VSPACE: A New Secret Sharing Scheme Using Vector Space

Prasenjit Choudhury[1], Alokparna Banerjee[2], V. Satyanarayana[2], and Gourish Ghosh[3]

[1] Department of Computer Applications, NIT Durgapur, West Bengal, India
[2] Department of Information Technology, NIT Durgapur, West Bengal, India
[3] Tata Consultancy Services, Kolkata, West Bengal, India
prasenjit0007@yahoo.co.in, speak2parna@yahoo.co.in, btech.sandeep@gmail.com, gourishghosh@gmail.com

Abstract. In this paper, a new secret sharing scheme utilizing the vector space is proposed. Initially, the basis vectors of each character of the secret are found on the basis of a random partition value, *N*. Next, the basis vectors of each partition are distributed in separate files and kept well distributed throughout the network. Only an authenticated person who knows the location of the shares can collect all the shares, distributed throughout the network, remove the redundancy and take original basis vectors to reconstruct the actual secret. The scheme is well secured against Brute Force attacks and partial discloser problem. Malicious behaviors are restricted by assigning dummy shares as long as they are not trustworthy.

Keywords: Secret sharing, vector space, basis vectors, MANET, security, cryptosystem.

1 Introduction

Secret sharing (threshold cryptography) is used to divide a cryptographic action, such as the generation and management of a secret key or the computation using secret keys, among several parties, in such a way that the action can be performed if and only if at least a certain number of parties collaborate.

In this paper, a new secret sharing scheme using vector space is proposed. It needs less computation and hence is suitable to meet the security requirements [2] of MANET [1]. It is more secure against eavesdropping, Brute force attack [8] and partial discloser problem.

The rest of the paper is organized as follows. In Section 2, background of the proposed scheme is discussed, followed by a brief introduction about vectors and vector space in Section 3, and vector space associated with a character in Section 4. In Section 5, a new secret sharing scheme is proposed with example. Security of the scheme is analyzed in Section 6, followed by concluding remarks in Section 7.

2 Background

In Shamir's secret sharing scheme [3], a secret (S) is divided into n pieces in such a way that it is easily reconstructable from any k pieces, but even complete knowledge of k-1 pieces reveals absolutely no information about secret. Here, the main goal is to divide S into n pieces S_1, ..., S_n in such a way that knowledge of any k or more S_i pieces of share makes S easily computable. It is not possible to reconstruct S from any k-1 or fewer S_i pieces. Such a scheme is called a (k, n) threshold scheme [4].

Problems with Shamir's scheme. Shamir's secret sharing paradigms suffers with the following problems:

- In (k, n), the value of threshold, k is fixed. If the adversary gains the knowledge of 'any k' number of secrets out of n, he can easily reconstruct the original secret.
- Secret sharing scheme does not depend on individual character of the secret.
- The computations are not feasible.
- The scheme suffers from partial disclosure problem.

In Shamir's (k, n) threshold scheme, the value of k was fixed, but in our approach it is variable. We have distributed shares depending on the reputation [5] [6] of each node. Our scheme is similar to (n, n) secret sharing scheme, although it satisfies the properties of threshold secret sharing scheme.

3 Vectors and Vector Space

A point in a Euclidean plane can be represented by an ordered pair of numbers $X = (X_1, X_2)$ [7]. Point X can also be regarded as a vector emanating from the origin $O = (0, 0)$ to the point (X_1, X_2). In GCF(2), every number can be either 0/1.

Definition. An n-dimensional vector space over the field F consists of:

1. A field F (with its element set S and two operations * and •)
2. A set W of k-tuples (numbers are taken from F)
3. A binary operation '$\oplus$' called vector sum between the elements of the set W, such that W is an abelian group under this operation $\oplus$
4. A binary operation '$\otimes$' called the scalar multiplication between the elements of the set W, such that W is an abelian group under this operation $\otimes$

4 Vector Space Associated with a Character

An ASCII character, C can be represented by 8 tuples, $C= (c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8)$, such that c_i is either 0 or 1[7]. For instance, the character A is represented by 01000001. Altogether, 2^8 or 256 such 8 tuples are possible. For example, say $J \leftarrow C_1$ and $r \leftarrow C_2$. J is represented as 01001010 and r is represented as 01110010.

The ring sum $C_1 \oplus C_2$ is represented by 00111000, which is the ASCII value of '8'.

5 Our Proposal of Secret Sharing

Motivation of our scheme. In the proposed secret sharing scheme, graph theory and basis vectors were used for the distribution of shares. Initially, the binary ASCII value of each character of the share was found. This binary representation of the secret string was then grouped into partitions, depending on a random partition value N, such that each partition P_i has N bits. P_i was then broken down into basis vectors. Let k_i be the number of basis vectors in each P_i. As the number of basis vectors in each P_i is variable, hence k_i is also variable. The basis vectors of each P_i were used to construct the secret shares. Hence, to reconstruct a secret, each partition, P_i must be restored, for which all the k_i basis vectors are needed. Thus, k_i is the threshold of each P_i. But k_i being variable for different partitions, it was more difficult for the adversary to guess all the values of k_i. Moreover, without knowing the value of N (which is a random number known by the dealer only), the adversary could not retrieve the individual characters of the secret. Thus Brute Force attack [8] became very difficult to achieve. Secondly, to reconstruct the shares into original secret, it needed less computation because basis vectors were introduced.

The shares are distributed depending on the reputation [5] [6] of each node. The shares which have more number of basis vectors are distributed among the set of trustworthy nodes, so that they can reconstruct the secret whenever required. The new nodes of the network, whose reputation is, yet to be determined, get the shares with more number of redundant vectors. Moreover, we have used partitioning so that malicious users cannot get partial information from his share or other shares.

Our scheme has the following 3 phases:

5.1 Initial Phase

In this phase, the individual characters of the secret were partitioned based on a random partition value N and then the basis vectors of each partition were found out. It consisted of two functions, *partition()* and *findbvarray()*. The function *partition()* takes *str* as input and divides it into T number of partitions, such that T=*str.length*. The function *findbvarray()* finds the secret shares (basis vectors along with redundancy) of each partition, P and stores them in an array.

Importance of Partitioning

- To eliminate partial disclosure problem a partition value was introduced to convert each character to an un-known format. So, malicious user cannot get partial information from his share or other shares.
- Maximum number of basis vectors in an ASCII character is only 7. So a maximum of 7 shares can be obtained without using the partition technique. By introducing the partition mechanism, any number of shares can be obtained based on the size of the random partition value. Partition value being random, number of shares for the same secret but different partition value is different.

Input. Suppose the secret string: alo & the partition value, N: 10
str:= alo. Fig 1 shows each character of *str* along with its binary ASCII value.

Each character of *str*:=alo	a	l	o
Binary ASCII Value	01100001	01101100	01101111

Fig. 1. ASCII value of each character of *str* for *str*:=alo

Taking N:=10, we find the partitions P_1, P_2 and P_3. We make the size of P_3 equal to 10 (N) by padding it with extra '1's. Fig 2 shows each partition P_i along with its binary ASCII value.

Each Partition P_i	P_1	P_2	P_3
Binary ASCII Value	0110000101	1011000110	1111111111

Fig. 2. ASCII Value of each partition, P_i

Maximum number of Basis Vectors in P_1, P_2 and P_3 is 10. So, M=10= Number of shares. Now we find the basis vectors of each partition. Fig 3 shows the secret shares (basis vectors and redundancy vectors) of each partition P_i.

Each Partition P_i		P_1	P_2	P_3	
Secret Shares of P_i	1	0000000001	0000000010	0000000001	→ To *File1*
	2	0000000100	0000000100	0000000010	→ To *File2*
	3	0010000000	0001000000	0000000100	→ To *File3*
	4	0100000000	0010000000	0000001000	→ To *File4*
	5	0000000101	1000000000	0000010000	→ To *File5*
	6	0010000100	0000000110	0000100000	→ To *File6*
	7	0110000000	0001000100	0001000000	→ To *File7*
	8	0100000101	0011000000	0010000000	→ To *File8*
	9	0010000001	1010000000	0100000000	→ To *File9*
	10	0100000100	1000000110	1000000000	→ To *File10*

□ Basis Vectors ▨ Redundant Vectors

Fig. 3. The Secret Shares corresponding to each partition, P_i

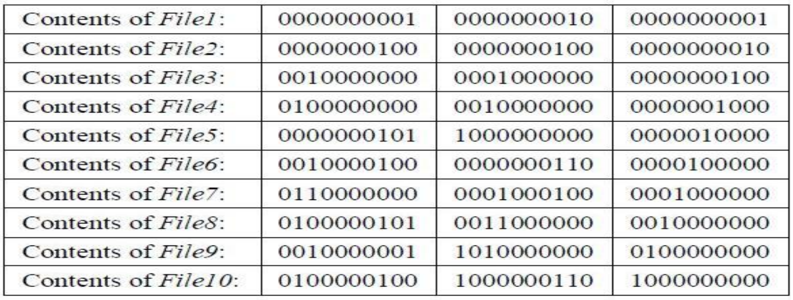
5.2 Distribution Phase

In this phase, the basis vectors of each partition was distributed in separate files and kept well distributed throughout the network. It consisted of the function *distribution()*, which takes *str* as the input and puts the secret shares in files.

The secret shares are then distributed in 10 various files all over the network. Fig 4 shows the contents of *File1* to *File10.*

5.3 Reconstruction Phase

In this phase, only an authenticated person, knowing the location of the shares and having the right authority can collect all the shares, distributed throughout the network, remove the redundancy and take the original basis vectors to reconstruct the



Contents of *File1*:	0000000001	0000000010	0000000001
Contents of *File2*:	0000000100	0000000100	0000000010
Contents of *File3*:	0010000000	0001000000	0000000100
Contents of *File4*:	0100000000	0010000000	0000001000
Contents of *File5*:	0000000101	1000000000	0000010000
Contents of *File6*:	0010000100	0000000110	0000100000
Contents of *File7*:	0110000000	0001000100	0001000000
Contents of *File8*:	0100000101	0011000000	0010000000
Contents of *File9*:	0010000001	1010000000	0100000000
Contents of *File10*:	0100000100	1000000110	1000000000

Fig. 4. Contents of the Secret Share Files

partitions. He can retrieve the individual characters of the secret using *N*. This phase consisted of two functions, *isBasisVector()* and *reconstruct()*. The *isBasisVector()* function takes an integer value and checks whether it is a basis vector or not. If the integer is a basis vector, it returns 1, otherwise it returns 0. The *reconstruct()* function reads the basis vectors from the secret files created by the *distribution()* function and outputs the secret message.

Output. The secret is: alo

6 Security Analysis

As the secret is shared between several users, each can get only a single secret key. Hence it is hard to attack by Brute Force [8]. Moreover, value of *k* being variable for different partitions, it becomes more difficult for the adversary to guess all the values of *k*. But there is chance of Brute Force attack by acquiring partial information from a shared secret. A malicious user can try to get another shared secret with the help of his share of secret. To overcome this problem, we have introduced a random partition value, *N*. Without knowing the value of *N* (which is a random number known by the dealer only), the adversary cannot retrieve the individual characters of the secret. Thus Brute Force attack is very difficult to achieve.

The scheme is also secured against partial discloser problem of secret sharing. Introduction of the random partition value, *N* converts each character of the secret string to an unknown format. So, malicious users cannot get partial information from his share or other shares. Moreover, we can restrict behaviour of malicious users by assigning dummy shares as long as they are not trustworthy.

7 Conclusions

In this paper, we have introduced a new secured scheme for sharing secrets in the network. The secret string was broken down into *shares* with the help of basis vectors. The sharing technique is well secured against partial discloser problem, as well as Brute Force attacks. We have also restricted the behaviour of malicious users by assigning dummy shares to them on the basis of multi-degree or weighted entropy, as long as they are not trustworthy. The scheme takes less computation, compared to the remaining approaches of secret sharing. Thus, it is very useful in MANET.

References

1. Corson, S., Macker, J.: Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations. RFC 2501 (January 1999)
2. Yang, H., Luo, H., Ye, F., Songwulu, Zhang, L.: Security in Mobile ADHOC NETWORKS: Challenges and Solutions. In: IEEE Wireless Communications and Networking conference (2004)
3. Shamir, A.: How to Share a Secret. Communications of ASM 22 (11) (November 1979)
4. Beimel, A., Tassa, T., Weinreb, E.: Characterizing Ideal Weighted Threshold Secret Sharing. SIAM Journal on Discrete Mathematics 22(1) (February 2008)
5. Michiardi, P., Molva, R.: CORE: A Collaborative Reputation Mechanism to Enforce Node Cooperation in Mobile Ad hoc Networks. In: Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security: Advanced Communications and Multimedia Security, pp. 107–121 (September 2002)
6. Bonnefoi, P.-F., Sauveron, D.: MANETS: An Exclusive choice between use and security? Computing and Informatics 22, 1001–1013 (2003) (October 22, 2007)
7. Graph Theory by Narshing Deo [PHI]
8. Brute Force Attack, http://en.wikipedia.org/wiki/Brute_force_attack