International Journal of Contemporary Research in Computer Science and Technology (IJCRCST) e-ISSN: 2395-5325
Volume 2, Issue 11 (November '2016)

RANDOM POLYNOMIAL BASED SECRET SHARING SCHEME FOR SECURING KEY IN THE CLOUD ENVIRONMENT

S.Ahamed Ali,
Assistant Professor,
Department of Information Technology,
Velammal Engineering College,
Chennai-66,India

Dr.M.Ramakrishnan,
Professor and Chairperson,
School of Information Technology,
Madurai Kamaraj University,
Madurai,India

Abstract: The concept of secret sharing has been used in the cloud environment to secure the key used for encryption and decryption. The keys are divided into shares and stored in different cloud storage nodes to prevent from unauthorized access. The "n" shares of the key can be regenerated to decrypt data stored in the cloud. The proposed system extends the concept of secret sharing by using our new one way collision resistant hash function which is based on random polynomial. This scheme makes our proposed system to secure against conspiracy attack even if the pseudo shares are compromised. Our proposed system analysis shows that it has better time and space complexity when compared to other secret sharing schemes.

Keywords: Secret sharing, Hash function, Random polynomial, Cloud, Conspiracy attack

I. INTRODUCTION

A secret kept in a single resource can be easily attacked or lost. Secret Sharing (SS) schemes have been proposed since late 1970s. The concept of SS is used to encode the secret into n pieces ("shadows" or "shares") and these n pieces are distributed to n participants or servers which are located at different location. SS has been generally used to overcome the issue regarding the security of key distribution in apublic key cryptosystem. Blakely and Shamir were pioneers in proposing SS solution for safeguarding cryptographic keys. In a (t, n) SS, the dealer divides the secret into n shares and distributes shares to n shareholders in such a way any t or more than t shares can reconstruct this secret; but any t - 1 or fewer than t - 1 sharescannot obtain any information of the secret.

There are a vast number of research papers on the concept of secret sharing. Recently, Harn and Lin (2010) introduced a notion ofstrong t-consistency of a (t, n) SS and proposed a strongverifiable secret sharing (VSS) scheme. The major disadvantage of the above method is that it has high complexity operations like exponentiation, inversion etc. Around the same time another method was introduced by Angsuman Das and Avishek Adhikari which used one way hash function to generate pseudo shares. The use of hash function and 'XOR' operation makes the scheme to overcome the above methods disadvantage. The proposed scheme implements the concept of secret sharing in databases. This scheme uses Shamir's method to divide the secret into shares and it uses one way collision resistant hash function to generate pseudo share from each share. Each of these pseudo shares are distributed to multiple databases located at different locations.

The rest of this work is organized as follows: The related works are discussed in Section 2. The proposed system is discussed in Section 3. The analysis and discussionof the proposed scheme are given in Section 4 and finally, the conclusion is given in Section 5.

II. RELATED WORKS

In Shamir's (t, n) SS scheme, it assumes that a mutually trusted dealer divides the secret into n shares and distributes each shareto corresponding shareholder secretly. Chor et al. (1985) presenteda notion of verifiable secret sharing (VSS). The propertyof verifiability enables shareholders to verify that their shares areconsistent.In Shamir's (t, n) SS (Shamir, 1979), a mutually trusted dealer is responsible to generate shares and distribute each share to correspondingshareholder secretly. Ingemarsson and Simmons (1991)introduced a new type of SS without the assistance of a mutuallytrusted dealer. The basic idea of this type of SS is that each shareholderalso acts as a dealer to select a sub-secret and generateshares for other shareholders. The master secret is the summationof all sub-secrets. However, there is one potential problemin their design. That is, the number of shares kept by each shareholderis proportional to the number of shareholders in the scheme.

Harn and Lin (2010) proposed an approach, (n, t, n) SS in which the first parameter, n refers to the number of dealers, the second parameter, t refers tothe threshold, and the third parameter, n refers to the number ofshareholders, in the SS. Harn and Lin also introduced a new notionof strong t-consistency of shares and proposed a strong (n, t, n) VSS. Angsuman Das and Avishek Adhikari (2010) proposed another method which used one way collision resistant hash function to generate. This method did not use high complexity operations such as interpolation, exponentiation etc., but provided high security due to the use of one way collision resistant hash function.Yan-Xiao Liua, LeinHarnb, Ching-NungYangc, Yu-Qing Zhang (2012) proposed a new efficient (n, t, n) Multi Secret Sharing scheme and Verifiable multi secret sharing scheme. This approach is more efficient than Harn and Lin's strong (n, t, n) Verifiable Secret Sharing.

III. PROPOSED SCHEME

The proposed scheme uses the concept of secret sharing to divide the key into shares and distributing them across different nodes in the cloud. We extend Shamir's (t, n) share generation concept to divide the secret into n shares. The proposed scheme uses one way collision resistant hash function to generate pseudo share from each of the shares.

A) Architecture:

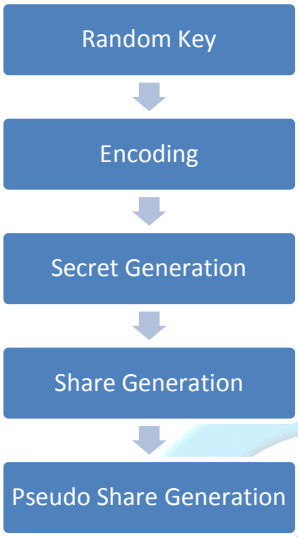

Figure 1: Share Generation Scheme

Initially, the key is encoded using erasure encoding which transforms the key into a longer data with n symbols such that the original message can be recovered from a subset of n symbols. The encoded data is then given as input to one way collision resistant hash function to generate the secret. Then a random polynomial of degree n is used to generate the shares S_i,i=1, 2, …, n, where n refers to the degree of polynomial. The degree of the polynomial depends on the number of share holder nodes used to distribute the key. The coefficients for the polynomial are provided by the user as a secret pin number to get access information from the cloud by decrypting the data block needed. The final step is the generation of pseudo shares. For the share S_i, the shares S_i and S_{i+1} are concatenated and given as input to the hash function. The output from the hash function is XORed with the share S_i to obtain the corresponding pseudo share. Once all the pseudo shares are obtained, they are distributed among n databases.

B) Implementation

Encoding the Key value:

Initially, the key is encoded using the **Erasure** encoding technique.The Erasure encoder takes a block of digital data and adds extra "redundant" bits. Similarly, the Erasure decoder processes each block and recovers the original data. A Erasure code is specified as RS(*n*,*k*) with *s*-bit symbols. The encoder takes *k* data symbols of *s* bits each and adds parity symbols to make an *n* symbol codeword.

There are *n-k* parity symbols of s bits each. AErasure decoder can correct up to *t* symbols, where *2t = n-k*. The following diagram shows a typical Erasure codeword:

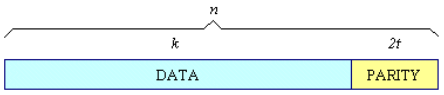

Figure 2: Encoding

The main advantage of using Erasure codes is that the probability of an error remaining in the decoded data is (usually) much lower than the probability of an error if Erasure is not used. This is often described as coding gain. For example, a popular Erasure code is RS(255,223) with 8-bit symbols. Each codeword contains 255 code word bytes, of which 223 bytes are data and 32 bytes are parity. For this code:

$$n = 255, k = 223, s = 8$$

$$2t = 32, t = 16$$

The decoder can correct any 16 symbol errors in the code word: i.e. errors in up to 16 bytes anywhere in the codeword can be automatically corrected.

Secret Generation using Cube Hashing:
The encoded key value is given as input to a one way collision resistant hash function. The proposed polynomial based hashing is used to generate the hash value for the encoded key. The proposed hashing is highly collision resistant and it is highly flexible which can be used to generate a message digest from 128 bits size to 512 bits size based on the requirement. The proposed scheme provides security standards similar to that of SHA hashing with much less computational complexity.

Share Generation using random polynomial:
Once the secret has been generated, it is divided into n secret shares by using a random polynomial of degree n, where n denotes the number of share holder nodes in the cloud in which the secret is distributed. The random polynomial is used because, for each secret to be distributed in the share holder nodes, the polynomial used for generating shares will be different making it difficult for unauthorised users to access the secret. For example, if the number of share holder nodes on which the secret is shared is n=4, then the random polynomial generated will be of the form:

$$\mathbf{f(x)=a_1x^3+a_2x^2+a_3x+a_4}$$

Where, a_1,a_2,a_3,a_4 are random coefficients which are given as input by the user. Then, using the above generated random polynomial, n shares is obtained with the help of the following equation given as:

$$\mathbf{S_i=f(i),}\ \text{where}\ \mathbf{i = 1,2,...,n}$$

Let us consider a random polynomial, say, $5x^3+6x^2+3x+9$, where 5, 6, 3, 9 are random values for a_1, a_2, a_3, a_4 respectively. Now, the secret shares generated are as follows:

$$\mathbf{S_1=2;\ S_2=79;\ S_3=207;\ S_4=437}$$

Generation of pseudo shares using hash function and XOR operation:

Once the shares S_1, S_2, …,S_n are obtained the next step is to compute the pseudo share for each of the sub shares. The pseudo share is computed by using the proposed hashing scheme which is already used in the first step for the generation of secret. For the share S_i, i=1, 2, …, n-1, the shares S_i and S_{i+1} are concatenated and is given as input to the hash function.

For the share S_n, shares S_n and S_1 are given as input to the hash function. Then the output from the hash function is XORed with the share S_i to obtain the pseudo share of share S_i. Once the pseudo share is obtained, the n pseudo shares are stored in n share holder nodes. The hashed values are used to generate n shares for the reconstruction of original data.

Let us consider the following shares

S_1=2; S_2=79; S_3=207; S_4=439

Input to Cube hash function for shares

For $S_1=H(X_1)=H(S_1||S_2)=H(279)$
For $S_2=H(X_2)=H(S_2||S_3)=H(79207)$
For $S_3=H(X_3)=H(S_3||S_4)=H(207439)$
For $S_4=H(X_4)=H(S_4||S_1)=H(4392)$

Once the hash values are obtained, the shares are XORed with the corresponding hash value to obtain the pseudo shares.

The pseudo shares for the above generated shares and hash values are as follows:

$S_{11}=S_1$ XOR $H(X_1)$
$S_{22}=S_2$ XOR $H(X_2)$
$S_{33}=S_3$ XOR $H(X_3)$
$S_{44}=S_4$ XOR $H(X_4)$

These n pseudo shares are stored in n share holder nodes.

Regeneration of the key:

Once all the shares are stored in the share holder nodes using the proposed method, the next step is to make sure that only the authenticated users can access their data. For this the user has to provide the key value and a secret n digit secret number which is the coefficient of the random polynomial of that key. This n digit number is provided to the user as soon as the user's logs on to the cloud. Once the user id and the n digit random number are provided, the user id is encoded using Reed Solomon encoding. The encoded value is given as input to the cube hash function.

The output from the hash function is the required secret. This secret is divided into n shares by using the n digit secret number as the coefficient of the random polynomial. Once the polynomial is generated, it is used to obtain the n shares of the secret. After obtaining the n shares, the pseudo share of each shares are generated. To obtain the pseudo share of share S_i, shares S_i and S_{i+1} are concatenated and is given as input to the cube hash function. The output from the hash function $H(X_i)$ is XORed with S_i to obtain the corresponding pseudo share.

Once the pseudo share is obtained, it is compared with the pseudo shares stored in n databases. If all the pseudo shares are alike, then the user is provided with access to his/her data.

IV. RESULT ANALYSIS:

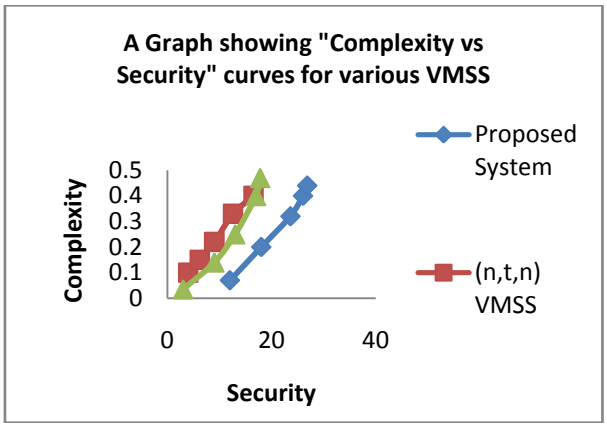

Figure 3: Performance Analysis

The Graph shows the Complexity vs. Security curves for various VMSS in which each curve represents the behavior of complexity and for that how security is varying for that particular Secret sharing system. There are totally 3 curves in the graph in which the Blue curve represents the proposed system, the Green curve represents the Secret Sharing using hashing, and finally, the Red curve represents the(n, t, n) verifiable Multi Secret Sharing(VMSS)

While comparing the proposed system with other VMSS, even for the lowest complexity, the security is higher and as complexity increases, the security level also increases gradually. The proposed system reaches the highest level of security (i.e., 28.5) whereas the rest are stuck at the security level of 20 which is a considerable advantage of the proposed system. Moreover, at the lowest complexity of 0.2 and below, only the proposed system has the largest level for security approximately 12 whereas the other VMSS struggle to go beyond 4.2. In other words, it is clear that the proposed system does not suffer from computation problems faced by other systems.

The high security of the proposed system even at low complexity is due to the use of one way collision resistant hash function. The Cube Hashing that is used in the proposed system has less computational complexity and provides security range of almost equal to Secure Hash Algorithm. The level of security in the proposed system can be improved by increasing the number of share holder nodes in which the pseudo shares are stored or by increasing the size of message digest of the Cube Hashing. This increases the computational complexity to some extent as shown in graph.

V. CONCLUSION

In this paper we have presented a method to improve the key security using secret sharing. It has been emphasized that, unlike in several other schemes, operations like modular multiplication; exponentiation and inversion are not used, thereby reducing the computational cost of the scheme to quite a large extent. As it applies a one-way hash function and the concatenation operation, the proposed scheme is secure against notorious conspiracy attacks even if the pseudo-secret shares are compromised. We hope our scheme will provide better security to key, which is used for encryption and decryption in an untrusted cloud environment

VI.REFERENCES

[1] G.R. Blakley, Safeguarding cryptographic keys, The National Computer Conference 1979, AFIPS 48, 1979, pp. 313_317.

[2] A. Shamir, How to share a secret, Communications of the ACM 22 (1979) 612_613.

[3] Harn, L., Lin, C., 2010. Strong (n, t, n) verifiable secret sharing scheme. Information Sciences 180 (16), 3059–3064.

[4] Ingemarsson, I., Simmons, G.J., 1991. A protocol to set up shared secret schemes without the assistance of a mutually trusted party. In: Advances in Cryptology,Proceedings of the Eurocrypt'90, vol. 473, 21–24 May, Aarhus, Denmark, LNCS.Springer-Verlag, Berlin, pp. 266–282.

[5] Yan-Xiao Liua, LeinHarnb, Ching-NungYangc, Yu-Qing ZhangEfficient (n, t, n) secret sharing schemes, The Journal of Systems and Software 1325-1332

[6] Angsuman Das, AvishekAdhikari, An efficient multi-use multi-secret sharing scheme based on hash functionApplied Mathematics Letters 23 (2010) 993_996.

[7] http://cubehash.cr.yp.to/security.html

[8] http://algo.epfl.ch/~didier/reed_solomon.html