Efficient Access Control Scheme with MAABE and Verification in Cloud Storage

Mr. VelpulaSundaraRatnam[1] & Ms. G V Rama Gayatri[2]
[1]*Associate professor*, [2]*M. Tech*
[12]*Dept. of CSE, Malla Reddy Engineering College for Women, Misammaguda, Dhulapally, Secundrabad-500100*

Abstract- Data access control is a challenging issue in publiccloud storage systems. Ciphertext-policy attribute-based encryption (CP-ABE) has been adopted as a promising technique toprovide flexible, fine-grained, and secure data access control forcloud storage with honest-but-curious cloud servers. However,in the existing CP-ABE schemes, the single attribute authoritymust execute the time-consuming user legitimacy verification andsecret key distribution, and hence, it results in a single-pointperformance bottleneck when a CP-ABE scheme is adopted ina large-scale cloud storage system. Users may be stuck in thewaiting queue for a long period to obtain their secret keys,thereby resulting in low efficiency of the system. Although multi-authority access control schemes have been proposed, theseschemes still cannot overcome the drawbacks of single-pointbottleneck and low efficiency, due to the fact that each of theauthorities still independently manages a disjoint attribute set.In this paper, we propose a novel heterogeneous frameworkto remove the problem of single-point performance bottleneckand provide a more efficient access control scheme with anauditing mechanism. Our framework employs multiple attributeauthorities to share the load of user legitimacy verification.Meanwhile, in our scheme, a central authority is introduced togenerate secret keys for legitimacy verified users. Unlike othermulti-authority access control schemes, each of the authoritiesin our scheme manages the whole attribute set individually. Toenhance security, we also propose an auditing mechanism todetect which attribute authority has incorrectly or maliciouslyperformed the legitimacy verification procedure.

Keywords- Cloud storage, access control, auditing, CP-ABE.

I. INTRODUCTION

To address the issue of data access control in cloud storage,there have been quite a few schemes proposed, among whichCiphertext-Policy Attribute-Based Encryption (CP-ABE) isregarded as one of the most promising techniques. A salientfeature of CP-ABE is that it grants data owners direct controlpower based on access policies, to provide flexible, fine-grained and secure access control for cloud storage systems.In CP-ABE schemes, the access control is achieved by usingcryptography, where an owner's data is encrypted with anaccess structure over attributes, and a user's secret key islabelled with his/her own attributes. Only if the attributesassociated with the user's secret key satisfy the access structure, can the user decrypt the corresponding ciphertext toobtain the plaintext. So far, the CP-ABE based access controlschemes for cloud storage have been developed into twocomplementary categories, namely, single-authority scenario[1]–[2], and multi-authority scenario [3]–[4].

A straightforward idea to remove the single-point bottleneckis to allow multiple authorities to jointly manage the universalattribute set, in such a way that each of them is able todistribute secret keys to users independently. By adoptingmultiple authorities to share the load, the influence of thesingle-point bottleneck can be reduced to a certain extent.However, this solution will bring forth threats on securityissues. Since there are multiple functionally identical authorities performing the same procedure, it is hard to find theresponsible authority if mistakes have been made or maliciousbehaviors have been implemented in the process of secret keygeneration and distribution. For example, an authority mayfalsely distribute secret keys beyond user's legitimate attributeset. Such weak point on security makes this straightforwardidea hard to meet the security requirement of access controlfor public cloud storage. Our recent work, TMACS [5], is athreshold multi-authority CP-ABE access control scheme forpublic cloud storage, where multiple authorities jointly managea uniform attribute set. Actually it addresses the single-pointbottleneck of performance and security, but introduces someadditional overhead. Therefore, in this paper, we present afeasible solution which not only promotes efficiency androbustness, but also guarantees that the new solution is assecure as the original single-authority schemes.

The similar problem has been considered and partly tackled in other related areas, such as public key infrastructure(PKI) for e-commerce [6]. To reduce the certificate authority (CA)'s load, one or more registration authorities (RAs)are introduced to perform some of administration tasks onbehalf of CA. Each RA is able to verify a user's legitimacyand determine whether the user is entitled to have a validcertificate. After the verification, it validates the credentialsand forwards the certificate request to CA. Then, CA willgenerate a certificate for the user. Since the heaviestwork of verification is performed by a selected RA, the loadof CA can be largely reduced. However, the security ofthe scheme with single-CA/multi-RAs partly depends on thetrustiness of multiple

RAs. In order to achieve traceability,CA should store some information to confirm which RA hasbeen responsible for verifying the legitimacy of a specificuser.

In this paper, inspired by the heterogeneous architecturewith single CA and multiple RAs, we propose a robust andauditable access control scheme for publiccloud storage to promote the performance while keepingthe flexibility and fine granularity features of the existingCP-ABE schemes. In our scheme, we separate the procedureof user legitimacy verification from the secret key generation,and assign these two sub-procedures to two different kindsof authorities. There are multiple authorities (named attributeauthorities, AAs), each of which is in charge of the wholeattribute set and can conduct user legitimacy verificationindependently. Meanwhile, there is only one global trustedauthority (referred as Central Authority, CA) in charge ofsecret key generation and distribution. Before performing asecret key generation and distribution process, one of the AAsis selected to verify the legitimacy of the user's attributesand then it generates an intermediate key to send to CA. CAgenerates the secret key for the user on the basis of the receivedintermediate key, with no need of any more verification. Inthis way, multiple AAs can work in parallel to share the loadof the time-consuming legitimacy verification and standbyfor each other so as to remove the single-point bottleneckon performance. Meanwhile, the selected AA doesn't takethe responsibility of generating final secret keys to users.Instead, it generates intermediate keys that associate withusers' attributes and implicitly associate with its own identity,and sends them to CA. With the help of intermediate keys,CA is able to not only generate secret keys for legitimacyverified users more efficiently but also trace an AA's mistakeor malicious behavior to enhance the security.

The main contributions of this work can be summarized asfollows.

A. To address the single-point performance bottleneck of key distribution existed in the existing schemes; we propose a robust and efficient heterogeneous framework with single CA (Central Authority) and multiple AAs (Attribute Authorities) for public cloud storage. The heavy load of user legitimacy verification is shared by multiple AAs, each of which manages the universal attribute set and is able to independently complete the user legitimacy verification, while CA is only responsible for computational tasks. To the best of our knowledge, this is the first work that proposes the heterogeneous access control framework to address the low efficiency and single-point performance bottleneck for cloud storage.
B. We reconstruct the CP-ABE scheme to fit our proposed framework and propose a robust and high-efficient access control scheme, meanwhile the scheme still preserves the fine granularity, flexibility and security features of CP-ABE.
C. Our scheme includes an auditing mechanism that helps the system trace an AA's misbehavior on user's legitimacy verification.

II. RELATED WORK

Recently, we considered the single-point performance bottleneck of CP-ABE based schemes and devised a threshold multi-authority CP-ABE access control scheme in our another work [5]. Different from other multi-authority schemes, in [5], multiple authorities jointly manage a uniformattribute set. Taking advantage of (t, n) threshold secret sharing, the master secret key can be shared among multipleauthorities, and a legal user can generate his/her secret keyby interacting with any t authorities. This scheme actually addressed the single-point bottleneck on both securityand performance in CP-ABE based access control in publiccloud storage. However, it is not efficient, because a user hasto interact with at least t authorities, and thus introduces higherinteraction overhead.

To meet some scenarios where users' attributes comefrom multiple authorities, some multi-authority schemes havebeen proposed. Based on the basic ABE [7] scheme,Chase et al. [8] proposed the first multi-authority schemewhich allows multiple independent authorities to monitor attributes and distribute corresponding secret keys, butinvolves a central authority (CA). Subsequently, some multi-authority ABE schemes without CA have been proposed, suchas [9] and [10]. Since the first construction of CP-ABE [11], agreat many multi-authority schemes have been conducted overCP-ABE. Muller et al. [12] proposed the first multi-authorityCP-ABE scheme in which a user's secret key was issuedby an arbitrary number of attribute authorities and a masterauthority. Then Lewko et al. [13] proposed a decentralized CPABE scheme where the secret keys can be generated fully bymultiple authorities without a central authority.

Ruj et al. [14]applied Lewko's work [13] for access control in cloud storagesystems, and also proposed a revocation method. Lin et al. [10]proposed a decentralized access control scheme based onthreshold mechanism. In [15] and [16], the authors proposedtwo efficient multi-authority CP-ABE schemes for data accesscontrol in cloud storage systems, where a central authorityis only needed in system initialization phase. Based on thebasic multi-authority architecture, some other literatures triedto address the user identity privacy issue [16] policyupdate and the accountability to prevent key abusing. However, in above multi-authority schemes, multiple authorities separately manage disjoint attribute sets. That isto say, for each attribute, only one authority could issue secretkeys associated with it. Therefore, in large-scale systems,the single-point performance bottleneck still exists in multi-authority schemes due to the property that each of the multipleauthorities maintains only a disjoint subset of attributes.

In this paper, we present an efficient heterogeneous framework with single CA/multiple AAs to address the problem ofsingle-point performance bottleneck. The novel idea of ourproposed scheme is that the complicated and time-consuminguser legitimacy verification is executed only once by oneselected AA. Furthermore, an auditing mechanism is proposedto ensure the traceability of malicious AAs. Thus our schemecan not only remove the single-point performance bottleneckbut also be able to provide a robust, high-efficient, and secureaccess control for public cloud storage.

III. SYSTEMIMPLEMENTATION

A. System Architecture:

The system model of our design is shown in Fig. 1, which involves five entities: a central authority (CA), multiple attribute authorities (AAs), many data owners (Owners), many data consumers (Users), and a cloud service provider.

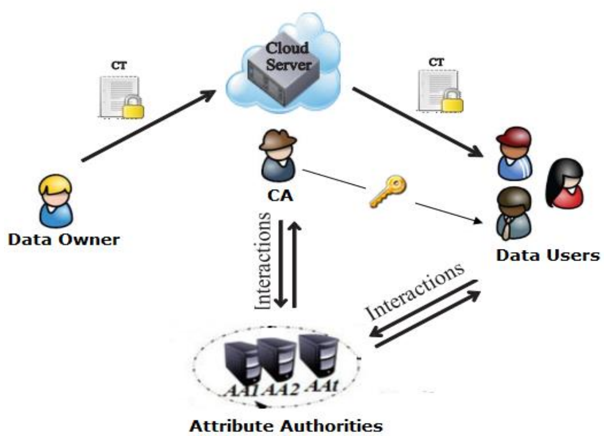

Fig.1: System model.

Central Authority (CA):

The central authority (CA) is the administrator of theentire system. It is responsible for the system constructionby setting up the system parameters and generating publickey for eachattribute of the universal attribute set. In thesystem initialization phase, it assigns each user a uniqueUid and each attribute authority a unique Aid. For akey request from a user, CA is responsible for generatingsecret keys for the user on the basis of the receivedintermediate key associated with the user's legitimateattributes verified by an AA. As an administrator of theentire system, CA has the capacity to trace which AA hasincorrectly or maliciously verified a user and has grantedillegitimate attribute sets.

Attribute Authorities:

The attribute authorities (AAs) are responsible forperforming user legitimacy verification and generatingintermediate keys for legitimacy verified users. Unlikemost of the existing multi-authority schemes where eachAA manages a disjoint attribute set respectively, ourproposed scheme involves multiple authorities to sharethe responsibility of user legitimacy verification and eachAA can perform this process for any user independently.When an AA is selected, it will verify the users' legitimateattributes by manual labor or authentication protocols,and generate an intermediate key associated with theattributes that it has legitimacy-verified. Intermediate keyis a new concept to assist CA to generate keys.

Data Owner:

The data owner (Owner) defines the access policy aboutwho can get access to each file, and encrypts the fileunder the defined policy. First of all, each owner encryptshis/her data with a symmetric encryption algorithm. Then,the owner formulates access policy over an attributeset and encrypts the symmetric key under the policyaccording to public keys obtained from CA. After that, theowner sends the whole encrypted data and the encrypted symmetric key (denoted as ciphertextCT) to the cloudserver to be stored in the cloud.

Data Consumer:

The data consumer (User) is assigned a global useridentity Uid by CA. The user possesses a set of attributesand is equipped with a secret key associated with his/herattribute set. The user can freely get any interestedencrypted data from the cloud server. However, the usercan decrypt the encrypted data if and only if his/herattribute set satisfies the access policy embedded in theencrypted data.

Cloud Server:

The cloud server provides a public platform for owners to store and share their encrypted data. The cloudserver doesn't conduct data access control for owners.The encrypted data stored in the cloud server can bedownloaded freely by any user.

IV. OUR PROPOSED SCHEME

To achieve a robust and efficient access control for publiccloud storage, we propose a hierarchical framework withsingle CA and multiple AAs to remove the problem of singlepoint performance bottleneck and enhance the system efficiency. In our proposed scheme, the procedure ofkey generation is divided into two sub-procedures: 1) theprocedure of user legitimacy verification; 2) the procedureof secret key generation and distribution. The user legitimacyverification is assigned to multiple AAs, each of which takesresponsibility for the universal attribute set and is able to verifyall of the user's attributes independently. After the successfulverification, this AA will generate an intermediate key andsend it to CA. The procedure of secret key generation anddistribution is executed by the CA that generates the secretkey associated with user's attribute set without any moreverification. The secret key is generated using the intermediatekey securely transmitted from an AA and the master secret key.

In our one-CA/multiple-AAs construction, CA participatesin the key generation and distribution for security reasons:To enhance auditability of corrupted AAs, one AA cannotobtain the system's master secret key in case it can optionallygenerate secret keys without any supervision. Meanwhile, theintroduction of CA for key generation and distribution isacceptable, since for a large-scale system, the most timeconsuming workload of legitimacy verification is offloadedand shared among the multiple AAs, and the computationworkload for key generation is very light. The procedure ofkey generation and distribution would be more efficient thanother existing schemes.

This section first gives an overview of our proposed scheme,and then describes the scheme in detail. Our scheme consistsof five phases, namely System Initialization, Encryption, KeyGeneration, Decryption, and Auditing & Tracing.

1) System Initialization:

Firstly, CA chooses two multiplicative cyclic groups G(the parameter g is a generator of G) andGT with the same prime order p, and defines a binary mape : G × G → GT on G. CA randomly chooses α, β, a andb ∈ Zp as the master secret key. CA also randomly generates public keys for each attribute Atti, (i = 1, 2, . . . , U):h1, h2, . . . , hU∈ G. Besides, let H : (0, 1)∗ → Zp be a hashfunction. The published public key is:

P K = GT, G, H, g, ga, e(g, g)α, h1, . . . , hU

and the master secret key is:M SK = α, β, a, b

This implicitly exists in the system, and doesn't need to be obtained by any other entity.Another task for CA in this operation is handling AAs'and users' registration. Here, CA generates a pair of keys(skCA, vkCA) to sign and verify, in which, vkCA is publiclyknown by each entity in the system. Each AA sends aregistration request to CA during the System Initialization.For each legal AA, CA assigns a unique identity Aid ∈Zp,randomly chooses a private key k Aid ∈Zp, and computes itscorresponding public key P K Aid = gkAid . Furthermore, CAgenerates a certificate CertAid which includes the public keyPK Aid, and sends it with the corresponding private key k Aidto the AA with the identity Aid. Meanwhile, each user getshis/her Uid, private key kUid and CertUid from CA.

2) Encryption:

The procedure of Encryption is performed by the data owner himself/herself. To improve the system's performance, the owner first chooses a random number κ ∈GT as the symmetric key and encrypts the plaintext message M using κ with the symmetric encryption algorithm. The encrypted data can be denoted as Eκ(M). Then the owner encrypts the symmetric key κ using CP-ABE under the access policy A defined by himself/herself. The owner defines an easy expressed monotonic boolean formula. Following the method defined in [41], the owner can turn it to an LSSS access structure, which can be denoted as (M, ρ). Here, M is an l × n matrix, where l is the scale of a specific attribute set associated with a specific access policy and n is a variable that is dependent on the monotonic Boolean formula definition and the LSSS turning method. The function ρ maps each row of M to a specific attribute, marked as ρ(i) ∈ {Att1, Att2, . . . , AttU }. A random secret parameter s is chosen to encrypt the symmetric key κ. To hide the parameter s, a random vector v= (s, y2, y3, . . . ,yn) ∈Znp is selected, where y2, y3, . . . , yn are randomly chosen and used to share the parameter s. Each λi = Mivis computed for i = 1, 2, . . . ,l, where Mi denotes the i-th row of the matrix M. Owner randomly selects r1,r2, . . . ,rl∈Zp and uses the public key generated by CA to compute:

(C = κe(g, g)αs, C = gs,∀i = 1 to l, Ci = (ga)λi · hρ(i)−ri , Di = gri).

3) Key Generation and Distribution:

This procedure istotally different from those existing CP-ABE schemes.It involves the given user, a selected AA and CA. We dividethe procedure into the following 4 steps.

STEP 1: U j → AAi. When a user U j with the identityUid j makes a secret key request, the user selects an AA (AAiwith the identity Aidi) by a certain scheduling algorithmand sends the ertUid to show the validity of his/heridentity, along with some proofs to show that he/she hasthe attribute set that he/she claims to have.

STEP 2: AAi → CA. The user legitimacy verificationprocess may involve manual labor or verification protocols performed by AAi. After successful verification, AAiobtains the current timestamp value T S, computes t1 =H (Uid j||T S||0) and t2 = H (Uid j||T S||1), and generatesan intermediate key ICAidi ,Uid j as follows:

ICAidi ,Uid j = {Kx = hkxAidi t1, Jx = htx2}∀x∈S j

Where S j is the verified legitimate attribute set for the userwith the identity Uid j. Finally this AA securely sends thefollowing message to CA:

{Uidj ,Aidi, S j , ICAidi ,Uid j , T S}

STEP 3&STEP 4: C A → AAi → U j . After receivingthe message from the AA, CA first uses Aidi to obtain thecorresponding stored public key PKAidi. Then CA checks whether the transmission delay is within the allowed time interval T. We assume that the current time is T. IfT − T S >T, CA stops here and sends RE J to the AA.Otherwise, CA continues to compute t1 = H (Uid j||T S||0),t2 = H (Uid j||T S||1), and makes sure t1 and t2 haven't yetbeen re-used from the same user. This can prevent AA'scollusion attack (We will discuss the collusion attack inSection VI.). CA continues to use its master secret keyM SK to generate a secret key SK j for the user.

4) Decryption: The procedure of Decryption is performedby the user. A user can freely query and download any interested encrypted data from the public cloud storage. However,he/she cannot decrypt data unless his/her attribute set satisfiesthe access structure embedded in the ciphertext.

5) Auditing & Tracing: Each AA may generate an intermediate key for any attribute set associated with a specific user,and then CA can generate the secret key for this user withoutany more verification. However, AAs can be compromisedand cannot be fully trusted. Meanwhile, the user legitimacyverification is conducted by manual labor, and therefore AAsmay maliciously or incorrectly generate an intermediate keyfor an unverified attribute set. A malicious user will try anypossible means to gain the secret key associated with thespecific attribute set to obtain the data access permission. Under this assumption, the user would often show abnormalbehaviors. Usually, we need to hold the accountability ofAAs to prevent the compromised or misbehaved ones fromfreely generating secret keys for malicious users.

The procedure of Auditing & Tracing is periodically performed or event-triggered by CA to mandatorily ask a suspected user to securely submit Kx of a given attribute, L and TS in his/her gained secret key. In order to continue to obtaindata, users have to cooperate to perform the process correctly.However, in order to deceive CA, a suspected user still hasthe motivation to submit a secret key component that doesn'tbelong to him/her. Thus, to implement an effective tracing, CAmust confirm the received secret key components really belongto the given user. Based on the reasons mentioned above, thetracing method should be executed as the following two subprocedures.

Secret key ownership confirming. This procedure is executed to confirm that the received secret key componentreally belongs to the user who has submitted it. We assumethat the user is U j with the identity Uidj . CA randomlyselects a suspected attribute x in S j , and asks U j to securelysubmit his/her secret key components Kx, L and T S. ThenCA computes t1 = H (Uid j||T S||0), t2 = H (Uid j||T S||1)and Kx = hαx t2 · g−b(t1+t2), and confirms whether thefollowing equation holds:

e(hx, L) = e(g, KxKx).

If it holds, CA will further continue to execute the nextsub-procedure. Otherwise, it indicates that the suspecteduser doesn't correctly submit his/her own secret keycomponents and the user will receive a severe punishment,such as kicking the user out of the system.

AA Tracing. This procedure is executed to trace andconfirm which AA has generated the suspected user's secretkey. CA takes its master secret key M SK to recover thepublic key associated with a specific AA as follows:

P K = (L · g−αt2)1/βt1 = gkAidi βt1/βt1 = gkAidi .

CA uses PK as an index to search its storage for the responsible AA. If some AA with the identity Aidi owns a public key that is equal to PK, it means that AA has maliciouslyor incorrectly verified the legitimacy of this user. The foundAA should implement security enhancement or be kickedout of the system as a severe punishment.

V. EXPERIMENT RESULTS

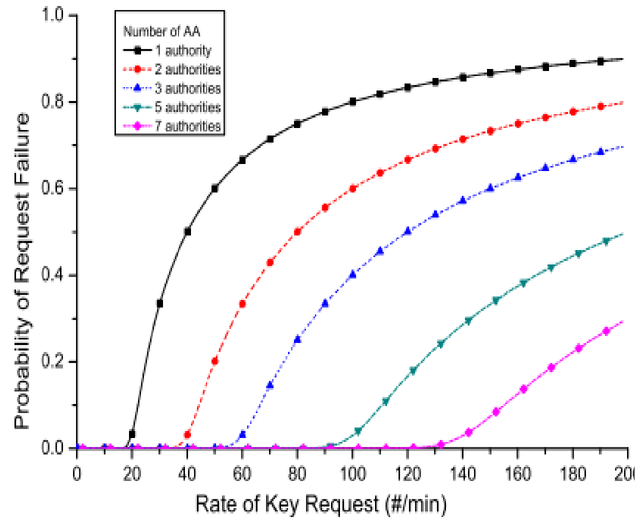

Fig.2: The failure rate in RAAC with µ1 = 20/min, µ2 = 200/min andK = 30.

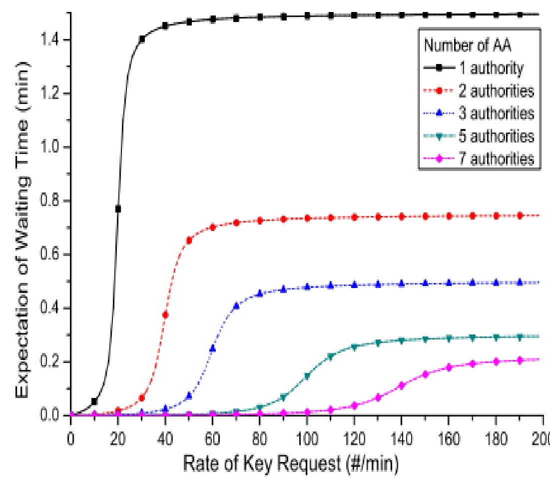

Fig.3: The average waiting time in RAACith µ1 = 20/min, µ2 =200/min and K = 30.

Whereas, with 7 AAs, the average waiting time is about15s. Moreover, from Fig. 2, with the arrival rate less than150, the failure rate is less than 5%. Although using moreworking AAs brings larger configuration cost, by combiningthe failure rate and the average waiting time, we can assurethat the configuration of multiple AAs can provide secretkey generation service with high quality as well as lowcost.

Fig. 6 shows the average waiting time versus the arrivalrate and the number of AAs when µ1 = 20/min, µ2 =200/min, K =

30. From the figure, we can see that theaverage waiting time increases rapidly with the increase ofarrival rate when the arrival rates are low. But later the averagewaiting time will become steady because newly arrival userswill be rejected by the system due to the limit length ofwaiting queue. More specifically, with single AA, the averagewaiting time increases rapidly and reaches 1.5 min, which isunbearable.

VI. CONCLUSION

In this paper, we proposed a new frameworkto eliminate the single-point performance bottleneckof the existing CP-ABE schemes. By effectively reformulatingCP-ABE cryptographic technique into our novel framework,our proposed scheme provides a fine-grained, robust and efficient access control with one-CA/multi-AAs for public cloudstorage. Our scheme employs multiple AAs to share the loadof the time-consuming legitimacy verification and standby forserving new arrivals of users' requests.We also proposed an auditing method to trace an attributeauthority's potential misbehavior. We conducted detailed security and performance analysis to verify that our scheme issecure and efficient. The security analysis shows that ourscheme could effectively resist to individual and colludedmalicious users, as well as the honest-but-curious cloudservers. Besides, with the proposed auditing & tracing scheme,no AA could deny its misbehaved key distribution. Furtherperformance analysis based on queuing theory showed thesuperiority of our scheme over the traditional CP-ABE basedaccess control schemes for public cloud storage.

VII. REFERENCES

[1]. Y. Wu, Z. Wei, and R. H. Deng, "Attribute-based access to scalablemedia in cloud-assisted content sharing networks," IEEE Trans. Multimedia, vol. 15, no. 4, pp. 778–788, Jun. 2013.

[2]. Y. Xue, J. Hong, W. Li, K. Xue, and P. Hong, "LABAC: A locationaware attribute-based access control scheme for cloud storage," in Proc.IEEE Global Commun. Conf. (GLOBECOM), Dec. 2016, pp. 1–6.

[3]. A. Lewko and B. Waters, "Decentralizing attribute-based encryption,"in Advances in Cryptology—EUROCRYPT. Berlin, Germany: Springer,2011, pp. 568–588.

[4]. J. Chen and H. Ma, "Efficient decentralized attribute-based accesscontrol for cloud storage with user revocation," in Proc. IEEE Int. Conf.Commun. (ICC), Jun. 2014, pp. 3782–3787.

[5]. W. Li, K. Xue, Y. Xue, and J. Hong, "TMACS: A robust and verifiablethreshold multi-authority access control system in public cloud storage,"IEEE Trans. Parallel Distrib. Syst., vol. 27, no. 5, pp. 1484–1496,May 2016.

[6]. S. Chokhani, W. Ford, R. Sabett, C. Merrill, and S. Wu, Internet x.509Public Key Infrastructure Certificate Policy and Certification PracticesFramework, document IETF RFC, RFC3647, 2003.

[7]. A. Sahai and B. Waters, "Fuzzy identity-based encryption," inAdvances in Cryptology—EUROCRYPT. Berlin, Germany: Springer,2005, pp. 457–473.

[8]. M. Chase, "Multi-authority attribute based encryption," in Proc. 4thTheory Cryptogr. Conf. (TCC), 2007, pp. 515–534.

[9]. M. Chase and S. S. M. Chow, "Improving privacy and security in multiauthority attribute-based encryption," in Proc. 16th ACM Conf. Comput.Commun.Secur.(CCS), 2009, pp. 121 130.

[10]. H. Lin, Z. Cao, X. Liang, and J. Shao, "Secure threshold multi authorityattribute based encryption without a central authority," Inf. Sci., vol. 180,no. 13, pp. 2618–2632, Jul. 2010.

[11]. J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attributebased encryption," in Proc. IEEE Symp.Secur. Privacy (S&P),May 2007, pp. 321–334.

[12]. S. Müller, S. Katzenbeisser, and C. Eckert, "Distributed attribute-basedencryption," in Information Security and Cryptology—ICISC. Berlin,Germany: Springer, 2009, pp. 20–36.

[13]. A. Lewko and B. Waters, "Decentralizing attribute-based encryption,"in Advances in Cryptology—EUROCRYPT. Berlin, Germany: Springer,2011, pp. 568–588.

[14]. S. Ruj, A. Nayak, and I. Stojmenovic, "DACC: Distributed accesscontrol in clouds," in Proc. 10th Int. Conf. Trust, Secur. Privacy Comput.Commun. (TrustCom), Nov. 2011, pp. 91–98.

[15]. K. Yang, X. Jia, K. Ren, and B. Zhang, "DAC-MACS: Effective dataaccess control for multi-authority cloud storage systems," in Proc. IEEEConf. Comput. Commun. (INFOCOM), Apr. 2013, pp. 2895–2903.

[16]. J. Han, W. Susilo, Y. Mu, J. Zhou, and M. H. A. Au, "Improving privacy and security in decentralized ciphertext-policy attributebased encryption," IEEE Trans. Inf. Forensics Security, vol. 10, no. 3,pp. 665–678, Mar. 2015.

Authors Profile

Mr. VelpulaSundaraRatnam Obtained his B.Tech from JNTUCEH, Kukatpally and received M-Tech in the stream of Computer Science and Engineering at JNTUCEH, Kukatpally, He has 13 years of Experience teaching for both under graduate and post graduate students .He published 11 technical papers in both the National and International Journals and conferences .He is now working as Associate professor at MallaReddy Engineering College for women JNTUH.

Ms. G V Rama Gayatri obtained her B Tech from Malla Reddy College of Engineering for Women JNTUH, with First Class Distinction. She is currently pursuing M. Tech from Malla Reddy Engineering College for Women JNTUH.