# How to Recycle Shares in Secret Sharing Schemes [1]

Yuliang Zheng
Thomas Hardjono
Jennifer Seberry

Department of Computer Science
University College
University of New South Wales
Australian Defence Force Academy
Canberra, ACT 2600, AUSTRALIA
E-mail: yuliang/thomas/jennie@cs.adfa.oz.au
Phone: (06) 268 8580

[1]Supported in part by Telecom Australia under the contract number 7027 and by the Australian Research Council under the reference number A48830241.

**How to Recycle 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 the pseudo-random function family and the universal hash function family. 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 in [7] and Blakley in [1]. Since then a number of $(t, w)$ *threshold schemes* have been suggested by researchers in the field of cryptography [8]. These schemes provide the property that by using any $t$ or more pieces of the shared secret, which are called *shares* hereafter, the whole secret can be derived, while at the same time maintaining that any $t - 1$ 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 organization.

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 [3] and the universal hash function family [2, 9]. 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 organized 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, followed by an example of the scheme in Section 4. Section 5 compares the scheme with that suggested by Shamir in [7] together with a discussion

on the advantages and disadvantages of the scheme. The paper is closed by some remarks and conclusion in Section 6.

# 2 Basic Constructs

Denote by $\mathcal{N}$ the set of all positive integers, $\Sigma$ the alphabet $\{0,1\}$ and $\sharp S$ the number of elements in a set $S$. Denote by $n$ a *security parameter* which determines many things such as the length of a shared secret, the length of shares, the security level of a secret sharing scheme and so on. 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 $\ell$ and $m$ will be used to denote polynomials from $\mathcal{N}$ to $\mathcal{N}$.

Let $F = \{F_n | n \in \mathcal{N}\}$ be an infinite family of functions, where $F_n = \{f | f : \Sigma^{\ell(n)} \rightarrow \Sigma^{m(n)}\}$. We call $F$ a function family mapping $\ell(n)$-bit input to $m(n)$-bit output strings. $F$ is *polynomial time computable* if there is a polynomial time algorithm (in $n$) computing 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 $f \in F_n$.

## 2.1 Pseudo-random Function Families

Intuitively, a function family $F = \{F_n | n \in \mathcal{N}\}$ 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 $f : \Sigma^{\ell(n)} \rightarrow \Sigma^{m(n)}$, outputs a bit 0 or 1. The algorithm $A$ can query the oracle only by writing on a special tape some $y \in \Sigma^{\ell(n)}$ and will read the oracle answer $f(y)$ on a separate answer-tape. The oracle prints its answer in one step.

**Definition 1** *Let $F = \{F_n | n \in \mathcal{N}\}$ be an infinite family of functions, where $F_n = \{f | f : \Sigma^{\ell(n)} \rightarrow \Sigma^{m(n)}\}$. 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 $n$,*

$$|p_k^f - p_k^r| < 1/Q(n),$$

*where $p_k^f$ denotes the probability that $A$ outputs 1 on input $k$ and access to an oracle $O_f$ for $f \in_R F_n$ and $p_k^r$ the probability that $A$ outputs 1 on input $k$ 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 $A$.*

In [3], it has been shown that pseudo-random function families can be constructed from pseudo-random string generators. By the result of [5, 4], the existence of one-way functions is sufficient for the construction of pseudo-random function families.

## 2.2 Universal Hash Function Families

*Universal hash function families* (UHFFs), which were first introduced in [2] and further developed in [9], have played an essential role in many recent major results in cryptography and theoretical computer science (see for example [4, 5, 6]). Let $H = \bigcup_n H_n$ 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 $H = \bigcup_n H_n$ be a family of functions that is polynomial time computable, samplable and maps $\ell(n)$-bit input into $m(n)$-bit output strings. Let $k$ be a fixed positive integer. $H$ is a* (strong) $k$-universal hash function family *if for all $n$, for all $k$ (distinct) strings $x_1, x_2, \ldots, x_k \in \Sigma^{\ell(n)}$ and all $k$ strings $y_1, y_2, \ldots, y_k \in \Sigma^{m(n)}$, there are $\sharp H_n / 2^{km(n)}$ functions in $H_n$ that map $x_1$ to $y_1$, $x_2$ to $y_2$, ..., $x_k$ to $y_k$.*

The following definition of the *collision accessibility property* is presented due to its importance to our secret sharing scheme.

**Definition 3** *Let $H = \bigcup_n H_n$ be a family of functions that is polynomial time computable, samplable and maps $\ell(n)$-bit input into $m(n)$-bit output strings. Let $k$ be a fixed positive integer. $H$ has the* collision accessibility property *if for all $n$ and for all $1 \leq j \leq k$, given any set $X = \{x_1, x_2, \ldots, x_j\}$ of $j$ initial 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 $x_j$ to the same strings in $\Sigma^{m(n)}$.*

Strong $k$-universal hash function families with the collision accessibility property can be obtained from polynomials over finite fields [2, 9]. We denote by $P_n$ the collection of all polynomials over $GF(2^{\ell(n)})$ with degree less than $k$. That is,

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

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 $H = \bigcup_n H_n$. Then $H$ is a strong $k$-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* $D$ 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 $D$ and the $w$ shareholders.
2. *Dividing Phase* in which the dealer $D$ divides the secret $K$ into $w$ pieces so that at least $t$ of the pieces are required to reconstruct the secret $K$.
3. *Recovering Phase* in which $t$ or more shareholders work together in order to reconstruct the shared secret.

## 3.1 Initial Status

Initially, the dealer $D$ 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 $D$ should determine a pseudo-random function family $F = \{F_n | n \in \mathcal{N}\}$ where $F_n = \{f_K | f_K : \Sigma^n \rightarrow \Sigma^{\ell(n)}, K \in \Sigma^n\}$ and each function $f_K \in F_n$ is specified by an $n$-bit string $K$. $D$ should also determine a $z$-universal hash function family $H = \{H_n | n \in \mathcal{N}\}$ which maps an $n$-bit input into $n$-bit output strings and has the collision accessibility property, where $z$, which is to be defined below, denotes the total number of combinations of the $w$ shareholders taken at least $t$ at a time.

## 3.2 Dividing Phase

Let

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

be defined as the number of combinations or selections of $w$ shareholders taken $i$ at a time ($0 \leq i \leq w$). From this definition, the number of combinations of the $w$ shareholders taken *at least* $t$ at a time will consist of the following summation:

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

Denote by $B_1, B_2, \ldots, B_z$ the $z$ different combinations of the shareholders taken at least $t$ at a time. Note 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 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 $D$:

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 exclusive-OR operation:
$$X_i = f_{K_{i_1}}(I_{i,i_1}) \oplus f_{K_{i_2}}(I_{i,i_2}) \oplus \cdots \oplus f_{K_{i_j}}(I_{i,i_j}) \tag{1}$$
where each $f_{K_{i_l}}(I_{i,i_l})$ is provided to $D$ by shareholder $U_{i_l}$, and $I_{i,i_l}$ is the concatenation of $ID_{i_l}$, $G_i$ and $N_K$. That is, $I_{i,i_l} = ID_{i_l} \parallel G_i \parallel N_K$. It is assumed that the length in bits of $I_{i,i_l}$ is $\ell(n)$. 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 $U_i$ is able to produce $K_i$, is introduced into the scheme. The key $K_i$ represents the share of the secret held by shareholder $U_i$.

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.
$$h(X_1) = h(X_2) = \cdots = h(X_z) = K \tag{2}$$

3. Make the function $h$ public along with the fact that $h$ is associated with the 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

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

Then they calculate

$$K = h(X_i)$$

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 $N_{K'}$.

### 3.4 Security of the Scheme

The security of our secret sharing scheme lies in its use of the pseudo-random function family and, to a certain extent, the universal hash function family. In the recovering phase, each shareholder $U_j$ must submit their piece of the secret $K_j$ in the form of the string $f_{K_j}(I_{i,j})$ before a group $B_i$ of (at least) $t$ shareholders could recover $K$. Hence, the main intention of attackers who want to recover the shared secret illegally would be to obtain the pieces of the secret belonging to the honest shareholders in the group.

More specifically, assume that there is an infinite subset $\mathcal{N}'$ of $\mathcal{N}$ and a polynomial $Q$, such that for each $n \in \mathcal{N}'$ there is a shareholder $U_i$ that can find with probability $1/Q(n)$ the key $K_j$ $(i \neq j)$ belonging to another shareholder $U_j$. This means that there is an probabilistic polynomial time algorithm that can predict with probability $1/Q(n)$ the pseudo-random function family for all $n \in \mathcal{N}'$. This is a contradiction to the definition of pseudo-random function families. Hence, the scheme is secure. A more extensive discussion on the security of the scheme will be provided in the final paper.

# 4 An Example

As an example consider the case where $w = 4$ shareholders $U_1, U_2, U_3$ and $U_4$ with keys $K_1, K_2, K_3$ and $K_4$ are involved in a $t = 2$ secret sharing scheme. Thus the combination of $w = 4$ shareholders taken (at least) $t = 2$ at a time will result in:

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

sets $B_1, B_2, \ldots, B_{11}$ where

$$\begin{aligned}
B_1 &= \{U_1, U_2\}, \\
B_2 &= \{U_1, U_3\}, \\
B_3 &= \{U_1, U_4\}, \\
B_4 &= \{U_2, U_3\}, \\
B_5 &= \{U_2, U_4\}, \\
B_6 &= \{U_3, U_4\}, \\
B_7 &= \{U_1, U_2, U_3\}, \\
B_8 &= \{U_1, U_2, U_4\}, \\
B_9 &= \{U_1, U_3, U_4\}, \\
B_{10} &= \{U_2, U_3, U_4\}, \\
B_{11} &= \{U_1, U_2, U_3, U_4\}.
\end{aligned}$$

From these sets of combinations, the input strings corresponding to these sets can be derived as follows:

$$\begin{aligned}
X_1 &= f_{K_1}(I_{1,1}) \oplus f_{K_2}(I_{1,2}) \\
X_2 &= f_{K_1}(I_{2,1}) \oplus f_{K_3}(I_{2,3}) \\
X_3 &= f_{K_1}(I_{3,1}) \oplus f_{K_4}(I_{3,4}) \\
X_4 &= f_{K_2}(I_{4,2}) \oplus f_{K_3}(I_{4,3}) \\
X_5 &= f_{K_2}(I_{5,2}) \oplus f_{K_4}(I_{5,4}) \\
X_6 &= f_{K_3}(I_{6,3}) \oplus f_{K_4}(I_{6,4}) \\
X_7 &= f_{K_1}(I_{7,1}) \oplus f_{K_2}(I_{7,2}) \oplus f_{K_3}(I_{7,2}) \\
X_8 &= f_{K_1}(I_{8,1}) \oplus f_{K_2}(I_{8,2}) \oplus f_{K_4}(I_{8,4}) \\
X_9 &= f_{K_1}(I_{9,1}) \oplus f_{K_3}(I_{9,3}) \oplus f_{K_4}(I_{9,4}) \\
X_{10} &= f_{K_2}(I_{10,2}) \oplus f_{K_3}(I_{10,3}) \oplus f_{K_4}(I_{10,4}) \\
X_{11} &= f_{K_1}(I_{11,1}) \oplus f_{K_2}(I_{11,2}) \oplus f_{K_3}(I_{11,3}) \oplus f_{K_4}(I_{11,4})
\end{aligned}$$

Let $H = \{H_n | n \in \mathcal{N}\}$ be a 11-universal hash function family with the collision accessibility property. Following the above calculation, a function $h$ is chosen uniformly and randomly from $H_n$ such that the eleven results $X_1, X_2, \ldots, X_{11}$ are mapped to $K$ in the following way:

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

# 5 Comparison with Shamir's Scheme

The scheme suggested by Shamir in [7] 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

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

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

$$K_1 = p(1), K_2 = p(2), \ldots, K_w = p(w)$$

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 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 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]. 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

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

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 the description of an 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 the number of shareholders to be small. This is particularly true in the case of a vault in a bank where the authority to open the vault of one 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.

# 6 Conclusion and Remarks

In this paper we have presented a simple secret sharing scheme based on the pseudo-random function family (PRFF) [3] and on the universal hash function family (UHFF) [2]. 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 individual inputs to the instance of a universal hash function family mapping to the desired shared secret. The scheme differs from the traditional approach suggested by Shamir in [7] in that no pieces of the secret are actually dispersed among the $w$ shareholders. The advantage of our approach lies in the freedom of each shareholder to choose their own secret key (corresponding to their "piece" of the shared secret) and in the life time of their secret key which need not be renewed each time the shared secret is recreated by $t$ or more shareholders. Our approach still maintains the important criteria that the collaboration of $t-1$ or less shareholders will not result in the compromise of the shares of the remaining shareholder(s).

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.

# References


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

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

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

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

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

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

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

[8] Simmons, G. J. Robust shared secret schemes. *Congressus Numerantium 68* (1989), 215–248.

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