SHORT COMMUNICATION REPORT

AN IMPROVED SECRET SHARING ALGORITHM

*ADEWUMI, S. E & GARBA, E. J. D

Department of Mathematics
University of Jos – Nigeria
*(Corresponding author)
adewumisa@unijos.edu.ng
adewumis@gmail.com

Assuming the desire of a 9-man Election Tribunal Judges is to protect their judgment from public knowledge before judgment is delivered. They decided to lock up their judgment in a cabinet. They agree that any 5 judges can open the cabinet to retrieve the document, but not less than 5 can do so. To achieve this, they will need ${}^{9}c_{5} = 126$ padlocks to secure the cabinet. This is prohibitive and impracticable in real life. A secret sharing algorithm will achieve the judges' desire without having to purchase 126 padlocks to secure the cabinet.

A secret sharing scheme is a method of sharing a key *S* into pieces (called shadows) among a set of *p* participants called the shareholders such that any *p* shareholders forming a quorum can reconstruct the value of *S* but no group of less than *p* participants can do so. For example, the control of nuclear weapons in Russia involves two-out-of-three participants (Time Magazine 1992). Usually, a person known as the Dealer (D) $D \notin S$ chooses the key *S*. The dealer gives some partial information called shares to each participant which must be secretly distributed such that no shareholder knows the share that has been given to another shareholder.

When there is need to reconstruct the key *S* some participants' ≤ *p* will pull together their shares to reconstruct *S*. This scheme is represented below by Shamir's (*p,n*) –threshold scheme in Z_m (Stinson 1995).

1. *D* chooses *w* distinct, non-zero elements of Z_m, denoted by $x_i \;\; 1 \le i \le w$. For $1 \le i \le w$, *D* gives the value x_i to participant p_i. The values x_i are public
2. Suppose *D* wants to share a key $S \in Z_m$. *D* secretly chooses (independently at random) *p-1* elements of $Z_m, a_1, \cdots, a_{p-1}$.
3. For $1 \le i \le w$, *D* computes $y_i = a(x_i)$ where $a(x) = S + \sum_{j=1}^{p-1} a_j x^j \bmod m$.
4. For $1 \le i \le w$, *D* gives the share y_i to p_i.

Suppose that participants $p_1, \cdots, p_n$ want to determine *S*. They know that $y_i = a(x_i), 1 \le i \le n$, where $a(x) \in Z_m[x]$ is the (secret) polynomial chosen by *D*. Since a(x) has degree at most p-*1*, a(x) can be written as $a(x) = a_0 + a_1 x + \cdots + a_{p-1} x^{p-1}$, where the coefficients $a_0, \cdots, a_{p-i}$ are unknown elements of Z_m, and $a_0 = S$ is the key.

For a *(p,p)*-threshold scheme, where all participants must pull shares together before *S* can be reconstructed. A *(p,p)*-threshold scheme is described below:

1. *D* secretly chooses (independently at random) p-*1* elements of Z_m, $y_1, \cdots, y_{p-1}$ using a chosen polynomial
2. *D* computes $y_p = S - \left[\sum_{i=1}^{p-1} y_i \right] \bmod m \;\; \cdots (1)$
3. For $1 \le i \le w$, *D* gives the shares y_i to p_i

The formula use by *p* participants to compute *S* can be represented by

$$S = \left[\sum_{i=1}^{p-1} y_i \right] \bmod m \qquad \cdots (2)$$

Shamir (1979) identified some properties of a (*p, n*) threshold scheme as follows:

(1) The size of each piece does not exceed the size of the original data.
(2) When p is kept fixed, D_i pieces can be dynamically added or deleted (e.g., when executives join or leave the company) without affecting the other *D*, pieces. (A piece is deleted only when a leaving executive makes it completely inaccessible, even to himself.)
(3) It is easy to change the D_i pieces without changing the original data *D*

THE METHOD

The method below is used to demonstrate this proposed improved sharing scheme:

1. Compress the document to be shared using the compression algorithm developed by (Adewumi & Garba 2008) into a single code.
2. Share that code amongst *p* participants with each having k[th] part of the code, an alphabet and a percentage representing the given alphabet.

3. To reconstruct the original values

i. Call all the *p* participants to submit their trusted shares, percentage and the alphabet they have
ii. Reconstruct the secret code.
iii. Reconstruct each alphabet representing the *n* splits.
iv. Carry out decompression

To obtain an efficient quadratic polynomial for generating shadow for an integer code, a prime p which must be bigger than the shared code must be obtained. The coefficients $a_0 + a_1 + \cdots + a_{p-1}$ in the polynomial are randomly chosen from a uniform distribution over the integers in [0, m), and the values $p_1, \ldots, p_n$ are computed modulo *m*

An example is considered to demonstrate the workability of this scheme. Assuming the word ATTACK is to be effected at a certain time. Assuming no single individual has the sole right to effect the commencement of an *attack* until a group comes together. To solve this problem, the word *attack* will have to be compressed with the method in (Adewumi & Garba 2008) and then shared as shown in Table 1.

From Table 1, the value to be shared among 4 shareholders is 40. In this example, 4 shareholders are used for convenience to correspond to each shareholder having only one letter (ATTACK can be formed from four letters A, C, K and T). In Table 1, A occurred in positions 1, 4; C appeared in positions 2, 3; with K and T appeared in positions 5 and 6 respectively. A polynomial is now choosing for sharing the code (40) amongst these members. If our polynomial is of the form $f(x) = a_1 x^2 + a_2 + S[\mathrm{mod}\, m]$, where a_1, a_2 are coefficients; *S* is the secret code (in this case 40) and *p* is prime.

If we choose a_1, a_2 to be 7 and 8 respectively with p as 41, then the polynomial will be

$$f(x) = 7x^2 + 8x + 40[\mathrm{mod}\, 41] \ldots (3)$$

TABLE 1. COMPRESSION OF SECRET DOCUMENT

Letter (l_i)	Position of occurrence (α)	Binary strings for th positions	Concatenated binary strings of each l_i	Decimal number equivalent of concatenated binary strings d_i	The length (k) of each binary string	Percentage of l_i in respect to the word being compressed
A	1, 4	001, 100	001100	12	3	30
C	2,3	010,011	010011	17	3	42.5
K	5	101	101	5	3	12.5
T	6	110	110	6	3	15
				$\sum d_i = 40$		

With this polynomial in place, we obtain the shares (shadows) up to p_3 using the (*p,n*) threshold scheme and p_4 can be created using a (*p,p*) threshold scheme and these shares can be given out to each shareholder. With this in place, the 4 shareholders must come together if they need to reconstruct the secret code 40. The shareholders are each given a letter from amongst A, C, K and T, the percentage representing letter share of the key value and the corresponding p_i of the letter. Specifically, the following are given out as shares to each shareholder: a share of the secret code, a single letter of the alphabet (in this case either A, C, K or T), a percentage of their shared alphabet and the string length (in this case 3).

Using the polynomial in equation (3), the shares (p_1, p_2 and p_3) can be generated as follows:

Using our polynomial $f(x) = 7x^2 + 8x + 40[\mathrm{mod}\, 41]$, we obtain

p_1 = 7+8+40 = 14 (mod 41)
p_2 = 28+16+40 = 2 (mod 41)
p_3 = 63+24+40 = 4 (mod 41)

To find the last share (p_4) we use equation (1) which results in

$p_4 = 40 - \left[\sum_{i=1}^{3} p_i\right] \mathrm{mod}\, 41$ = 40 –(14+2+4) mod 41 = 40 – (20 mod 41) = 40-20 = 20

p_4 = 20.

The following shares are handed out:

p_1 = 14
p_2 = 2
p_3 = 4
p_4 = 20.

Specifically, each shareholder receives the following shares as shown in Table 2.

TABLE 2. SHARE TABLE

P_1	14	A	30	3
P_2	2	C	42.5	3
P_3	4	K	12.5	3
P_4	20	T	15	3

If at any point the members want to reconstruct the message, all they need to do is to come together and submit their shares to a trusted individual (may be the dealer of the secret code) who will activate the reconstruction as follows:

First, he uses their secret shares of 14, 2, 4 and 20 to recover the secret code (in this case 40), using equation (2) which is: $S = \left[\sum_{i=1}^{p-1} y_i \right] \bmod p$. We obtain the secret code (S) as 14+2+4+20 (mod 41) = 40.

We now calculate decimal number equivalent of the concatenated binary strings of each letter A, C, K, and T through their percentages as follows: A is $\frac{30}{100} x40 = 12$, C is $\frac{42.5}{100} x40 = 17$, K is $\frac{12.5}{100} x40 = 5$, and T is $\frac{15}{100} x40 = 6$.

Without the knowledge of the secret code 40, the positions of A, C, K, and T cannot be reconstructed.

Having recover those for each alphabet, we decompress them to recover the document using our decompression technique in (Adewumi & Garba 2008)

The scheme provides an improved algorithm for sharing secrets. The scheme can be implemented to secure most government classified documents in public and private places by first compressing the document and then share the code among trusted officers within government ministry or agency. Reconstruction will happen when the participant come together and the dealer reconstruct their shared code, decompress it, only then can they unlock the content of the documents otherwise it will not be possible for a small number of participants to do so.

REFERENCES

Adewumi, S. E. & Garba, E. J. D. 2008. Text Compression Algorithm: A new approach. *Journal of Institute of Mathematics and Computer Sciences,* 19 (1) (in press)

Menezes, P.; Van Oorshot, P. & Vanstone, S. 1997. *Handbook of Cryptography* CRC Press, New York

Time Magazine, May 4 1992:13.

Shamir, A. 1979. *How to share a secret.* Communications of the Association of Computer Machinery 22 (11)

Stinson, D. 1995. *Cryptography: Theory and Practice.* CRC Press, New York