Reusing Shares in Secret Sharing Schemes

Yuliang Zheng
Thomas Hardjono
Jennifer Seberry

The Centre for Computer Security Research
Department of Computer Science
University of Wollongong
Wollongong, NSW 2522, Australia

E-mail: yuliang/thomas/jennie@cs.uow.edu.au
Phone: +61 42 21 4327
Fax: +61 42 21 4329

Submission for *The Computer Journal*

Reusing Shares in Secret Sharing Schemes

Abstract

A (t, w) threshold scheme is a method for sharing a secret among w shareholders so that the collaboration of at least t shareholders is required in order to reconstruct the shared secret. This paper is concerned with the re-use of shares possessed by shareholders in threshold schemes. We propose a simple (t, w) threshold scheme based on the use of cryptographically strong pseudo-random functions and universal hash functions. A remarkable advantage of the scheme is that a shareholder can use a single string in the share of many different secrets, in particular, a shareholder need not be given a new share each time a new secret is to be shared.

Keywords: Cryptography, Information Security, Secret Sharing.

1 Introduction

The problem of maintaining a secret among w shareholders whereby at least t of them are required to cooperate before the secret can be reproduced was first posed by Shamir [10] and Blakley [3]. Since then a number of (t, w) *threshold schemes* have been suggested by researchers in the field of cryptography [12]. These schemes provide the property that by using any t or more pieces of the shared secret, which are called *shares* hereafter, the whole shared secret can be derived, while at the same time maintaining that any $t-1$ or less shares will be insufficient to derive the shared secret. The shared secret itself can be a master key to a cryptographic system, a vault-lock combination, or even a decision which must be arrived at by at least t members in an organisation.

A common drawback of these proposed schemes is that each time when a shared secret is recovered, all shares of the secret including those which did not participate in the recovering process become useless. Therefore each shareholder has to be given a new share when a new secret is to be shared. In this paper we propose a simple (t, w) threshold scheme based on the use of the pseudo-random function family [5] and the universal hash function family [4, 13]. This scheme can remedy the above mentioned drawback. Another remarkable advantage of the scheme is that a shareholder can use a single string in the share of many different secrets.

This paper is organised as follows. Section 2 will discuss the background in the basic constructs necessary for the foundation of the secret sharing scheme. In particular, this will consist of the definitions of pseudo-random function families and universal hash function families. Using these basic constructs, the secret sharing scheme is presented in Section 3, which is followed by an example of the scheme in Section 4. Section 5 discusses security and recycleability of the scheme and Section 6 compares the scheme with that suggested by Shamir [10] together with a discussion on the advantages and disadvantages of the scheme. The paper is closed by some remarks and conclusion in Section 7.

2 Basic Constructs

Denote by $\mathcal{N}$ the set of all positive integers, Σ the alphabet $\{0, 1\}$ and $\#S$ the number of elements in a set S. Denote by n an integer in $\mathcal{N}$ that determines many parameters such

as the length of a shared secret, the length of shares, the security level of a secret sharing scheme and so on. In the literature such an interger n is called a *security parameter*. By $x \in_R S$ we mean that x is chosen randomly and uniformly from the set S. The composition of two functions f and g is defined as $f \circ g(x) = f(g(x))$. Throughout the paper ℓ and m will be used to denote polynomials from $\mathcal{N}$ to $\mathcal{N}$.

We are concerned with collections of functions induced by the security paraneter n. In particular, we are interested in $F = \bigcup_{n \in \mathcal{N}} F_n$, an infinite family of functions, where F_n is a collection of functions from $\Sigma^{\ell(n)}$ to $\Sigma^{m(n)}$, namely, $F_n = \{f | f : \Sigma^{\ell(n)} \to \Sigma^{m(n)}\}$. We call F a function family mapping $\ell(n)$-bit input to $m(n)$-bit output strings. F is said to be *polynomial time computable* if there is a polynomial time (in n) algorithm that computes all $f \in F$, and *samplable* if there is a probabilistic polynomial time algorithm that on input $n \in \mathcal{N}$ outputs uniformly at random a description of $f \in F_n$. Note that if $F = \bigcup_{n \in \mathcal{N}} F_n$ is samplable, then the description of a function in F_n is "compact" in the sense that the length of the description is bounded by a polynomial in n.

2.1 Pseudo-random Function Families

This subsection introduces the concept of pseudo-random functions. Intuitively, a function family $F = \bigcup_{n \in \mathcal{N}} F_n$ is a pseudo-random function family (PRFF) if to a probabilistic polynomial time algorithm, the output of a function f chosen randomly and uniformly from F_n, whose description is unknown to the algorithm, appears to be totally uncorrelated to the input of f, even if the algorithm can choose input for f. The formal definition is described in terms of *(uniform) statistical tests for functions*. A (uniform) statistical test for functions is a probabilistic polynomial time algorithm A that, given n as input and access to an oracle O_f for a function $f : \Sigma^{\ell(n)} \to \Sigma^{m(n)}$, outputs a bit 0 or 1. The algorithm A can query the oracle only by writing on a special tape some $x \in \Sigma^{\ell(n)}$ and will read the oracle answer $f(x)$ on a separate answer-tape. The oracle prints its answer in one step.

Definition 1 *Let $F = \bigcup_{n \in \mathcal{N}} F_n$ be an infinite family of functions, where $F_n = \{f | f : \Sigma^{\ell(n)} \to \Sigma^{m(n)}\}$. Assume that F is both polynomial time computable and samplable. F is a* pseudo-random function family *iff for any statistical test A, for any polynomial Q, and for all sufficiently large n,*

$$|p_n^f - p_n^r| < 1/Q(n) \tag{1}$$

where p_n^f denotes the probability that A outputs 1 on input n and access to an oracle O_f for $f \in_R F_n$ and p_n^r the probability that A outputs 1 on input n and access to an oracle O_r for a function r chosen randomly and uniformly from the set of all functions from $\Sigma^{\ell(n)}$ to $\Sigma^{m(n)}$. The probabilities are computed over all the possible choices of f, r and the internal coin tosses of A.

The concept of pseudo-random functions were first introduced by Goldreich, Goldwasser and Micali in [5]. In the same paper they have also shown that pseudo-random function families can be constructed from pseudo-random string generators [5]. By a result of Impagliazzo, Levin and Luby [7, 6], the existence of one-way functions is sufficient for the construction of pseudo-random function families.

We are interested in a particular type of pseudo-random function families $F = \bigcup_{n \in \mathcal{N}} F_n$, where F_n can be represented by $F_n = \{f_{idx} | idx \in \Sigma^n, f_{idx} : \Sigma^{\ell(n)} \to \Sigma^{m(n)}\}$. For such a

pseudo-random function family, each function in F_n is uniquely indexed by an n-bit string in Σ^n. Thus to select a function from F_n uniformly at random, we only have to choose a random n-bit string.

In practice, such a pseudo-random function family can be easily constructed from a strong data encryption algorithm, plus a bit imagination. The most widely used data encryption algorithm is perhaps the Data Encryption Standard (DES) proposed by FIPS [1]. DES is a strong encryption algorithm that transforms a 64-bit plaintext into a 64-bit ciphertext using a 56-bit key. Without knowing the key, the output of DES appears to be indistinguishable with a random 64-bit string. We can consider DES as a set of functions, $DES_{56} = \{des_{idx} | idx \in Sigma^{56}, des_i dx : \Sigma^{64} \rightarrow \Sigma^{64}\}$, each of which is uniquely specified by a 56-bit string and implements a mapping from $Sigma^{64}$ to $Sigma^{64}$. To finish the construction, we can assume that there is an infinite family of DES-like algorithms, say DES_1, ..., DES_{55}, DES_{56}, DES_{57}, ..., where each DES_i is an encryption algorithm that is designed using the same priciples as for designing DES (=DES_{56}), and transforms an $(i+8)$-bit plaintext into an $(i+8)$-bit ciphertext using an i-bit key.

2.2 Universal Hash Function Families

Universal hash function families (UHFFs) [4, 13] play an essential role in many recent major results in cryptography and theoretical computer science [6, 7, 9]. Let $H = \bigcup_{n \in \mathcal{N}} H_n$ be a family of functions mapping $\ell(n)$-bit input into $m(n)$-bit output strings. For two strings $x, y \in \Sigma^{\ell(n)}$ with $x \neq y$, we say that x and y collide with each other under $h \in H_n$ or x and y are siblings under $h \in H_n$, if $h(x) = h(y)$.

Definition 2 *Let $H = \bigcup_{n \in \mathcal{N}} H_n$ be a family of functions that is polynomial time computable, samplable and maps $\ell(n)$-bit input into $m(n)$-bit output strings. Let $D_n = \{x \in \Sigma^{\ell(n)} | \exists h \in H_n, \exists y \in \Sigma^{m(n)}$ such that $h(x) = y\}$ and $R_n = \{y \in \Sigma^{m(n)} | \exists h \in H_n, \exists x \in \Sigma^{\ell(n)}$ such that $y = h(x)\}$. Let $k \geq 2$ be a positive integer. H is a* (strongly) k-universal hash function family *if for all n, for all k (distinct) strings $x_1, x_2, \ldots, x_k \in D_n$ and all k strings $y_1, y_2, \ldots, y_k \in R_n$, there are $\#H_n/(\#R_n)^k$ functions in H_n that map x_1 to y_1, x_2 to y_2, ..., and x_k to y_k.*

An equivalent definition for the (strongly) k-universal hash function family is that for all k (distinct) strings $x_1, x_2, \ldots, x_k \in D_n$, when h is chosen uniformly at random from H_n, the concatenation of the k resultant strings $y_1 = h(x_1), y_2 = h(x_2), \ldots, y_k = h(x_k)$ is distributed randomly and uniformly over the k-fold Cartesian product R_n^k of R_n. The following *collision accessibility property* is a useful one.

Definition 3 *Let $H = \bigcup_{n \in \mathcal{N}} H_n$ be a family of functions that is polynomial time computable, samplable and maps $\ell(n)$-bit input into $m(n)$-bit output strings. Let $k \geq 1$ be a positive integer. H has the* k-collision accessibility property, *or simply* the collision accessibility property, *if for all n and for all $1 \leq i \leq k$, given any set $X = \{x_1, x_2, \ldots, x_i\}$ of i strings in $\Sigma^{\ell(n)}$, it is possible in probabilistic polynomial time to select randomly and uniformly functions from H_n^X, where $H_n^X \subset H_n$ is the set of all functions in H_n that map x_1, x_2, ..., and x_i to the same strings in $\Sigma^{m(n)}$.*

k-universal hash function families with the collision accessibility property can be obtained from polynomials over finite fields [4, 13]. First we note that there is a natrual one-to-one correspondence between strings in $\Sigma^{\ell(n)}$ and elements in $GF(2)^{\ell(n)}$. This allows us to interchange a $\ell(n)$-bit string and an element in $GF(2)^{\ell(n)}$. Now denote by P_n the collection of all polynomials over $GF(2^{\ell(n)})$ with degrees less than k, i.e.,

$$P_n = \{a_0 + a_1 x + \cdots + a_{k-1} x^{k-1} | a_0, a_1, \ldots, a_{k-1} \in GF(2^{\ell(n)})\} \tag{2}$$

For each $p \in P_n$, let h_p be the function obtained from p by chopping the first $\ell(n) - m(n)$ bits of the output of p whenever $\ell(n) \geq m(n)$, or by appending a fixed $m(n) - \ell(n)$-bit string to the output of p whenever $\ell(n) < m(n)$. Let $H_n = \{h_p | p \in P_n\}$, and $H = \bigcup_{n \in \mathcal{N}} H_n$. Then H is a strongly k-universal hash function family, which maps $\ell(n)$-bit input into $m(n)$-bit output strings and has the collision accessibility property.

3 A New Secret Sharing Scheme

This section describes a new (t, w) threshold scheme for $w = O(\log n)$, where n is the length of a secret to be shared. We assume that each secret K to be shared has a *serial number* N_K. We also assume that the w shareholders have identities $ID_1, ID_2, \ldots, ID_w$ respectively. For simplicity the w shareholders will be denoted by $U_1, U_2, \ldots, U_w$ respectively. In describing the scheme, we assume that there is a trusted *dealer* who holds a secret K to be shared. The scheme will be described in terms of the following three aspects:

1. *Initial Status* of the dealer and the w shareholders.
2. *Dispersing Phase* in which the dealer splits the secret K into w pieces, each of which corresponds to a shareholder, in such a way that at least t of the pieces are required to reconstruct the shared secret K.
3. *Recovering Phase* in which t or more shareholders work together in order to reconstruct the shared secret K.

3.1 Initial Status

Initially, the dealer holds an n-bit secret K to be shared and each shareholder U_i has a n-bit secret key K_i which is randomly chosen by the shareholder. The dealer should determine a pseudo-random function family $F = \{F_n | n \in \mathcal{N}\}$ where $F_n = \{f_{idx} | idx \in \Sigma^n, f_{idx} : \Sigma^{\ell(n)} \rightarrow \Sigma^n\}$ and each function $f_{idx} \in F_n$ is specified by an n-bit string idx. The dealer should also determine a z-universal hash function family $H = \{H_n | n \in \mathcal{N}\}$ which is based on polynomials over finite fields and maps an n-bit input into n-bit output strings and has the collision accessibility property (see Section 2.2), where z, which is to be determined below, denotes the total number of combinations of the w shareholders taken at least t at a time.

3.2 Dispersing Phase

Recall that the number of ways to choose an i-element subset ($0 \leq i \leq w$) from a w-element set is

$$C(w,i) = \left(\begin{array}{c} w \\ i \end{array} \right) = \frac{w!}{i!(w-i)!} \tag{3}$$

From this equation, the number of combinations of the w shareholders taken *at least* t at a time will consist of the following summation:

$$z = \sum_{i=0}^{w-t} \left(\begin{array}{c} w \\ t+i \end{array} \right) = \left(\begin{array}{c} w \\ t \end{array} \right) + \left(\begin{array}{c} w \\ t+1 \end{array} \right) + \cdots + \left(\begin{array}{c} w \\ w \end{array} \right) \tag{4}$$

Denote by $B_1, B_2, \ldots, B_z$ the z different combinations of the w shareholders taken at least t at a time. Note that $w = O(\log n)$ and that $z = O(2^w) = O(2^{c \log n}) = O(n^c)$ for some constant c. For each B_i, we associate it with a w-bit identity G_i. The j-th bit ($1 \leq j \leq w$) of G_i corresponds to the shareholder U_j, and it is set to 1 if and only if U_j is a member of B_i.

The core part of the secret sharing scheme is the following steps taken by the dealer:

1. For each set B_i ($1 \leq i \leq z$), merge the keys $K_{i_1}, K_{i_2}, \ldots, K_{i_j}$ of the shareholders $U_{i_1}, U_{i_2}, \ldots, U_{i_j}$ in B_i together by the use of the following bit-wise exclusive-OR operation:
$$X_i = f_{K_{i_1}}(I_{i,i_1}) \oplus f_{K_{i_2}}(I_{i,i_2}) \oplus \cdots \oplus f_{K_{i_j}}(I_{i,i_j}) \tag{5}$$
where j denotes the number of shareholders in B_i, each $f_{K_{i_l}}(I_{i,i_l})$ is provided to the dealer by shareholder U_{i_l}, and I_{i,i_l} is the concatenation of ID_{i_l}, G_i and N_K (i.e. $I_{i,i_l} = ID_{i_l} \parallel G_i \parallel N_K$). It is assumed that the length in bits of I_{i,i_l} is $\ell(n)$. Note that the secret key K_i of shareholder U_i is used as an index to specify a function in F_n. One reason for the need to use the function f is to ensure that only actual shareholders are able to derive the string X_i. Hence an element of authenticity, in that only shareholder U_i knows K_i, is introduced into the scheme. The key K_i held by shareholder U_i represents a share of the shared secret K.

2. Choose uniformly and randomly from H_n a function h such that the z resulting values $X_1, X_2, \ldots, X_z$ corresponding to the sets $B_1, B_2, \ldots, B_z$ are mapped to the secret K, i.e.
$$h(X_1) = h(X_2) = \cdots = h(X_z) = K \tag{6}$$

3. Make the function h public along with the fact that h is associated with the shared secret with serial number N_K.

3.3 Recovering Phase

When the shareholders $U_{i_1}, U_{i_2}, \ldots, U_{i_j}$ in the set B_i want to reconstruct the shared secret K, they put together $f_{K_{i_1}}(I_{i,i_1}), f_{K_{i_2}}(I_{i,i_2}), \ldots, f_{K_{i_j}}(I_{i,i_j})$, and calculate

$$X_i = f_{K_{i_1}}(I_{i,i_1}) \oplus f_{K_{i_2}}(I_{i,i_2}) \oplus \cdots \oplus f_{K_{i_j}}(I_{i,i_j}) \tag{7}$$

Then they calculate

$$K = h(X_i) \tag{8}$$

which is the shared secret to be recovered.

Using this method any combination of at least t out of the w shareholders can get together corresponding to one of the sets B_i $(i \leq z)$ while maintaining secret their own keys through the use of f. After the shared key K has been used, and thus known to the shareholders in set B_i, it is discarded and a new key K' is selected together with a new function h' from H_n that maps $X'_1, X'_2, \ldots, X'_z$ to K'. Here $X'_1, X'_2, \ldots, X'_z$ represents the new values derived from f due to the change in the serial number N_K to the new serial number $N_{K'}$. More details on these issues are presented in Section 5.

4 An Example

This section presents a practical example that employs the encryption algorithm DES as a pseudo-random function. We consider the case where $w = 4$ shareholders U_1, U_2, U_3 and U_4 are involved in a $t = 2$ secret sharing scheme. The total number of combinations of four shareholders taken (at least) two at a time will be

$$z = \sum_{i=0}^{2} \begin{pmatrix} 4 \\ 2+i \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \end{pmatrix} + \begin{pmatrix} 4 \\ 3 \end{pmatrix} + \begin{pmatrix} 4 \\ 4 \end{pmatrix} = 6 + 4 + 1 = 11. \tag{9}$$

More specifically, we have the following eleven different groups $B_1, B_2, \ldots, B_{11}$:

$$\begin{array}{ll}
B_1 = \{U_1, U_2\}, & G_1 = 1100 \\
B_2 = \{U_1, U_3\}, & G_2 = 1010 \\
B_3 = \{U_1, U_4\}, & G_3 = 1001 \\
B_4 = \{U_2, U_3\}, & G_4 = 0110 \\
B_5 = \{U_2, U_4\}, & G_5 = 0101 \\
B_6 = \{U_3, U_4\}, & G_6 = 0011 \\
B_7 = \{U_1, U_2, U_3\}, & G_7 = 1110 \\
B_8 = \{U_1, U_2, U_4\}, & G_8 = 1101 \\
B_9 = \{U_1, U_3, U_4\}, & G_9 = 1011 \\
B_{10} = \{U_2, U_3, U_4\}, & G_{10} = 0111 \\
B_{11} = \{U_1, U_2, U_3, U_4\} & G_{11} = 1111.
\end{array}$$

In the above table $G_1, G_2, \ldots, G_{11}$ are the idenitities of the groups.

Each shareholder U_i, $1 \leq i \leq 4$, selects a 56-bit random string K_i and keeps it secret. K_1, K_2, K_3 and K_4 will participate in the secret sharing and recovering procedure.

Now suppose that the dealer has a 64-bit key K which he wants to disperse into the four shareholders in such a way that two or more of the shareholders can recover the secret key K at a later stage. We further suppose that K has a 10-bit serial number N_K associated with it, and each shareholder U_i has a unique 50-bit identity ID_i. Thus the length of the concatenation of a 50-bit shareholder identity, a 4-bit group identity and a 10-bit serial number is precisely 64 bits.

The dealer initiates the secret sharing procedure by informing the four shareholders the 10-bit serial number N_K. Note that U_1 is a member of

$$B_1, B_2, B_3, B_7, B_8, B_9, B_{11},$$

U_2 a member of

$$B_1, B_4, B_5, B_7, B_8, B_{10}, B_{11},$$

U_3 a member of

$$B_2, B_4, B_6, B_7, B_9, B_{10}, B_{11},$$

and U_4 a member of

$$B_3, B_5, B_6, B_8, B_9, B_{10}, B_{11}.$$

Upon recieving N_K from the dealer, the four shareholders use the encryption algorithm DES to calculate the following 64-bit strings and pass them over to the dealer.

U_1:

$$\begin{aligned}
X_{1,1} &= des_{K_1}(ID_1||G_1||N_K) \\
X_{1,2} &= des_{K_1}(ID_1||G_2||N_K) \\
X_{1,3} &= des_{K_1}(ID_1||G_3||N_K) \\
X_{1,7} &= des_{K_1}(ID_1||G_7||N_K) \\
X_{1,8} &= des_{K_1}(ID_1||G_8||N_K) \\
X_{1,9} &= des_{K_1}(ID_1||G_9||N_K) \\
X_{1,11} &= des_{K_1}(ID_1||G_{11}||N_K)
\end{aligned}$$

U_2:

$$\begin{aligned}
X_{2,1} &= des_{K_2}(ID_2||G_1||N_K) \\
X_{2,4} &= des_{K_2}(ID_2||G_4||N_K) \\
X_{2,5} &= des_{K_2}(ID_2||G_5||N_K) \\
X_{2,7} &= des_{K_2}(ID_2||G_7||N_K) \\
X_{2,8} &= des_{K_2}(ID_2||G_8||N_K) \\
X_{2,10} &= des_{K_2}(ID_2||G_{10}||N_K) \\
X_{2,11} &= des_{K_2}(ID_2||G_{11}||N_K)
\end{aligned}$$

U_3:

$$\begin{aligned}
X_{3,2} &= des_{K_3}(ID_3||G_2||N_K) \\
X_{3,4} &= des_{K_3}(ID_3||G_4||N_K) \\
X_{3,6} &= des_{K_3}(ID_3||G_6||N_K) \\
X_{3,7} &= des_{K_3}(ID_3||G_7||N_K) \\
X_{3,9} &= des_{K_3}(ID_3||G_9||N_K) \\
X_{3,10} &= des_{K_3}(ID_3||G_{10}||N_K) \\
X_{3,11} &= des_{K_3}(ID_3||G_{11}||N_K)
\end{aligned}$$

U_4:

$$\begin{aligned}
X_{4,3} &= des_{K_4}(ID_4||G_3||N_K) \\
X_{4,5} &= des_{K_4}(ID_4||G_5||N_K) \\
X_{4,6} &= des_{K_4}(ID_4||G_6||N_K) \\
X_{4,8} &= des_{K_4}(ID_4||G_8||N_K) \\
X_{4,9} &= des_{K_4}(ID_4||G_9||N_K) \\
X_{4,10} &= des_{K_4}(ID_4||G_{10}||N_K) \\
X_{4,11} &= des_{K_4}(ID_4||G_{11}||N_K)
\end{aligned}$$

After recieving all the 28 strings from the four shareholders, the dealer derives from them eleven new strings, each corresponds to one of the eleven groups:

$$\begin{aligned}
X_1 &= X_{1,1} \oplus X_{2,1} \\
X_2 &= X_{1,2} \oplus X_{3,2} \\
X_3 &= X_{1,3} \oplus X_{4,3} \\
X_4 &= X_{2,4} \oplus X_{3,4} \\
X_5 &= X_{2,5} \oplus X_{4,5} \\
X_6 &= X_{3,6} \oplus X_{4,6} \\
X_7 &= X_{1,7} \oplus X_{2,7} \oplus X_{3,7} \\
X_8 &= X_{1,8} \oplus X_{2,8} \oplus X_{4,8} \\
X_9 &= X_{1,9} \oplus X_{3,9} \oplus X_{4,9} \\
X_{10} &= X_{2,10} \oplus X_{3,10} \oplus X_{4,10} \\
X_{11} &= X_{1,11} \oplus X_{2,11} \oplus X_{3,11} \oplus X_{4,11}
\end{aligned}$$

Since there are eleven different combinations (groups), the dealer chooses a 11-universal hash function family $H = \{H_n | n \in \mathcal{N}\}$ with the collision accessibility property. As is mentioned earlier, this can be done by letting H_n be the set of all uni-variable polynomials on $GF(2^n)$ with degree not larger than ten. Now the dealer chooses from H_{64} a random function $h(x) = a_0 + a_1 x + \cdots + a_{10} x^{10}$ such that the eleven results $X_1, X_2, \ldots, X_{11}$ are mapped to K in the following way:

$$h(X_1) = h(X_2) = \cdots = h(X_{11}) = K. \tag{10}$$

In determinating h, the dealer should consider two cases: (1) the eleven strings $X_1, X_2, \ldots, X_{11}$ are distinct. (2) the eleven strings $X_1, X_2, \ldots, X_{11}$ are *not* distinct.

In the first case, the function $h(x) = a_0 + a_1 x + \cdots + a_{10} x^{10}$ is uniquely determined by solving the following linear equations on a_0, a_1, ..., a_{10}:

$$\begin{aligned}
a_0 + a_1 X_1 + \cdots + a_{10} X_1^{10} &= K \\
a_0 + a_1 X_2 + \cdots + a_{10} X_2^{10} &= K \\
&\vdots \\
a_0 + a_1 X_{11} + \cdots + a_{10} X_{11}^{10} &= K
\end{aligned}$$

As the eleven strings $X_1, X_2, \ldots, X_{11}$ are distinct, the linear equations have a unique solution.

In the second case, not all the eleven strings $X_1, X_2, \ldots, X_{11}$ are distinct. Without loss of generality, suppose that only the first $1 \leq r < 11$ strings $X_1, X_2, \ldots, X_r$ are distinct. The dealer now chooses $11 - r$ random 64-bit strings $X'_{r+1}, \ldots, X'_{11}$ so that $X_1, X_2, \ldots, X_r$, $X'_{r+1}, \ldots, X'_{11}$ are all distinct. Then the dealer uses the same method as for the first case to solve a set of eleven linear equations on a_0, a_1, ..., a_{10}. This give the function $h(x) = a_0 + a_1 x + \cdots + a_{10} x^{10}$.

Once the function h is determined, the dealer passes over the description of h, namely the eleven coeeficients a_0, a_1, ..., a_{10}, to all the four shareholders. This completes the dispersing phase.

Later when two or more shareholders, say U_1 and U_2, want to recover the shareed secret key K, they put together

$$X_{1,1} = des_{K_1}(ID_1||G_1||N_K)$$

and

$$X_{2,1} = des_{K_2}(ID_2||G_1||N_K)$$

and compute

$$X_1 = X_{1,1} \oplus X_{2,1}.$$

Now the secret key can be recovered by calculating

$$K = h(X_1).$$

After the key K is recovered, X_1, X_2, ..., X_{11} all become un-wanted strings. Nevertherless, shareholder U_1's secret string K_1 which has participated in the sharing procedure of the key K remains un-copromised and is only known to U_1. Similarly, K_2, K_3 and K_4 remain known to U_2, U_3 and U_4 respectively. Hence K_1, K_2, K_3 and K_4 can be used in the sharing of the dealer's new secret keys.

5 Security and Recycleability of the Scheme

This section discusses the following two issues on the scheme: *security* and *recycleability*. Security is mainly concerned with the uncompromisability of the shared secret against the illegal collaboration of $t - 1$ shareholders, while recycleability is concerned with the unpredictability of the keys of shareholders after the reconstruction of the shared secret.

Researchers distinguish two levels of security of a secret sharing scheme. One is called *information theoretic security*, and the other *computational security*. A (t, w) threshold scheme is *information theoretically secure* if the collaboration of $t - 1$ shareholders does not reveal any additional information on the shared secret in the sense of Shannon [11]. Note that in this definition no limitation is imposed on the computational power of shareholders. For this reason, information theoretic security is also called *unconditional security* or *perfect security*. For many practical applications, a looser security, computational security, is enough. A formal definition of computational security is introduced in the following.

Consider a (t, w) threshold scheme, where $w = w(n)$, $1 \leq t = t(n) \leq w$ and n is the security parameter. Denote by $\mathcal{K}_n$ the key space indexed by n and D_n the probability

distribution over $\mathcal{K}_n$ according to which keys from $\mathcal{K}_n$ are chosen. Assume that the computational power of shareholders is bounded by probabilistic polynomial time. Let K be a shared secret which is chosen from $\mathcal{K}_n$ according to D_n. Denote by $p_i(D_n)$ the probability that i $(1 \leq i < t)$ shareholders succeed in extracting the shared secret K by collaboration, and by $p_0(D_n)$ the maximum probability that a probabilistic polynomial time algorithm, which is aware of the probability distribution D_n, succeeds in obtaining K on input n. , Informally, a threshold scheme is computationally secure if the difference between the probabilities p_i and p_0 is negligible. In other words, the illegal collaboration of $i < t$ shareholders brings no advantage in the extraction of the shared secret K.

Definition 4 *Let D_n be a probability distribution on a key space $\mathcal{K}_n$. A (t, w) threshold scheme is* computationally secure with respect to D_n *if for any polynomial Q, for any $1 \leq i < t$ and for all sufficiently large n, $|p_i(D_n) - p_0(D_n)| < 1/Q(n)$. A (t, w) threshold scheme is* computationally secure *if it is computationally secure with respect to the uniform distribution on the key space $\mathcal{K}_n$.*

Our secret sharing scheme uses one-way functions in an essential way, therefore it can not be information theoretically secure. We now prove that it is computationally secure.

Theorem 1 *The (t, w) threshold scheme presented in Section 3 is computationally secure for $w = O(\log n)$.*

Proof : Note that for this scheme, the key space is $\mathcal{K}_n = \Sigma^n$. Therefore, when a shared secret K is chosen uniformly at random from $\mathcal{K}_n$, we have $p_0(D_n) = 1/2^n$, where D_n denotes the uniform distribution on $\mathcal{K}_n$. Now we show that for any $i < n$ and for any polynomial Q we have $p_i(D_n) < 1/Q(n)$ for all sufficiently large n. Suppose for contradiction that there are $i < t$ shareholders who can extract by collaboration the shared secret K with probability $1/Q(n)$. Since instances of the UHFF based on polynomials over finite fields are easily invertible, the i shareholders can obtain the exclusive-OR of the outputs of f specified by keys unknown to the i shareholders. (These keys are possessed by shareholders not collaborating with the i shareholders.) This contradicts the un-predictability of the PRFF. From $p_0(D_n) = 1/2^n$ and $p_i(D_n) < 1/Q(n)$ we have $|p_i(D_n) - p_0(D_n)| < 1/Q(n)$. This completes the proof. □

A secret sharing scheme has recycleability property if no information on the keys of shareholders is released after the re-construction of shared secrets. The recycleability property of our secret sharing scheme follows from the fact that the outputs of instances of PRFF on different inputs look un-correlated to probabilistic polynomial time algorithms. Note that recycleability is equivalent to the property that shareholders can use the same keys in sharing different secrets simultaneously.

6 Comparison with Shamir's Scheme

The scheme suggested by Shamir [10] consists of the division of a shared secret K into w pieces $K_1, K_2, \ldots, K_w$ and the use of a polynomial

$$p(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{t-1} x^{t-1} \tag{11}$$

of degree $t-1$ to disperse the pieces. By placing $a_0 = K$ and evaluating

$$K_1 = p(1), K_2 = p(2), \ldots, K_w = p(w) \tag{12}$$

any subset of t ($t \leq w$) of the K_i values can be used to find the coefficients of $p(x)$ by interpolation and the shared secret K contained in a_0 can then be recovered by simply calculating $p(0) = a_0 = K$. The calculations are done modulo a prime P where $P > w$ and $P > K$, and the coefficients of the polynomial are chosen randomly from the elements of the finite field $GF(P)$.

In our scheme the concept of sharing a secret that is only retrievable by the collaboration of at least t shareholders is fundamentally the same as that suggested by Shamir. However, an important difference lies in the fact that the shareholders do not hold pieces of the secret in the sense of Shamir's scheme. Rather, each shareholder holds a key, any t (at least) of which can be combined together to recreate the shared secret. The keys of the shareholders are maintained as a secret by each shareholder in the same manner that he or she would maintain the secrecy of his or her share in Shamir's scheme. Inherent in our approach is the advantage that the secret key of a shareholder can be selected by him or her, and can be used many times independent of the shared secret.

Another advantage of our approach is the variable length in bits of the shared secret K. In general the shared secret K can be polynomially longer than that of the secret key K_i of each shareholder U_i. This compares favorably with Shamir's scheme where the shared key K and the key K_i of the shareholder U_i are of equal length.

A further advantage lies in the fact that our scheme can be easily adapted to a *general access structure*. The notion of a general access structure refers to the situation where a secret can be divided among a set of shareholders such that any "qualified subset" of the shareholders can reconstruct the secret while the unqualified subsets cannot [8, 2]. The (t, w) threshold scheme is in fact only a special case of the general access structure. It is not clear how Shamir's threshold scheme can be adapted to a general access structure.

Our scheme has a disadvantage in the small number of shareholders w, namely $w = O(\log n)$, where n is the length in bits of the shared secret K. Recall that the number of combinations of the w shareholders taken at least t at a time is

$$z = \sum_{i=0}^{w-t} \left(\begin{array}{c} w \\ t+i \end{array} \right) \tag{13}$$

which is of order $O(2^w)$. In general, for a z-universal hash function family $H = \{H_n | n \in \mathcal{N}\}$, the size of a description of a function $h \in H_n$ is of order $O(n^c z) = O(n^c 2^w)$ which grows exponentially with w, where c is a constant. For practical purposes we must maintain the size of the description of h to be of order $O(n^d)$ for some constant d. This means that we must keep w to be of order $w = O(\log n)$ for the scheme to be practical. However, this restriction does not render the scheme unusable since many practical situations require a small number of shareholders. This is particularly true in the case of a vault in a bank where the authority to open the vault of a bank director may be distributed among a small number of w managers in the form of shares of the key K to the vault. Then at least t of the w ($t \leq w$) managers would be required in order to open the vault when the director is unavailable.

7 Conclusion and Remarks

In this paper we have presented a simple secret sharing scheme based on the pseudo-random function family (PRFF) [5] and on the universal hash function family (UHFF) [4]. The scheme employs combinations of w shareholders taken at least t at a time. These different combinations form a number of sets of shareholders, each of which represents an individual input to an instance of the universal hash function family which maps the inputs to the desired shared secret. The advantage of our approach lies in the freedom of each shareholder to choose his or her own secret key (corresponding to his or her "piece" of the shared secret) and in the re-usability of his or her secret key which is not compromised even when the shared secret is recreated by t or more shareholders.

Our approach to secret sharing has opened a number of avenues for further research. These include research into finding schemes that will remove the restrictions on the size of w and into other mathematical constructs suitable for the formation of secret sharing schemes having recycleable shares.

Acknowledgements

This work was supported by Telecom Australia under the contract number 7027 and by the Australian Research Council under the reference numbers A48830241, A49130102, A9030136, A49131885 and A49232172. The authors would like to thank the anonymous referee for his or her helpful comments.

References

[1] FIPS PUB 46-1. Data encryption standard (DES), 1977. NBS.

[2] J. Benaloh and J. Leichter. Generalized secret sharing and monotone functions. In S. Goldwasser, editor, *Advances in Cryptology - Proceedings of Crypto'88,* Lecture Notes in Computer Science, Vol. 403, pages 27–35. Springer-Verlag, 1990.

[3] G. R. Blakley. Safeguarding cryptographic keys. In *Proceedings of the National Computer Conference,* AFIPS Conference Proceedings, Vol. 48, pages 313–317, 1979.

[4] J. Carter and M. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18:143–154, 1979.

[5] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of ACM*, 33(4):792–807, 1986.

[6] J. Håstad. Pseudo-random generation under uniform assumptions. In *Proceedings of the 22-nd ACM Symposium on Theory of Computing*, pages 395–404, 1990.

[7] R. Impagliazzo, L. Levin, and M. Luby. Pseudo-random generation from one-way functions. In *Proceedings of the 21-st ACM Symposium on Theory of Computing*, pages 12–24, 1989.

[8] M. Ito, A. Saito, and T. Nishizeki. Secret sharing scheme realizing general access structure. In *Proceedings of IEEE Globecom'87, Tokyo, Japan*, pages 99–102, 1987.

[9] J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proceedings of the 22-nd ACM Symposium on Theory of Computing*, pages 387–394, 1990.

[10] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[11] C. E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28:656–715, 1949.

[12] G. J. Simmons. Robust shared secret schemes. *Congressus Numerantium*, 68:215–248, 1989.

[13] M. Wegman and J. Carter. New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, 22:265–279, 1981.