FileReCrypt: An Authenticated And Revocable Scheme For Personal File Sharing Using Cloud Storage

Bharati Mishra, Debasish Jena

Abstract: Personal Records such as Identity proofs, birth certificates, and qualification and experience certificates, social claim certificates are essential in numerous scenarios like applying for employment, education, and land and vehicle registration and so on. Cloud platform can be used for this purpose with a number of advantages such as storage-on-demand, robustness, access anywhere anytime from any device. At the same time, the cloud storage provider must ensure confidentiality, integrity, access control and authentication for the documents. Towards this end, an authenticated revocable symmetric proxy re-encryption scheme has been designed in the proposed work using All-Or-NOthing-Transform (AONT) and key homomorphic encryption (KHE) in the elliptic curve group setting. The implementation results show that the performance measures are acceptable for small storage requirements while achieving the security goals.

Index Terms: All-Or-NOthing-Transform, Bilinear Pairing, Cloud Storage, Key Homomorphic Encryption, Personal File Sharing, Symmetric proxy re-encryption, revocation

1 INTRODUCTION

IN personal life, people share several files with various service providers for verification purposes. These personal data files are vulnerable to misuse if not handled with ethics. Cloud storage platform can be used to share such files with the service providers. Sharing the files in plaintext format through cloud may lead to security attacks. Therefore, these files should be encrypted and shared with only authorized service providers. Hybrid encryption schemes can be used for this purpose. In hybrid encryption schemes, files are encrypted using a symmetric key and in turn, the symmetric key is encrypted using a public key scheme. These schemes provide confidentiality of files. However, the service providers should be able to verify the integrity of the shared files. Moreover, the personal files should be shared for a limited time period with the service providers to avoid malicious use. Hence, the encryption key for the file should be changed so that the access to the file can be revoked from the service provider after expiry of the time period. This can be achieved using symmetric key proxy re-encryption In the proposed work, an authenticated and revocable proxy re-encryption scheme has been proposed for personal file sharing using cloud storage. AONT transformation and key-homomorphic encryption operations are used to design the proposed scheme.

The paper is organized as follows. In section 2, existing schemes related to symmetric proxy re-encryption has been discussed. The various techniques used in the proposed work are briefed in section 3. In section 4, the proposed scheme is presented. In section 5 the performance and security analysis of the proposed work is carried out. Concluding remarks is discussed in section 6.

- *Bharati Mishra is currently working as Assistant Professor at IIIT Bhubaneswar, India, PH-916742653364. E-mail: bharati@iiit-bh.ac.in*
- *Debasish Jena is currently working as Associate Professor at IIIT Bhubaneswar, India, PH-916742653319. E-mail: debasish@iiit-bh.ac.in*

2 RELATED WORK

In this section, the existing AONT schemes and proxy re-encryption schemes have been discussed with their relative merits and demerits. Rivest [1], proposed the first AONT scheme. In his scheme, an adversary needs to obtain all the blocks of the ciphertext in order to decrypt a single message block. They proved that, using AONT, brute force key search on symmetric encryption slows down by a factor of the number of blocks in the ciphertext. Later, Stinson [2] proposed an unconditionally secure all-or-nothing-transform. V. Card and T. Van proposed an extension of Rivest AONT[3], where the security of AONT is proved to be independent of the number of blocks in the ciphertext. Anand Desai [4] proposed a new and efficient AONT scheme which has the resistance to exhaustive key search property. Mabo et al. proposed the idea of public key proxy re-encryption [5]. But, it was first formally defined by Blaze et al.[6]. There after a number of public key proxy re-encryption schemes have been proposed [7], [8], [9] where the secret key used to encrypt the file can be re-encrypted for many users using their public key. But, the files encryptd with the symmetric key is not re-encrypted. If the users have downloaded the the encrypted symmetric key, they can still be able to decrypt the file. In order to revoke the access to the file, it needs to be downloaded and encrypted with a new symmetric key. To overcome this problem, Cook et al. [10] proposed the first symmetric proxy re-encryption scheme. They suggested using two encryption keys. The first key is used to encrypt the file by the file owner. The second key is used by the proxy to re-encrypt the file. Later, when key revocation happens, the proxy first decrypts the file using second key and then again re-encrypts with the new key. This involves two costly operations, one for decryption and another for encryption. Horose [11] tried to improve the above scheme by providing a faster method to transform encryption from second key to new key. However, this is not an efficient method, because it also involves two encryption operations. Amril Syalin et al. proposed a symmetric proxy re-encryption scheme using a weak encryption (transform one permutation

to other). They first modified the AONT scheme by Rivest [1]. They suggested using the modified AONT on the file. Then they used a mapping function, that has the ability to transform one permutation two another. They proved the security of their scheme based on the assumption that output of AONT is always random. But this assumption may not hold good, when a user which has access to previous encryption keys. Steven Myers and Adam Shull proposed a hybrid proxy re-encryption scheme [13] using AONT transformation, symmetric encryption and public proxy re-encryption. In the proposed scheme, a key homomorphic encryption is performed on the AONT blocks to achieve the randomness. Later during re-encryption, the key homomorphic property of the encryption is used to change the encryption key.

3 PRELIMINARIES

All-or-Nothing Transformation [1]: All-or-Nothing transform(AONT) is a pre-processing technique before encrypting the message in order to make the brute force search attack on the chosen symmetric key encryption algorithm difficult. Let a message m divided into b1, b2,...bs blocks. A random key K is chosen from the key space 2^n where n is the number of bits in K. The output block sequence is generated as

$$b_i' = b_i \oplus E(K,i) \text{ for i=1,2,..s}$$

$$h_i = b_i' \oplus E(K_0,i) \text{ for i=1,2,..s}$$

$$b_{s+1}' = K \oplus h_1 \oplus h_2..h_s$$

Here K_0 is shared with the authorized user. During decryption, the key is recovered as follows:

$$K = b_{s+1}' \oplus h_1 \oplus h_2..h_s$$

The file blocks are recovered using the key obtained in the above step as follows.

$$b_i = b_i' \oplus E(K,i)$$

The original file is recovered by combining the blocks obtained in above step.

Key Homomorphic Encryption[14]: Let F : K x X be a pseudo random function(PRF). F is called Key-Homomorphic PRF if:

- Given $F(K_1, x)$ and $F(K_2, x)$, there is an efficient polynomial time algorithm to compute $F(K_1 + K_2,x)$.
- Further, $F(K_1 + K_2,x) = F(K_1, x). F(K_2, x)$

4 THE PROPOSED SCHEME

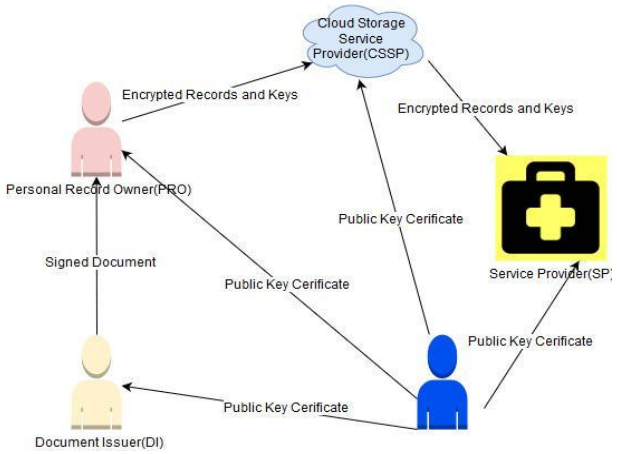

Fig. 1. Health Record Sharing through Cloud Storage

The proposed scheme consists of five types of entities as shown in figure 1.

1) Trusted Authority(TA)
   The TA issues public key certificates to all other entities in the system.
2) Personal Record Owner(PRO)
   The PRO encrypts its personal records and the keys using the proposed scheme and uploads the files into cloud storage. The PRO can be an applicant to an educational institute and needs to submit its date of birth proof and qualifying examination records.
3) Cloud Storage Provider(CSSP)
   The CSSP stores the encrypted files and keys. It also performs the re-encryption operation for the PRO.
4) Service Provider(SP)
   The SP downloads the encrypted personal record blocks and the keys. It uses the decryption algorithm of the proposed scheme to recovers the original personal record. The SP can be a potential employer who would like to verify the academic records of the applicant.
5) Document Issuer(DI)
   The DI is the authority to issue the documents to the PRO. i.e., the Dean of an institution

A. Threat Model

There are primarily three types of adversaries in the threat model of the proposed scheme.

1) The Unauthorized User
   Any user who do not have the valid keys i.e. symmetric key and KH-Encryption key may try to decrypt the file.
2) The Revoked User
   A revoked user who has a previous KH-Encryption key and the symmetric key may try to obtain the plaintext of the file.
3) The Cloud Storage Provider

The Cloud storage provider may try to recover the original file from the encrypted blocks.

The security requirements of the proposed scheme are:

1) File Confidentiality
   Any adversary should not be able to obtain the original file.
2) File Integrity
   Any authorized user should be able to verify the integrity of the file.
3) Re-encryption Indistinguishability
   An adversary should not be able to distinguish an encrypted ciphertext from a re-encrypted one.

System Overview

The proposed scheme consists of following phases.

1) Setup Phase
   In this phase, each user of the system obtains a public key certificate from the TA. The DI issues digitally signed documents to the PRO.
2) Encryption Phase
   This phase consists of two sub phases.
   a) AONT Phase
      In this phase, the PRO uses the AONT encryption algorithm[1] to encrypt its personal records which is a file R of any type. The owner specifies the number of AONT blocks it wants, say n. AONT splits the record into n number of blocks and performs the encryption with a symmetric key K. This results in AONT file blocks b_0, b_1,..b_{n-1} and the canary block b_n calculated as

$$b_n = H(H(b_0)||H(b_1)...||H(b_{n-1}) \oplus K$$

      to verify the integrity of the individual blocks and share the key.

   b) Homomorphic Encryption Phase
      In this sub phase, each AONT file block is passed through the key homomorphic encrypter(KHE) which takes a key K1 as input as shown in figure 2. The KHE transforms the AONT file blocks to a set of points on the chosen elliptic curve E. For this, each AONT file block is padded using PKCS-7 padding [15]. Then the padded block is divided into l message blocks each of size p where p is the bitsize of the chosen elliptic curve. The encryption algorithm is given in detail in algorithm 2.

Algorithm 1 Homomorphic PRF

Require: Elliptic curve E, public parameter X, key K
Ensure: G(X,K)
  P = H(X)
  Compute T = K*P
  Output T

Algorithm 2 Encryption

Require: Elliptic curve E, public parameter X, AONT Block File b_i, key K_1
Ensure: A set of random points on the elliptic curve E
  $b_i^{'}$ = Pad(b_i)
  m[] = Fragment($b_i^{'}$)
  for i: 1 to l **do**
    Q = MessageToPoint($m[i]$)
    P[i] = Q $\oplus$ G(X$\oplus$ i,K_1)
    Output P
  end for

Algorithm 3 Decryption

Require: Elliptic curve E, public parameter X, data points R, key K_1
Ensure: AONT Block File b_i
  for i: 1 to l **do**
    Q = R[i] $\oplus$ G(X$\oplus i$,K_1)
    $m[i]$ = PointToBlock(Q)
  end for
  Output $b_i = m[1]||m[2]..m[l]$

3) Decryption Phase
In the decryption phase, the health service provider recovers the health records of the PRO. It uses the same homomorphic pseudo random function with the key K_1shared to it using asymmetric proxy re-encryption. The output of the KH-PRF is subtracted from the encrypted point to obtain the corresponding AONT block. All the AONT blocks thus obtained, combined to reconstruct the health record as shown in figure 3. The detailed algorithm for decryption process is provided in algorithm 3.

4) Re-encryption Phase
In this phase, if the PRO wants to revoke the access to its personal records from the service provider A, it needs to re-encrypt the files with anew homomorhic key K2. The PRO shall generate a proxy key and provide it to the CSSP. The CSSP shall re-encrypt the ciphertexts with the re-encryption key K=K1- K2 as shown in figure 4. The detailed re-encryption algorithm is presented in algorithm

Algorithm 4 Re-Encryption

Require: Elliptic curve E, public parameter X, data points $R_i[]$,Re-Encryption key ΔK

Ensure: Set of Elliptic Curve Points $R_i'[]$

for i: 1 to l **do**

$R'[i] = R[i] \oplus G(X,\Delta K)$

end for

Output $R'[]$

5 IMPLEMENTATION

The proposed scheme is implemented using Charm frame-work version 10.04 [16]. The security analysis and performance analysis is given in the following sub sections.

A. Security Analysis

1)File Confidentiality:
Here, the file confidentiality requirement is analyzed with respect to the three adversaries.

of the threat model i.e. unauthorized user, revoked user and The cloud storage provider. An unauthorized user does not posses the symmetric key as well as the KH-encryption key. Hence, it cannot recover the AONT blocks and subsequently the original file. The AONT blocks are encrypted using key homomorphic PRF. But the output of PRF is random. Adversary cannot guess the encrypted block.

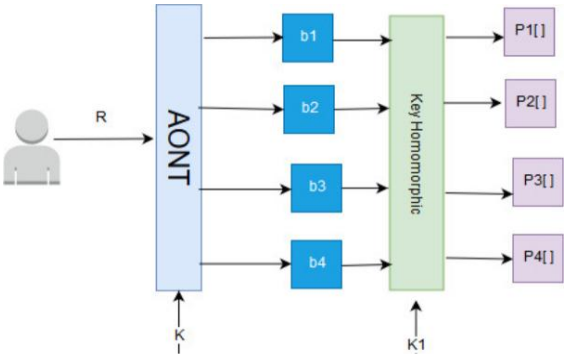

Fig. 2. Encryption Phase

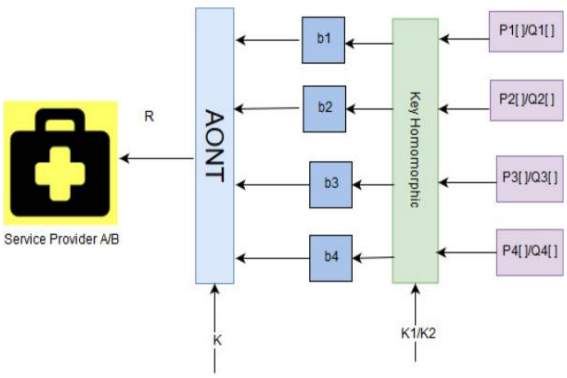

Fig. 3. Decryption Phase

the Decision Diffie-Hellman assumption holds in the chosen EC group G. The revoked user may possess the symmetric key and the revoked KH-Encryption key Ki. Let the present KH-Encryption key be ki+t . Since, the KH-Encryption keys are randomly generated, there is no algorithm to derive ki+t from Ki except the brute force search or breaking the security of the chosen PRF. Neither the symmetric key nor the PRF key is stored in unencrypted form in the cloud storage. The CSSP has only access to the re-encryption token which is an element obtained from subtracting two random numbers in the field Zp.

2) File Integrity: The authorized user first obtains the using the PRF key. It concatenates the AONT blocks and applies the SHA-256 to get a hash output hF . Then it XORs the hF with the encrypted symmetric key, which gives the original symmetric key K used for AONT. If there is a using the PRF key. It concatenates the AONT blocks and applies the SHA-256 to get a hash output hF . Then it XORs the hF with the encrypted symmetric key, which gives the original symmetric key K used for AONT. If there is a result in an invalid key. Therefore, the original file cannot be re-recovered from a modified ciphertext. Hence, integrity of the file can be verified by the authorized service provider. Again, the AONT encryption ensures that, none of the original AONT blocks are missing.

3) Re-encryption Indistinguishably: Let a file block 'b' be encrypted under two keys K1 and K2 and C1 and C2 be the corresponding ciphertext blocks. Let r1 and r2 be the outputs of G(X,K1) and G(X,K2) respectively. Since r1 and r2 are random, hence when added to the data block shall result in two random blocks. Therefore the adversary cannot find out the encryption keys corresponding the encrypted blocks.

B. Performance Analysis

The symbols used in the analysis of the proposed scheme is shown in table 1 with their description.

Table 1 SYMBOLS WITH DESCRIPTION

Symbol	Meaning
Tb p	Time to map a data block to a point on the EC
Tpb	Time to map a a point on the EC to a data block
Tm	Time for Point multiplication with a scalar
Ta	Time for Point addition
TZ	Time for addition or subtraction in the group ZP
Ta	Time for Point addition
a	Key size of the symmetric key encryption algorithm
b	public Key size of the asymmetric key encryption algorithm
c	private Key size of the asymmetric key encryption algorithm
h	no of bits in the output of the hash function
n	Total no of data blocks

The time taken for Encryption operation involves time to map each block of K-bits to a point on elliptic curve, time to execute the homomorphic function (scalar point multiplication), time for point addition and mapping the encrypted point back to a block. During the decryption operation by SP, the same set of operation is performed with additive inverse of the homorphic key. For Re-key generation, subtraction of two integers in prime modulus is performed by the PRO. For proxy re-encryption by CSSP, the operations are same as encryption except the key to be used is the re-encryption key. In table 2, the computation time for various operations are shown.

Table 2 COMPUTATIONAL COMPLEXITY

Operation	Computational Complexity
Encryption	O((Tb p + Tm + Ta + Tpb)n)
Decryption	O((Tb p + Tm + Ta + Tpb)n)
Re-Encryption	O((Tb p + Tm + Ta + Tpb)n)
Re-Key Generation	O(TZ)

Table 3 PROCESSING TIME TAKEN BY VARIOUS OPERATIONS

	File Size		
FileReCrypt Operation	1KB	1MB	1GB
Encrypt	4.28 ms	4.94s	1.15 hours
Decrypt	4.34ms	6.32s	1.85 hours
ReEncrypt	4.35ms	6.64s	1.87 hours
ReKeyGen	3:08 s		

Table 4 STORAGE COST ANALYSIS

Entity	Storage Cost(bits)
PRO	a+b+c
CSSP	F+a+3c+2h
SP	b+c
DI	b+c
TA	b+c

Table 5 COMMUNICATION COST ANALYSIS

Communicating Entities	Encrypt and Upload(bits)	Re-Encrypt (bits)	Download and Decrypt(bits)
PRO-CSSP	F+a+3c+2h	2c+h	F
CSSP-SP	0	0	F+a+3c+2h

The time taken by the Encrypt, Decrypt,Re-Encrypt and Decrypt operation in the proposed FileReCrypt scheme is shown in the table 3. The SECG curve 'prime256v1' is used in for implementation. It is observed that, time taken by various operations scales linearly with the no of blocks in the file. The timings for various operations are higher than AES encryption. It is due to the fact that ECC is a public key cryptography where the crypto-primitives are costlier than symmetric key counterpart. Therefore, the proposed scheme is suitable for sharing files up to few MB. Usually files shared for verification purpose are limited to few MB. So, the proposed scheme meets the performance requirements of the application scenario at hand. The table 4 depicts the storage cost at different entities. Each entity shall store their public and private key for Public key operations. The cloud storage shall store the encrypted file and the encrypted key files. The AONT encryption shall result in n AONT blocks and one canary block. All the AONT blocks together shall be F bits. The canary block is same as the key size of the symmetric encryption which is 'a' bits. The homorphic key is the private key of ECC which is 'c' bits. The caranary block is encrypted and stored in a file. The ECC homomorphic key which is 228 bits for our implementation is stored in another file. In the proposed scheme, cloud storage is used to share these key files. Hybrid encryption with Elgamal public key on ECC group is used for this purpose. Therefore, the encrypted key file becomes [a(AES Key)+3c(homomorphic key+2 * size of point on ECC)+2h(output of hashes for AES key and homomorphic key)] bits which is the additional storage overhead in the proposed scheme. The communication cost incurred between PRO-CSSP and CSSP-SP is shown in table 5. The communication cost involves uploading the encrypted file along with the encrypted key files during encryption scenario. In re-encryption scenario, only the homemorphic key is updated. Hence, additional 2c+h bits needs to be transmitted to the cloud storage for the homomorphic key file. Another 2c+h bits are sent to the CSSP from PRO for the generated re-encryption Key.

6 CONCLUSION

Personal File sharing is essential in numerous real life scenarios for document verification purposes. Confidentiality of these documents is essential. To achieve this security property, files should be encrypted. At the same time to achieve secure sharing and revocation, proxy re-encryption of the files is essential. In the proposed work AONT transformation with homomorphic proxy re-encryption technique has been used. The performance analysis shows that this technique can be used for sharing files with lesser volume up to some MBs. In futur work, these techniques can be improved to work for high volume files.

REFERENCES

[1] R. L. Rivest, "All-or-nothing encryption and the package transform," in International Workshop on Fast Software Encryption. Springer, pp. 210–218, 1997.

[2] D. R. Stinson, "Something about all or nothing (transforms)," Designs, Codes and Cryptography, vol. 22, no. 2, pp. 133–138, 2001.

[3] Canda, Valer, and Tran Van Trung. "A new mode of using all-or-nothing transforms." In Proceedings IEEE International Symposium on Information Theory, p. 296. IEEE, 2002.

[4] Desai, Anand. "The security of all-or-nothing encryption: Protecting against exhaustive key search." In Annual International Cryptology Conference, pp. 359-375. Springer, Berlin, Heidelberg, 2000.

[5] M. Mambo and E. Okamoto, "Proxy cryptosystems: Delegation of the power to decrypt ciphertexts," IEICE transactions on fundamentals of electronics, Communications and computer sciences, vol. 80, no. 1, pp. 54–63, 1997.

[6] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in International Conference on the Theory and Applications of Cryptographic Techniques. Springer, pp. 127– 144, 1998.

[7] M. Jakobsson, "On quorum controlled asymmetric proxy re-encryption," in International Workshop on Public Key Cryptography. Springer, pp. 112–121,1999.

[8] Y. Dodis, "Proxy cryptography revisited," in Proc. 10th Annual Network and Distributed System Security Symposium-NDSS'03, 2003.

[9] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," ACM Transactions on Information and System Security (TISSEC), vol. 9, no. 1, pp. 1–30, 2006

[10] Cool, D. L., and Angelos D. Keromytis. "Conversion and proxy functions for symmetric key ciphers." In International Conference on Information Technology: Coding and Computing (ITCC'05)-Volume II, vol. 1, pp. 662-667. IEEE, 2005.

[11] S. Hirose, "On re-encryption for symmetric authenticated encryption," in Computer Security Symposium (CSS), 2010

[12] A. Syalim, T. Nishide, and K. Sakurai, "Realizing proxy re-encryption in the symmetric world," in International Conference on Informatics Engineering and Information Science. Springer, pp. 259–274, 2011.

[13] S. Myers and A. Shull, "Efficient hybrid proxy re-encryption for practical revocation and key rotation." IACR Cryptology ePrint Archive, vol. 2017, p. 833, 2017.

[14] D. Boneh, K. Lewi, H. Montgomery, and A. Raghunathan, "Key homo-morphic prfs and their applications," in Annual Cryptology Conference. Springer, pp. 410–428,2013.

[15] B. Kaliski, "Pkcs# 7: Cryptographic message syntax version 1.5," Tech. Rep., 1998.

[16] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, and A. D. Rubin, "Charm: a framework for rapidly proto-typing cryptosystems," Journal of Cryptographic Engineering, vol. 3, no. 2, pp. 111–128, 2013.