Towards Optimal Double-Length Hash Functions

Mridul Nandi

Applied Statistics Unit,
Indian Statistical Institute,
Kolkata, India
mridul_r@isical.ac.in

Abstract. In this paper we design several double length hash functions and study their security properties in the random oracle model. We design a class of double length hash functions (and compression functions) which includes some recent constructions [4, 6, 10]. We also propose a secure double length hash function which is as efficient as the insecure concatenated classical hash functions [7].

1 Introduction

An n-bit compression function or hash function is said to be "ideal" or "maximally secure" if the best collision attack requires $\Omega(2^{n/2})$ many queries which is same as the complexity of the *birthday attack*. To increase the security level, one can design $2n$-bit compression functions and hash functions (also known as *double length compressions or hash functions* respectively). Potentially one can expect a security level of $\Omega(2^n)$ for a $2n$-bit hash function. But the trivial concatenated hash functions $H \parallel G$ is not secure where one of the H and G is a classical hash function [7]. In this paper we design several double length compression functions and double length hash functions based on single length compression functions. More precisely we consider the following problem;

Problem : *Given a secure compression function,* $f : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$ *(or* s *compression functions* $f_1, \cdots, f_s : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$*),* $m > 0$*, design a secure compression function* $F : \{0,1\}^{2n} \times \{0,1\}^{N-2n} \to \{0,1\}^{2n}$ *and a hash function* $H : \{0,1\}^{\leq L} \to \{0,1\}^{2n}$*, where* $N > 2n$ and L is sufficiently large.

Designing a double length hash function from a single length compression or hash function is also important in the hardware point of view. As crypto-hardware are expensive, the construction which allows an existing hardware would be meaningful while we are looking for more security. There were several attempts to construct a secure block cipher based double length compression functions. Again, most of these have several collision and preimage attacks much better than the birthday attack [5, 6, 8, 10, 17]. Recently, Lucks [4, 10] designed a secure double length compression function. A similar designed is proposed by Hirose [6] by using a secure block ciphers of the form $E : \{0,1\}^n \times \{0,1\}^{2n} \to \{0,1\}^n$. But the efficiency of their design is fairly low. A more efficient double

S. Maitra et al. (Eds.): INDOCRYPT 2005, LNCS 3797, pp. 77–89, 2005.

length compression function based on three independent random compression functions has been proposed by Nandi *et. al.* [13]. But the security of the compression function is not maximum. So it would be interesting to design both maximally secure and efficient double length hash function.

1.1 Our Contribution

In this paper we design several new double length hash functions and compute their security level and the rate (measurement of efficiency).

Our first design is a generalization of Lucks's [10] and Hirose's [6] constructions. Given a permutations $p(\cdot)$ on the set of all N-bit strings and a compression function $f : \{0,1\}^N \to \{0,1\}^n$, define $f^p(X) = f(X) \parallel f(p(X))$. We show that the double length function f^p is maximally secure provided the permutation p does not have any fixed point.

Next, we study the security level for the double length hash function defined by the classical iteration of a compression function f^p defined as above. We show that, along with secure compression functions there are many more compression functions which extends to a secure double length hash functions. Thus, we have a wide class of maximally secure double length hash functions. Lucks and Hirose's construction belong to this class.

Then we design a construction similar to the concatenated hash function. We show the collision security (or preimage security) level of the double length hash function $\Omega(2^n/in^{i-1})$ (or $\Omega(2^{2n}/in^{i-1})$ respectively), where i is the number of iterations of underlying compression function to invoke a double length compression function. It determines the efficiency of the hash function.

2 Preliminaries

In this section we briefly recall some preliminaries of hash functions. We first give a brief introduction of classical hash functions. Then we explain random oracle model and the behavior of adversary in the random oracle model. Next, we explain Joux's attack and its application to the collision attack on concatenated hash function. Finally we state the recent constructions of double length hash functions and we compare their efficiencies.

2.1 The Classical Iterated Hash Function

We briefly explain a simplified version of Merkle-Damgård method [3, 11]. Let $f : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$ be an underlying compression function. Given a message $M \in \{0,1\}^{\leq L} = \cup_{i=0}^{L} \{0,1\}^i$, where $L = 2^m - 1$ and an *initial value* $h_0 \in \{0,1\}^n$ we apply the following padding rule;

- *padding rule*: Choose smallest $i \geq 0$ such that $|M| + i + 1$ is multiple of m. Let $\langle |M| \rangle$ be the m-bit binary representation of $|M|$. We write $M \parallel 10^i \parallel \langle |M| \rangle = m_1 \parallel m_2 \cdots \parallel m_l$ for some positive integer l and $|m_i| = m$, $1 \leq i \leq l$.

Each m_i is known as a *message block* of the underlying compression function. We define the *classical iterated hash function* $H^f(M)$, or simply $H(M)$, by the following method;

$$H(M) = h_l, \text{ where } h_i = f(h_{i-1} \parallel m_i),\ 1 \le i \le l.$$

the h_i's are known as *intermediate hash values*, $i \neq 0, l$. For simplicity, we ignore the padding rule.

We use the notation $h \rightarrow_x h'$ (a labeled arc) to mean $f(h, x) = h'$, where $|h| = |h'| = n$ and $|x| = m$. Thus, the computation of $H(M)$ can be represented by a labeled path from h_0 to h_l as follows;

$$h_0 \rightarrow_{m_1} h_1 \rightarrow_{m_2} h_2 \cdots h_{l-1} \rightarrow_{m_l} h_l \quad \text{or} \quad h_0 \Rightarrow_M h_l.$$

Thus, $h_0 \Rightarrow_M h_l$ if and only if $H(M) = h_l$.

2.2 The Random Oracle Model

Let $\mathcal{F}^{D \rightarrow R}$ be a set of all functions from D to R. A randomly chosen function f from $\mathcal{F}^{D \rightarrow R}$ is known as a *random function* [2]. One can define a random function in the following equivalent way and equivalence can be checked in a straightforward way;

Definition 1. (Random Function)
A random function*, f from D to R, takes values as random variables, such that for any $x \in D$, $f(x)$ has uniform distribution on R and for any $k > 0$ and k distinct elements $x_1, \cdots x_k \in D$, the random variables $f(x_1), \cdots, f(x_k)$ are independently distributed.*

Proposition 1. *Let $f : D \rightarrow R$ be a random function and $\{X_1, Y_1\} \neq \{X_2, Y_2\}$ be two subsets of D. Then* $\mathsf{Pr}[\, f(X_1) = f(Y_1)] = 1/|R|$ *and* $\mathsf{Pr}[\, f(X_1) = f(Y_1)$ *and* $f(X_2) = f(Y_2)] = 1/|R|^2$.

Proof. Since $X_1 \neq Y_1$, $f(X_1)$ is uniformly distributed on R given $f(Y_1)$ and hence $\mathsf{Pr}[\, f(X_1) = f(Y_1)] = 1/|R|$.

For the second part of the proposition, without loss of generality, let us assume that $X_1 \notin \{X_2, Y_2\}$. Thus $f(X_1)$ is uniformly distributed on R given the random variables $f(Y_1), f(X_2)$ and $f(Y_2)$. Thus, the conditional probability Pr $[f(X_1) = f(Y_1) \mid f(Y_1), f(X_2), f(Y_2)]$ is $1/|R|$ and hence so is unconditional. So we have,

$$\begin{aligned}
&\mathsf{Pr}\,[f(X_1) = f(Y_1) \text{ and } f(X_2) = f(Y_2)\,] \\
&= \mathsf{Pr}\,[f(X_1) = f(Y_1) \;\mid\; f(X_2) = f(Y_2)\,] \times \mathsf{Pr}\,[\, f(X_2) = f(Y_2)\,] \\
&= \mathsf{Pr}\,[\, f(X_1) = f(Y_1)\,] \times \mathsf{Pr}\,[\, f(X_2) = f(Y_2)\,] \\
&= 1/|R|^2.
\end{aligned}$$

□

We call $f_1, \cdots, f_s$ *independent random functions* if f_i's are chosen independently and randomly from the set $\mathcal{F}^{D \rightarrow R}$. We state an equivalent definition of independent random functions.

Definition 2. (Independent Random Functions)
We say a family of random functions $f_1, \cdots, f_s : D \to R$ *are* independent random functions *if for any s subsets* $\{x_1^1, \cdots, x_{k_1}^1\}$, $\cdots$, $\{x_1^s, \cdots, x_{k_s}^s\}$, *the random vectors* $(f_1(x_1^1), \cdots, f_1(x_{k_1}^1)), \cdots, (f_s(x_1^s), \cdots, f_s(x_{k_s}^s))$ *are independently distributed.*

The Adversary in the Random Oracle Model : When a hash function $H(\cdot)$ is designed based on a random compression function f, an attack algorithm is an oracle algorithm, $\mathcal{A}^f$, with the oracle f. Thus, adversary can ask several queries of f adaptively and based on the query-response pairs adversary finally outputs a message or a pair of messages depending on the nature of the attack. It can choose $x_1, \cdots x_q$ adaptively and get responses $y_1, \cdots, y_q$, where $y_i = f(x_i)$. We can think that y_i as a realization of the random variable $f(x_i)$ which is observed by the adversary. Define the complete list of query-response pairs $\mathcal{Q} = ((x_1, y_1), \cdots, (x_q, y_q))$ by *view* of the adversary. Any output produced by the adversary should only depend on the view. Moreover, if the adversary is finding collisions for a hash function, $H(\cdot)$, based on the compression function, $f(\cdot)$, and it outputs a pair of distinct messages $M \neq N$ then the values of $H(M)$ and $H(N)$ should be computed from the view. When we have two or more underlying compression functions $f_1, f_2, \cdots$, we have a set of lists of pairs $\{\mathcal{Q}_1, \mathcal{Q}_2, \cdots\}$ called view of the adversary, where $\mathcal{Q}_i$ is the view due to the random compression function f_i, $i \geq 1$.We define the complexity of an attack algorithm by size of the view to be required to have non negligible probability of success or advantage [1]. The minimum complexity of an attack algorithm is a measurement of security of the hash function.

2.3 The Birth-Day Attack

A set $\{M_1, \cdots, M_r\}$ is said to be an *r-way collision set* of $g : D \to R$, if $g(M_1) = \cdots = g(M_r)$. The above event is known as a *multicollision.*

BirthdayAttack(g, q, r) **:**

1. Choose $x_1, \cdots, x_q$ randomly from the domain D and compute $y_i = g(x_i)$ for $1 \leq i \leq q$.
2. Return a subset (if any) $C \subseteq \{x_1, \cdots, x_q\}$ of size r such that C is an r-way multicollision subset for the function g. Otherwise return the output "failure".

The next proposition gives an estimate of the complexity of the birthday attack in finding an r-way collision with significant probability. See [15, 16] for a detail discussion.

Proposition 2. (Complexity of the Birthday Attack) [14]
For a random function $g : D \to \{0,1\}^n$, *the birthday attack with complexity* q *finds an* r*-way collision with probability* $O(q^r/2^{(r-1)n})$. *Thus, the birthday attack requires* $\Omega(2^{n(r-1)/r})$ *queries to find an* r*-way collision with significant probability. For* $r = 2$, *the birthday attack requires* $O(2^{n/2})$ *queries.*

2.4 Joux's Multicollision Attack

In a recent paper by Joux [7], it was shown that there is a 2^r-way collision attack for the classical iterated hash function based on a compression function, $f : \{0,1\}^{m+n} \to \{0,1\}^n$, where the attack has complexity $O(r\, 2^{n/2})$. This complexity is much less than $\Omega(2^{\frac{n(2^r-1)}{2^r}})$, which is the complexity for the birthday attack (see Proposition 2).

Here is the basic idea of the attack. Consider a set of vertices $V = \{0,1\}^n$. We use the notation $h \to_M h'$ (a labeled arc) to mean $f(h, M) = h'$. Here, $|h| = |h'| = n$ and $|M| = m$. The strategy is to first find r successive collisions (see Figure 1) by performing r successive birthday attacks, as follows:

$$\begin{aligned} &f(h_0, m_1) = f(h_0, n_1) = h_1 \text{ (say), where } m_1 \neq n_1 \\ &f(h_1, m_2) = f(h_1, n_2) = h_2 \text{ (say), where } m_2 \neq n_2 \\ &\qquad\qquad \vdots \\ &f(h_{r-1}, m_r) = f(h_{r-1}, n_r) = h_r \text{ (say), where } m_r \neq n_r. \end{aligned}$$

For $1 \leq i \leq r$, we apply BirthdayAttack$(f(h_{i-1}, \cdot), 2^{n/2}, 2)$ to find $m_i \neq n_i$ such that $f(h_{i-1}, m_i) = f(h_{i-1}, n_i)$. Thus the set

$$\{x_1 \parallel \cdots \parallel x_r : x_i = m_i \text{ or } n_i, 1 \leq i \leq r\}$$

is a 2^r-way collision set. The complexity of the attack is $O(r\, 2^{n/2})$. Figure 1 is a diagram illustrating the attack.

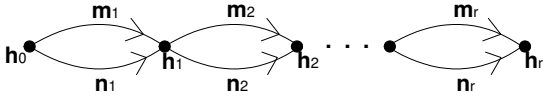

Fig. 1. Graphical representation of Joux's multicollision attack

Applications of Multicollision Attacks : A natural and efficient approach to produce large output hash values is the concatenation of several smaller output hash values. For example, given two classical iterated hash functions, H and G, one can define a hash function $H(M) \parallel G(M)$. This idea has been frequently used because it is efficient and simple to implement. However, due to the attacks of Joux [7], there exists a collision attack that is more efficient than the birthday attack. The complexity of the attack is roughly the maximum of the complexity of the multicollision birthday attack on H and the complexity of the standard birthday attack on G.

We briefly describe the attack (see [7] for more details). Let H and G have output hash values of n_H and n_G bits in length, respectively.

1. By using Joux's multicollision attack, find $2^{n_G/2}$ messages which have common output hash value (say h^*) on H.

2. Find two messages, say M and N where $M \neq N$, which are members of the set of $2^{n_G/2}$ messages found in step 1, such that they have same output hash value (say g^*) on G. Note that we expect to be able to find a collision on an n_G-bit function from a set of $2^{n_G/2}$ messages using the standard birthday attack.

Thus, we have $H(M) \parallel G(M) = H(N) \parallel G(N) = h^* \parallel g^*$. The overall complexity of this attack is $\mathrm{O}(n_G\, 2^{n_H/2} + 2^{n_G/2})$. Note that we only assume that H is a classical iterated hash function; G can be any hash function at all.

2.5 Rate Function (Efficiency Measurement) of Known Designs

We have underlying compression functions $f_1, f_2, \cdots f_k : \{0,1\}^n \times \{0,1\}^m \rightarrow \{0,1\}^n$. We design a double length compression function, $F : \{0,1\}^N \rightarrow \{0,1\}^{2n}$, based on $f_1, f_2, \cdots f_k$. We define a measurement of the efficiency of the compression function, $F(\cdot)$, called the rate function of F. Roughly, it measures the number of message blocks that are hashed per underlying compression function.

Definition 3. (Rate Function)
Let a double length compression function, F, be based on $f_1, \cdots, f_k$. Define the rate function of F by $\frac{N-2n}{m \times s}$, where s is the number of invocations of all f_i's are required to compute $F(X)$, $X \in \{0,1\}^N$.

Since F is a compression function, $N > 2n$. Thus, the rate function is always positive. When the rate function is constant, we only use the term *rate* instead of rate function. We define rate of the classical iterated hash function by the rate of the underlying compression function.

Example 1. The underlying double length compression function of the concatenated hash function $H^{f_1} \parallel H^{f_2}$ is $F(H_1, H_2, M) = f_1(H_1, M) \parallel f_2(H_2, M)$, where $|H_1| = |H_2| = n$ and $|M| = m$. The rate function of F is $1/2$.

Example 2. **(Nandi et. al. [13])** Let $f_i : \{0,1\}^{2n} \rightarrow \{0,1\}^n$ be three underlying compression functions, $1 \leq i \leq 3$. Define, $F : \{0,1\}^{3n} \rightarrow \{0,1\}^{2n}$, where $F(x,y,z) = (f_1(x,y) \oplus f_2(y,z)) \parallel (f_2(y,z) \oplus f_3(z,x))$ with $|x| = |y| = |z| = n$. The rate of this compression function is $1/3$. The best collision attack on F requires $\Omega(2^{2n/3})$ many queries of f_i's in the random oracle model [13].

Example 3. **(Lucks [10])** Let $f : \{0,1\}^{n+m} \rightarrow \{0,1\}^n$ be an underlying compression function, $m > n$. Define $F(H_1, H_2, M) = f(H_1, H_2, M) \parallel f(H_2, H_1, M)$, $|H_1| = |H_2| = n$ and $|M| = m - n$. The rate function of the compression function is $\frac{n+m-2n}{2m} = \frac{1}{2} - \frac{n}{2m}$. When $m = 2n$, the rate of the compression function is $\frac{1}{4}$. The compression function is not secure. But we show later (also see [10]) that the classical hash function based on it is maximally secure in the random oracle model.

3 A Class of Double Length Compression Functions

Fix a compression function, $f : \{0,1\}^{n+m} \to \{0,1\}^n$, and define a class of double length compression functions

$$\mathcal{C} = \{f^p = f(\cdot) \parallel f(p(\cdot)) : p \text{ is a "simple" permutation on } \{0,1\}^{n+m}\}.$$

By a simple permutation we mean both the permutations, p and p^{-1}, are easy to compute. One can also consider a compression function $f^{p_1,p_2}(X) = f(p_1(X)) \parallel f(p_2(X))$ where $|X| = n + m$ for two simple permutations p_1 and p_2. Since p_1 and p_2 are simple, it is sufficient to study the security properties of f^p where $p = p_2 \circ p_1^{-1}$. All these compression function have rate $\frac{1}{2} - \frac{n}{2m}$ (as in Example 3).

3.1 Security Analysis of the Compression Functions from $\mathcal{C}$

In this section, we study the security properties of the compression functions from the class, $\mathcal{C}$, in the random oracle model of f. Let us first consider the Example 3. In this example, $F = f^p$, where $p(H_1, H_2, M) = H_2 \parallel H_1 \parallel M$, $|H_1| = |H_2| = n$ and $|M| = m - n$. By using the birthday attack, find H, G and M_1, M_2, such that $(H, M_1) \neq (G, M_2)$ and $f(H, H, M_1) = f(G, G, M_2)$. Now, it is easy to check that $f^p(H, H, M_1) = f^p(G, G, M_2)$. Here, we need $O(2^{n/2})$ many queries.

The reason for having the above attacks is that the permutation p has many "fixed points". X is called a *fixed point* of a function $p(\cdot)$, if $p(X) = X$. We write $\mathcal{F}_p$ for the set of all fixed points of p. In the above example, $\mathcal{F}_p = \{H \parallel H \parallel M : |H| = n, |M| = m - n\}$ is the set of fixed points of the permutation p and $|\mathcal{F}_p| > 2^n$. Thus, one can apply birthday attack to find a collision (or a preimage) on the compression function f from the fixed point set. Similar attack can be done for any compression function based on a permutation with more that 2^n many fixed points. In the light of the above discussion, one should use a permutation, p, which does not have many fixed points. In fact, there are many permutations where the set of fixed points are the empty set. We give two classes of examples of that kind, in below.

Example 4. For $A \in \{0,1\}^N \setminus \{\mathbf{0}\}$, define a permutation $p : \{0,1\}^N \to \{0,1\}^N$ such that $p(X) = X \oplus A$. It is easy to check that $\mathcal{F}_p$ is empty.

Example 5. We can map any N-bit string to an integer modulo 2^N. We use " $+$ " to denote addition modulo 2^N. Let $p(X) = X + A$ where $A \neq 0$. Note that, $p(X) \neq X$ for all X. Moreover, if $A \neq 2^{N-1}$ then the fixed point of $p(p(X)) = X + 2A$ (in notation, p^2) is also empty.

Suppose, f^p is a double length compression function based on a permutation, p, where $\mathcal{F}_p$ is the empty set. Then a collision, $f^p(X) = f^p(Y)$ with $X \neq Y$ implies $f(X) = f(Y)$ and $f(p(X)) = f(p(Y))$. Thus, $\{X, Y\}$ and $\{p(X), p(Y)\}$ are collision sets of f. Now, we have the following two cases.

- **Case-1 :** Two collision sets are identical $\{X, Y\} = \{p(X), p(Y)\}$. Since p does not have any fixed point, we have $Y = p(X)$ and $X = p(Y)$. Thus, we should have a collision set $\{X, p(X)\}$, where $p(X) \neq X$ and $p(p(X)) = X$. Let $\Omega(K_1(n))$ (or in short K_1) be the complexity of the best attack to find a collision set of the form $\{X, p(X)\}$.
- **Case-2 :** $\{X, Y\} \neq \{p(X), p(Y)\}$. Let $\Omega(K_2(n))$ (or in short K_2) be the complexity of the best attack to find two distinct collision sets of the form $\{X, Y\}$ and $\{p(X), p(Y)\}$.

Thus a collision on f^p reduces to the one of the above two events and hence the complexity of best collision attack is $\min\{K_1, K_2\}$. If p^2 does not have any fixed point then we can exclude the first case also and the complexity of the best collision attack is $K_2(n)$. We summarize the above discussion as follows;

Proposition 3. *The complexity of the best collision attack on f^p is* $\min\{\Omega(K_1(n)), \Omega(K_2(n))\}$ *where p is a permutation with no fixed point and K_1 and K_2 are defined as above. Moreover, if the permutation p^2 does not have any fixed point (like in the Example 5) then the best collision attack on f^p is $\Omega(K_2(n))$.*

Now we give some evidences why K_1 and K_2 would be large for a good compression function f. Suppose an adversary tries to find two collision sets $\{X, Y\} \neq \{p(X), p(Y)\}$. After finding a collision set $\{X, Y\}$, he does not have any freedom to choose for the second collision set and he is forced to check whether $\{p(X), p(Y)\}$ is a collision set or not. Thus K_2 would be large and may be close to 2^n for a good underlying compression function.

Next, an adversary tries to find a collision set $\{X, p(X)\}$. For each message X, $p(X)$ is completely determined (also vice-versa) and hence the adversary has to check equality of two values, $f(X)$ and $f(p(X))$, instead of comparing several values like in the birthday attack. Thus we expect K_1 to be large. In the random oracle model of f, we can prove that, $K_1(n) = K_2(n) = 2^n$. We first introduce a new notion called *computable message*.

Definition 4. (Computable Message)
Let the double length compression function, F, be based on the compression functions, $f_1, \cdots, f_k$. Let $\mathcal{Q}_j = \{(x_1^j, y_1^j), \cdots, (x_{q_j}, y_{q_j})\}$ be the view *of f_j, $1 \leq j \leq k$ and let $\mathcal{Q} = (\mathcal{Q}_1, \cdots, \mathcal{Q}_k)$. We say, an input X is* computable message *of F with respect to the view $\mathcal{Q}$, if the value of $F(X)$ can be determined from $\mathcal{Q}$.*

For example, when $F = f^p$, an input X is computable message of F with respect to $\{(x_1, y_1), \cdots, (x_q, y_q)\}$, view of f, if $X = x_i$ and $p(X) = x_j$ for some $i, j \in [1, q]$. Thus, $f^p(X) = f(x_i) \parallel f(x_j) = y_i \parallel y_j$, which can be computed from $\mathcal{Q}$.

Theorem 1. *Under the assumption of the random oracle model of f, $K_1(n) = K_2(n) = 2^n$. Thus, for any permutation p where $\mathcal{F}_p$ is the empty set, any attack algorithm finding collisions requires $\Omega(2^n)$ many queries of f in the random oracle model of f.*

Proof. If an adversary can ask at most q many queries then he can have at most q many computable messages and hence at most $\binom{q}{2}$ 2-sets $\{X, Y\}$. Hence the probability that the adversary finds $X \neq Y$ with $\{X, Y\} \neq \{p(X), p(Y)\}$ such that $f(X) = f(Y)$ and $f(p(X)) = f(p(Y))$ is at most $\binom{q}{2}/2^{2n}$ (by using Proposition 1). Thus, to have a significant success probability, q should be $\Omega(2^n)$.

Similarly, from a set of q queries one can get $O(q)$ many pairs of the form $(X, p(X))$, where X and $p(X)$ both are computable. For fixed X, $\mathsf{Pr}\,[f(X) = f(p(X))] = 1/2^n$ provided $p(X) \neq X$ (see Proposition 1). Thus success probability is at most $q/2^n$ and hence $q = \Omega(2^n)$ for significant success probability. □

Remark 1. Let p be a permutation where $|\mathcal{F}_p| << 2^{n/2}$ such that there are no two elements $X \neq Y \in \mathcal{F}_p$ with $f(X) = f(Y)$. We can prove Theorem 1 if the permutation p satisfies the above condition instead of the condition given in the Theorem.

4 A Class of Double Length Hash Functions

Now we study the double length hash functions defined by the classical iteration of the compression functions f^p stated in Sect. 3.

Definition 5. *Let p be a permutation on the set of $(n+m)$-bits strings, $m \geq n$. Define $\mathcal{F}_p[2n] = \{Z \in \{0,1\}^{2n} : \exists\ M \in \{0,1\}^{m-n}$ such that $Z \parallel M \in \mathcal{F}_p\ \}$. It is a projection of $\mathcal{F}_p$ onto the the first $2n$-bits of it. We say the permutation, $p(\cdot)$, is* good *if $|\mathcal{F}_p[2n]| = O(2^n)$. In particular, when p does not have any fixed point it is also a good permutation.*

Now we define the following attack. Find M and $H \notin \mathcal{F}_p[2n]$, such that $f^p(H, M) \in \mathcal{F}_p[2n]$, where $|M| = m - n$ and $|H| = 2n$. Let the complexity of the best attack be $\Omega(K_3(n))$ (or in short K_3).

Proposition 4. *The classical hash function, H^{f^p}, based on a good permutation and an initial value $H_0 \notin \mathcal{F}_p[2n]$ has collision security* $\min\{K_1, K_2, K_3\}$.

Proof. Let (M, M') be a collision pair of H^{f^p} and $H_0 \notin \mathcal{F}_p[2n]$. We denote H_i and G_i for internal hash values while computing the final hash value for messages $M = M_1 \parallel M_2 \cdots$ and $M' = M'_1 \parallel M'_2 \cdots$ respectively. One of the following holds:

1. There is an i such that $H_i \notin \mathcal{F}_p[2n]$ but $f^p(H_i \parallel M_{i+1}) \in \mathcal{F}_p[2n]$ or there is a j such that $G_j \notin \mathcal{F}_p[2n]$ but $f^p(G_j \parallel M'_{j+1}) \in \mathcal{F}_p[2n]$. To achieve this we need $\Omega(K_3)$ many queries.
2. There are $H_i, G_j \notin \mathcal{F}_p[2n]$ such that $X = (H_i, M_{i+1}) \neq (G_j, M') = Y$ and $f^p(X) = f^p(Y)$. Since $H_i, G_j \notin \mathcal{F}_p[2n]$, $p(X) \neq X$ and $p(Y) \neq Y$. Thus either $\{X, Y\} \neq \{p(X), p(Y)\}$ are collision two sets or $\{X, p(X)\}$ is a collision set for the compression function f. To achieve this adversary requires $\min\{K_1, K_2\}$ many queries of f.

Combining both the adversary needs $\min\{K_1, K_2, K_3\}$ queries. □

Theorem 2. *For a good permutation p and random compression function f, $K_3(n) = 2^n$ and hence H^{f^p} is maximally secure against collision attack.*

Proof. We have already seen that after q many queries the adversary can have at most q many computable messages for f^p. Given a computable message $H \parallel M$ with $H \notin \mathcal{F}_p[2n]$, we have $p(H \parallel M) \neq H \parallel M$ and hence $f^p(H \parallel M)$ is uniformly distributed over the set $\{0,1\}^{2n}$. But $|\mathcal{F}_p[2n]| < 2^n$ since the permutation $p(\cdot)$ is good. Thus we have, $\mathsf{Pr}\ [f^p(H \parallel M) \in \mathcal{F}_p[2n]] \leq 1/2^n$. Since we have at most q computable message the success probability of the adversary is less than $q/2^n$. This proves the fact that $K_3(n) = 2^n$ under the random oracle model of $f(\cdot)$. By Theorem 1 and Proposition 4, H^{f^p} is maximally secure under the random oracle model of f. □

5 An Efficient Double Length Hash Function

Let the compression function be $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$. For $i > 1$, define $f^{(i)} : \{0,1\}^{(i+1)n} \to \{0,1\}^n$ by using the classical iteration. Thus, for $x_0 \parallel \cdots \parallel x_i$ with $|x_j| = n$, $0 \leq j \leq i$ and $h_0 = x_0$,

$$f^{(i)}(x_0 \parallel \cdots \parallel x_i) = h_i, \text{ where } h_j = f(h_{j-1}, x_j),\ 1 \leq j \leq i.$$

We call $f^{(i)}$ is the *i-iterated compression function*. Now we can observe that the multicollision on this compression function is not as easy as the classical hash function, since we restrict the number of message blocks. Any r^i-way collision on $f^{(i)}$ reduces to at least r-way collision on the underlying compression function f (by using pigeon-hole principle). Thus, if we assume that $(r+1)$-way collision on f is infeasible then we can have at most r^i-way collision on $f^{(i)}$. Recall that, in the random oracle model of f, r-way collision requires $\Omega(2^{n(r-1)/r})$ queries. Now we summarize this by the following lemma.

Lemma 1. *(r^i+1)-way collision on $f^{(i)}$ reduces to at least $(r+1)$-way collision on f. In particular, when f is a random function, the complexity of (r^i+1)-way collision attack on $f^{(i)}$ is $\Omega(2^{nr/(r+1)})$ and the complexity of (n^i+1)-way collision attack on $f^{(i)}$ is $\Omega(2^n)$.*

Like the concatenation of two independent hash functions we can define the concatenation of two independent i-iterated compression functions. Thus, given two independent compression functions, f_1 and f_2, we can define a double length compression function, $F_i(X) = f_1^{(i)}(X) \parallel f_2^{(i)}(X)$, $|X| = n(i+1)$. Obviously, in this construction, we need to assume $i \geq 2$. Otherwise, for $i = 1$, it does not compress the input. Now we can study the security property of this concatenated compression function in the random oracle model.

Lemma 2. *If f is a random function then for any two distinct $(i+1)$-block inputs X and Y, $\mathsf{Pr}\,[f^{(i)}(X) = f^{(i)}(Y)] \leq i/2^n$. If f_1 and f_2 are two independent random functions then $\mathsf{Pr}\,[F_i(X) = F_i(Y)] = i^2/2^{2n}$.*

Proof. Let j be the round number where collision of f occurs. Call this event by C_j. Thus, $f^{(i)}(X) = f^{(i)}(Y)$ implies $\cup_{j=1}^{i} C_j$. Now, $\mathsf{Pr}\,[C_j] = \mathsf{Pr}\,[f(X_j) = f(Y_j)$ and $X_j \neq Y_j] = 1/2^n$, X_j and Y_j denote the input of f at j^{th} invocation for messages X and Y respectively. Thus, $\mathsf{Pr}\,[\cup_{j=1}^{i} C_j] \leq i/2^n$.

$$\begin{aligned}\mathsf{Pr}\,[F_i(X) = F_i(Y)] &= \mathsf{Pr}\,[f_1^{(i)}(X) = f_1^{(i)}(Y), f_2^{(i)}(X) = f_2^{(i)}(Y)] \\ &= \mathsf{Pr}\,[f_1^{(i)}(X) = f_1^{(i)}(Y)] \times \mathsf{Pr}\,[f_2^{(i)}(X) = f_2^{(i)}(Y)] \\ &\leq i^2/2^{2n}.\end{aligned}$$

The second equality follows from the fact that f_1 and f_2 are independent random functions and the last inequality is immediate from the first half of the Lemma. □

Thus to find the collision probability for any adversary we need to compute the number of pairs (X, Y) it can get from any possible set of queries. Note that the adversary should compute the F-values of both X and Y. Now we state the computable message which means the message whose hash value can be computed from the set of queries the adversary made. We fix $i \geq 2$.

Definition 6. (Computable message) *Let $\mathcal{Q}_j$ be the set of query response tuples for the random function f_j, $j = 1, 2$. X is said to be a computable message for $f_j^{(i)}$ (also for F_i) with respect to $\mathcal{Q}_j$ if the value of $f_j^{(i)}(X)$ (or $F_i(X)$) can be computed from $\mathcal{Q}_j$ (or $\mathcal{Q}_1 \cup \mathcal{Q}_2$ respectively).*

More precisely, if $X = x_0 \parallel \cdots \parallel x_i$ then X is computable for $f_1^{(i)}$ with respect to $\mathcal{Q}_1$ if $(x_0 \parallel x_1, h_1)$, $(h_1 \parallel x_2, h_2)$,$\cdots$,$(h_{i-1} \parallel x_i, h_i) \in \mathcal{Q}_1$. Thus the $f_1^{(i)}$-value of X is h_i. Similarly one can define computable messages for $f_2^{(i)}$. A message X is computable with respect to $\mathcal{Q}_1 \cup \mathcal{Q}_2$ for the compression function F_i, if X is computable for both $f^{(i)}$ with respect to $\mathcal{Q}_j$, $j = 1, 2$.

Let q be the number of queries. We assume that $q = \mathrm{o}(2^n)$. Thus there is no n-way collision on both f_1 and f_2. Note that, the complexity of n-way collision on a random function is $\Omega(2^{n(n-1)/n}) = \Omega(2^n)$. Thus we can have at most n^{i-1}-way collision on $f_1^{(i-1)}$ or $f_2^{(i-1)}$. The number of computable messages for F_i is at most qn^{i-1}. Thus, the total number of pairs of the form (X, Y) where $X \neq Y$ are $(i+1)$-block inputs and both X and Y are computable messages is at most $q^2 n^{2(i-1)}/2$. Thus, the probability that we have a collision among these pairs is bounded by $i^2 q^2 n^{2(i-1)}/2^{2n+1}$. To have non-negligible probability we need $q = \Omega(2^n/in^{i-1})$. Thus we have the following theorem :

Theorem 3. *If f_1 and f_2 are two independent random functions then the complexity for finding a collision on F_i requires $\Omega(2^n/(in^{i-1}))$ queries.*

Efficiency of the compression function. The rate function of the compression function, F_i, is $((i+1)n - 2n)/2ni = \frac{1}{2} - \frac{1}{2i}$. Thus, the rate of the compression function is close to $1/2$ provided i is large. So we have a trade-off between the security level and the efficiency.

For $s \geq 2$, define a double length hash function $H : (\{0,1\}^n)^* \rightarrow \{0,1\}^{2n}$. We can define the hash function on arbitrary domain by applying some standard

padding rule. Let $M = m_1 \parallel \cdots \parallel m_l$ be l-block message, $|m_i| = n$ for each i. Let $l = (s-1)b+r$, where $0 \le r < s-1$. Thus, we divide the message $M = M_1 \parallel \cdots \parallel M_b \parallel M_{b+1}$, where $|M_i| = (s-1)n$, $1 \le i \le b$ and $|M_{b+1}| = rn$. In case of $r = 0$ we do not have any message block M_{b+1}. Let H_0 be an initial two block message that is $|H_0| = 2n$. Now define the hash function $H(H_0, M) = F_s^{(b)}(M_1 \parallel \cdots \parallel M_{b+1})$ if $r = 0$, otherwise $H(H_0, M) = F_r(F_s^{(b)}(M_1 \parallel \cdots \parallel M_{b+1}) \parallel M_{b+1})$.

Thus, the hash function is the classical iterated hash function using two underlying compression functions F_s and F_{r+1}. Thus any collision on H reduces to a collision on one of the compression functions. Thus we have the following theorem;

Theorem 4. *For any $s \ge 2$, collision on $H^{(s)}$ requires $\Omega(2^n/s^2n^{s-1})$ complexity.*

6 (2nd) Preimage Security Analysis

Similar to the previous section we can study the (2nd) preimage security. Recall that we say a message X is computable from the set of queries $\mathcal{Q}$ if $f^{(i)}(X)$ can be computed from the set $\mathcal{Q}$. We have already observed that if q is the maximum number of queries and at most r-way collision is possible then we can have qr^{i-1} computable messages. Now given M, $F_i(M)$ is a $2n$-bit random string. We also have observed that $\Pr[F(M) = F(N)] = i^2/2^{2n}$, where $M \ne N$. So, if $q = o(2^n)$ then the number of computable messages for N is at most $n^{i-1}q$. Thus, there will be a computable message $N \ne M$ such that $F_i(M) = F_i(N)$ is bounded by $qn^{i-1}/i^2 2^{2n}$. Thus complexity for any attack algorithm of 2nd preimage attack is $\Omega(2^{2n}/i^2n^{i-1})$.

Note that, in preimage or collision attack on F_i, we do not count the complexity for multicollision attack. In fact, it is likely that if the number of computable messages is qr^{i-1} then the number of queries is at least $q2^{n(r-1)/r}$. Thus one can try to prove the following statement in the random oracle model;

The complexity of the best collision (or preimage) attack on the above double length compression function F_i is $\Omega(2^n/i)$ (or $\Omega(2^{2n}/i)$ respectively).

7 Conclusion

We have studied several new double length compression functions. We first introduced a class of double length compression function which contains recently known constructions [6, 10]. We studied their security levels in the random oracle model. We also designed a double length compression function F_i of rate close to $1/2$ (the rate of concatenated hash function). The design is very much similar to the concatenated hash functions. It has almost maximal security level. In fact, we believe that the complexity of the best collision attack on the above double length compression function F_i is $\Omega(2^n/i)$. It would be interesting to prove our belief. A possible future research would be to design efficient as well as secure double length hash functions.

References

1. M. Bellare. *A Note on Negligible Function.* Journal of Cryptology, Springer-Verlag, vol **15**, No. 4, pp. 271-284, September 2002.
2. R. Canetti, O. Goldreich and S. Halvei. *The random oracle methodology, revisited.* 30^{th} Annual ACM Symposium on Theory of Computing (STOC), pp. 209-218, 1998.
3. I. B. Damgård. *A Design Principle for Hash Functions.* Advances in Cryptology - Crypto'89, Lecture Notes in Computer Sciences, vol **435**, Springer-Verlag, pp. 416-427, 1989.
4. H. Finney. *More problems with hash functions.* The cryptographic mailing list, 24 Aug 2004. Available at http://lists.virus.org/cryptography-0408/msg00124.html.
5. M. Hattori, S. Hirose and S. Yoshida. *Analysis of Double Block Lengh Hash Functions.* 9th IMA International Conference Cryptographi and Coding, 2003, Lecture Notes in Computer Science, vol **2898**, 2003.
6. S. Hirose. *Provably Secure Double-Block-Length Hash Functions in a Black-Box Model*, 7th International Conference on Information Security and Cryptology, 2004.
7. A. Joux. *Multicollision on Iterated Hash Functions. Applications to Cascaded Constructions.* Advances in Cryptology - Crypto'04, Lecture Notes in Computer Science, vol **3152**, 2004.
8. L. Knudsen, X. Lai and B. Preneel. *Attacks on fast double block length hash functions.* Journal of Cryptology, vol **11**, no-1, winter, 1998.
9. L. Knudsen and B. Preneel. *Construction of Secure and Fast Hash Functions Using Nonbinary Error-Correcting Codes.* IEEE transactions on information theory, vol **48**, No. 9, Sept-2002.
10. S. Lucks. *Design principles for Iterated Hash Functions.* ePrint Archive Report, 2004. Available at http://eprint.iacr.org/2004/253.
11. R. Merkle. *One Way Hash Functions and DES.* Advances in Cryptology - Crypto'89, Lecture Notes in Computer Sciences, Vol. **435**, Springer-Verlag, pp. 428-446, 1989.
12. C. H. Meyer and M. Schilling. *Secure program load with manipulation detection code.* Proceedings Securicom, pp. 111-130, 1988.
13. M. Nandi, W. Lee, K. Sakurai and S. Lee. *Security Analysis of a 2/3-rate Double Length Compression Function in The Black-Box Model.* Fast Software Encryption'05, 2005.
14. M. Nandi and D. R. Stinson. *Multicollision Attacks on Generalized Hash Functions.* Cryptology ePrint Archive, 2004. Available at http://eprint.iacr.org/2004/330.
15. D. R. Stinson. *Cryptography : Theory and Practice*, Second Edition, CRC Press, Inc.
16. D. R. Stinson. *Some observations on the theory of cryptographic hash functions.* ePrint Archive Report, 2001. Available at http://eprint.iacr.org/2001/020/.
17. T. Satoh, M. Haga and K. Kurosawa. *Towards Secure and Fast Hash Functions.* IEICE Trans. vol **E82-A**, No. 1 January, 1999.