

# Implementation of MA-ABE for Better Data Security in Cloud

<sup>1</sup>Nallamalli Ramesh, <sup>2</sup>Gorantla Praveen, <sup>3</sup>V.P. Krishna Anne, <sup>4</sup>Dr. Rajasekhara Rao Kurra

<sup>1,2,3,4</sup>Dept. of CSE, KL University, Green Fields, Vaddeswaram, Guntur, AP, India

## Abstract

The more sensitive data is shared and stored by the third party references on the Internet, the need of encrypting data stored at these sites. The one drawback of encrypting data, is that it can be selectively shared at a coarse-grained level (i.e; issuing another party our private key), Here we develop a new cryptosystem concept for fine grained sharing of encrypted data that we call multi authority attribute based encryption MA-ABE. The scheme proposed allows any polynomial number of independent authorities to monitor attributes and distribute secret keys. The encryption choose, each authority, a number  $dk$  and a set of attributes, can encrypt a message such that a user can only decrypt if he had at least  $dk$  of the given attributes from each authority of  $k$ . The scheme proposed can tolerate an arbitrary member of corrupt authorities. Here, the focus is done and showed how to apply the techniques to achieve a multi-authority version of the small system.

## Key words

Sensitive Data, Encrypting Data, Fine Grained, Sharing, Tolerate Multi Authority.

## 1. Introduction

Cloud computing is a promising computing paradigm which recently has drawn extensive attention from both academia and industry. By combining a set of existing and new techniques from research areas such as Service-Oriented Architectures (SOA) and virtualization, cloud computing is regarded as such a computing paradigm in which resources in the computing infrastructure are provided as services over the Internet. Along with this new paradigm, various business models are developed, which can be described by terminology of "X as a Service (XaaS)" [1] where X could be software, hardware, data storage, and etc. Successful examples are Amazon's EC2 and S3 [2], Google App Engine [3], and Microsoft Azure [4] which provide users with scalable resources in the pay-as-you- use fashion at relatively low prices.

For example, Amazon's S3 data storage service just charges \$0.12 to \$0.15 per gigabyte- month. As compared to building their own infrastructures, users are able to save their investments significantly by migrating businesses into the cloud. With the increasing development of cloud computing technologies, it is not hard to imagine that in the near future more and more businesses will be moved into the cloud. As promising as it is, cloud computing is also facing many challenges that, if not well resolved, may impede its fast growth. Data security, as it exists in many other applications, is among these challenges that would raise great concerns from users when they store sensitive information on cloud servers. These concerns originate from the fact that cloud servers are usually operated by commercial providers which are very likely to be outside of the trusted domain of the users. Data confidential against cloud servers is hence frequently desired when users outsource data for storage in the cloud. In some practical application systems, data confidentiality is not only a security/privacy issue, but also of juristic concerns. For example, in healthcare application scenarios use and disclosure of Protected

Health Information (PHI) should meet the requirements of Health Insurance Portability and Accountability Act (HIPAA) [5], and keeping user data confidential against the storage servers is not just an option, but a requirement. Furthermore, we observe that there are also cases in which cloud users themselves are content providers. They publish data on cloud servers for sharing and need fine-grained data access control in terms of which user (data consumer) has the access privilege to which types of data. In the healthcare case, for example, a medical center would be the data owner who stores millions of healthcare records in the cloud. It would allow data consumers such as doctors, patients, researchers, medical-practitioners and etc, to access various types of healthcare records under policies admitted by HIPAA.

To enforce these access policies, the data owners on one hand would like to take advantage of the abundant resources that the cloud provides for efficiency and economy; on the other hand, they may want to keep the data contents confidential against cloud servers. As a significant research area for system protection, data access control has been evolving in the past thirty years and various techniques [6-9] have been developed to effectively implement fine-grained access control, which allows flexibility in specifying differential access rights of individual users. Traditional access control architectures usually assume the data owner and the servers storing the data are in the same trusted domain, where the servers are fully entrusted as an omniscient reference monitor [10] responsible for defining and enforcing access control policies. This assumption however no longer holds in cloud computing since the data owner and cloud servers are very likely to be in two different domains. On one hand, cloud servers are not entitled to access the outsourced data content for data confidentiality; on the other hand, the data resources are not physically under the full control of the owner. For the purpose of helping the data owner enjoy fine-grained access control of data stored on untrusted cloud servers, a feasible solution would be encrypting data through certain cryptographic primitive (s), and disclosing decryption keys only to authorized users. Unauthorized users, including cloud servers, are not able to decrypt since they do not have the data decryption keys. This general method actually has been widely adopted by existing works [11-14] which aim at securing data storage on untrusted servers. One critical issue with this branch of approaches is how to achieve the desired security goals without introducing a high complexity on key management and data encryption. These existing works, as we will discuss in section V-C, resolve this issue either by introducing a per file Access Control List (ACL) for fine-grained access control, or by categorizing files into several file groups for efficiency. As the system scales, however, the complexity of the ACL-based scheme would be proportional to the number of users in the system. The file based group scheme, on the other hand, is just able to provide coarse-grained data access control. It actually still remains open to simultaneously achieve the goals of fine-grainedness, scalability, and data confidentiality for data access control in cloud computing. In this paper, we address this open issue and propose a secure and scalable fine-grained data access control scheme for cloud computing. Our proposed scheme is partially based on our

observation that, in practical application scenarios each data file can be associated with a set of attributes which are meaningful in the context of interest. The access structure of each user can thus be defined as a unique logical expression over these attributes to reflect the scope of data files that the user is allowed to access. As the logical expression can represent any desired data file set, fine-grainedness of data access control is achieved. To enforce these access structures, prior to this they define a public key component (KP-ABE) for each attribute and here we propose a MA-ABE. User secret keys are defined to reflect their access structures so that a user is able to decrypt a ciphertext if and only if the data file attributes satisfy his access structure. Such a design also brings about the efficiency benefit, as compared to previous works, in that, 1) the complexity of encryption is just related to the number of attributes associated to the data file, and is independent to the number of users in the system; and 2) data file creation/deletion and new user grant operations just affect current file/user without involving system-wide data file update or re-keying. One extremely challenging issue with this design is the implementation of user revocation, which would inevitably require re-encryption of data files accessible to the leaving user, and may need update of secret keys for all the remaining users. If all these tasks are performed by the data owner themselves, it would introduce a heavy computation overhead on them and may also require the data owner to be always online. To resolve this challenging issue, our proposed scheme enables the data owner to delegate tasks of data file re-encryption and user secret key update to cloud servers without disclosing data contents or user access privilege information. We achieve our design goals by exploiting a novel cryptographic primitive, namely Multi Authority Attribute Based Encryption without a Central Authority (MA-ABE), and uniquely combine it with the technique of Proxy Re-Encryption (PRE) [16] and lazy re-encryption [11].

Main contributions of this paper can be summarized as follows.

1. To the best of our knowledge, this paper is the first that simultaneously achieves fine-grainedness, scalability and data confidentiality for data access control in cloud computing;
2. Our proposed scheme enables the data owner to delegate most of computation intensive tasks to cloud servers without disclosing data contents or user access privilege information;
3. The proposed scheme is provably secure under the standard security model. In addition, our proposed scheme is able to support user accountability with minor extension.

The rest of this paper is organized as follows. Section II discusses models and assumptions. Section III Previous Work. Section IV presents our Proposed Work. In section V, we analyze our proposed scheme in terms of its security and performance. We conclude this paper in Section VI.

## II. Models And Assumptions

### A. System Models

Similar to [17], we assume that the system is composed of the following parties: the Data Owner, various Data Consumers, various Cloud Servers, and a Third Party Auditor if necessary. To access data files shared by the data owner, Data Consumers, or users for brevity, download data files of their interest from Cloud Servers and then decrypt. Neither the data owner nor users will be always online. They come online just on the necessity basis. For simplicity, we assume that the only access privilege for users is data file reading. Extending our proposed scheme to support data file writing is trivial by asking the data writer to sign the new data

file on each update as [12] does. From now on, we will also call data files by files for brevity. Cloud Servers are always online and operated by the Cloud Service Provider (CSP). They are assumed to have abundant storage capacity and computation power. The Third Party Auditor is also an online party which is used for auditing every file access event. In addition, we also assume that the data owner can not only store data files but also run his own code on Cloud Servers to manage his data files. This assumption coincides with the unified ontology of cloud computing which is recently proposed by Youseff et al. [18].

### B. Security Models

In this work, we just consider Honest but Curious Cloud Servers as [14] does. That is to say, Cloud Servers will follow our proposed protocol in general, but try to find out as much secret information as possible based on their inputs. More specifically, we assume Cloud Servers are more interested in file contents and user access privilege information than other secret information. Cloud Servers might collude with a small number of malicious users for the purpose of harvesting file contents when it is highly beneficial. Communication channel between the data owner/users and Cloud Servers are assumed to be secured under existing security protocols such as SSL. Users would try to access files either within or outside the scope of their access privileges. To achieve this goal, unauthorized users may work independently or cooperatively. In addition, each party is preloaded with a public/private key pair and the public key can be easily obtained by other parties when necessary.

### C. Design Goals

Our main design goal is to help the data owner achieve fine-grained access control on files stored by Cloud Servers. Specifically, we want to enable the data owner to enforce a unique access structure on each user, which precisely designates the set of files that the user is allowed to access we also want to prevent Cloud Servers from being able to learn both the data file contents and user access privilege information. In addition, the proposed scheme should be able to achieve security goals like user accountability and support basic operations such as user grant/revocation as a general one-to-many communication system would require. All these design goals should be achieved efficiently in the sense that the system is scalable.

## III. Previous Work

To our best knowledge, there is only one existing multi authority attribute based encryption scheme proposed by Chase [17]. Her scheme allows any polynomial number of independent authorities to monitor attributes and distribute secret keys. An encryption chooses, for each authority, a number and a set of attributes. He can then encrypt a message such that a user could only decrypt if he has at least of the given attributes for each authority  $k$ . Chase's scheme mainly employs the following two techniques: The first is to require that every user has a Global Identifier (GID), and the second is to use a fully trusted central authority.

The GID is used to prevent a collision attack between different users. More specifically, a user who only has enough secret keys from a certain set of authorities might collude with another user who only has enough keys from the rest authorities to decrypt a ciphertext. The GID could be considered as a randomness embedding to each user's secret keys. This technique adaptation is done in this paper. The secret keys for each user could be considered as an evaluation of a function on GID in Chase's

scheme. This evaluation is ephemeral and decoupled from the master secret key but the ability to decrypt is required to depend on the users' attribute rather than their individual GID. The central authority is the second tool used in Chase's scheme in order to guarantee the above property. The central authority is responsible to provide a final secret key to integrate the secret keys from the other attribute authorities such that the decryption process would be independent of GID. The central authority would be able to decrypt all the cipher text in the Chase's system because it masters the system secret key. In other words, now that the central authority has to be fully trusted, the trust is not totally distributed through all the authorities. The security of the whole system would be totally broken if the central authority is corrupted. Furthermore, it would also increase the computation and communication cost to run and maintain such a fully trusted authority in the system. There is a possible attack, other than the collusion attack, which is ignored in Chase's paper. Any qualified user couldn't be able to decrypt the ciphertext in their system even if there exists only one authority who doesn't distribute the correct secret keys. This attack is possible because all the attribute authorities could be corrupted in any way in the system. There are some other ABE schemes proposed recently, such as [1, 3, 8, 14-13]. The first key policy attribute based encryption scheme (KP-ABE) is proposed in [8]. Both ABE schemes described in this paper are KP-ABE.

#### IV. Proposed Work

This paper focuses on removing the central authority from multi authority ABE scheme. It's difficult to remove the central authority while preventing the collusion attack and keeping the decryption process independent of the identifier of each user. As indicated in the above, it's the central authority which is responsible to integrate the secret keys from the other attribute authorities in Chase's scheme, and thus to integrate these secret keys without the central authority would be an obstacle in our system. Another difficulty is that the integration must be accompanied with the last decryption step as Chase's scheme did. The integration aims to emancipate the users from the restriction of individual identifier, which means this integration shouldn't be completed before the final decryption step because the collusion attack might be possible if the users are free from the bondage of GID.

In this paper, we replace the pseudo random function used in Chase's scheme by a polynomial. After that, we apply the key distribution technique and the joint zero secret sharing technique [13] to the proposed scheme. We'll show that all the above difficulties could be overcome by these simple modifications and the first secure threshold multi authority fuzzy identity based encryption scheme without a central authority could be constructed.

In the proposed scheme, an encrypter can encrypt a message such that a user could only decrypt if he has at least of the given attributes about the message for at least  $t + 1$ ,  $t \leq n/2$  honest authorities in the proposed scheme. All the authorities only need to communicate with each other without revealing any private information to the others during the initialization (or the periodical update step when it comes to the proactive multi authority attribute scheme), and they could operate independently and keep autonomous in the left steps of the scheme. In other words, the whole relatively costly process is transparent to the users, and won't cause any additional inconvenience to them. The trust is truly distributed between each authority in the proposed system, and the expense due to the central authority also disappears since the central authority is removed. The security

of the proposed scheme could be reduced to the secrecy of the underlying Distributed Key distribution protocol (DKG) [13] and Joint Zero Secret Sharing protocol (JZSS) and standard Decisional BDH assumption.

Our adversary is allowed to distribute incorrect secret keys deliberately. This fault behavior is possible since the trust laid upon each authority is now reduced and each one could be corrupted in any way. Our proposed system always contains at least  $t + 1$  honest authorities who would distribute correct secret keys (due to the restriction for the  $(t - 1, n)$ -threshold adversary, here  $t \leq n/2$ ), and those who obtain enough keys from these honest authorities could still decrypt the respective ciphertext. In other words, the faulty behavior of authorities wouldn't disrupt our proposed system.

#### V. Technical Preliminaries

##### A. Threshold MA-ABE without a central authority

Since the MA-ABE scheme is actually the generalization of the MA-FIBE scheme, we'll show how to employ the method introduced in [14] to convert the proposed MA-FIBE scheme into MA-ABE schemes without a central authority in this section. Basically, there are two ways which would realize this kind of conversion since there are actually two ABE schemes introduced in [14]. Both of them could be applied to the proposed MA-FIBE scheme. The major difference between the proposed MA-FIBE and MA-ABE schemes lies in the Secret Key Distribution (SKD) algorithm and the other rest algorithms. Two MA-ABE schemes without a central authority are given in this section. The difference between these two schemes is the size of their public parameters. The first scheme corresponds to the construction for access trees in [14] which we denote as construction for small universe here.

The first MA-ABE scheme is shown as below:

The proposed multi authority system consists of  $n_k$  attribute authorities. Each attribute authority monitors a set of attributes in the universe of attributes. The universe consists of attributes where  $= \sum_{k=1}^n n_k$ .

The proposed multi authority scheme consists of the following algorithms:

##### 1. Setup

A randomized algorithm takes as input the security parameter  $\kappa$  and outputs all the system public parameters PK and the secret key SK for attribute authorities.

SKD (SK, GID,  $\square_u$ ): The selection of GID is the same to that of MA-FIBE. For each GID, set MK as  $MK = y_{k,u} = a_{k,0} + GID + a_{k,m} GID^m$ .

Employ the Key Generation  $(\tau_k, MK)$  algorithm [22]. For each node  $x$  in the tree, set the degree of the polynomial to be one less than the threshold value  $k_x$  of that node, that is,  $d_x = k_x - 1$ . Choose a polynomial  $q_x(x)$  for the root node of the access tree  $\tau_k$  of authority  $AA_k$  such that  $(0) = MK$ , then select other points of the polynomial  $q_x$  randomly to define it completely.

For any other node  $x$ ,  $q_x$  set  $(0) = q_{\text{parent}}(x)(\text{index}(x))$  and choose other points randomly to completely define  $q_x$ . Once the polynomials have been decided for the leaf node  $x$ , the authority could distribute the secret key for each user which is  $D_{k,x} = \{g_2^{q_x(0)/t_{x,j}}\}_{j \in \{1, \dots, k\}}$ , here,  $j$  denotes the respective attribute for the leaf  $x$ . The secret keys  $D_u$  for  $u$  are  $\{D_{k,x}\}_{k \in \{1, \dots, n\}}$ .

## 2. Encryption

(ENC)( $\theta_u, M, PK$ ): choose a random value  $s \in \mathbb{Z}_q$ . for the attribute set  $\theta_c = \{\theta_c^k\}, k \in \{1, \dots, n\}$ , generate the ciphertext

$$C = \{\theta_c, E = e(g_1, g_2)^s \cdot M || 0^l, \{E_{k,j} = T_{k,j}^s\}_{j \in \theta_c^k, \forall k}\}$$

( $k$  corresponds to the authority  $AA_k$ . In order to check whether a decryption is valid, prior to encryption, we append  $M$   $0_s$  trailing  $0^l$  denoted message  $M \in \{0,1\}^l$ ).

## 3. Decryption (DEC)( $C, \theta_u$ )

(i) For  $AA_k, k = 1, 2, \dots, t+1$  run the following two steps.

(a). For each attributes  $j \in \square_c^k \cap \square_c^k$  compute  $e(E_{k,j}, D_{k,x}) =$

$$e(g_1^{t_{k,j}s}, g_2^{q_x(\theta_j)/t_{x,j}}) = e(g_1, g_2)^{q_x(\theta_j)s}.$$

(b). The left steps run the first Decryption algorithm in [14], i.e. run DecryptNode algorithm on the root of the access tree  $T_k$  (as shown in appendix E). At each level of the tree, if the decryptor has successfully computed Decrypt Node for sufficient children, then it combines the results to obtain  $e(g_1, g_2)^{q_x(\theta_j)s}$  for the parent node  $x$ . Finally, if the decryptor has the attributes set which satisfies the tree  $T_k$ , then it can compute.

$$e(g_1, g_2)^{q_x(\theta_j)s} = e(g_1, g_2)^{y_{x,u}s} = e(g_1, g_2)^{(a_{k,0} + a_{k,1}GID + \dots + a_{k,m}GID^m)s}$$

(ii).

$$\pi_{k=1}^{t+1} e(g_1, g_2)^{y_{k,u}s}$$

$$= e(g_1, g_2)^{\sum_{k=1}^{t+1} (\gamma_k a_{k,0} + \gamma_k a_{k,1}GID + \dots + \gamma_k a_{k,m}GID^m)s}$$

$$= e(g_1, g_2)^{a_0s}$$

$$= e(g_1, g_2)^s$$

(iii).  $E / e(g_1, g_2)^s = M || 0^l$ .

Where,  $e : G_1 \times G_2 \rightarrow G_t$  be a bilinear map.

Let  $g_1$  and  $g_2$  be generators of  $G_1$  and  $G_2$ , respectively.

The map  $e$  is an admissible bilinear map if  $e(g_1, g_2)$  generates  $G_t$  and  $e$  is efficiently computable.

## VI. Analysis

The proposed two MA-ABE schemes can both be proven SAS CPA secure under the decisional BDH assumption. The steps of the proofs are similar to that of the basic MA-FIBE scheme. Since the ABE scheme rather than the FIBE scheme is adopted here, then the respective reduction about the part of ABE scheme would follow the proof shown in [14]. Due to the space limit, the detailed proofs of would be shown in the full version of this paper.

## A. Security Analysis

We first analyze security properties of our proposed scheme, starting with the following immediately available properties.

### 1. Fine-grainedness of Access Control

In our proposed scheme, the data owner is able to define and enforce expressive and flexible access structure for each user. Specifically, the access structure of each user is defined as a logic formula over data file attributes, and is able to represent any desired data file set.

## 2. User Access Privilege Confidentiality

Our proposed scheme just discloses the leaf node information of a user access tree to Cloud Servers. As interior nodes of an access tree can be any threshold gates and are unknown to Cloud Servers, it is hard for Cloud Servers to recover the access structure and thus derive user access privilege information.

## 3. User Secret Key Accountability

This property can be immediately achieved by using the enhanced construction of KP-ABE which can be used to disclose the identities of key abusers. Now we analyze data confidentiality of our proposed scheme by giving a cryptographic security proof.

## 4. Data Confidentiality

We analyze data confidentiality of our proposed scheme by comparing it with an intuitive scheme in which data files are encrypted using symmetric DEKs, and DEKs are directly encrypted using standard KP-ABE. In this intuitive scheme just ciphertexts of data files are given to Cloud Servers. Assuming the symmetric key algorithm is secure, e.g., using standard symmetric key algorithm such as AES, security of this intuitive scheme is merely relied on the security of KP-ABE. Actually, the standard KP-ABE is provably secure under the attribute-based Selective-Set model given the Decisional Bilinear Diffie-Hellman (DBDH) problem is hard. Therefore, the intuitive scheme is secure under the same model.

## B. Performance Analysis

This section numerically evaluates the performance of our proposed scheme in terms of the computation overhead introduced by each operation as well as the cipher text size.

### 1. Computation Complexity

We analyze the computation complexity for the following six operations: system setup, new file creation, file deletion, new user grant, user revocation, and file access.

### 2. Cipher text Size

The cipher text is composed of an ID, a header, and a body. The body is just the data block. The header for each data file is composed of an attribute set  $I$ , one group element on  $G_2$ , and group elements on  $G_1$ .

## C. Related Work

Existing work close to ours can be found in the areas of “shared cryptographic file systems” and “access control of outsourced data”. In [18], Kallahalla et al proposed Plutus as a cryptographic file system to secure file storage on untrusted servers. Plutus groups a set of files with similar sharing attributes as a file-group and associates each file-group with a symmetric lockbox-key. Each file is encrypted using a unique file-block key which is further encrypted with the lockbox-key of the file group to which the file belongs. If the owner wants to share a file-group, he just delivers the corresponding lockbox-key to users. As the complexity of key management is proportional to the total number of file-groups, Plutus is not suitable for the case of fine-grained access control in which the number of possible “file-groups” could be huge. In [19], Goh et al proposed SiRiUS which is layered over existing file systems such as NFS but provides end-to-end security. For the purpose of access control, SiRiUS attaches each file with a meta data file that contains the file’s Access Control List (ACL), each entry of which is the encryption of the file’s File

Encryption Key (FEK) using the public key of an authorized user. The extension version of SiRiUS uses NNL broadcast encryption algorithm to encrypt the FEK of each file instead of encrypting it with each individual user's public key. As the complexity of the user revocation solution in NNL is proportional to the number of revoked users, SiRiUS has the same complexity in terms of each meta data file's size and the encryption overhead, and thus is not scalable. Ateniese et al [20] proposed a secure distributed storage scheme based on proxy re-encryption. Specifically, the data owner encrypts blocks of content with symmetric content keys. The content keys are all encrypted with a master public key, which can only be decrypted by the master private key kept by the data owner. The data owner uses his master private key and user's public key to generate proxy re-encryption keys, with which the semi-trusted server can then convert the ciphertext into that for a specific granted user and fulfill the task of access control enforcement. The main issue with this scheme is that collusion between a malicious server and any single malicious user would expose decryption keys of all the encrypted data and compromise data security of the system completely. In addition, user access privilege is not protected from the proxy server. User secret key accountability is neither supported In [21], Vimercati et al proposed a solution for securing data storage on untrusted servers based on key derivation methods. In this proposed scheme, each file is encrypted with a symmetric key and each user is assigned a secret key. To grant the access privilege for a user, the owner creates corresponding public tokens from which, together with his secret key, the user is able to derive decryption keys of desired files. The owner then transmits these public tokens to the semi-trusted server and delegates the task of token distribution to it. Just given these public tokens, the server is not able to derive the decryption key of any file. This solution introduces a minimal number of secret key per user and a minimal number of encryption key for each file. However, the complexity of operations of file creation and user grant/revocation is linear to the number of users, which makes the scheme unscalable. User access privilege accountability is also not supported.

## D. Discussion

According to the above analysis, we can see that our proposed scheme is able to realize the desired security goals, i.e., fine-grained access control, data confidentiality, user access privilege confidentiality, and user secret key accountability. The goal of scalability is also achieved since the complexity for each operation of our proposed scheme, is no longer dependent to the number of users in the system. Therefore, our proposed scheme can serve as an ideal candidate for data access control in the emerging cloud computing environment. On the contrary, existing access control schemes in related areas either lack scalability [19, 21] and fine-grainedness [18], or do not provide adequate proof of data confidentiality [20].

## VII. Conclusion

The usage and dealing with the cloud computing is drastically demanding is increased. The maintenance, overall cost estimations are increasing abundantly. To overcome above pitfalls, the data that is shared and stored by the other parties, encrypting of data, sharing selectively etc are overcome by the process of monitoring all through the cloud and sharing the encrypted data by multi authority attribute based encryption (MA-ABE). The pitfalls and cost had been come down and performance increase anonymously.

## VIII. Acknowledgement

We are greatly delighted to place our most profound gratitude to Er. K. Satyanarayana, Chancellor of K.L. University for motivating us towards research emphasis. We#1 also extend our deep regards to our friends Vanga Nagi Reddy, Nathani Sushma and Simhadri Chinna Gopi who helped us in documenting this paper. We also thank the anonymous references of this paper for their valuable comments.

## References

- [1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, M. Zaharia, "Above the clouds: A Berkeley view of cloud computing", University of California, Berkeley, Tech. Rep. USB-EECS-2009-28, 2009.
- [2] "Amazon Web Services (AWS)", [Online] Available: <http://www.aws.amazon.com>.
- [3] "Google aapple engine", [Online] Available: <http://www.code.google.com/appengine/>.
- [4] "Microsoft-Azure", [Online] Available: <http://www.microsoft.com/azure/>.
- [5] 104<sup>th</sup> United States Congress, "Health Insurance Portability and Accountability Act of 1996 (HIPPA)", [Online] Available: <http://www.aspe.hhs.gov/admsimp/pl104191.htm>, 1996.
- [6] H. Harney, A. Colgrove, P. D. McDaniel, "Principles of policy in secure groups", in Proc. of NDSS'01, 2001.
- [7] P. D. McDaniel, A. Prakash, "Methods and limitations of security policy reconciliation", in Proc. of SP'02, 2002.
- [8] T. Yu, M. Winslett, "A unified scheme for resource protection in automated trust negotiation", in Proc. of SP'03, 2003.
- [9] J. Li, N. Li, W. H. Winsborough, "Automated trust negotiation using cryptographic credentials", in Proc. of CCS'05, 2005.
- [10] J. Anderson, "Computer Security Technology Planning Study", Air Force Electronic Systems Division, Report ESD-TR-73-51, 1972, [Online] Available: <http://www.seclab.cs.ucdavis.edu/projects/history>
- [11] J. Bzethencourt, A. Sahai, B. Waters, "Ciphertext-policy attribute-based encryption", In IEEE Symposium on Security and Privacy, pp. 321-334, 2007.
- [12] L. Cheung C. Newport, "Provably secure ciphertext policy abe", In ACM Conference on Computer and Communications Security, pp. 456-465, 2007.
- [14] V. Goyal, O. Pandey, A. Sahai, B. Waters, "Attribute-based encryption for ne-grained access control of encrypted data", In ACM Conference on Computer and Communications Security, pp. 89-98, 2006.
- [15] R. Gennaro, S. Jarecki, H. Krawczyk, T. Rabin, "Secure distributed key generation for discrete-log based cryptosystems", J. Cryptology, 20(1), pp. 51-83, 2007.
- [16] R. Ostrovsky, A. Sahai, B. Waters, "Attribute-based encryption with non-monotonic access structures", In ACM Conference on Computer and Communications Security, pp. 195-203, 2007.
- [17] M. Pirretti, P. Traynor, P. McDaniel, B. Waters, "Secure attribute-based systems", In ACM Conference on Computer and Communications Security, pp. 99-112, 2006.
- [18] M. Chase., "Multi-authority attribute based encryption. In TCC, pp. 515-534, 2007.
- [19] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, K. Fu, "Scalable secure file sharing on untrusted storage", in Proc. of FAST'03, 2003.

- [20] E. Goh, H. Shacham, N. Modadugu, D. Boneh, "Sirius: Securing remote untrusted storage", in Proc. of NDSS'03, 2003.
- [21] G. Ateniese, K. Fu, M. Green, S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage", in Proc. of NDSS'05, 2005.
- [22] S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, "Over-encryption: Management of access control evolution on outsourced data", in Proc. of VLDB'07, 2007.
- [23] Huang Lin, Zhenfu Cao, Xiaohui Liang, Jun Shao, "Secure threshold multi authority attribute based encryption without a central authority", Elsevier, 2010.



Dr. Rajasekhara Rao Kurra is a Professor of Computer Science and Engineering at K L University and presently holding several key positions in KL University, as Dean (Faculty & Student Affairs) & Principal KL College of Engineering (Autonomous). Having more than 26 years of teaching and research experience, Prof. Rao is actively engaged in the research related to

Embedded Systems, Software Engineering and Knowledge Management. He obtained Ph.D in Computer Science & Engineering from Acharya Nagarjuna University (ANU), Guntur, Andhra Pradesh and produced 35 publications in the International/National Journals and Conferences.

Prof. Rao was awarded "Patron Award" by CSI in 2011. He has been adjudged as best teacher and has been honored with "Best Teacher Award", six times. Dr. Rao is a Fellow of IETE, Life Member's of IE, ISTE, ISCA & CSI (Computer Society of India). He has been the past Chairman of the Koneru Chapter of CSI. Presently, Prof. K.R.Rao is the CSI State Student Coordinator of Andhra Pradesh.



Nallamalli Ramesh received his B.Tech in Computer Science and Engineering from Mekapati Rajamohan Reddy Institute of Technology and Science, Andhra Pradesh, India, in 2010. He is Pursuing M.Tech in Computer Science and Engineering in K L University, A.P, India during 2010-2012. His research invites include Cloud Computing, Data Mining and Knowledge Discovery, Data Warehousing and Database Systems.



Gorantla Praveen received his B.Tech in Computer Science and Engineering from Parkasam Engineering College, Andhra Pradesh, India, in 2010. He is Pursuing M.Tech in Computer Science and Engineering in K L University, A.P, India during 2010-2012. His research invites include Cloud Computing, Data Mining and Knowledge Discovery, Data Warehousing and Database Systems.



Mr. V. Praveen Krishna Anne currently working as Assistant Professor in K L University and pursuing the Ph.D in Computer Science & Engineering. He is actively engaged in research and publications in the areas of Network Security, Data Mining, Green Computing & Cloud Computing. He completed his M.Tech (CSE) from K.L College of Engineering, Acharya Nagarjuna

University and secured First Rank. He is topper of the university and honoured with gold medal for his extraordinary performance in MCA from SCSVMV University, Tamilnadu. He was awarded "Significant Contribution Award" by CSI in 2011. In KL University he has feedback as an excellent teacher. He has to his credit of research with 11 publications in reputed International/National Journals and Conferences.

Mr. Krishna is the Life member of CSI, IEEE & ISCA. Presently, he is Student Branch Counselor of CSI-K L University Student Branch.