PRIVACY PRESERVING DATA SHARING WITH CP-ABE

Neeta S. Nipane, Nutan M. Dhande

Abstract--- **With the recent adoption and diffusion of the data sharing paradigm in distributed systems such as online social networks or cloud computing, there have been increasing demands and concerns for distributed data security. One of the most challenging issues in data sharing systems is the enforcement of access policies and the support of policies updates. With the development of cryptography, the attribute-based encryption (ABE) draws widespread attention of the researchers in recent years. The ABE scheme, which belongs to the public key encryption mechanism, takes attributes as public key and associates them with the ciphertext or the user's secret key. It is an efficient way to solve open problems in access control scenarios, for example, how to provide data confidentiality and expressive access control at the same time. Ciphertext policy attribute-based encryption (CP-ABE) is becoming a promising cryptographic solution to this issue. It enables data owners to define their own access policies over user attributes and enforce the policies on the data to be distributed. Therefore, in this study, we propose a novel CP-ABE scheme for a data sharing system by exploiting the characteristic of the system architecture. The proposed scheme features the following achievements: 1) the key escrow problem could be solved by escrow-free key issuing protocol, which is constructed using the secure two-party computation between the key generation center and the data-storing center, and 2) fine-grained user revocation per each attribute could be done by proxy encryption which takes advantage of the selective attribute group key distribution on top of the ABE. The performance and security analyses indicate that the proposed scheme is efficient to securely manage the data distributed in the data sharing system.**

Index Terms-- **Data sharing, attribute-based encryption, revocation, access control, removing escrow.**

I. INTRODUCTION

With the development of the Internet and the distributed computing technology, there is a growing demand for data sharing and processing in an open distributed computing environment. RECENT development of the network and computing technology enables many people to easily share their data with others using online external storages. People can share their lives with friends by uploading their private photos or messages into the online social networks such as Facebook and MySpace; or upload highly sensitive personal health records (PHRs) into online data servers such as Microsoft HealthVault, Google Health for ease of sharing with their primary doctors or for cost saving. As people enjoy the advantages of these new technologies and services, their concerns about data security and access control also arise. Improper use of the data by the storage server or unauthorized access by outside users could be potential threats to their data. People would like to make their sensitive or private data only accessible to the authorized people with credentials they specified. The data provider needs to provide expressive access control and data confidentiality when communicating with customers. Recently, much attention has been attracted by a new public key primitive called Attribute-based encryption (ABE). ABE has significant advantage over the traditional PKC primitives as it achieves flexible one-to-many encryption instead of one-to-one. ABE is envisioned as an important tool for addressing the problem of secure and fine-grained data sharing and access control. In an ABE system, a user is identified by a set of attributes.

Attribute-based encryption (ABE) is a promising cryptographic approach that achieves a fine-grained data access control [4], [5], [6], [7]. It provides a way of defining access policies based on different attributes of the requester, environment, or the data object. Especially, ciphertext policy attribute-based encryption (CP-ABE) enables an encryptor to define the attribute set over a universe of attributes that a decryptor needs to possess in order to decrypt the ciphertext, and enforce it on the contents [6]. Thus, each user with a different set of attributes is allowed to decrypt different pieces of data per the security policy. This effectively eliminates the need to rely on the data storage server for preventing unauthorized data access, which is the traditional access control approach of such as the reference monitor.

Traditionally, this type of expressive access control is enforced by employing a trusted server to store data locally. The server is entrusted as a reference monitor that checks that a user presents proper certification before allowing him to access records or files. However, services are increasingly storing data in a distributed fashion across many servers. Replicating data across several locations has advantages in both performance and reliability. The drawback of this trend is that it is increasingly difficult to guarantee the security of data using traditional methods; when data is stored at several locations, the chances that one of them has been compromised increases dramatically. For these reasons we would like to require that sensitive data is stored in an encrypted form so that it will remain private even if a server is compromised.

Modern distributed information systems require flexible access control models which go beyond discretionary,

***Manuscript received May, 2015**.*

***First Author name**, Neeta S. Nipane, Nagpur University, Wardha-442001, Maharashtra, India*
***Second Author name**, Nutam M. Dhande, Nagpur University, Wardha-442001, Maharashtra, India .*

mandatory and role-based access control. Recently proposed models, such as attribute-based access control, define access control policies based on different attributes of the requester, environment, or the data object. On the other hand, the current trend of service-based information systems and storage outsourcing require increased protection of data including access control methods that are cryptographically enforced. The concept of Attribute-Based Encryption(ABE) fulfills the aforementioned requirements. It provides an elegant way of encrypting data such that the encryptor data such that the attribute set that the decryptor needs to posses in order to decrypt the cipher-text. This is useful for applications where the data provider knows specifically which user he wants to share with, in many applications the provider will want to share data according to some policy based on the receiving user's credentials.

II. Literature Review

ABE comes in two flavours called key-policy ABE (KP-ABE) and ciphertext-policy ABE (CP-ABE). In KP-ABE, attributes are used to describe the encrypted data and policies are built into users' keys; while in CP-ABE, the attributes are used to describe users' credentials, and an encryptor determines a policy on who can decrypt the data. Between the two approaches, CP-ABE is more appropriate to the data sharing system because it puts the access policy decisions in the hands of the data owners. Most of the existing ABE schemes are constructed on the architecture where a single trusted authority, or KGC has the power to generate the whole private keys of users with its master secret information. Thus, the key escrow problem is inherent such that the KGC can decrypt every ciphertext addressed to users in the system by generating their secret keys at any time.

Junbeom Hur [1], proposed CP-ABE schemes with slight changes in attribute rather than from the scratch and remove the problem which is encountered in Bethencourt et al. [5], like key escrow and revocation. But in this system there is computational overhead.

M. Chase and S.S.M. Chow [2], proposed a distributed KP-ABE scheme that solves the key escrow problem in a multi-authority system. Most of the existing ABE schemes are constructed on the architecture where a single trusted authority, or KGC has the power to generate the whole private keys of users with its master secret information. Thus, the key escrow problem is inherent such that the KGC can decrypt every ciphertext addressed to users in the system by generating their secret keys at any time. One disadvantage of this kind of fully distributed approach is the performance degradation.

S.S.M. Chow [3], proposed an anonymous private key generation protocol in identity-based literature such that the KGC can issue a private key to an authenticated user without knowing the list of users' identities. However, found that this cannot be adapted to ABE systems due to mainly two reasons. First, in Chow's protocol, identities of users are not public anymore, at least to the KGC, because the KGC can generate users' secret keys otherwise. Second, the collusion attack between users is the main security threat in ABE.

J. Bethencourt, A. Sahai, and B. Waters [4], and A. Boldyreva, V. Goyal, and V. Kumar [5], proposed first key revocation mechanisms in CP-ABE and KP-ABE settings, respectively. These schemes enable an attribute key revocation by encrypting the message to the attribute set with its validation time. These attribute-revocable ABE have the security degradation problem in terms of the backward and forward secrecy.

N. Attrapadung and H. Imai [6], suggested another user-revocable ABE schemes addressing the key revocation problem by combining broadcast encryption schemes with ABE schemes. However, in this scheme, the data owner should take full charge of maintaining all the membership lists for each attribute group to enable the direct user revocation. This scheme is not applicable to the data sharing system, because the data owners will no longer be directly in control of data after storing their data to the external storage server.

S. Yu, C. Wang, K. Ren, and W. Lou [7], recently addressed the user revocation in the ABE-based data sharing system. In this scheme, the user revocation is realized using proxy reencryption by the data server. However, in order to revoke users, the KGC should generate all secret keys including the proxy key on behalf of the data server. Then, the server would reencrypt the ciphertext under the proxy key received from the KGC to prevent revoked users from decrypting the ciphertext. Thus, the key escrow problem is also inherent in this scheme.

S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati [8], proposed a solution for securing outsourced data on semi-trusted servers based on symmetric key derivation methods which can achieve fine-grained access control. Unfortunately, the complexities of file creation and user grant/revocation operations are linear to the number of authorized users, which is less scalable.

Shilpa Elsa Abraham and R. Gokulavanan [10], proposed The system scalability is enhanced using ABE and MA-ABE. The expressibility of our encryptor's access policy is somewhat limited by that of MA-ABE's. In practice, the credentials from different organizations may be considered equally effective, in that case distributed ABE schemes will be needed. The following drawbacks are identifying from the existing system. User identity bases access control mechanism is not supported under the situation. Dynamic policy management is yet another issue.

Recently, the research community has proposed Attribute-based Encryption (ABE) systems where encryption and decryption are determined by the attributes of the data and the recipients. An ABE cryptosystem is designed to enable fine-grained access control of the encrypted data. It allows the encryptor to attach attributes or policies to a message being encrypted so that only the receiver(s) who is (are) assigned compatible policies or attributes can decrypt it. Formally, the attributes can be considered as boolean variables with arbitrary labels, and the policies are expressed as conjunctions and disjunctions of attribute variables. The ABE systems can be viewed as a generalization of Identity Based Encryption (IBE) systems [10]. In IBE systems, only one attribute is used which is the identity of the receiver, whereas ABE systems enable the use of multiple attributes simultaneously. In fact, current ABE schemes are built by cleverly combining the basic techniques of IBE with a linear secret sharing scheme. In these schemes, the access policies in the form of a monotonic boolean formula over the attribute variables [11]. There are two alternatives in enforcing the access policy. The access policy can be embedded in the private key of a user, which results in a cryptosystem called Key Policy ABE (KP-ABE) [11]. Alternatively, the access policy can be embedded in the ciphertext, which results in the Ciphertext Policy ABE (CP-ABE) system. Both KP-ABE and CP-ABE systems ensure that a group of users cannot access any unauthorized data by colluding with each other.

III. Security Analysis Of The Proposed System

According to the existing schemes, the functionalities in an ideal ABE scheme is listed as follows:

1. Data confidentiality: Unauthorized users who do not have enough attributes satisfying the access policy should be prevented from accessing the plaintext of the data. Additionally, the KGC is no longer fully trusted in the data sharing system. Thus, unauthorized access from the KGC as well as the data-storing center to the plaintext of the encrypted data should be prevented.

2. Collusion resistance: Collusion resistance is one of the most important security property required in ABE systems [4], [5], [6]. If multiple users collude, they may be able to decrypt a ciphertext by combining their attributes even if each of the users cannot decrypt the ciphertext alone. Do not want these colluders to be able to decrypt the private data in the server by combining their attributes. Since we assume the KGC and data-storing center are honest, consider any active attacks from them by colluding with revoked users as in [3], [12].

3. User/attribute revocation: If a user quits the system, the scheme can revoke his access right. Similarly, attribute revocation is inevitable.

4. Scalability: The number of authorized users cannot affect the performance of the scheme. That is to say, the scheme can deal with the case that the number of the authorized users increases dynamically.

IV. Proposed Methodology

The first CP-ABE scheme proposed by Bethencourt et al. [5], which are mostly motivated by more rigorous security proof in the standard model. However, most of the schemes failed to achieve the expressiveness of the Bethencourt et al.'s scheme. Therefore, in this proposed, develop a variation of the CP-ABE algorithm partially based on (but not limited to) Bethencourt et al.'s construction in order to enhance the expressiveness of the access control policy instead of building a new CP-ABE scheme from scratch. The attribute based encryption scheme is enhanced to handle distributed attribute based encryption process. Data update and key management operations are tuned for multi user access environment.

Its key generation procedure is modified for removing key escrow. The proposed scheme is then built on this new CP-ABE variation by further integrating it into the proxy reencryption protocol for the user revocation. To handle the fine-grained user revocation, the data storing center must obtain the user access (or revocation) list for each attribute group, since otherwise revocation cannot take effect after all. This setting where the data-storing center knows the revocation list does not violate the security requirements, because it is only allowed to reencrypt the ciphertexts and can by no means obtain any information about the attribute keys of users. The two parties engage in the arithmetic 2PC protocol with master secret keys of their own, and issue independent key components to users during the key issuing phase. The 2PC protocol deters them from knowing each other's master secrets so that none of them can generate the whole set of secret keys of users individually. Thus, we take an assumption that the KGC does not collude with the data-storing center since they are honest.

A. Data Sharing Architecture

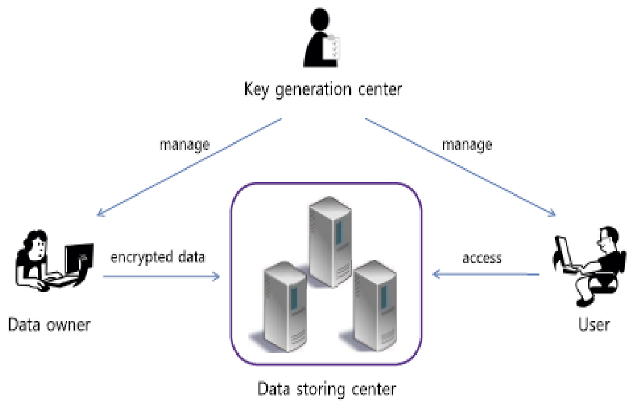

Fig.1 Architecture of data sharing system

Fig.1 shows the architecture of the data sharing system, which consists of the following system entities:

1) Key generation center: It is a key authority that generates public and secret parameters for CP-ABE. It is in charge of issuing, revoking, and updating attribute keys for users. It grants differential access rights to individual users based on their attributes. That is, it will honestly execute the assigned tasks in the system; however, it would like to learn information of encrypted contents as much as possible. Thus, it should be prevented from accessing the plaintext of the encrypted data even if it is honest.

2) Data-storing center: It is an entity that provides a data sharing service. It is in charge of controlling the accesses from outside users to the storing data and providing corresponding contents services. The data-storing center is another key authority that generates personalized user key with the KGC, and issues and revokes attribute group keys to valid users per each attribute, which are used to enforce a fine-grained user access control.

3) Data owner: It is a client who owns data, and wishes to upload it into the external data-storing center for ease of sharing or for cost saving. A data owner is responsible for defining (attribute-based) access policy, and enforcing it on its own data by encrypting the data under the policy before distributing it.

4) User: It is an entity who wants to access the data. If a user possesses a set of attributes satisfying the access policy of the encrypted data, and is not revoked in any of the valid attribute groups, then he will be able to decrypt the ciphertext and obtain the data.

A. Ciphertext-policy ABE (CP-ABE)

In the CP-ABE scheme described in [1], each user is associated with a set of attributes and her private key is generated based on these attributes. When encrypting a message *M*, the encryptor specifies an access structure which is expressed in terms of a set of selected attributes for *M*. The message is then encrypted based on the access structure such that only those whose attributes satisfy this access structure can decrypt the message. Unauthorized users are not able to decrypt the ciphertext even if they collude. In [1], the access structure is sent in plaintext. A CP-ABE scheme consists of the following four algorithms:

1. Setup: This is a randomized algorithm that takes a security parameter as input, and outputs the public parameters *PK* and a master key *MK*. *PK* is used for encryption and *MK* is used to generate user secret keys and is known only to the central authority.
2. Encryption: This is a randomized algorithm that takes as input a message *M*, an access structure *T*, and the public parameters *PK*. It outputs the ciphertext *CT*.
3. KenGen: This is a randomized algorithm that takes as input the set of a user (say *X*)'s attributes *SX*, the master key *MK* and outputs a secret key *SK* that identifies with *SX*.
4. Decryption: This algorithm takes as input the ciphertext *CT*, a secret key *SK* for an attribute set *SX*. If *SX* satisfies the access structure embedded in *CT*, it will return the original message *M*.

The security authority holds the master key *MK* and publishes the public parameters *PK*. Using *PK*, a user can encrypt an arbitrary message, *M* using any arbitrary access structure *T* based on a set of attributes, *S*.

A CP-ABE scheme which supports the use of positive and negative attributes. Specifically, modify Bethencourt et al.'s CP-ABE construction to accommodate a non-monotonic access structure. In a CP-ABE cryptosystem, a plaintext *M* is encrypted based on an access tree structure *T* to generate a ciphertext *CT*. The private key of user *X* is identified with the set of *X*'s attributes, *SX*. A private key, *SK* will be able to decrypt *CT* if the set of attributes, *SX* assigned to *SK* satisfies the access tree *T*.

B. RSA Technology

Here, used the RSA technology for encryption purpose. RSA is one of the first practicable public-key cryptosystems and is widely used for secure data transmission. Public-Key encryption is a powerful mechanism for protecting the confidentiality of stored and transmitted information. In such a cryptosystem, the encryption key is public and differs from the decryption key which is kept secret. RSA stands for Ron Rivest, Adi Shamir and Leonard Adleman, who first publicly described the algorithm in 1977.

To generate the encryption and decryption keys, we can proceed as follows.
1. Generate randomly two "large" primes p and q.
2. Compute $n = pq$ and $\Phi = (p - 1)(q - 1)$.
3. Choose a number e so that

$$\gcd(e, \Phi) = 1.$$

4. Find the multiplicative inverse of e modulo Φ, i.e., find d so that

$$ed \equiv 1 \pmod{\Phi}.$$

This can be done efficiently using Euclid's Extended Algorithm.

The encryption public key is $K_E = (n, e)$ and the decryption private key is $K_D = (n, d)$.

The encryption function is

$$E(M) = M^e \bmod n.$$

The decryption function is

$$D(M) = M^d \bmod n.$$

These functions satisfy

$$D(E(M)) = M \text{ and } E(D(M)) = M \text{ for any } 0 \leq M < n.$$

To describe construction, use the terminology and language as in [13, 14], with describing the access tree, *T*. Each leaf node of the access tree *T* represents either a positive or a negative attribute. Each internal node of *T* represents a threshold gate, which can be an 'AND' gate or an 'OR' gate in special cases. If *numx* is the number of children of a node *x* and *kx* is its threshold value, then $0 < kx \cdot numx$. A threshold gate represents an OR gate if $kx = 1$, and an AND gate if $kx = numx$. Denote the parent of node *x* by *parent*(*x*). The function *att*(*x*), which is defined only for the leaf nodes, denotes the attribute associated with a leaf node *x*. In the access tree, also define an ordering among the child nodes of every node. The children of node *x* are numbered from 1 to *numx*. Ddenote such a number associated with node *x* by *index*(*x*).

Encryption (*PK*, *M*, *T*). The encryption algorithm encrypts a message *M*, $M \varepsilon G1$ under the tree access structure *T*. The algorithm first chooses a polynomial *qx*for each node *x* (including the leaves) in the tree *T*. These polynomials are chosen in the following way in a top-down manner, starting from the root node *R*. For each node *x* in the tree, set the degree *dx* of the polynomial *qx* to be one less than the threshold value *kx* of that node, that is, $dx = kx - 1$. Starting with the root node *R* the algorithm chooses a random $s \varepsilon Zp$ and sets $qR(0) = s$. Then, it chooses *dR* other points of the polynomial *qR* randomly to define it completely. For any other node *x*, it sets $qx(0) = qparent(x)(index(x))$ and chooses *dx* other points randomly to completely define *qx*.

Decrypt(*CT, SK*). Decryption procedure as a recursive algorithm. For ease of exposition we present the simplest form of the decryption algorithm.The optimizations reported in [13] to design a more efficient decryption algorithm; first define a recursive algorithm DecryptNode (*CT, SK, x*) that takes as input a ciphertext *CT,* a private key *SK,* which is associated with a set *S* of attributes, and a node *x* from *T* . If the node *x* is a leaf node, then we let *i* = *att*(*x*).

C. Keeping the Access Policy Hidden

The construction the ciphertext, *CT* contains the access tree, *T* which is in the clear text, make a little modification, which could keep most of the access policy used in the ciphertext hidden. The modification of our construction is only in stating the access policy, which is described below while the rest of the construction remains the same. Our modified access tree, *T* does not contain any specific value of an attribute—instead, *T* specifies only the attribute name. However, the ciphertext fields (e.g., *C*1*y*, *C*4*y*) are computed using both the attribute names and the specific values. During decryption, the receiver *X* appends to each matched k attribute name in *T* its own value. So, *X* is able to decrypt if and only if *X* is an authorized user. Assuming an attribute could have multiple possible values in the application domain, the modified scheme does not reveal the policy to the adversary yet enables the intended retrieval.
As an example, say a bank having the 10 branches in the city. Customer A's having account in branch 1, and he is performing the transaction with the branch 5 of the same bank. Then this branch having the specific access rights, only summarized information not the whole data about the customer. The access rights are specified by the owner to the user. The public key, private key, ciphertext is of size *O*(*N*), where *N* is the total number of attributes present in the application domain. All operations such as Setup, Encryption, KeyGen, and Decrypt require *O*(*N*) computations. On the other hand, our scheme not only supports general access policy but hides most of the access policy. The public and private key is of size *O*(*d*), where *d* is the number of attributes present in each private key, which is typically a small constant. Operations such as Setup, and KeyGen require *O*(*d*) computations while the cost of Encryption and Decrypt operations is application specific.

D. Advantages of using keys more than 1024 to 2048 bits

- Some hardware(many smart cards, some card readers and some other devices such as Polycom phones) don't support anything bigger than 2048 bits.
- Uses less CPU than a longer key during encryption and authentication.
- Using less CPU means using less battery power (important for mobile devices).
- Uses less storage space: while not an issue on disk, this can be an issue in small devices like smart cards that measure their RAM in kilobytes rather than gigabytes.

E. Efficiency of Construction

As of 2003 RSA Security claims that 1024-bit RSA keys are equivalent in strength to 80-bit symmetric keys, 2048-bit RSA keys to 112-bit symmetric keys and 3072-bit RSA keys to 128-bit symmetric keys.

RSA claims that 1024-bit keys are likely to become crackable some time between 2006 and 2010 and that 2048-bit keys are sufficient until 2030. An RSA key length of 3072 bits should be used if security is required beyond 2030.

V. Advantages

- Distributed environment**.**
- Security of sensitive fields**.**
- Break glass access for emergency situations**.**
- On-demand revocation

VI. Conclusions

A novel framework of secure sharing of personal records under distributed environment with the CP-ABE encryption technique has been proposed in this paper . Public and personal access models are designed with security and privacy enabled mechanism. The framework addresses the unique challenges brought by multiple transaction records of owners and users, in that the complexity of key management is greatly reduced while guaranteeing the privacy compared with previous works. The attribute-based encryption model is enhanced to support distributed ABE operations with CP-ABE. The system is improved to support dynamic policy management model.

REFERENCES

[1] Junbeom Hur, "Improving Security and Efficiency in Attribute-Based Data Sharing" *IEEE Transactions On Knowledge And Data Engineering* Vol:25 No:10 Year 2013

[2] M. Chase and S.S.M. Chow, "Improving Privacy and Security in Multi-Authority Attribute-Based Encryption," *Proc. ACM Conf. Computer and Comm. Security*, pp. 121-130, 2009.

[3] S.S.M. Chow, "Removing Escrow from Identity-Based Encryption," *Proc. Int'l Conf. Practice and Theory in Public Key Cryptography (PKC '09),* pp. 256-276, 2009.

[4] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext- Policy Attribute-Based Encryption," *Proc. IEEE Symp. Security and Privacy*, pp. 321-334, 2007.

[5] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-Based Encryption with Efficient Revocation," *Proc. ACM Conf. Computer and Comm. Security*, pp. 417-426, 2008.

[6] N. Attrapadung and H. Imai, "Conjunctive Broadcast and Attribute-Based Encryption," *Proc. Int'l Conf. Palo Alto on Pairing-Based Cryptography (Pairing)*, pp. 248-265, 2009.

[7] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute Based Data Sharing with Attribute Revocation," Proc. ACM Symp. Information, Computer and Comm. Security (ASIACCS '10), 2010.

[8] S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi and P. Samarati, "Overencryption: management of access control evolution on outsourced data," in VLDB '07, 2007, pp. 123–131.

[9] Shilpa Elsa Abraham and R. Gokulavanan "Ensuring Privacy and Security in Data Sharing under Cloud Environment" *International Journal of Computer Applications Technology and Research Volume 2– Issue 2*, 188 - 194, 2013

[10] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *SIAM Journal of Computing*, 2003.

[11] V. Goyal and others. Attribute-based encryption for finegrained access control of encrypted data. In *Proceedings of ACM CCS*, 2006.

[12] S.D.C. Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Over-Encryption: Management of Access Control Evolution on

Outsourced Data," Proc. Int'l Conf. Very Large Data Bases (VLDB '07), 2007.

[13] J. Bethencourt and others. Ciphertext-policy attribute based encryption. In *Proceedings of IEEE SP, Oakland*, 2007.

BIOGRAPHIES

Neeta S. Nipane, pursuing M.Tech 2nd year, Computer Science & Engineering in Nagpur Universty. Two years of experience as an Assistant Lecturer in an engineering college.