Melioration Process for Image Sharing Security and Secured Image Login

[1]Sushmita A. Kale, [2]Prof. P. S. Malge

[1]Student, [2]Assistant Professor
Department of Electronics Engineering,
WIT, Solapur, India

***Abstract*: In distributed system people are having access to data if they possess certain set of credentials. Cipher text policy attribute based encryption (CPABE) is a scheme which enforces such policies to employ trusted server to store data. CP-ABE enables data owner to define their own access policies over user attributes and enforce the policies on the data to be distributed. Attribute based data sharing scheme enforce a fine-grained data access control by exploiting the characteristic of the data sharing system. The existing scheme use alphanumeric data to share and generates secret keys through a secure two-party computation such that any curious key generation center or data-storing center cannot derive the private keys individually. Here time is required for acknowledging and processing. Thus here a system is developed to share multimedia data i.e. image (jpeg) online which improves efficiency by storing image in text format instead of storing image and improve security by setting attributes by owner with time function.**

***Index Terms*: Image Security, Image Encryption, CP-ABE, Image Sharing System, Attribute Based Encryption.**

I. INTRODUCTION

Attribute-based encryption (ABE) is a recent approach that reconsiders the concept of public-key cryptography. In public-key cryptography, a message is encrypted for a specific receiver using the receiver's public-key. Identity-based cryptography and in particular identity-based encryption (IBE) changed the traditional understanding of public-key cryptography by allowing the public-key to be an arbitrary string, e.g., the email address of the receiver. ABE defines the identity as a set of attributes, e.g., roles, and messages can be encrypted with respect to subsets of attributes (key-policy ABE) or policies defined over a set of attributes (ciphertext-policy ABE). The issue is someone should be able to decrypt a ciphertext if the person holds a key for "matching attributes" where user keys are always issued by some trusted party. ABE comes in two flavors called key-policy ABE and ciphertext-policy ABE. In KeyPolicy, attributes are used to describe the encrypted data and policies are built into users‟ keys; while in CiphertextPolicy, the attributes are used to describe users‟ credentials and an encrypter determines policy on who can decrypt the data. Between the two approaches, CiphertextPolicy was more appropriate to the data-sharing system because it puts the access policy decisions in the hands of the data owners [2], [3]. In CiphertextPolicy, the key generation center (KGC) generates private keys of users by applying the KGC‟s master secret keys to users‟ associated set of attributes. Attribute-based encryption can be used for log encryption. Instead of each part of a log encryption with the keys of all recipients, it is possible to encrypt the log only with attributes which match recipients' attributes. This primitive can be used for broadcast encryption in order to decrease the number of keys used. CiphertextPolicy is more appropriate to the image-sharing system because it puts the access policy decisions in the hands of the image owners.

II. LITERATURE REVIEW

Sahai and Waters [4] first introduced attribute-based encryption (ABE) for encrypted access control. In an ABE system, both the user secret key and the ciphertext are associated with a set of attributes. Only if at least a threshold number of attributes overlap between the ciphertext and his secret key, can the user decrypt the ciphertext. Goyal et al. [5] first introduced the concept of CP-ABE based on [4]. The idea of a CPABE scheme is as follows: the user secret key is associated with a set of attributes and each ciphertext is embedded with an access structure. EndUser is able to decrypt a ciphertext if and only if his attributes satisfy the access structure of the ciphertext. Bethencourt et al. [6] proposed the first CP-ABE construction under the generic group model. Cheung et al. The [7] proposed the first provably secure CP-ABE under a standard assumption (the DBDH assumption) while only permitting AND gates in the access structure. Goyal recently proposed a bounded CiphertextPolicy scheme with expressive access structure and provable security under the standard model. However, the complexity of the construction is extremely high and can just serve as theoretical feasibility. The issue of attribute revocation, a.k.a. key revocation, in CP-ABE, was first addressed in [8] as a rough idea. In this paper, it suggests extending each user attribute with an expiration date. This idea, as the authors pointed out, requires the users to periodically go to the authority for key reissuing and thus is inefficient. [6] Enhances this solution by associating the user secret key with a single expiration date. As is compared to [8], this solution places a lower load on the authority as users need to update their keys less frequently. And this method is not able to realize user attribute change in a timely fashion. These solutions can just disable a user secret key at a designated time but are not able to revoke a user attribute/key on the ad hoc basis. In, Boldyreva et al. [9] proposed an efficient revocation scheme for IBE, and the proposed scheme is also applicable to KP-ABE and fuzzy IBE. However, its applicability to CP-ABE is not clear. In 1998, Blaze et al.[10], a semi-trustable proxy is able to convert a ciphertext encrypted under Alice's public key into another ciphertext that can be opened by Bob's private key without seeing the underlying plaintext. This cryptographic primitive is called Proxy Re-Encryption (PRE). It allows the proxy, given the proxy re-encryption key rka↔b, to

translate ciphertext under public key pka into ciphertext under public key pkb and vise Versa. We refer to [10] for more details on proxy re-encryption schemes. Enhancements to proxy re-encryption scheme can be found in [11].

III. PROPOSED SYSTEM

A novel concept is introduced as an image sharing system to secure image sharing by using CP-ABE scheme. Image is encrypted and key for decryption is stored in database. Image is decrypted using key when attribute matches otherwise user can send request of particular image to the owner of image then the key distribution is takes place with notification.

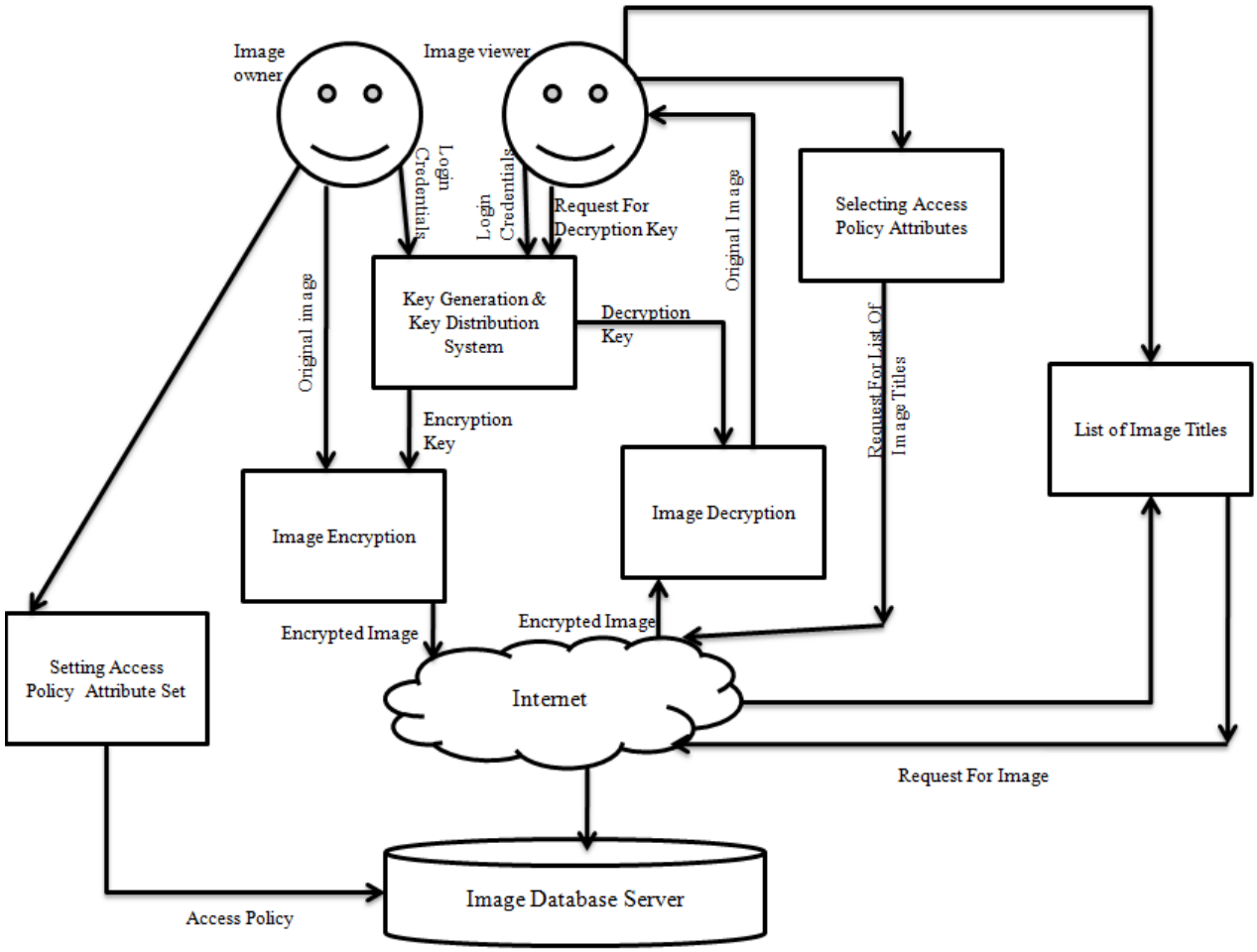

Fig 3.1 Proposed systems

System uses methods such as generation of keys, encryption of image, and decryption. Stepwise description of these methods are:
Algorithm for Key_generation :

a. Owner select image to upload
b. The system set attribute Pra and randomly get number as another attribute Ra.

Then these attributes generate key for encryption with time function T.

i.e. Pbk= E(Pra ,Ra ,T).

Where, Pbk is public key and E is encryption.

c. Ra and T together generate image ID.

Algorithm for Encryption:

a. Image first converted to binary
b. Encryption is done using Pbk and base64 encryption algorithm.

Ci=Eb(Pi,Pbk)

Where, Ci is ciphertext of image,
Eb is encryption,
Pi is plaintext of image,
Pbk is public key.

Algorithm for Decryption:

a. User select image to download.
b. System checks attributes of user with image attribute.
c. If attributes matches, generate key for decryption Prk and decrypt image

Pi=Db (Ci,Prk)

Where, Pi is plain text of image,
Db is decryption,
Ci is ciphertext of image,
Prk is private key.
d. If attributes does not matches, send request to owner. If owner allow then key will be sent to user to download image.

IV. IMPLIMENTATION DETAILS

As per our problem statement we are presenting sharing system for images. For accessing this system stakeholder need to register first. Stakeholder is either image owner or image user. The image owner store image on database server. The image user access image online from database server. For storing image in database, image owner has to login first with username and password. If login is successful, the image owner select the image file that he want to upload. Then the image is inputted for encryption. Encryption used is CP-ABE. When image is encrypted it is uploaded online on database server. And if login is unsuccessful then the error message is displayed. For accessing image from database, image user has to login first with username and password. If login is successful, the image user gets the list of uploaded images. User selects the image from the list. If users attributes matches then file download by decrypting using decryption key. But if users attributes doesn't match then user need to send request to its owner just by clicking on that file. Then if that image owner allow to user to access that image then user get the message which contains image ID and decryption key. Then the image file is decrypted when image user enter the decryption key and download that image file. And if login is unsuccessful the error message is displayed. Flow chart of the developed system is depicted in Fig-4.1.

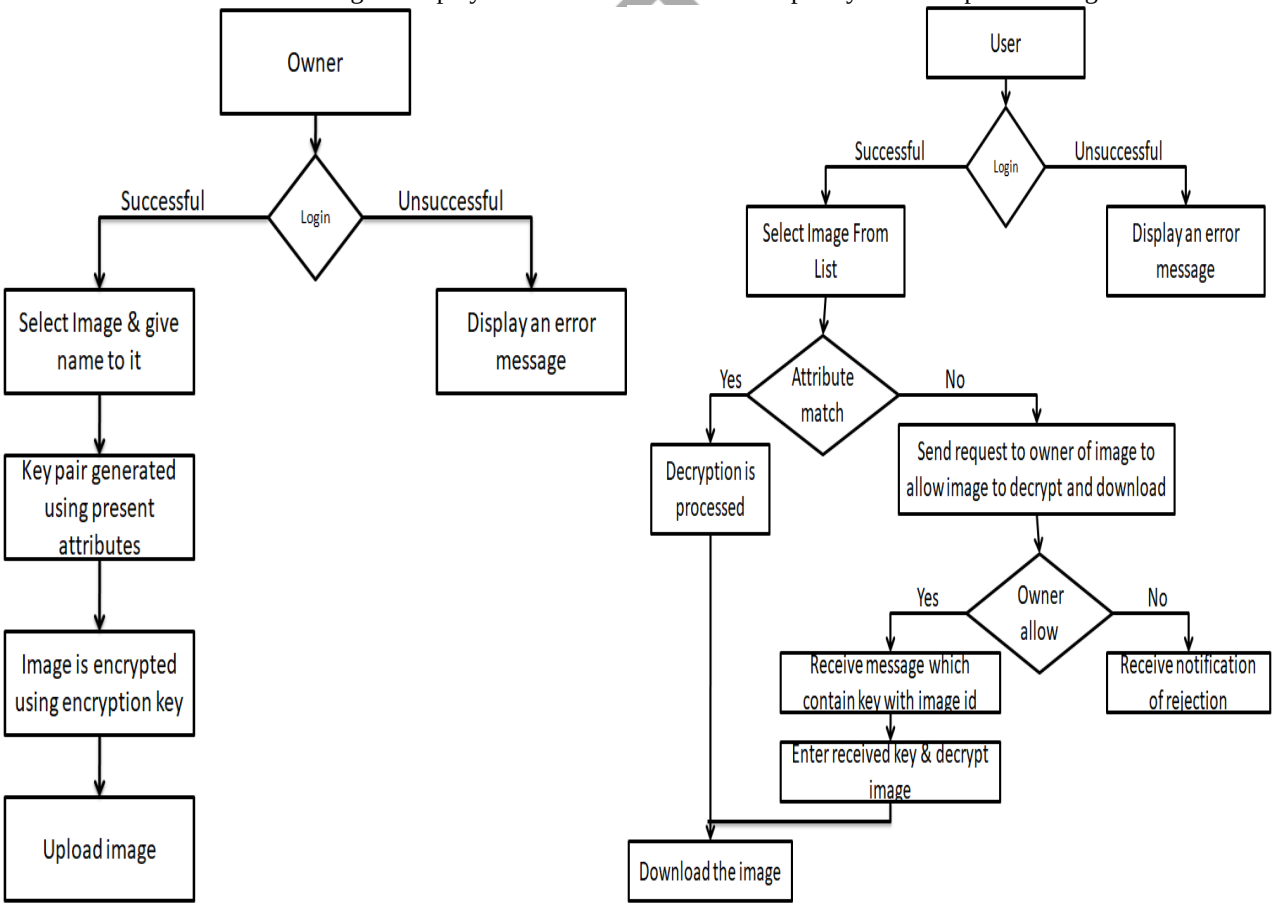

Fig4.1. Flowchart of Proposed System

This is the flow of the developed system which describes how the system work. The developed system has two parts, that is, Image Owner and Image User. Generating key, Encrypting image and setting access policy are the processes of image owner side. Whereas matching attribute, decrypting image, requesting for image are the processes of image user side. Person registers with system by providing information such as user name, password, mobile number and email. This information is store online in database by encrypting the password so nobody can identify the password. That's why there is a security for login.

Algorithm for Owner:

1. Owner Register.
2. Get Login.
3. Select Image to upload.
4. Encrypted image will get converted to text file & get stored on database.
5. System will generate unique key for the image using Key generation algorithm.
6. Owner will approve the request.
7. The generated key will be given to the requested person through mail.

8. Owner Logout.

Algorithm for User

1. User registers.
2. Get Login.
3. Request for image to Owner.
4. User will get key from Owner through Mail.
5. User will enter the key.
6. The image will get decrypt & get downloaded.
7. User Logout.

1. Algorithm for Key Generation

• Owner select image to upload.

• The system set attribute A and randomly generate number as another attribute B. Then these attributes generate key for encryption with time function T.

i.e. pk= E(A ,B ,T).

where, pk is public key of the image owner and E is encryption technique [MD5].

• B and T together generate image ID.

2. Algorithm for Encryption

• Image first converted to binary .

• Encryption is done using key and base64 encryption algorithm.

Ci=Eb(Pi ,pk)

Where, Ci is ciphertext of image, Eb is base64 encryption, Pi is plaintext of image, pk is public key of image owner .

3. Algorithm for Decryption

• User select image to download.

• System compares attributes of user with attribute of encrypted image.

• If image user attributes matches, generate key for decryption Prk and decrypt image Pi=Db(Ci ,Prk)

• Where, Pi is plain text of image, Db is base64 decryption, Ci is ciphertext of image, Prk is private key of image user.

• If image user attributes does not matches, send request to image owner.

• If owner allow then key (Prk) will be sent to user to download image.

4. Algorithm for MD5

• Step1 : Append padding bits

The input message is extended ["padded"] so that its length (in bits) equals to 448 mod 512. The padding is performed, even if the length of the message is already 448 mod 512. Padding is performed as : a single "1" bit is appended to the message, and then "0" bits are appended so that the length in bits of the padded message becomes congruent to 448 mod 512. Minimum one bit and at most 512 bits are appended.

• Step2 : Append length

The representation of the length of the message(64 bit) is appended to the result of step1. If the message length is above 2^64, only the low-order 64 bits will be used. The output message (after padding with bits and with b) has a length that is an exact multiple of 512 bits. Then the input message will have a length that is an exact multiple of 16(32-bit) words.

• Step3 : Initialize MD buffer

A 4-word buffer is used to compute the message digest. Each of word is a 32-bit register. These 32-bit registers are initialized to the following values in hexadecimal, low-order bytes first):

word A: 01 23 45 67
word B: 89 ab cd ef
word C: fe dc ba 98
word D: 76 54 32 10

• Step4 : Process message in 16-word blocks.

The functions is defined as each function takes an input of three 32-bit words and produces a 32-bit word output.

F (L, K, M) = LK or not (L) M
G ((L, K, M) = LM or K not (M)
H (L, K, M) = L xor K xor M
I (L, K, M) = K xor (L or not (M))

5. Algorithm for base64

• Step 1: Divide the input bytes stream into blocks of 3 bytes.

• Step 2: Divide 24 bits of each 3-byte block into 4 groups of 6 bits.

• Step 3: Map each group of 6 bits to 1 printable character, based on the 6-bit value using the Base64 character set map.

• Step 4: If the last 3-byte block has only 1 byte of input data, pad 2 bytes of zero (\x0000). After encoding it as a normal block, override the last 2 characters with 2 equal signs (==), so the decoding process knows 2 bytes of zero were padded.

• Step 5: If the last 3-byte block has only 2 bytes of input data, pad 1 byte of zero (\x00). After encoding it as a normal block, override the last 1 character with 1 equal signs (=), so the decoding process knows 1 byte of zero was padded.

• Step 6: Carriage return (\r) and new line (\n) are inserted into the output character stream. They will be ignored by the decoding process.

V. RESULTS

Fig 5.1.Image before Encryption

1. Fig 5.1 shows the original image.
2. Owner uploads the image it will be clear and it is in the original form.

Fig5. 2.1 & 5.2.2 Image after Applying Bluer and Encryption Algorithm
Fig 5.2.1 Image after Blurring Algorithm

1. Fig 5. 2.1 show the image after the encryption.
2. When User login in to the account then the blurred image will be display.
3. When User wanted to download or save the image then User have to send the request to the Owner.

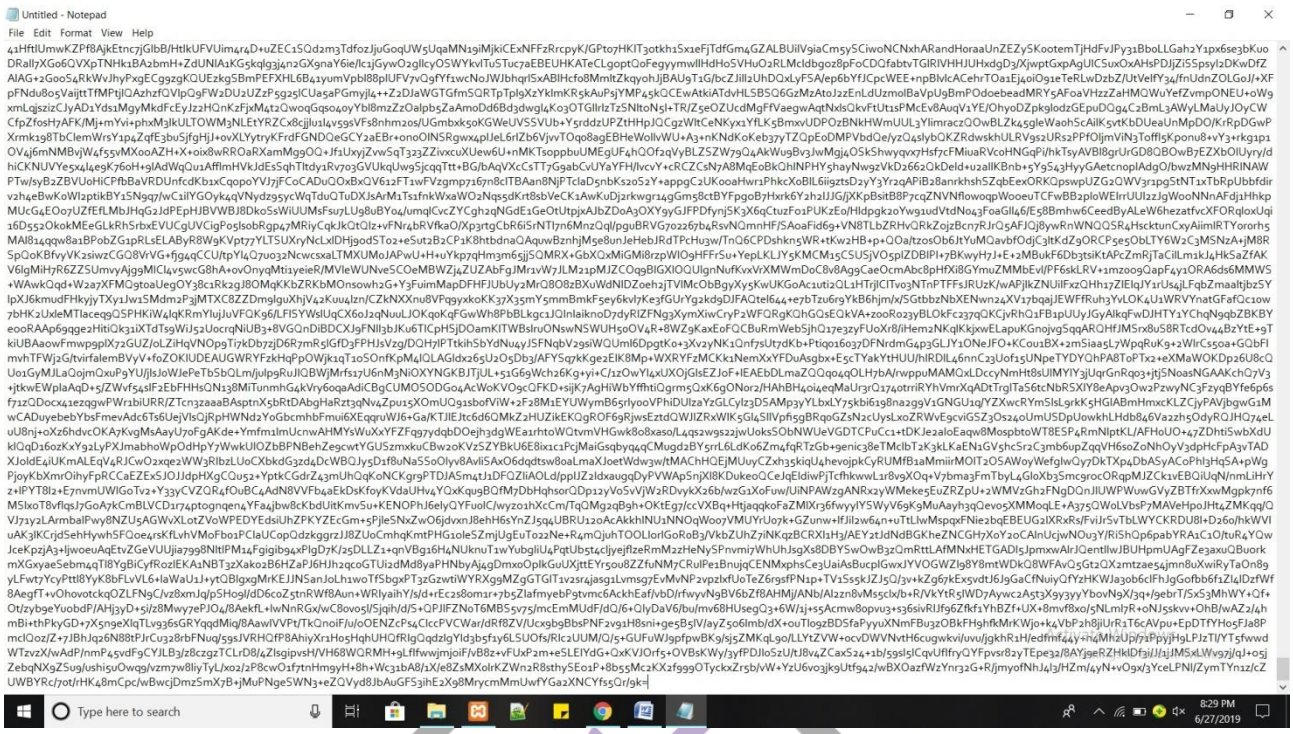

Fig 5.2.2 Image after Encryption Algorithm

1. Fig 5. 2.2 show the encrypted form of the original image.
2. When User wanted to download the image then the original image cannot be download.
3. The original image gets converted into the text form.

Fig 5. 3. Image after Decryption

1. Fig 5. 3 show the original image.
2. When Owner accepts the request from User then and then only User can see the original image.

VI. CONCLUSION

In a newly developed system Ciphertext Policy - Attribute Based Encryption (CP-ABE) is used to share an image in jpeg format. Developed a new algorithm to generate a pair of keys used for encryption and decryption of an image in jpeg format. Three attributes namely domain name in email id, random number and time at the uploading an image are used to generate keys. Confidentiality of the generated keys is improved. It is found that the size of an encrypted image is smaller than an original image in jpeg format. Due to this, there is an improvement in uploading and storage of an encrypted image on a server. If a hacker invades into the image distribution server then he will not be able to get the original image. In a system described in a base paper alpha-numeric data access policy is based on the user role profile. In newly developed system image access policy is based on image user email id and mobile number. Using this system image owner can securely share an image to many not known image user who qualifies image access policy. The users who do not qualify image access policy can send a request to an image owner. Image owner approves the requests. The approved the only requester will get access to the encrypted image.

REFERENCES

[1] Rinki Pakshwaretal, A Survey On Different Image Encryption and Decryption Techniques, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 4 (1) , 2013, 113 – 116

[2] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker, "Mediated Ciphertext-Policy Attribute-Based Encryption and Its Application," Proc.Int‟l Workshop Information Security Applications (WISA „09), pp. 309323,2009.

[3] 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.

[4] A. Sahai and B. Waters. Fuzzy Identity-Based Encryption. In Proc. of EUROCRYPT'05, Aarhus, Denmark, 2005.

[5] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-Based Encryption for Fine-grained Access Control of Encrypted Data. In Proc. of CCS'06, Alexandria, Virginia, USA, 2006.

[6] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-Policy Attribute-Based Encryption. In Proc. of SP'07, Washington, DC, USA, 2007.

[7] L. Cheung and C. Newport. Provably Secure Ciphertext Policy ABE. In Proc. of CCS'07, New York, NY, USA, 2007.

[8] M. Pirretti, P. Traynor, P. McDaniel, and B. Waters. Secure Atrribute-Based Systems. In Proc. of CCS'06, New York, NY, USA, 2006.

[9] A. Boldyreva, V. Goyal, and V. Kumar. Identity-based Encryption with Efficient Revocation. In Proc. of CCS'08, Alexandria, Virginia, USA, 2008.

[10] M. Blaze, G. Bleumer, and M. Strauss. Divertible Protocols and Atomic Proxy Cryptography. In Proc. of EUROCRYPT '98, Espoo, Finland, 1998.

[11] R. H. Deng, J. Weng, S. Liu, and K. Chen. Chosen-Ciphertext Secure Proxy Re-encryption without Pairings. In Proc. of CANS'08, Berlin, Heidelberg, 2008.