Journal of Universal Computer Science, vol. 5, no. 1 (1999), 20-31
submitted: 13/5/98, accepted: 7/1/99, appeared: 28/1/99 © Springer Pub. Co.

Fast Hashing and Rotation-Symmetric Functions

Josef Pieprzyk and Cheng Xin Qu
Centre for Computer Security Research
School of Information Technology and Computer Science
University of Wollongong
Wollongong, NSW 2522, AUSTRALIA
email: josef/cxq01@cs.uow.edu.au

Abstract: Efficient hashing is a centerpiece of modern Cryptography. The progress in computing technology enables us to use 64-bit machines with the promise of 128-bit machines in the near future. To exploit fully the technology for fast hashing, we need to be able to design cryptographically strong Boolean functions in many variables which can be evaluated faster using partial evaluations from the previous rounds. We introduce a new class of Boolean functions whose evaluation is especially efficient and we call them rotation symmetric. Basic cryptographic properties of rotation-symmetric functions are investigated in a broader context of symmetric functions. An algorithm for the design of rotation-symmetric functions is given and two classes of functions are examined. These classes are important from a practical point of view as their forms are short. We show that shortening of rotation-symmetric functions paradoxically leads to more expensive evaluation process.

1 Introduction

Hashing algorithms are important cryptographic primitives which are indispensable for an efficient generation of both signatures and message authentication codes [23]. They are also widely used as one-way functions in key agreement and key establishment protocols [9]. Hashing can be designed using either block encryption algorithms or computationally hard problems or substitution-permutation networks (S-P networks).

Parameters of hashing algorithms based on block encryption algorithms, are restricted by properties of underlying encryption algorithms. Assume that an encryption algorithm operates on n-bit strings. A single use of the cipher produces n-bit hash value. This means that the n-bit strings have to be at least 128-bit long. Otherwise, the hash algorithm is subject to the birthday attack. The attack finds colliding messages in $2^{n/2}$ steps with a high probability (larger than 0.5). If the hash algorithm applies more than one encryption, it becomes slower than underlying cipher. The use of a "strong" encryption algorithm does not guarantee a collision-free hash algorithm. There have been many spectacular failures that prove the point [13].

Design of hashing algorithms using intractable problems can be attractive as the security evaluation can sometimes be reduced to the proof that finding a collision is as difficult as solving an instance of a computationally hard problem. Numerous examples have shown that the application of hard problems does not automatically produce sound hash algorithms. The misunderstanding springs from the general characterisation of the problem. For example, a problem is considered to be difficult if belongs to the **NP-complete** class [6]. Any problem is a collection of instances. Some of them are intractable but some are easy. If a hash algorithm applies easy instances, it is simply insecure. The main shortcoming of this class of hash algorithms is that they are inherently slow.

The class of hash algorithms based on S-P networks includes fastest algorithms. They apply the well-known concept of *confusion* and *diffusion* introduced by Shannon [22]. Representatives of this class are MD4 [15], MD5 [16], SHA [17] and many others [19]. Despite of demolishing MD4 and weakening MD5 by Dobbertin [2, 3], their structural properties look sound and they are frequently used as benchmarks for efficiency evaluation.

2 Motivation

The MD family of hash algorithms uses the Feistel structure [3, 11]. The structure can be defined as follows. Let the input be (L_{i-1}, R_{i-1}) and the output be (L_i, R_i). Then $L_i = R_{i-1}$ and $R_i = L_{i-1} \oplus f(R_{i-1}, K_{i-1})$, where the function f is controlled by the subkey K_{i-1} and $\oplus$ stands for bitwise XOR. Rivest used a modification of the structure for his MD4 and MD5 algorithms. A single iteration is described as

$$\begin{array}{l} A_i = D_{i-1};\, B_i = A_{i-1} + F(B_{i-1}, C_{i-1}, D_{i-1}) + m_{i-1}; \\ C_i = B_{i-1};\; D_i = C_{i-1}, \end{array}$$

where (A_i, B_i, C_i, D_i) is a 128-bit string split into four 32-bit words defined for the i-th iteration, $F : \{0,1\}^{96} \to \{0,1\}^{32}$ is a function which takes three 32-bit words and generates a 32-bit output word, m_i is the message hashed in the i-th iteration and "+" stands for addition modulo 2^{32}. In fact, the function F is a collection of 32 Boolean functions evaluated in parallel using bitwise binary operations. Note that rotation has been ignored. For efficiency reasons, the function F is generated on the fly by using bitwise operations such as ˆ, &, | accessible in C/C++ languages.

In general, we can view a hashing algorithm as a sequence of iterations. A single iteration takes an input $X = (X_k, \ldots, X_0)$ and a message word (block) M (for the sake of simplicity we assume that M has been already merged with the corresponding constant) and produces the output $Y = (Y_k, \ldots, Y_0)$ according to

$$Y_0 = M + F(X_{k-1}, \ldots, X_0) + ROT(X_k, s) \text{ and } Y_{i+1} = X_i \qquad (1)$$

for $i = 0, \ldots, k-1$, where words or blocks are n-bit sequences ($n = 32, 64, 128, \ldots$), + stands for addition modulo 2^n and $ROT(X_k, s)$ is circular rotation of the word X_k by s position to the left. Assume that we have a parallel machine and we wish to examine how fast the iteration (1) can be produced. Parallel implementations of MD4/MD5 are used as benchmarks. For the sake of clarity, we assume that all bitwise operations, addition modulo 2^n and the rotation ROT take one instruction. In our analysis, we ignore all initial steps necessary to setup hashing.

The computational complexity of a single iteration (1) equals the number of instructions necessary to produce Y_0. The evaluation of the function F seems to be the major component. Note that the function can be evaluated after X_0 is known. The evaluation of X_0 can be done concurrently with the evaluation of two parts of the function F as

$$F(X_{k-1}, \ldots, X_0) = G_1(X_{k-1}, \ldots, X_1) \oplus X_0 G_0(X_{k-1}, \ldots, X_1).$$

where XOR and logical multiplication are done bitwise. The correctness of the above representation is justified at the beginning of the next section. When X_0, G_0 and G_1 are available then the function F can be evaluated using two instructions: one to produce X_0G_0 and the second to generate the final evaluation. To obtain Y_0, one would need a single addition only as the rotation and $M + ROT(X_k, s)$ can be executed in parallel. All together, a single iteration of any member of the MD family takes three instructions assuming that the evaluation of G_1 and G_0 can be done in parallel [1]. This is the absolute upper bound for efficiency of hashing with members of the MD family. Can we do better ?

Before we answer the question, let the efficiency of hashing algorithms be expressed by the number of bits of a compressed message per instruction. The MD4 speed is then $\frac{512}{48*3} = 3.55$ bits of compressed message per instruction. The length of message block in MD4 is 512 bits, the number of instructions is 144 (48 rounds and each round takes 3 instructions). Consider an algorithm implemented on a 64-bit machine. Assume that the algorithm takes 4096-bit messages and compresses them into 1024-bit digests using 3 passes with 64 iterations each. Its speed is $\frac{4096}{192*3} = 7.1$ so twice as fast as MD4 (and seems to be much more secure as it employs 192 iterations). The crucial issue becomes the design of the function F which needs to be based on a Boolean function in 15 variables.

3 Definition of Rotation-Symmetric Boolean Functions

Let n be a positive integer and $V_n = \{0,1\}^n$ be the space of binary vectors. Consider a Boolean function $f : V_n \to V_1$. The function can be uniquely represented as

$$f(x) = f(x_1, \ldots, x_n) = g_1(x_1, \ldots x_{n-1}) \oplus x_n g_0(x_1, \ldots x_{n-1})$$

where
$g_1(x_1, \ldots x_{n-1}) = f(x_1, \ldots, x_{n-1}, 0)$ and $g_0(x_1, \ldots x_{n-1}) = f(x_1, \ldots, x_{n-1}, 1) \oplus g_1(x_1, \ldots x_{n-1})$. Clearly, to check that the representation is correct it is enough to check whether it holds for both $x_n = 0$ and $x_n = 1$. Let us study the relation between functions $f(x)$ and $f(y)$ used in two consecutive iterations. The rotation operation binds variables y_i with x_i according to the following assignments $y_{i+1} = x_i$ for $i = 1, 2, \ldots, n-1$. Note that y_1 is evaluated after the final evaluation of $f(x)$ and is equal to $y_1 = m + f(x) + c$ where m is a binary message and c is a bit extracted from a block X_n. After substituting $y_{i+1} = x_i$ for $i = 1, 2, \ldots, n-1$, the function $f(y)$ becomes

$$f(y) = f(y_1, \ldots, y_n) = h_1(x_1, \ldots, x_{n-1}) + y_1 h_0(x_1, \ldots, x_{n-1}).$$

The conclusions of the above considerations, can be formulated as the following corollary.

Proposition 1. *Given two consecutive iterations of a hashing algorithm from the MD family (an MD-type hash algorithm) based on the function $f(x_1, \ldots, x_n)$. Then the evaluation of the function $f(y_1, \ldots, y_n)$ in the second iteration may use some terms of $f(x_1, \ldots, x_n)$ evaluated in the previous iteration. Ideally, the evaluation $f(y)$ may take as little as three operations if*

1. *the partial functions $g_1(x_1, \ldots x_{n-1}) = h_1(x_1, \ldots, x_{n-1})$ and*

2. the partial functions $g_0(x_1, \ldots x_{n-1}) = h_0(x_1, \ldots, x_{n-1})$,

assuming that y_1 *is given.*

Consider some examples. Let our function f be one of the functions used in MD4. Namely, $f(x_1, x_2, x_3) = x_1x_2 + \bar{x}_1x_3 = x_1x_2 \oplus x_1x_3 \oplus x_3$. If we apply rotation $y_2 = x_1$, $y_3 = x_2$ then $f(y) = x_2 \oplus y_1(x_1 \oplus x_2)$. The evaluation of $f(y)$ cannot be supported by partial evaluations of $f(x)$ as $g_1 = x_1x_2$ is different from $h_1 = x_2$ and $g_0 = x_1 \oplus 1$ is different from $h_0 = x_1 \oplus x_2$. The situation will vary from iteration to iteration.

Let a function $f(x_1, x_2, x_3, x_4, x_5) = x_1x_2 \oplus x_2x_3 \oplus x_3x_4 \oplus x_4x_5 \oplus x_5x_1$. It can be represented as $f(x) = x_1x_2 \oplus x_2x_3 \oplus x_3x_4 \oplus x_5(x_1 \oplus x_4)$ The function $f(y)$ with $y_2 = x_1$, $y_3 = x_2$, $y_4 = x_3$, $y_5 = x_4$ becomes $f(y) = x_1x_2 \oplus x_2x_3 \oplus x_3x_4 \oplus y_1(x_1 \oplus x_4)$ The evaluations of both $x_1x_2 \oplus x_2x_3 \oplus x_3x_4$ and $x_1 \oplus x_4$ done for the function $f(x)$, can be reused for the evaluation of $f(y)$.

To avoid a confusion, we have to stress that it is necessary to run two (or more) concurrent processes. One for evaluation of the function $f(y)$ and the others to prepare partial evaluations for the next iteration or in other words $f(y) = g_1(y_1, \ldots y_{n-1}) \oplus y_n g_0(y_1, \ldots y_{n-1})$.

It can be argued that if the "infinite" parallelism is allowed then the form of the function f does not matter. In practice, however, this is never the case. Most of the computers are still using a single processor architecture and for them an efficient evaluation of the function f is crucial. If a tradeoff between processing time and memory is allowed, all partial evaluations could be stored and reused.

Definition 1. *The class of* rotation-symmetric *functions includes all Boolean functions* $f : V_n \to V_1$ *such that* $f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$, *where* $y_{i+1} = x_i$ *for* $i = 1, 2, \ldots, n-1$ *and* $y_1 = x_n$ *or shortly* $f(x) = f(ROT(x))$.

The aim of this work is to investigate cryptographic properties of rotation-symmetric functions and discuss how to construct such functions.

4 Cryptographic Characteristics of Boolean Functions

Given the space V_n of binary vectors. Denote $\alpha_0 = (0, 0, \ldots, 0, 0)$, $\alpha_1 = (0, 0, \ldots, 0, 1)$, and so forth until $\alpha_{2^n-1} = (1, 1, \ldots, 1, 1)$. Vectors α may be treated as integers and then they can be ordered as $\alpha_0 < \alpha_1 < \cdots < \alpha_{2^n-1}$. Let $\alpha = (a_1, \ldots, a_n)$ and $x = (x_1, \ldots, x_n)$, we say that $x = \alpha$ if $x_i = a_i$ for all i. Boolean functions will be considered in their normal forms so

$$f(x) = \bigoplus_{\alpha \in V_n} c_\alpha x^\alpha = \bigoplus_{\alpha \in V_n} c_\alpha x_1^{a_1} \ldots x_n^{a_n} \tag{2}$$

where $\oplus$ stands for binary XOR operation (or addition modulo 2) and $c_\alpha \in \{0, 1\}$. The *truth table* of the function f is the binary sequence $(f(\alpha_0), \ldots, f(\alpha_{2^n-1}))$. A function f is *balanced* if its truth table consists of 2^{n-1} ones and zeros. The *Hamming weight* of a binary vector is defined as the number of ones it contains. In particular, the Hamming weight of a function f is the number of ones in its truth table and is denoted by $wt(f)$. The *Hamming distance* between two functions $f, g : V_n \to V_1$ is the Hamming weight of $f \oplus g$ or $d(f, g) = wt(f \oplus g)$.

Consider the function from Equation (2). If $c_\alpha = 0$ for all $wt(\alpha) > 1$, then f is called *affine* function. An affine function is *linear* if $c_{\alpha_0} = 0$ [8].

Definition 2. *Let $f(x)$ be a Boolean function on V_n. The nonlinearity of the function is defined by the minimum Hamming distance between the function and an affine function φ so $N_f = min\{wt(f \oplus \varphi) \mid \varphi \text{ is an affine function on } V_n\}$.*

Definition 3. *[5, 14] Let $f(x)$ be a function on V_n and α be a vector in V_n. We say that the function $f(x)$ satisfies the propagation criterion of degree k if $f(x) \oplus f(x \oplus \alpha)$ is balanced for all α such that* $0 < wt(\alpha) \leq k$. *If $k = 1$, we say that $f(x)$ satisfies the* Strict Avalanche Criterion *or SAC.*

Given a set $\mathcal{A} = \{a_1, \ldots, a_n\}$. The set $Sym(\mathcal{A})$ consists of all permutations which can be defined on the set $\mathcal{A}$. Note that $\pi \in Sym(\mathcal{A})$ operates on $\mathcal{A}$ and induces the permutation on indices $\{1, \ldots, n\}$ so $\pi(a_1, \ldots, a_n) = (a_{\pi(1)}, \ldots, a_{\pi(n)})$. Typically, a permutation $\pi(1, \ldots, n)$ can be written in the form of a sequence $(\pi(1), \ldots, \pi(n))$. So if $\mathcal{A} = \{1, 2, 3, 4\}$, then $\pi(1, 2, 3, 4) = (\pi(1), \pi(2), \pi(3), \pi(4)) = (2, 4, 1, 3)$ where $\pi(1) = 2$, $\pi(2) = 4$, $\pi(3) = 1$ and $\pi(4)) = 3$. The collection of permutations over the set $\{1, \ldots, n\}$ creates a symmetric group $\mathcal{S}_n$ where the group operation is the composition of permutations.

Definition 4. *A Boolean function $f(x) : V_n \to V_1$ is called symmetric with respect to the permutation π if $\pi(f(x)) = f(x_{\pi(1)}, \ldots, x_{\pi(n)}) = f(x_1, \ldots, x_n)$.*

5 Properties of Rotation-Symmetric Functions

The class of symmetric functions can be defined as a collection of all Boolean functions $f(x) : V_n \to V_1$ which are symmetric with respect to all permutations $\pi \in \mathcal{S}_n$ (see [18]). For every $\mathcal{S}_n$ and each degree $k = 1, \ldots, n$, there is a homogeneous symmetric function $e_k(x) : V_n \to V_1$ such that

$$e_k(x) = \bigoplus_{\substack{i_1, \ldots, i_k \in \mathcal{N};\\ i_1 \neq \ldots \neq i_k}} x_{i_1} \ldots x_{i_k} \tag{3}$$

where $\mathcal{N} = \{1, \ldots, n\}$. In other words, each term in a homogeneous function has the same degree. The functions $e_k(x) = e_k(\pi(x))$ for any $\pi \in \mathcal{S}_n$. Assume that $e_0 = 1$, then the function

$$\prod_{i=1}^{n} (1 \oplus x_i) = \bigoplus_{k=0}^{n} e_k(x)$$

For example, let $n = 4$, the functions be: $e_1(x) = x_1 \oplus x_2 \oplus x_3 \oplus x_4$, $e_2(x) = x_1x_2 \oplus x_1x_3 \oplus x_1x_4 \oplus x_2x_3 \oplus x_2x_4 \oplus x_3x_4$, $e_3(x) = x_1x_2x_3 \oplus x_1x_2x_4 \oplus x_1x_3x_4 \oplus x_2x_3x_4$, $e_4(x) = x_1x_2x_3x_4$. Clearly, $e_0 \oplus e_1 \oplus e_2 \oplus e_3 \oplus e_4 = (1 \oplus x_1)(1 \oplus x_2)(1 \oplus x_3)(1 \oplus x_4)$.

Let $m_k(x) = x_{i_1} \ldots x_{i_k}$ be a term where all indices $i_1, \ldots, i_k$ are different. Given a permutation $\pi \in \mathcal{S}_n$, then $\pi(m_k) = x_{\pi(i_1)} \ldots x_{\pi(i_k)}$, where $1 \leq k \leq n$. Observe that the permutation π generates a cyclic group $\mathcal{C}_r$ of order $r \leq n$ and $\mathcal{C}_r = \{\varepsilon, \pi, \pi^2, \ldots, \pi^{r-1}\}$ where ε is the identity permutation. The cyclic group acts on the term $m_k(x)$ and produces a homogeneous Boolean function of degree k in the following form:

$$f_k(x) = m_k \oplus \pi(m_k) \oplus \ldots \oplus \pi^{r-1}(m_k) \tag{4}$$

Note that rotation $\rho \in \mathcal{S}_n$ is defined as $\rho(i) = i+1$ for $i = 1, \ldots, n-1$ and $\rho(n) = 1$. Equation (4) can be used to generate a homogeneous rotation-symmetric Boolean function of degree k and

$$f_k(x) = m_k \oplus \rho(m_k) \oplus \ldots \oplus \rho^{n-1}(m_k) \tag{5}$$

Lemma 1. *Given a rotation-symmetric Boolean function in the form of Expression (5). Then its nonlinearity is $N_{f_k} \geq 2^{n-k}$ for $k = 2, \ldots, n$.*

Proof. Clearly, the weight of the term m_k is $wt(m_k) = 2^{n-k}$, the nonlinearity $N_{m_k} = \min(2^{n-k}, 2^n - 2^{n-k})$. Without the loss of the generality, the function (5) can be rewritten as $f_k(x) = x_1 ... x_k \oplus x_2 ... x_{k+1} \oplus \ldots \oplus x_n x_1 ... x_{k-1}$. Take an arbitrary affine function

$$\varphi(x) = \bigoplus_{i=1}^{n} a_i x_i \oplus c$$

where $x = (x_1, \ldots, x_n)$ and $c \in V_1$. Then

$$f_k \oplus \varphi = \bigoplus_{i=1}^{n} x_i(a_i \oplus x_{i+1} ... x_{i+k-1}) \oplus c$$

As $N_{f_k} = \min_\varphi d(f_k, \varphi) = \min_\varphi wt(f_k \oplus \varphi)$, so according to the result given in [7] we have

$$N_{f_k} \geq wt(x_i(a_i \oplus x_{i+1} \ldots x_{i+k-1})) = 2^{n-k}.$$

Consider an example. Let $n = 6$ and $m_3 = x_2 x_3 x_5$. Then the corresponding rotation-symmetric function (of degree 3) is generated as follows

$$f_3(x) = (x_2x_3x_5) \oplus \rho(x_2x_3x_5) \oplus \rho^2(x_2x_3x_5) \oplus \rho^3(x_2x_3x_5) \oplus \rho^4(x_2x_3x_5) \oplus$$
$$\rho^5(x_2x_3x_5) = x_2x_3x_5 \oplus x_3x_4x_6 \oplus x_4x_5x_1 \oplus x_5x_6x_2 \oplus x_6x_1x_3 \oplus x_1x_2x_4.$$

Equation (5) produces simple rotation-symmetric functions for two following cases. When $k = 1$, the corresponding homogeneous rotation-symmetric function of degree 1 is

$$f_1(x) = e_1 = \bigoplus_{i=0}^{n-1} \rho^i[m_1(x)] = x_1 \oplus x_2 \oplus \ \ldots \ \oplus x_n$$

which is a linear function and is symmetric with respect to all permutations from $\mathcal{S}_n$. If $k = n$, the function (5) becomes $f_n(x) = e_n(x) = x_1 x_2 ... x_n$ which is symmetric with respect to all permutations from $\mathcal{S}_n$ and has the lowest Hamming weight which equals 1.

Consider homogeneous rotation-symmetric Boolean functions of the degree 2. Assume that an initial term is $m_2(x) = x_j x_{j+\ell}$ for some ℓ $(\ell + j \leq n)$ and the rotation is $\rho \in \mathcal{S}_n$. Then the corresponding homogeneous rotation-symmetric Boolean functions is

$$f_2(x) = \bigoplus_{i=0}^{n-1} \rho^i(x_j x_{j+\ell}) = x_1 x_{\ell+1} \oplus \ \dots \ \oplus x_i x_{\ell+i} \oplus \ \dots \ \oplus x_n x_{\ell+n}, \qquad (6)$$

where each subscript w is taken as $((w-1) \bmod n) + 1$.

Theorem 1. *Let $f_2(x) : V_n \to V_1$ be a homogeneous rotation-symmetric Boolean function of degree 2 which is generated from a term of degree 2 using the rotation $\rho \in \mathcal{S}_n$. The function has the following properties:*

(i). the Hamming weight of $f_2(x)$ is $2^{n-2} \leq wt(f_2) \leq 2^n + 2^{n-2}$,
(ii). the nonlinearity of the function is $N_f \geq 2^{n-2}$,
(iii). if n is odd $(n > 2)$, the function $f_2(x)$ is balanced,
(iv). the functions satisfy the propagation criterion with respect to all vectors $\alpha \in V_n$ such that $0 < wt(\alpha) < n$ and satisfies the SAC criterion.

Proof. (i). Since $f_2(x) \oplus f_2(x \oplus \alpha)$ is a constant or an affine function, we can observe that the auto-correlation of $f_2(x)$ defined as $\Delta(\alpha) = \bigoplus_{\alpha \in V_n} (-1)^{f_2(x) \oplus f_2(x \oplus \alpha)}$ is equal to ([20])

$$\Delta(\alpha) = \begin{cases} 2^n \text{ if } \alpha = \alpha_0 \text{ or } \alpha = \alpha_{2^n-1} \\ 0 \text{ otherwise} \end{cases}$$

For any vector $\alpha \in V_n$, $wt(f_2(x)) = wt(f_2(x \oplus \alpha))$. The auto-correlation of two sequences of the same weight cannot be 0 or 2^n if either the weight $wt(f_2) < 2^{n-2}$ or $wt(f_2) > 2^{n-1} + 2^{n-2}$, hence $2^{n-2} \leq wt(f_2) \leq 2^{n-1} + 2^{n-2}$.
(ii). Let $\varphi(x)$ be an affine function on V_n. The Hamming distance between $f_2(x)$ and $\varphi(x)$ is $wt(f_2 \oplus \varphi)$ and

$$f_2 \oplus \varphi = \bigoplus_{i=1}^{n} x_i(a_i \oplus x_{i+\ell}) \oplus c.$$

The term $x_i(a_i \oplus x_{i+\ell})$ constitutes a Boolean function whose Hamming weight is 2^{n-2}. Since $wt(f_2) \geq wt(m_2)$, then $wt(f_2 \oplus \varphi) \geq wt(x_i(a_i \oplus x_{i+l}))$. Therefore, we can conclude that $N_f \geq 2^{n-2}$.

(iii). By contradiction. Assume that $wt(f_2(x)) \neq 2^{n-1}$. Let $y_i = 1 \oplus x_i$ for all $i = 1, \dots, n$ ($n > 2$ and odd). Note that the functions $f_2(y)$ and $f_2(x)$ have to be of the same weight as the relation between x and y is one to one, or $wt(f_2(x)) = wt(f_2(y))$. So that without the loss of generality, we can take $\ell = 1$ in (6) and take a closer look at the function $f_2(y)$ which is

$$\begin{aligned} f_2(y) &= y_1 y_2 \oplus \dots \oplus y_{n-1} y_n \oplus y_n y_1 \\ &= (1 \oplus x_1)(1 \oplus x_2) \oplus \dots \oplus (1 \oplus x_{n-1})(1 \oplus x_n) \oplus (1 \oplus x_n)(1 \oplus x_1) \\ &= 1 \oplus f_2(x) \end{aligned}$$

As $f_2(y) = 1 \oplus f_2(x)$, it means that $wt(f_2(y)) = 2^n - wt(f_2(x))$. From the assumption $(wt(f_2(x)) \neq 2^{n-1})$, we conclude that $wt(f_2(y)) \neq wt(f_2(x))$ which is the requested contradiction which proves the claim.

(iv). Let $\alpha = (a_1, a_2, \ ..., \ a_n)$, Then

$$f_2(x) \oplus f_2(x \oplus \alpha) = (a_n \oplus a_2)x_1 \oplus \ldots \oplus (a_{n-2} \oplus a_n)x_{n-1} \oplus (a_{n-1} \oplus a_1)x_n \oplus C$$

where the constant $C = a_1a_2 \oplus a_2a_3 \oplus \ldots \oplus a_na_1$. When $\alpha = \alpha_0 = \mathbf{0}$ and $\alpha = \alpha_{2^n-1} = \mathbf{1}$, $f_2(x) \oplus f_2(x \oplus \alpha)$ is constant. For $\alpha \neq \{\mathbf{0}, \mathbf{1}\}$, $f_2(x) \oplus f_2(x \oplus \alpha)$ is a balanced affine function. This means that $f_2(x)$ satisfies propagation criterion of the order k where $k = 1, \ldots, n-1$. Clearly, the function satisfies the SAC criterion.

The nonlinearity of the function $f_2(x)$ was considered in [12] and proved that it attains a high nonlinearity. More precisely, the following lemma is true.

Lemma 2. *[12] Given $f_2(x) : V_n \to V_1$ for n odd, then the nonlinearity of the function is $N_{f_2} = 2^{n-1} - 2^{(n-1)/2}$.*

Consider two classes of functions

$$\begin{aligned} f_2^{(n)} &= x_1x_2 \oplus x_2x_3 \oplus \ldots \oplus x_{n-1}x_n \oplus x_nx_1 \\ g_2^{(n)} &= x_1x_2 \oplus x_2x_3 \oplus \ldots \oplus x_{n-1}x_n \end{aligned}$$

for $n = 0, 1, \ldots$. If we assume that $wt(g_2^{(0)}) = wt(f_2^{(0)}) = 0$, then the following equations are satisfied

$$\begin{aligned} wt(g_2^{(n)}) &= 2^{n-2} + 2wt(g_2^{(n-2)}); \\ wt(f_2^{(n}) &= wt(g_2^{(n-1)}) + wt(x_1 \oplus g_2^{(n-2)})wt(1 \oplus x_1 \oplus x_{n-2} \oplus g_2^{(n-2)}), \end{aligned}$$

where $(x_1 \oplus g_2^{(n-2)})$ and $(1 \oplus x_1 \oplus x_{n-2} \oplus g_2^{(n-2)})$ are two functions on V_{n-2}.

Given two rotation-symmetric functions $f(x)$, $g(x)$ on V_n. The next corollary is useful to create a combined function which preserves the rotation symmetry.

Corollary 2. *Given two functions $f(x)$, $g(x)$ on V_n and the rotation $\rho \in \mathcal{S}_n$. If $\rho(f(x)) = f(x)$ and $\rho(g(x)) = g(x)$, then $\rho(f(x) \oplus g(x)) = f(x) \oplus g(x)$.*

6 Balanced Rotation-Symmetric Boolean Functions

The function $f_2(x)$ of degree 2 is an ideal candidate for hashing round function. It is balanced, highly nonlinear and satisfies the propagation criterion (including the SAC). To get other cryptographically strong rotation-symmetric functions, we may to apply Corollary (2) which states that sum of rotation-symmetric functions is a rotation-symmetric function as well. A general construction for rotation symmetric functions can be done using the following algorithm.

1. select requested collection of terms of degrees $k_1, \ldots, k_j$,
2. generate homogeneous rotation-symmetric functions of degrees $k_1, \ldots, k_j$,
3. compose the functions into the compound rotation-symmetric function $f(x) = f_{k_1}(x) \oplus \ldots \oplus f_{k_j}(x)$.

Clearly, the evaluation of the function $f(x)$ will be faster when the number of terms used to generate homogeneous functions is restricted. In practice, there are two most interesting cases when the number is limited to two and three. We are going to investigate the two cases.

Class 1 generated by two terms. Consider the case when $m_1(x)$ and $m_2(x) = x_1 x_\ell$ where $m_1 : V_{n+s} \to V_1$ and $m_2 : V_n \to V_1$. The the class of rotation-symmetric function is expressible as

$$f(x) = f_2 \oplus f_1 = \bigoplus_{i=0}^{n-1} \rho^i(m_2(x)) \oplus \bigoplus_{i=0}^{n+s-1} \rho^i(m_1(x)) \tag{7}$$

for $\rho \in \mathcal{S}_n$. Note that terms $m_1(x)$ do not need to be evaluated so the function $f(x)$ is especially attractive for a fast evaluation. The explicit form of the function is

$$f(x) = x_1(1 \oplus x_\ell) \oplus x_2(1 \oplus x_{\ell+1}) \oplus \ldots \oplus x_n(1 \oplus x_{\ell+n-1}) \oplus x_{n+1} \oplus \ldots \oplus x_{n+s}$$

The function $f(x)$ is balanced, its nonlinearity is $N_f \geq 2^{n+s-2}$, and the function satisfies the propagation criterion with respect to α such that $\alpha = (\beta_1, \beta_2)$ and $\beta_1 \neq \{\mathbf{0}, \mathbf{1}\}$, where $\beta_1 \in V_n$ and $\beta_2 \in V_s$.

Consider an example. Let $k = 3$ $n = 4$ and $s = 1$, then the function $f(x)$ can be written as

$$\begin{aligned} f(x) &= (x_1x_2x_3 \oplus x_2x_3x_4 \oplus x_3x_4x_1 \oplus x_4x_1x_2) \oplus (x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5) \\ &= x_1(1 \oplus x_2x_3) \oplus x_2(1 \oplus x_3x_4) \oplus x_3(1 \oplus x_4x_1) \oplus x_4(1 \oplus x_1x_2) \oplus x_5. \end{aligned}$$

The function is balanced and nonlinearity is at least 8.

Class 2 generated by three terms. Consider the case when $m_1(x)$ is a term of the degree 1 over V_{n+s}, $m_2(x)$ is a term of the degree 2 over V_{n+m} and $m_k(x)$ is a term of the degree k over V_n, where $n > k > 2$ and $(s \geq m)$. The function

$$\begin{aligned} f(x) &= f_k(x) \oplus f_2(x) \oplus f_1(x) \\ &= \bigoplus_{i=0}^{n-1} \rho^i(m_k(x)) \oplus \bigoplus_{i=0}^{n+m-1} \rho^i(m_2(x)) \oplus \bigoplus_{i=0}^{n+s-1} \rho^i(m_1(x)) \end{aligned}$$

where $\rho \in \mathcal{S}_n$, $\rho_1 \in \mathcal{S}_{n+s}$ and $\rho_2 \in \mathcal{S}_{n+m}$. The function $f(x)$ is balanced, has the nonlinearity $N_f \geq 2^{n+s-k}$ and satisfies the propagation criterion with respect to $n < wt(\alpha) < n + s$. Observe that from an efficient evaluation point of view, the homogeneous rotation-symmetric function $f_k(x)$ generated by $m_k(x)$ is the most expensive so that is why it should be kept relatively short $\rho \in \mathcal{S}_n$ (see [4]). For instance $n = 4$, $s = m = 1$ and $k = 3$, the balanced rotation-symmetric function is

$$\begin{aligned} f(x) = {} & x_1x_2x_3 \oplus x_2x_3x_4 \oplus x_3x_4x_1 \oplus x_4x_1x_2 \oplus x_1x_2 \oplus x_2x_3 \oplus \\ & \oplus x_3x_4 \oplus x_4x_5 \oplus x_5x_1 \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \end{aligned}$$

7 Evaluation of Functions

Consider functions from Class 1, i.e. rotation-symmetric functions of degree two. We are going to analyse bounds for the number of necessary operations needed to evaluate a round function when it is used for m consecutive rounds. Let our rotation-symmetric function over V_n be

$$f(x) = x_1x_2 \oplus x_2x_3 \oplus \ldots \oplus x_{n-1}x_n \oplus x_nx_1,$$

where n is odd. In the first round, the whole function needs to be evaluated from scratch. This will consume no more than $2n$ operations. This number can be reduced to $\frac{3n-1}{2}$ if the evaluation is done in pairs $f(x) = x_1x_2 \oplus x_3(x_2 \oplus x_4) \oplus \ldots \oplus x_n(x_{n-1} \oplus x_1)$. For the next round, if we keep the evaluation of $h(x_1, \ldots, x_{n-1}) = x_1x_2 \oplus x_2x_3 \oplus \ldots \oplus x_{n-2}x_{n-1}$ then we need to evaluate the new term $x_0(x_1 \oplus x_{n-1})$ which takes 2 operations. Evaluation of $f(x_0, x_1, \ldots, x_{n-1})$ takes at most three operations, where x_0 is a "new variable" which was not used in the previous round. To be able to use the same technique in next rounds, we need to evaluate the function $h(x_0, \ldots, x_{n-2}) = x_0x_1 \oplus x_1x_2 \oplus \ldots \oplus x_{n-3}x_{n-2}$ from $f(x_0, \ldots, x_{n-1})$. The "correction" of $h(x)$ will cost at most three operations as $h(x) = f(x) \oplus x_{n-1}(x_{n-2} \oplus x_0)$ and the term $x_{n-1}(x_{n-2} \oplus x_0)$ needs to be generated. In conclusion, the evaluation of $f(x)$ for m consecutive rounds will take no more than $\frac{3n-1}{2} + 6(m-1)$ operations.

What we can gain if we use shorter function which is not rotation symmetric but is obtained from a one by removing some of the terms. Let this function be

$$f(x_1, \ldots, x_n) = x_1x_2 \oplus x_3x_4 \oplus \ldots \oplus x_{n-2}x_{n-1} \oplus x_{n-1}x_n$$

In the first round the function needs $(n-1)$ operations for its evaluation. In the second round, the same number of operations is necessary as all terms need to be generated. This costs $(n-1)$ operations. In the third round, we can use partial evaluation from the first round. This consumes at most 3 operations. The evaluation of the expression for the 5-th round takes at most 3 operations. All together, the evaluation takes at most $2(n-1) + 6(m-2)$ operations.

Paradoxically, shorter functions require more steps for their evaluation. This phenomenon relates to the fact that rotation will generate all terms of the rotation-symmetric function gradually round by round with no chances for optimisation. Starting from a rotation-symmetric function allows optimal evaluation of terms which can be reused further in the consecutive rounds. The designers of the HAVAL hashing algorithm [24] fell into the trap. The first round function they used is $f_1(x_6, x_5, x_4, x_3, x_2, x_1, x_0) = x_1x_4 \oplus x_2x_5 \oplus x_3x_6 \oplus x_0x_1 \oplus x_0$ which is a shortened version of a rotation-symmetric function $f_2(x_1, \ldots, x_7)$.

8 Extensions and Further Research

The paper suggest a novel framework for designing cryptographically strong Boolean functions which can be efficiently evaluated when they are applied as round functions in a MD hashing with rotation as the round mixing operation. Clearly any symmetric Boolean function (in respect to any permutation) is also rotation symmetric. The reverse is not true as a rotation-symmetric function is not symmetric in general. Rotation-symmetric functions are much shorter than their symmetric equivalents. This is especially visible for bigger n. For instance, a rotation-symmetric function $f_2(x)$ over V_n includes n terms of degree 2 while its symmetric equivalent consists of $\frac{n(n-1)}{2}$ terms. Symmetric functions could be useful if the round mixing operation is an arbitrary permutation controlled by either cryptographic key (as for keyed hashing) or messages.

The round mixing operation can be viewed as a linear transformation of the input variables. Rotation is an especially simple case. Note that linear transformation of input variables does not increase the degree of the function. Similarly, it is possible to extend our considerations to the case of linear transformations.

The concept of efficient evaluation can be extended for permutations $p : V_n \to V_n$. This is not directly applicable in MD hashing but certainly is of interest for other cryptographic algorithms where the S-boxes are evaluated on the fly instead of using their lookup tables. The idea is to design a cryptographically strong permutation whose component output functions share as many common terms as possible so partial evaluations can be shared among the functions. The confirmation that such permutations exist can be found in the papers [12, 10].

Finally, it can be argued that an efficient evaluation may actually contradict the security of hashing. This argument may or may not be valid depending on other components used in the single round (shifting, addition modulo 2^n, etc.). Also the number of different functions together with the total number of rounds plays a significant role in getting a secure (collision free) hash algorithm.

Acknowledgement

Authors wish to thank Dr Xian-Mo Zhang and anonymous referees for their critical comments.

References

[1] Antoon Bosselaers, René Govaerts, and Joos Vandewalle. Fast hasing on the Pentium. In L. Koblitz, editor, *Advances in Cryptology - CRYPTO'96*, pages 298–312. Springer, 1996. Lecture Notes in Computer Science No. 1109.

[2] H. Dobbertin. Cryptanalysis of MD4. In *Fast Software Encryption, Lecture Notes in Comp uter Science, Vol. 1039, D.Gollmann (Ed.)*, pages 71–82. Springer-Verlag, 1996.

[3] H. Dobbertin. Cryptanalysis of MD5 compress. Announcement on Internet, May 1996.

[3] H. Feistel. Cryptography and computer privacy. *Scientific American*, 228:15–23, May 1973.

[4] C. Fontaine. The nonlinearity of a class of boolean functions with short representation. In J. Pribyl, editor, *Proceedings of PRAGOCRYPT96*, pages 129–144. CTU Publishing House, 1996.

[5] R. Forré. The strict avalanche criterion: Spectral properties of boolean functions and an extended definition. In S. Goldwasser, editor, *Advances in Cryptology - CRYPTO'88*, pages 450–468. Springer-Verlag, 1988. Lecture Notes in Computer Science No. 403.

[6] M. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, 1979.

[7] F.J. MacWilliams and N.J.A. Sloane. *The theory of error-correcting codes.* North-Holland, Amsterdam, 1977.

[8] W. Meier and O. Staffelbach. Nonlinearity criteria for cryptographic functions. In J.-J. Quisquater and J. Vandewalle, editors, *Advances in Cryptology - EUROCRYPT'89*, pages 549–562. Springer-Verlag, 1990. Lecture Notes in Computer Science No. 434.

[9] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography.* CRC Press, Boca Raton, 1997.

[10] K. Nyberg. On the construction of highly nonlinear permutations. In R.A. Rueppel, editor, *Advances in Cryptology — Eurocrypt '92*, pages 92–98, Berlin, 1993. Springer-Verlag.

[11] K. Nyberg. Generalised Feistel networks. In K. Kim and T. Matsumoto, editors, *Advances in Cryptology – ASIACRYPT'96*, volume 1163 of *Lecture Notes in Computer Science*, pages 91–104, Berlin, 1996. Springer.

[12] J. Pieprzyk. Bent permutations. In G. Mullen and P. Shiue, editors, *Lecture Notes in Pure and Applied Mathematics, Vol 141, Proceedings of 1st International Conference on Finite Fields, Coding Theory, and Advances in Communications and Computing, Las Vegas, 1991*, 1992.

[13] B. Preneel. *Analysis and design of cryptographic hash functions.* PhD thesis, Katholieke Universiteit Leuven, 1993.

[14] B. Preneel, W. Van Leekwijck, L. Van Linden, R. Govaerts, and J. Vandewalle. Propagation characteristics of Boolean functions. In I.B. Damgård, editor, *Advances in Cryptology — Eurocrypt '90*, pages 161–173, Berlin, 1991. Springer-Verlag.

[15] Ronald L. Rivest. The MD4 message digest algorithm. Technical Report MIT/LCS/TM-434, MIT Laboratory for Computer Science, October 1990.

[16] Ronald L. Rivest. The MD5 message-digest algorithm. Internet Request for Comments, April 1992. RFC 1321.

[17] M.J.B. Robshaw. MD2, MD4, MD5, SHA and other hash functions. Technical Report TR 101, RSA Laboratories, July 1994.

[18] B.E. Sagan. *The Symmetric Group: Representations, Combinatorial Algorithms, and Symmetric Functions.* Wadsworth & Brooks, 1991.

[19] Bruce Schneier. *Applied Cryptography.* John Wiley & Sons, 1996.

[20] Jennifer Seberry, Xian-Mo Zhang, and Yuliang Zheng. Nonlinearly balanced boolean functions and their propagation characteristics. In Douglas R. Stinson, editor, *Advances in Cryptology - CRYPTO'93*, pages 49–60. Springer, 1994. Lecture Notes in Computer Science No. 773.

[22] C. E. Shannon. Communication theory of secrecy systems. *Bell Sys. Tech. J.*, 28:657–715, 1949.

[23] D.R. Stinson. *Cryptography: Theory and Practice.* CRC Press, 1995.

[24] Y. Zheng, J. Pieprzyk, and J. Seberry. HAVAL - a one-way hashing algorithm with variable length of output. In J. Seberry and Y. Zheng, editors, *Advances in Cryptology — AUSCRYPT '92*, pages 83–104, Berlin, 1993. Springer-Verlag.