Ultra Secured and Authentic Key Distribution Protocol using a Novel Secret Sharing Technique

Prabir Kr. Naskar Hari Narayan Khan Ayan Chaudhuri Atal Chaudhuri

Department of Computer Science & Engineering
JADAVPUR UNIVERSITY
Kolkata 700032, West Bengal, India

ABSTRACT

Protection of sensitive data is an important issue, precisely during transmission over internet. Efficient cryptographic methods are there to protect data but everything depends on the protection of the encryption key. Threshold cryptography enables the construction of reliable and robust key management system which can reconstruct the key even in the destroy of some shares and on the contrary the key cannot be reconstructed unless a predefined set of shares are been accumulated. The earlier schemes available in literature lead to high computational complexity during both sharing and reconstructing. Here we are suggesting a scheme which employs simple graphical masking method, done by simple ANDing for share generation and reconstruction can be done by simple ORing the qualified set of shares. We are proposing to append secret sharing technique along with conventional cryptography technique for key management to make key management more robust mainly against the chance of compromise and single point failure. Nevertheless it confirms authenticity, confidentiality, non-repudiation and integrity as well.

General Terms: Secret key distribution, Security.

Keywords: Shared cryptography, Key management, Message digest, RSA Crypto system.

1. INTRODUCTION

The secure storage of the private keys in cryptography is an important issue. The possession of an extremely sensitive key by an individual may not be advisable as the key can easily be lost or as the individual may not be fully trusted. Again given copies of key to more than one individual increases the risk of compromise. The obvious solution is to give different shares of the key to several individuals, forcing them to co-operate to find the secret key. This not only reduces the risk of loosing the key but also makes compromising the key more difficult.

In threshold cryptography, secret sharing deals with such problem, namely, sharing a highly sensitive secret among a group of n users so that only when a sufficient number k of them come together, the secret can be constructed. Well known secret sharing schemes (SSS) in the literature include Shamir[1] based on polynomial interpolation, Blakley[2] based on hyper

plane geometry and Asmuth-Bloom[3] based on Chinese Remainder theorem.

A shortcoming of above secret sharing schemes is the need to reveal the secret shares during the reconstruction phase. The system would be more secure if the subject function can be computed without revealing the secret shares or reconstructing the secret back. This is known as function sharing problem where the function's computation is distributed according to underlying SSS such that distributed parts of computation are carried out by individual user and then the partial results can be combined to yield the final result without disclosing the individual secrets. Various function sharing protocols are been proposed [4], [5], [6], [7], [8], [9], [10] mostly based on Shamir's secret sharing as the underlying scheme. Some work [11] is also available on Blakley's secret sharing scheme, and Asmuth-Bloom scheme[12] as well.

In this paper we have suggested an efficient threshold sharing scheme absolutely different from any of the schemes discussed so far, where simple graphical masking (ANDing) technique is used for share generation and reconstruction can be done simply ORing the predefined minimal set of shares. The techniques ANDing and ORing are used for share generation and reconstruction only, they have little contribution towards ciphering. We are proposing to add our novel and simplified method of secret sharing technique along with conventional cryptography technique for key management to make key management more robust mainly against the chance of compromise and single point failure. Our contribution is mainly on designing a new information sharing technique which has substantially less computational overhead compared to all available secret sharing techniques.

Our scheme can be implemented in applications where very low end processors are normally used. The success of the scheme depends upon the mask generation, a step wise algorithm is suggested for such mask design for any (n, k) scheme where n number of masks are designed to generate n different shares and any k shares on ORing reconstruct the original secret.

Here we are suggesting a protocol using our secret sharing scheme and RSA protocol for secured key distribution among a group of users. The scheme also ensures authenticity, confidentiality and non-repudiation for individual shares and finally the integrity of the reconstructed secret key through signature matching using some standard (SHA5/MD5) hash technique. Obviously one is advised to use our secret sharing scheme for total security or some simplified variants of our proposed scheme if one has provision for compromising with the security in terms of authenticity, integrity etc.

2. SECRET SHARING ALGORITHM

The proposed work is based upon an efficient secret sharing scheme which employs simple graphical masking method using simple ANDing for share generation and reconstruction can be done by simple ORing the predefined minimal number of shares. It may be noted that ANDing and ORing are used for share generation and reconstruction only, it do not contribute much towards ciphering. Our purpose of adding secret sharing technique along with conventional cryptography technique for key management is to make key management more robust mainly against the chance of compromise and single point failure. Our contribution is mainly on designing a new information sharing technique which has substantially less computational overhead compared to all available secret sharing techniques.

2.1 Concept

For better understanding let us consider any secret as a binary bit file (i.e. bit is the smallest unit to work upon, in actual implementation one can consider a byte or group of bytes or group of pixels as the working unit). The secret could be an image, an audio or text etc. We shall decompose the bit file of any size onto n shares in such a way that the original bit file can be reconstructed only ORing any k number of shares where $k \leq n \geq 2$ but in practice we should consider $2 \leq k < n \geq 3$.

Our basic idea is based on the fact that every share should have some bits missing and those missing bits will be replenished by exactly (k-1) other shares but not less than that. So every individual bit will be missed from exactly (k-1) shares and must be present in all remaining (n-k+1) shares, thus the bit under consideration is available in any set of k shares but not guaranteed in less than k shares. Now for a group of bits, for a particular bit position, (k-1) number of shares should have the bit missed and (n-k+1) number of shares should have the bit present and similarly for different positions there should be different combinations of (k-1) shares having the bits missed and (n-k-1) number of shares having the bits present. Clearly for every bit position there should be ${}^{n}C_{k-1}$ such combinations and in our scheme thus forms the mask of size ${}^{n}C_{k-1}$, which will be repeatedly ANDed over the secret in any regular order. Different mask will produce different shares (*The style of placing the mask over the secret could be anything but it will be same for every share. It may also be noted that the knowledge of positioning the masks over the secret is not at all required for reconstruction of the secret*.) from the secret. Thus 0 on the mask will eliminate the bit from the secret and 1 in the mask will retain the bit forming one share. Different masks having different 1 and 0 distribution will thus generate different shares. Next just ORing any k number of shares we get the secret back but individual share having random numbers of 1's & 0's reflect no idea about the secret.

As an example a possible set of masks for 5 shares with threshold of 3 shares is shown below:

Share 1: 1 1 1 1 1 1 0 0 0 0
Share 2: 1 1 1 0 0 0 1 1 1 0
Share 3: 1 0 0 1 1 0 1 1 0 1
Share 4: 0 1 0 1 0 1 1 0 1 1
Share 5: 0 0 1 0 1 1 0 1 1 1

One can easily check that ORing any three or more shares we get all 1's but with less than three shares some positions still have 0's i.e. remain missing.

2.2 Algorithm

Here we are presenting the algorithm for designing the masks for n shares with threshold k.

Step-1:
List all row vectors of size n having the combination of (k-1) numbers of 0's and (n-k+1) numbers of 1's and arrange them in the form of a matrix. Obvious dimension of the matrix will be ${}^{n}C_{k-1} \times n$.

Step-2:
Transpose the matrix generated in Step-1. Obvious dimension of the transposed matrix will be $n \times {}^{n}C_{k-1}$. Each row of this matrix will be the individual mask for n different shares. The size of each mask is ${}^{n}C_{k-1}$ bits, i.e. the size of the mask varies with the value of n and k.

(It may be noted that the masking patterns are not unique. Different arrangements of the row vectors in Step-1 leads to different sets of masks but for a particular set, the masks are unique and they satisfy the requirements.)

Let us consider the previous example where n=5 and k=3.

Step-1: List of row vectors of size 5 bits with 2 numbers of 0's and 3 numbers of 1's.

1 1 1 0 0
1 1 0 1 0
1 1 0 0 1
1 0 1 1 0
1 0 1 0 1
1 0 0 1 1
0 1 1 1 0
0 1 1 0 1
0 1 0 1 1
0 0 1 1 1

Dimension of the matrix is ${}^{5}C_{2} \times 5$ i.e. 10×5

Step-2: Take the transpose of the above matrix and we get the desired masks for five shares as listed above in the form of matrix of dimension $5 \times {}^{5}C_{2}$ i.e. 5×10. There are five masks each of size 10 bits.

3. KEY MANAGEMENT PROTOCOL

Here we are presenting stepwise protocol for key management system using the concept of collating a predefined minimal number of shares. We are presuming the situation that a sender is sending multiple shares of a key to multiple recipients and the key can be constructed from a qualified set of shares received by the recipients. For transfer of shares from the sender to any of the recipients is performed by using asymmetric cryptography namely RSA technique. Thus it is presumed that sender and all the recipients have their individual private and public key pairs and the public counterparts are been distributed to others. For integrity, authentication and non-repudiation the concept of

digital signature is implemented using some standard Hash technique.

3.1 Share Generation

Step-1:
Sender encrypts the key, which he wants to send using his private key. The key may be of any length, longer the key better the security.

Step-2:
Sender finds the digest of the encrypted key using any hash algorithm and concatenates at the end of the encrypted key as signature.

Step-3:
Sender is advised to pad some extra random bits at the end of signature to confuse the size of the key (because the size of the signature is always 16 bytes) for strengthening against crypto-analysis. Add two bytes at the beginning to define the size of the key (assumed the maximum size of the key be $\mathbf{2^{16}}$ bits i.e. 8192 bytes). Now onwards we shall refer this file as "SECRET". (*The key size will also be shared, thus individual share will not disclose the size unless the SECRET is reconstructed*)

Step-4:
Select the mask for the first share and repeatedly place it on the SECRET and go on ANDing with the SECRET up to the last bit, which generates the first share. Now encrypt this share again with the public key of the first recipient and send it as first share.

Step-5:
Repeat Step-4 for all other (n-1) shares one after another.

3.2 Key Reconstruction

Step-1:
Any k or more number of recipients out of n recipients should join in key reconstruction with their individual shares.

Step-2:
Each of the recipients joined for key reconstruction will decrypt their individual shares using their individual private keys.

Step-3:
The decrypted shares from k or more recipients will be bit by bit ORed one after another and we shall get the SECRET back.

Step-4:
Once we get the SECRET back, we separate out the two bytes header, the encrypted key and the signature.

Step-5:
Next one will calculate the digest of the encrypted key and match with the signature. If no match (may be due to damage or tampering of some shares during transmission) we can try with some other set of k shares starting from Step-1 otherwise go to Step-6 to get the secret key.

Step-6:
Decrypt the encrypted key using the public key of the sender and that gives the secret key for use by the recipients.

4. EXAMPLE

Let us consider a key 7C235A77 to be shared among 5 recipients in such a way that any 3 of them can reconstruct the key.

Sender's Key pair	Private (169,137)	Public (169,41)
Recipient 1 Key pair	Private (143,179)	Public (143,59)
Recipient 2 Key pair	Private (143,139)	Public (143,19)
Recipient 3 Key pair	Private (143,251)	Public (143,11)
Recipient 4 Key pair	Private (143,227)	Public (143,83)
Recipient 5 Key pair	Private (169,149)	Public (169,29)

The key after encryption using the private key of the sender 8D7881A2

The digest of the encrypted key E3A6A0193AA3A0E7307A5E3C820391AE *(here MD5 is used as Hashing algorithm for ready availability)*

The SECRET	0004	8D7881A2	E3A6A0193AA3A0E7307A5E3C820391AE	F
	Header	Encrypted Key	MD5 Digest of the Encrypted Key	Padding

1st Share 000080780002C3A0A0190A83A0E4300A423080030182F
2nd Share 000405188120C026000130802005007 05C340003118C7
3rd Share 00048D280082A2A4A0081AA2A0A3205A162C820290A6A
4th Share 0004896081A2618220193221806610324C180201902C9
5th Share 00000C5080A023068010282300C330681A0C8200812A4

Next the above shares are individually encrypted by the corresponding public key of the recipient and sent to the concerned recipient.

Encrypted 1st Share	*0000*	13570148	756565672B8A656A032B420313300168	7C
Encrypted 2nd Share	*0045*	561C7062	4B40000171296256005 30E750051863E	13
Encrypted 3rd Share	*0072*	61280082	3F6D7F601A3F7F774C133763822C0138	2B
Encrypted 4th Share	*0048*	0F8A7432	665D4C2654847B473054201355010163	03
Encrypted 5th Share	*0000*	2613A324	9F43A323769F00000300002600003364	31

Next one can verify that any three encrypted shares after decryption (using the corresponding private key) on ORing among themselves give back the SECRET.

Finally from SECRET the encrypted key and the signature are separated out. The same Hash algorithm as used by sender is used to find the digest of the now encrypted key and compared with the signature. After verification the encrypted key is decrypted by the public key of the sender which is available with any recipient, to get finally the original key back.

5. CORRECTNESS OF OUR PROTOCOL

If one looks at the shares in the above example it is evident that the shares are all different from the actual key and there is no one to one relation between the shares and the actual key. Longer the length of the key, better the security and the shares will also be sparser.

The key is first encrypted by the sender's private key and finally opened by sender's public key, which confirms both authenticity and non-repudiation.

The shares are encrypted by individual receiver's public key before transmission and opened by individual receiver's private key at the receiving end, which ensures confidentiality.

Each share contains the share of the signature along with the share of the key. After reconstruction, the signature is compared with the digest of the key so generated to ensure integrity.

It can also be noticed that the n shares are generated only by ANDing n different masks with the SECRET and reconstructed simply by ORing the predefined minimal k number of shares where n and k can be anything $k \leq n > 2$.

6. CONCLUSION

We presented an efficient secret sharing approach with minimal computational overhead. To the best of our knowledge this is the simplest threshold sharing scheme, practically having no computational overhead during both share generation and secret reconstruction. Like other group of researchers the shares can also be sent in some cover images to hide from the active attention of the intruders. However, all of these methods need cover image size bigger than the secret but in our case cover image of same size as the secret is good enough. Our future effort will try to reduce the size of the cover image further i.e. cover image size may be lesser than the secret.

7. ACKNOWLEDGMENTS

We are thankful to the department of Computer Science & Engineering of Jadavpur University, Kolkata, for giving us the platform for planning and developing this work in departmental laboratories.

8. REFERENCES

[1] A. Shamir: "How to share a secret ?" Comm ACM, 22(11):612-613, 1979.

[2] G. Blakley : "Safeguarding cryptographic keys " Proc. of AFIPS National Computer Conference, 1979.

[3] C. Asmuth and J. Bloom :"A modular approach to key safeguarding" IEEE transaction on Information Theory, 29(2):208-210, 1983.

[4] Y. Desmedt "Some recent research aspects of threshold cryptography" Proc of ISW'97 1st International Information Security Workshop vol.1196 of LNCS paper 158-173 Springer-Verlag 1997.

[5] Y. Desmedt and Y. Frankel "Threshold cryptosystems" Proc of CRYPTO'89 volume 435 of LNCS, paper 307-315 Springer Verlag 1990.

[6] Y. Desmedt and Y. Frankel "Shared generation of authenticators and signatures" Proc. of CRYPTO'91 volume 576 of LNCS pages 457-469 Springer Verlag 1992.

[7] Y. Desmedt and Y. Frankel "Homomorphic zero knowledge threshold schemes over any finite abelian group" SIAM journal on Discrete Mathematics 7(4): 667-675, 1994.

[8] H. F. Hua ng and C.C. Chang "A novel efficient (t, n) threshold proxy signature scheme" Information Sciences 176(10): 1338-1349, 2006.

[9] A. De Santis, Y. Desmedt,Y. Frankel and M. Yung "How to share a function securely ?" In proc of STOC 94, paper 522-533, 1994.

[10] V. Shoup "Practical threshold signatures" In Proc of Eurocrypt 2000. volume 1807 of LNCS paper 207-220, Springer-Verlag 2000.

[11] Bozkurt, Kaya, Selcuk, Guloglu "Threshold Cryptography Based on Blakely Secret Sharing" Information Sciences.

[12] K. Kaya and A. A. Selcuk "Threshold Cryptograhy based on Asmuth-Bloom Secret Sharing" Information Sciences 177(19) 4148-4160, 2007