ')).

• **Decrypt.** This algorithm takes the public parameter pars, a ciphertext (M, ρ), E, B, C, {Ci, Di, Ei}i€ [1,l] with the corresponding label L and aprivate key skA for an attribute set A as the input.Suppose that an attribute set A satisfies the accessstructure (M, ρ).If gf1(M)hf0(β)= L, it outputs M. Otherwise, it outputs a failure symbol 0.

VI. CONCLUSION

Cloud Computing refers manipulating, configuring, and accessing the applications through online. It offers online data storage, infrastructure and application. Cloud Computing is both a combination of software and hardware based computing resources delivered as a network service.Attribute-based encryption (ABE) has been widely used in cloud computing where data providers outsource their encrypted data to the cloud and can share the data with users possessing specified credentials. On the other hand, deduplication is an important technique to save the storagespace and network bandwidth, which eliminates duplicatecopies of identical data. However, the standard ABE systemsdo not support secure deduplication, which makes themcostly to be applied in some commercial storage services.In this paper, we presented a novel approach to realize anattribute-based storage system supporting secure deduplication. Our storage system is built under a hybrid cloudarchitecture, where a private cloud manipulates the computation and a public cloud manages the storage. The privatecloud is provided with a trapdoor key associated with thecorrespondingciphertext, with which it can transfer theciphertext over one access policy into ciphertexts of the sameplaintext under any other access policies without beingaware of the underlying plaintext. After receiving a storagerequest, the private cloud first checks the validity of theuploaded item through the attached proof. If the proof isvalid, the private cloud runs a tag matching algorithm tosee whether the same data underlying the ciphertexthas been stored. If so, whenever it is necessary, it regeneratestheciphertext into a ciphertext of the same plaintext over anaccess policy which is the union set of both access policies.The proposed storage system enjoys two major advantages.Firstly, it can be used to confidentially share data with otherusers by specifying an access policy rather than sharing thedecryption key. Secondly, it achieves the standard notionof semantic security while existing

deduplicationschemesonly achieve it under a weaker security notion.The comparison between existing and proposed system is only in time complexity. By using CP-ABE algorithm, the secret key is generated and wastes a lots of time. While reducing this time complexity, we use Advanced Encryption Standard (AES), which is encrypt the data by using symmetric key K as key is fastly generates and encrypt the data in file. This will reduce the lots of time.

VII.REFERENCES

[1] S. Bugiel, S. Nurnberger, A. Sadeghi, and T. Schneider, "Twin ¨ clouds: Secure cloud computing with low latency - (full version)," in Communications and Multimedia Security, 12th IFIP TC 6 / TC 11 International Conference, CMS 2011, Ghent, Belgium, October 19- 21,2011. Proceedings, ser. Lecture Notes in Computer Science, vol. 7025 Springer, 2011, pp. 32–44.

[2] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof-systems (extended abstract)," in Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA, ACM, 1985, pp. 291– 304.

[3] M. Fischlin and R. Fischlin, "Efficient non-malleable commitment schemes," in Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA,August 20-24, 2000, Proceedings, ser. Lecture Notes in Computer Science, vol. 1880. Springer, 2000, pp. 413–431.

[4] M. Abadi, D. Boneh, I. Mironov, A. Raghunathan, and G. Segev, "Message-locked encryption for lock-dependent messages," in Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I, ser. Lecture Notes in Computer Science, vol. 8042. Springer,2013, pp. 374–391.

[5] S. Goldwasser and S. Micali, "Probabilistic encryption," J. Comput.Syst Sci., vol. 28, no. 2, pp. 270–299, 1984.

[6] M. Bellare, S. Keelveedhi, and T. Ristenpart, "Message-locked encryption and secure deduplication," in Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings, ser. Lecture Notes in Computer Science, vol. 7881 Springer, 2013, pp. 296–312.

[7] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings, ser. Lecture Notes in Computer Science, vol. 3494. Springer, 2005, pp. 457–473.

[8] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, I October 30 - November 3, 2006, ser. Lecture Notes in Computer Science, vol. 5126. Springer, 2006, pp. 89–98.

[9] R. Ostrovsky, A. Sahai, and B. Waters, "Attribute-based encryption with non-monotonic access structures," in Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007. ACM, 2007, pp. 195–203.

[10] A. B. Lewko and B. Waters, "Unbounded HIBE and attributebased encryption," in Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings, ser. Lecture Notes in Computer Science, vol. 6632 Springer, 2011, pp. 547–567.

[11] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in 2007 IEEE Symposium on Security and Privacy (S&P 2007), 20-23 May 2007, Oakland, California, USA IEEE Computer Society, 2007, pp. 321–334.

[12] L. Cheung and C. C. Newport, "Provably secure ciphertext policy ABE," in Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007. ACM, 2007, pp. 456–465.

[13] Y. Rouselakis and B. Waters, "Practical constructions and new proof methods for large universe attribute-based encryption," in 2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013. ACM, 2013, pp. 463–474.

[14] M. W. Storer, K. M. Greenan, D. D. E. Long, and E. L. Miller, "Secure data deduplication," in Proceedings of the 2008 ACM Workshop On Storage Security And Survivability, StorageSS 2008, Alexandria, VA, USA, October 31, 2008. ACM, 2008, pp. 1–10.

[15] D. Quick, B. Martini, and K. R. Choo, "Cloud Storage Forensics" Syngress Publishing/Elsevier 2014.

[16] K. R. Choo, J. Domingo-Ferrer, and L. Zhang, "Cloud cryptography: Theory, practice and future research directions," Future Generation Comp. Syst., vol. 62, pp. 51-53, 2016.

[17] K. R. Choo, M. Herman, M. Iorga, and B. Martini, "Cloud forensics: State-of-the-art and future directions," Digital Investigation, vol. 18, pp. 77-78, 2016.
[18] Y. Yang, H. Zhu, H. Lu, J. Weng, Y. Zhang, and K. R. Choo, "Cloud based data sharing with fine-grained proxy re-encryption," Pervasive and Mobile Computing, vol. 28, pp. 122–134, 2016.
[19] D. Quick and K. R. Choo, "Google drive: Forensic analysis of data remnants," J. Network and Computer Applications, vol. 40, pp. 179-193, 2014.
[20] B. Zhu, K. Li, and R. H. Patterson, "Avoiding the disk bottleneck in the data domain deduplication file system," in 6th USENIX Conference on File and Storage Technologies, FAST 2008, February 2629, 2008, San Jose, CA, USA, USENIX, 2008, pp. 269-282.
[21] S. Keelveedhi, M. Bellare, and T. Ristenpart, "Dupless: Server aided encryption for deduplicated storage," in Proceedings of the 22th USENIX Security Symposium, Washington, DC, USA, August 14-16, 2013. USENIX Association, 2013, pp. 179-194.
[22] M. Bellareand, S. Keelveedhi, "Interactive message-locked encryption and secure deduplication," in Public-Key Cryptography - PKC 2015 - 18th IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, March 30 - April 1, 2015, Proceedings, ser. Lecture Notes in Computer Science, vol. 9020. Springer, 2015, pp. 516–538.
[23] V. Goyal, A. Jain, O. Pandey, and A. Sahai, "Bounded ciphertext policy attribute based encryption," in Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik,Iceland,July7-11,2008,Proceedings,PartII-TrackB:Logic,Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations, ser. Lecture Notes in Computer Science, vol. 5126, Springer, 2008, pp. 579-591.
[24] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer, "Reclaiming space fromduplicate files in a server less distributed file system," in ICDCS, 2002, pp. 617–624.
[25] P. Anderson and L. Zhang, "Fast and secure laptop backups with encrypted de-duplication," in Uncovering the Secrets of System Administration: Proceedings of the 24th Large Installation System Administration Conference, LISA 2010, San Jose, CA, USA, November 7-12, 2010 USENIX Association, 2010.
[26] A. Rahumed, H. C. H. Chen, Y. Tang, P. P. C. Lee, and J. C. S. Lui, "A secure cloud backup system with assured deletion and version control," in 2011 International Conference on Parallel Processing Workshops, ICPPW 2011, Taipei, Taiwan, Sept. 13-16, 2011. IEEE Computer Society, 2011, pp. 160–167.
[27] P. Puzio, R. Molva, M. Onen, and S. Loureiro, "Cloudedup: Secure deduplication with encrypted data for cloud storage," in IEEE 5th International Conference on Cloud Computing Technology and Science, CloudCom2013, Bristol, United Kingdom, December 2-5,2013,Volume 1. IEEE Computer Society, 2013, pp. 363–370.
[28] J. Stanek, A. Sorniotti, E. Androulaki, and L. Kencl, "A secure data deduplication scheme for cloud storage," in Financial Cryptography and Data Security - 18th International Conference, FC 2014, Christ Church, Barbados, March 3-7, 2014, Revised Selected Papers, ser. Lecture Notes in Computer Science, vol. 8437. Springer, 2014, pp. 99–118.
[29] D. Boneh and M. Franklin, "Identity-based encryption from the weilpairing," in CRYPTO, ser. Lecture Notes in Computer Science, vol. 2139. Springer-Verlag, 2001, pp. 213–219.
[30]E. Fujisaki and T. Okamoto, "Secure integration of asymmetric and symmetric encryption schemes," J. Cryptology, vol. 26, no.1, pp. 80–101, 2013
[31] A. B. Lewko and B. Waters, "Decentralizing attribute-based encryption," in Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings, ser. Lecture Notes in Computer Science, vol. 6632 Springer, 2011, pp. 568–588.
[32] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in Public Key Cryptography - PKC 2011 - 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, March 6-9, 2011. Proceedings, ser. Lecture Notes in Computer Science, vol. 6571 Springer, 2011, pp. 53–70.
[33] A. Beimel, "Secure schemes for secret sharing and key distribution," Ph.D. dissertation, Israel Institute of Technology, Israel Institute of Technology, June 1996.
[34] J. Lai, R. H. Deng, Y. Yang, and J. Weng, "Adaptable ciphertext policy attribute-based encryption," in Pairing-Based Cryptography Pairing 2013 - 6th International Conference, Beijing, China, November 22-24, 2013, Revised Selected Papers, ser. Lecture Notes in Computer Science, vol. 8365. Springer, 2013, pp. 199–214.
[35] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, and A. D. Rubin, "Charm: a framework for rapidly prototyping cryptosystems," J. Cryptographic Engineering, vol. 3, no. 2, pp. 111–128, 2013.