A Lightweight Hash Function Resisting Birthday Attack and Meet-in-the-middle Attack*

Shenghui Su [1, 2], Tao Xie [3], and Shuwang Lü [4]

[1] College of Computers, Beijing University of Technology, Beijing 100124
[2] College of Information Engineering, Yangzhou University, Yangzhou 225009
[3] School of Computers, National University of Defense Technology, Changsha 410073
[4] Graduate School, Chinese Academy of Sciences, Beijing 100039

Abstract: To be paired with a lightweight digital signing scheme of which the modulus length is between 80 and 160 bits, a new non-Merkle-Damgård structure (non-MDS) hash function is proposed by the authors based on a multivariate permutation problem (MPP) and an anomalous subset product problem (ASPP) to which no subexponential time solutions are found so far. It includes an initialization algorithm and a compression algorithm, and converts a short message of n bits treated as only a block into a digest of m bits, where $80 \le m \le 232$ and $80 \le m \le n \le 4096$. Analysis shows that the new hash is one-way, weakly collision-free, and strongly collision-free along with a proof, and its security against existent attacks such as birthday attack and meet-in-the-middle attack gets the $O(2^m)$ magnitude. Running time of its compression algorithm is analyzed to be $O(nm^2)$ bit operations. A comparison with the Chaum-Heijst-Pfitzmann hash based on a discrete logarithm problem is made. Especially, the new hash with short input and small computation may be used to reform a classical hash with an m-bit output and an $O(2^{m/2})$ magnitude security into a compact hash with an $m/2$-bit output and the same security. Thus, it opens a door to convenience for utilization of lightweight digital signing schemes.

Keywords: Hash function; Compression algorithm; Merkle-Damgård structure; Provable security; Birthday attack; Meet-in-the-middle attack

1 Introduction

In recent years, the ECC-160 digital signing scheme, an analogue of the ElGamal digital signing scheme based on a discrete logarithm problem (DLP) in an elliptic curve group over a finite field [1][2], and some lightweight digital signing schemes — the optimized version of the REESSE1+ digital signing scheme [3] for example have been utilized for RF ID (Radio Frequency Identity) tags or non-RF ID tags [4][5][6].

While a RF ID tag contains an IC chip which is used to store signatures and other data, an non-RF ID tag contains no IC chip because a short signature from a lightweight or ultra-lightweight signing scheme may be symbolized in short length, and printed directly on a papery tag or label. Now, such tags are applied to the identification, authentication, or anti-forgery of financial-notes, certificates, diplomas, and commodities, particularly including food and drug.

It is well understood that we first need to extract the digest of a message by employing a hash function before signing the message. Traditionally, a hash function consists of a compression function and the Merkle-Damgård structure (MDS) [7][8]. Let $\hat{h}$ be a hash function, and generally, it has the following four properties [9][10]:

① given a message $\underline{m}$, it is very easy to calculate the message digest $\underline{g} = \hat{h}(\underline{m})$, where $\underline{g}$ is also called a hash output;
② given a digest $\underline{g}$, it is very hard to calculate the message $\underline{m}$ according to $\underline{g} = \hat{h}(\underline{m})$, namely $\hat{h}$ is one-way;
③ given any arbitrary message $\underline{m}$, it is computationally infeasible to find another message $\underline{m}'$ such that $\hat{h}(\underline{m}) = \hat{h}(\underline{m}')$, namely $\hat{h}$ is weakly collision-free;
④ it is computationally infeasible to find two arbitrary messages $\underline{m} \neq \underline{m}'$ such that $\hat{h}(\underline{m}) = \hat{h}(\underline{m}')$, namely $\hat{h}$ is strongly collision-free.

The word "infeasible" means that some problem cannot be solved at least in polynomial time. Sometimes, ④ is optional with some users of a hash function because ①, ②, and ③ are enough for most of applications of the users.

At present, SHA-1, SHA-256, and SHA-384 announced by NIST are among the hash functions that are believed to be secure [9][11] though they each cannot resist birthday attack of which the time

* This work is supported by MOST with Project 2007CB311100 and 2009AA01Z441. Corresponding email: reesse@126.com.

complexity is $O(2^{m/2})$ that means that the security of each them is nearly the $O(2^{m/2})$ magnitude, where m is the bit-length of a message digest namely a hash output. It is well known that the output bit-lengths of these three functions are 160, 256, and 384 respectively.

When any of the three is practically paired with a lightweight signing scheme of which the modulus bit-length is between 80 and 160, its output must be adjusted to the range of the modulus bit-length of the singing scheme with its security unchanged or corresponding to the signing scheme.

The modulus bit-length of the optimized REESSE1+ signing scheme based on a transcendental logarithm problem and a polynomial root finding problem is 80 [3], and its security is the 2^{80} magnitude. When SHA-1 is paired with this signing scheme, the output of SHA-1 must be adjusted to 80 bits with its security unchanged. Again when SHA-256 is paired with ECC-160, the output of SHA-256 must be adjusted to 160 bits with its security being at least the 2^{80} magnitude.

Therefore, it is a problem in practice how to adjust a message digest from a classical hash function to the range of the modulus bit-length of a host signing scheme and to keep the security of the message digest being unchanged or corresponding to the host signing scheme.

To settle this problem, the authors design a new non-MDS hash function called JUNA which includes two algorithms: an initialization algorithm and a compression algorithm, converts a short message or a message digest of n bits into an output string of m bits, where $80 \le m \le 232$ and $80 \le m \le n \le 4096$, and moreover ensures that the security of the output string against collision attacks gets the $O(2^m)$ magnitude.

The paper has two dominant novelties: ① designing the initialization algorithm based on a multivariate permutation problem which only has an exponential time solution currently, and makes the new hash function be able to resist a birthday attack; ② designing the compression algorithm based on an anomalous subset product problem which also only has an exponential time solution currently, and makes the new hash function be able to resist other classical attacks, especially including a meet-in-the-middle attack. The significance of the paper lies in the thing that a new non-MDS hash function with an m-bit output and the $O(2^m)$ magnitude security is first proposed by the authors while a classical iterative hash function is with an m-bit output and only the $O(2^{m/2})$ magnitude security.

In Section 2 of the paper, several relevant definitions are given. In Section 3, the two algorithms of the new hash function are described. In Section 4, the security of the new hash function is analyzed. In Section 5, the running time of the compression algorithm of the new hash is dissected, a comparison with another non-MDS hash, the Chaum-Heijst-Pfitzmann hash based on a discrete logarithm problem, is made, and the reformation of a classical hash function is illustrated.

Throughout the paper, unless otherwise specified, an even number $n \ge 80$ is the bit-length of a short message (a message digest) or the item-length of a sequence, the sign % denotes "modulo", $\bar{M}$ does "$M - 1$" with M prime, $\lg x$ denotes a logarithm of x to the base 2, $\neg b_i$ does NOT operation of a bit b_i, $\mathbb{P}$ does the maximal prime allowed in a coprime sequence, $|x|$ does the absolute value of a number x, $\|x\|$ does the order of x % M, $\|S\|$ does the size of a set S, and $\gcd(x, y)$ represents the greatest common divisor of two integers x and y. Without ambiguity, "% M" is usually omitted in expressions.

2 Several Definitions

Before the two algorithms of the new non-MDS hash function are described, three relevant definitions are presented.

2.1 A Coprime Sequence

Definition 1: If $A_1, \ldots, A_n$ are n pairwise distinct positive integers such that $\forall\ A_i$ and A_j $(i \ne j)$, either $\gcd(A_i, A_j) = 1$ or $\gcd(A_i, A_j) = F \ne 1$ with $(A_i / F) \nmid A_k$ and $(A_j / F) \nmid A_k\ \forall\ k\ (\ne i, j) \in [1, n]$, these integers are called a coprime sequence, denoted by $\{A_1, \ldots, A_n\}$, and shortly $\{A_i\}$.

Notice that the elements of a coprime sequence are not necessarily pairwise coprime, but a sequence of which the elements are pairwise coprime is a coprime sequence.

For example, {21, 15, 29, 23, 11, 17, 19, 13} and {23, 7, 11, 3, 19, 13, 5, 17} are two coprime sequences separately.

Property 1: Let $\{A_1, \ldots, A_n\}$ be a coprime sequence. If randomly select $k \in [1, n]$ elements $A_{x_1}, \ldots, A_{x_k}$ from the sequence, then the mapping from a subset $\{A_{x_1}, \ldots, A_{x_k}\}$ to a subset product $G = \prod_{i=1}^{k} A_{x_i}$ is one-to-one, namely the mapping from $b_1 \ldots b_n$ to $G = \prod_{i=1}^{n} A_i^{b_i}$ is one-to-one, where $b_1 \ldots b_n$ is a bit string.

Refer to [3] for its proof.

2.2 A Bit Shadow and a Bit Long-Shadow

Definition 2: Let $b_1 \ldots b_n \neq 0$ be a bit string. Then $\dot{b}_i$ with $i \in [1, n]$ is called a bit shadow if it comes from such a rule: ① $\dot{b}_i = 0$ if $b_i = 0$; ② $\dot{b}_i = 1$ + the number of successive 0-bits before b_i if $b_i = 1$; or ③ $\dot{b}_i$ = 1 + the number of successive 0-bits before b_i + the number of successive 0-bits after the rightmost 1-bit if b_i is the leftmost 1-bit.

Notice that (3) of this definition is slightly different from that in [3].

For example, let $b_1 \ldots b_8 = 01010110$, then $\dot{b}_1 \ldots \dot{b}_8 = 03020210$.

Fact 1: Let $\dot{b}_1 \ldots \dot{b}_n$ be the bit shadow string of $b_1 \ldots b_n \neq 0$. Then there is $\sum_{i=1}^{n} \dot{b}_i = n$.

Proof:

According to Definition 2, every bit of $b_1 \ldots b_n$ is considered into $\sum_{i=1}^{k} \dot{b}_{x_i}$, where $\dot{b}_{x_1}$, …, $\dot{b}_{x_k}$ are 1-bit shadows in the string $\dot{b}_1 \ldots \dot{b}_n$, and there is $\sum_{i=1}^{k} \dot{b}_{x_i} = n$.

On the other hand, there is $\sum_{j=1}^{n-k} \dot{b}_{y_j} = 0$, where $\dot{b}_{y_1}$, …, $\dot{b}_{y_{n-k}}$ are 0-bit shadows.

In total, there is $\sum_{i=1}^{n} \dot{b}_i = n$. □

Property 2: Let $\{A_1, \ldots, A_n\}$ be a coprime sequence, and $\dot{b}_1 \ldots \dot{b}_n$ be the bit shadow string of $b_1 \ldots b_n \neq 0$. Then the mapping from $b_1 \ldots b_n$ to $G = \prod_{i=1}^{n} A_i^{\dot{b}_i}$ is one-to-one.

Proof:

Step 1.

Let $b_1 \ldots b_n$ and $b'_1 \ldots b'_n$ be two different nonzero bit strings, and $\dot{b}_1 \ldots \dot{b}_n$ and $\dot{b}'_1 \ldots \dot{b}'_n$ be the two corresponding bit shadow strings.

If $\dot{b}_1 \ldots \dot{b}_n = \dot{b}'_1 \ldots \dot{b}'_n$, then by Definition 2, there is $b_1 \ldots b_n = b'_1 \ldots b'_n$.

In addition, for any arbitrary bit shadow string $\dot{b}_1 \ldots \dot{b}_n$, there always exists a preimage $b_1 \ldots b_n$. Thus, the mapping from $b_1 \ldots b_n$ to $\dot{b}_1 \ldots \dot{b}_n$ is one-to-one.

Step 2.

Obviously the mapping from $\dot{b}_1 \ldots \dot{b}_n$ to $\prod_{i=1}^{n} A_i^{\dot{b}_i}$ is surjective.

Again presuppose that $\prod_{i=1}^{n} A_i^{\dot{b}_i} = \prod_{i=1}^{n} A_i^{\dot{b}'_i}$ for $\dot{b}_1 \ldots \dot{b}_n \neq \dot{b}'_1 \ldots \dot{b}'_n$.

Since $\{A_1, \ldots, A_n\}$ is a coprime sequence, and $A_i^{\dot{b}_i}$ either equals 1 with $\dot{b}_i = 0$ or contains the same prime factors as those of A_i with $\dot{b}_i \neq 0$, we can obtain $\dot{b}_1 \ldots \dot{b}_n = \dot{b}'_1 \ldots \dot{b}'_n$ from $\prod_{i=1}^{n} A_i^{\dot{b}_i} = \prod_{i=1}^{n} A_i^{\dot{b}'_i}$, which is in direct contradiction to $\dot{b}_1 \ldots \dot{b}_n \neq \dot{b}'_1 \ldots \dot{b}'_n$.

Therefore, the mapping from $\dot{b}_1 \ldots \dot{b}_n$ to $\prod_{i=1}^{n} A_i^{\dot{b}_i}$ is injective [12].

In summary, the mapping from $\dot{b}_1 \ldots \dot{b}_n$ to $\prod_{i=1}^{n} A_i^{\dot{b}_i}$ is one-to-one, and further the mapping from $b_1 \ldots b_n$ to $\prod_{i=1}^{n} A_i^{\dot{b}_i}$ is also one-to-one. □

Definition 3: Let $\dot{b}_1 \ldots \dot{b}_n$ be the bit shadow string of $b_1 \ldots b_n \neq 0$. Then $\underline{\dot{b}}_i = \dot{b}_i 2^{\partial_i}$ with $i \in [1, n]$ is called a bit long-shadow, where $\partial_i = b_{i+(-1)^{\lfloor 2(i-1)/n \rfloor}(n/2)} = 0$ or 1.

According to Definition 3, it is not difficult to understand that for every $\underline{\dot{b}}_i$, there is $0 \leq \underline{\dot{b}}_i \leq n$ when $b_1 \ldots b_n \neq 0$.

For example, let $b_1 \ldots b_8 = 01010110$, then $\underline{\dot{b}}_1 \ldots \underline{\dot{b}}_8 = 06020410$.

Fact 2: Let $\underline{\dot{b}}_1 \ldots \underline{\dot{b}}_n$ be the bit long-shadow string of $b_1 \ldots b_n \neq 0$. Then there is $n \leq \sum_{i=1}^{n} \underline{\dot{b}}_i \leq 2n$.

Proof:

By Definition 3 and Fact 1, we have

$$\sum_{i=1}^{n} \underline{\dot{b}}_i = \sum_{i=1}^{n} \dot{b}_i 2^{\partial_i} \text{ and } \sum_{i=1}^{n} \dot{b}_i = n.$$

If every $b_i = 1$, namely every $\partial_i = 1$, then

$$\sum_{i=1}^{n} \underline{\dot{b}}_i = \sum_{i=1}^{n} \dot{b}_i 2^{\partial_i} = 2\sum_{i=1}^{n} \dot{b}_i = 2n.$$

Again, by Definition 3, not all the bits of $b_1 \ldots b_n$ are zero.

If there exists only one nonzero bit in $b_1 \ldots b_n$ — $b_x = 1$ with $x \in [1, n]$ for example, then

$$\sum_{i=1}^{n} \underline{\dot{b}}_i = \sum_{i=1}^{n} \dot{b}_i 2^{\partial_i} = \dot{b}_x 2^{\partial_x} = \dot{b}_x = n,$$

where $\partial_x = b_{x+(-1)^{\lfloor 2(x-1)/n \rfloor}(n/2)} = 0$ due to b_x being the unique nonzero bit.

Thus, it holds that $n \leq \sum_{i=1}^{n} \underline{\dot{b}}_i \leq 2n$. □

Property 3: Let $\underline{\dot{b}}_1 \ldots \underline{\dot{b}}_n$ be the bit long-shadow string of $b_1 \ldots b_n \neq 0$. Then the mapping from $b_1 \ldots b_n$ to $\underline{\dot{b}}_1 \ldots \underline{\dot{b}}_n$ is one-to-one.

Proof:

On one hand, assume that a bit string $b_1 \ldots b_n \neq 0$ is known.

It is understood from Definition 3 that $\underline{b}_i = b_i 2^{\partial_i}$ for each i, where $\partial_i = b_{i+(-1)^{\lfloor 2(i-1)/n \rfloor}(n/2)}$.

Because when $b_1 \ldots b_n$ is known, $b_1 \ldots b_n$ and $\partial_1 \ldots \partial_n$ are respectively determined, $\underline{b}_1 \ldots \underline{b}_n$ can also be determined uniquely.

On the other hand, assume that a bit long-shadow string $\underline{b}_1 \ldots \underline{b}_n$ is known.

According to $\underline{b}_i = b_i 2^{\partial_i}$ and $\underline{b}_i = 0$ with $b_i = 0$, where $\partial_i = b_{i+(-1)^{\lfloor 2(i-1)/n \rfloor}(n/2)}$, we can determinate b_i for $i = 1, \ldots, n$ as follows.

① Case of $\underline{b}_i = 0$

If $\underline{b}_i = 0$, then $b_i = 0$, and set $b_i = 0$.

② Case of $\underline{b}_i \neq 0$

If $\underline{b}_i \neq 0$, then $b_i \neq 0$, and set $b_i = 1$.

In this way, the value of every b_i can be determined uniquely.

In summary, the mapping from $b_1 \ldots b_n$ to $\underline{b}_1 \ldots \underline{b}_n$ is one-to-one. □

2.3 A Lever Function

The designing of the initialization algorithm of the new hash function is based on the hard problem $C_i \equiv (A_i W^{\ell(i)})^{\delta} \ (\% \ M)$ for $i = 1, \ldots, n$ which is first used for the REESSE1+ asymmetric cryptosystem, where the exponent $\ell(i)$ is called a lever function [3].

In the paper, we still borrow the concept of the lever function but a public key is regarded as an initial value, and a private key (parameter) is only used for the generation of the initial value, not for decryption.

Definition 4*:* The secret parameter $\ell(i)$ in the transform of a non-MDS hash function is called a lever function, if it has the following features:

① $\ell(.)$ is an injection from the domain $\{1, \ldots, n\}$ to the codomain $\Omega \subset \{5, \ldots, \bar{M}\}$, where $\bar{M}$ is large;

② the mapping between i and $\ell(i)$ is established randomly without an analytical expression;

③ an attacker has to be faced with all the permutations of elements in Ω when inferring a related private parameter from an initial value;

④ the owner of the private parameter only need to consider the polynomial arithmetic of elements in Ω when doing a certain computation.

Feature ③ and ④ make it clear that if n is large enough, it is infeasible for the attacker to search all the permutations of elements in Ω exhaustively while the computation by the owner of the private parameter is feasible in polynomial time in n. Thus, the amount of calculation on $\ell(.)$ is large at "a public terminal", and is small at "a private terminal".

Notice that the number of all the elements of Ω, namely the size of Ω is not less than n.

Property 4 **(Indeterminacy of $\ell(.)$)*:*** Let $\delta = 1$ and $C_i \equiv (A_i W^{\ell(i)})^{\delta} \ (\% \ M)$ with $\ell(i) \in \Omega = \{5, \ldots, n+4\}$ and $A_i \in \Lambda = \{2, \ldots, \bar{P} \mid 863 \leq \bar{P} \leq 1201\}$ for $i = 1, \ldots, n$. Then $\forall \ W \ (\|W\| \neq \bar{M}) \in (1, \bar{M})$, and $\forall \ x, y, z \ (x \neq y \neq z) \in [1, n]$,

① when $\ell(x) + \ell(y) = \ell(z)$, there is

$$\ell(x) + \|W\| + \ell(y) + \|W\| \neq \ell(z) + \|W\| \ (\% \ \bar{M});$$

② when $\ell(x) + \ell(y) \neq \ell(z)$, there always exist

$$C_x \equiv A'_x W'^{\ell'(x)} \ (\% \ M), \ C_y \equiv A'_y W'^{\ell'(y)} \ (\% \ M), \text{ and } C_z \equiv A'_z W'^{\ell'(z)} \ (\% \ M)$$

such that $\ell'(x) + \ell'(y) \equiv \ell'(z) \ (\% \ \bar{M})$ with $A'_z \leq \bar{P}$.

Proof:

① It is easy to understand that

$$W^{\ell(x)} \equiv W^{\ell(x)+\|W\|}, \ W^{\ell(y)} \equiv W^{\ell(y)+\|W\|}, \text{ and } W^{\ell(z)} \equiv W^{\ell(z)+\|W\|} \ (\% \ M).$$

Due to $\|W\| \neq \bar{M}$, $2\|W\| \neq \|W\|$, and $\ell(x) + \ell(y) = \ell(z)$, it follows that

$$\ell(x) + \|W\| + \ell(y) + \|W\| \neq \ell(z) + \|W\| \ (\% \ \bar{M}).$$

However, it should be noted that when $\|W\| = \bar{M}$, there is $\ell(x) + \|W\| + \ell(y) + \|W\| \equiv \ell(z) + \|W\| \ (\% \ \bar{M})$.

② Let $\bar{O}_{\mathrm{d}}$ be an oracle on solving a discrete logarithm problem.

Suppose that $W' \in [1, \bar{M}]$ is a generator of $(\mathbb{Z}^*_M, \cdot)$.

In light of group theories, $\forall \ A'_z \in \{2, \ldots, \bar{P}\}$, the congruence

$$C_z \equiv A'_z W'^{\ell'(z)} \ (\% \ M)$$

has a solution. Then, $\ell'(z)$ may be taken through $\bar{O}_{\mathrm{d}}$.

$\forall \ \ell'(x) \in [1, \bar{M}]$, and let

$$\ell'(y) \equiv \ell'(z) - \ell'(x) \ (\% \ \bar{M}).$$

Further, from the congruences $C_x \equiv A'_x W'^{\ell'(x)} \ (\% \ M)$ and $C_y \equiv A'_y W'^{\ell'(y)} \ (\% \ M)$, we can obtain many

distinct pairs (A'_x, A'_y), where $A'_x, A'_y \in (1, M)$, and $\ell'(x)+\ell'(y) \equiv \ell'(z)$ (% $\bar{M}$).

In this way, Property 4.2 is proven. □

Notice that letting $\Omega = \{5, \ldots, n + 4\}$, namely every $\ell(i) \geq 5$ makes seeking W from $W^{\ell(i)} \equiv A_i^{-1} C_i$ (% M) face an unsolvable Galois group when the value of $A_i \leq \bar{P}$ is guessed [13], and moreover Property 4 still holds when Ω is any subset containing n elements from $\{1, \ldots, \bar{M}\}$.

Property 4 manifests that will continued fraction attack on $C_i \equiv A_i W^{\ell(i)}$ (% M) by Theorem 12.19 in Section 12.3 of [14] be utterly ineffectual only if elements in Ω are fitly selected [15].

3 Design of the New Non-MDS Hash Function

The Chaum-Heijst-Pfitzmann hash function, a non-MDS one, is appreciable. It is based on a discrete logarithm problem, and proved to be strongly collision-free [16].

The new non-MDS hash function is composed of two algorithms which contain two main parameters m and n, where m denotes the bit-length of a modulus used in the new hash, n denotes the bit-length of a short message or a message digest from a classical hash function, and there are $80 \leq m \leq 232$ with $80 \leq m \leq n \leq 4096$.

Additionally, Λ and Ω are two integral sets, and their lengths should be selected in conformity to the values of m and n such that $2n^5 |\Omega|_{\|}^{\|} |\Lambda|^5 \geq 2^m$ with $2^{10} \leq |\Lambda| \leq 2^{32}$ and $n \leq \tilde{n} \leq 2^{32}$ (see Section 4.2.1), where $\tilde{n} = |\Omega|$, and $2^{10} \leq |\Lambda| \leq 2^{32}$ means $10 \leq \lceil \lg \bar{P} \rceil \leq 32$.

For example, as $m = 80 \leq n$, there should be $|\Lambda| = 2^{10}$ and $|\Omega| = n$; as $m = 96 \leq n$, should $|\Lambda| = 2^{12}$ and $|\Omega| = n$; as $m = 112 \leq n$, should $|\Lambda| = 2^{14}$ and $|\Omega| = n$; as $m = 128 \leq n$, should $|\Lambda| = 2^{16}$ and $|\Omega| = 2^{12}$; as $m = 232 \leq n$, should $|\Lambda| = 2^{32}$ and $|\Omega| = 2^{32}$.

Notice that in the arithmetic modulo $\bar{M}$, $-x$ represents $\bar{M} - x$.

3.1 Initialization Algorithm

This algorithm is employed by an authoritative third party or the owner of a key pair, and only needs to be executed one time.

INPUT: the bit-length m of a modulus with $80 \leq m \leq 232$;
the item-length n of a sequence with $80 \leq m \leq n \leq 4096$;
the maximal prime $\bar{P}$ with $10 \leq \lceil \lg \bar{P} \rceil \leq 32$;
the size $\tilde{n}$ of the set Ω with $2\tilde{n}n^5\bar{P}^5 \geq 2^m$ and $n \leq \tilde{n} \leq 2^{32}$.

S1: Produce $\Lambda \leftarrow \{2, 3, \ldots, \bar{P}\}$.
Produce a random coprime sequence $\{A_1, \ldots, A_n \mid A_i \in \Lambda\}$.

S2: Find a prime M with $\lceil \lg M \rceil = m$ such that $\bar{M} / 2$ is a prime,
or the least prime factor of $\bar{M} / 2 > 4n(2\tilde{n} + 3)$.

S3: Pick $W \in (1, \bar{M})$ making $\|W\| \geq 2^{m - \lceil \lg \bar{P} \rceil}$.
Pick $\delta \in (1, \bar{M})$ making $\gcd(\delta, \bar{M}) = 1$.

S4: Randomly yield $\Omega \leftarrow \{+/-5, +/-7, \ldots, +/-(2\tilde{n} + 3)\}$.
Randomly select a distinct $\ell(i) \in \Omega$ for $i = 1, \ldots, n$.

S5: Compute $C_i \leftarrow (A_i W^{\ell(i)})^\delta$ % M for $i = 1, \ldots, n$.

OUTPUT: an initial value $(\{C_i\}, M)$ which is public to the people.

A private parameter $(\{A_i\}, \{\ell(i)\}, W, \delta)$ may be discarded, but must not be divulged.

Assume that there is $C_i = C_j$ with $i \neq j$. Then $(A_i W^{\ell(i)})^\delta \equiv (A_j W^{\ell(j)})^\delta$ (% M), and $W^{\ell(i)-\ell(j)} \equiv A_j A_i^{-1}$ (% M). Because of $\bar{M} / 2$ = a prime or the least prime factor of $\bar{M} / 2 > 4n(2\tilde{n} + 3)$, the probability that the case $W^{\ell(i)-\ell(j)} \equiv A_j A_i^{-1}$ (% M), namely $C_i = C_j$ occurs is $1 / 2^m$.

At S3, to seek W, let $W \equiv g^{\bar{M}/F}$ (% M), where g is a generator of $(\mathbb{Z}_M^*, \cdot)$ obtained through Algorithm 4.80 in Section 4.6 of [9], and $F < 2^{\lceil \lg \bar{P} \rceil}$ is a factor of $\bar{M}$.

At S4, $\Omega = \{+/-5, +/-7, \ldots, +/-(2\tilde{n} + 3)\}$ indicates that Ω is one of $2^{\tilde{n}}$ potential sets, indeterminate, and unknown to the public, where "+/–" means the selection of the "+" or "–" sign.

Definition 5: Given $(\{C_i\}, M)$, seeking the original $(\{A_i\}, \{\ell(i)\}, W, \delta)$ from $C_i \equiv (A_i W^{\ell(i)})^\delta$ (% M) with $A_i \in \{2, 3, \ldots, \bar{P} \mid 10 \leq \lceil \lg \bar{P} \rceil \leq 32\}$ and $\ell(i) \in \{+/-5, +/-7, \ldots, +/-(2\tilde{n} + 3) \mid n \leq \tilde{n} \leq 2^{32}\}$ for $i = 1, \ldots, n$ is referred to as a multivariate permutation problem, shortly MPP [3].

Property 5: The MPP $C_i \equiv (A_i W^{\ell(i)})^\delta$ (% M) with $A_i \in \{2, 3, \ldots, \bar{P} \mid 10 \leq \lceil \lg \bar{P} \rceil \leq 32\}$ and $\ell(i) \in \{+/-5, +/-7, \ldots, +/-(2\tilde{n} + 3) \mid n \leq \tilde{n} \leq 2^{32}\}$ for $i = 1, \ldots, n$ is computationally at least equivalent to the

DLP in the same prime field.

See Section 4.1 for its proof.

3.2 Compression Algorithm

This algorithm is employed by a person who wants to obtain a short message digest.

INPUT: an initial value $(\{C_1, \ldots, C_n\}, M)$, where $\lceil \lg M \rceil = m$ with $80 \le m \le n \le 4096$;
a short message (or a message digest from a classical hash function) $b_1 \ldots b_n \ne 0$.

S1: Set $k \leftarrow 0$, $i \leftarrow 1$.

S2: If $b_i = 0$ then
S2.1: let $k \leftarrow k + 1$, $\underline{b}_i \leftarrow 0$
else
S2.2: if $i = k + 1$ then let $\bar{s} \leftarrow i$;
S2.3: let $\underline{b}_i \leftarrow k + 1$, $k \leftarrow 0$.

S3: Let $i \leftarrow i + 1$.
If $i \le n$ then go to S2.

S4: Compute $\underline{b}_{\bar{s}} \leftarrow \underline{b}_{\bar{s}} + k$.

S5: Compute $\underline{d} \leftarrow \prod_{i=1}^{n} C_i^{\underline{b}_i} \% M$,
where $\underline{b}_i = b_i 2^{\partial_i}$ with $\partial_i = b_{i+(-1)^{\lfloor 2(i-1)/n \rfloor}(n/2)}$.

OUTPUT: a digest $\underline{d} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} (\% M)$ of which the bit-length is m.

It is easily known from Definition 3 that the max of $\{\underline{b}_1, \ldots, \underline{b}_n\}$ is less than or equal to n when $b_1 \ldots b_n \ne 0$.

Definition 6: Given $(\underline{d}, M)$, seeking the original $\underline{b}_1 \ldots \underline{b}_n$ from $\underline{d} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} (\% M)$, where $\underline{b}_i = \underline{b}_i 2^{\partial_i}$ with $\partial_i = b_{i+(-1)^{\lfloor 2(i-1)/n \rfloor}(n/2)}$ and $\underline{b}_i$ being a bit shadow is referred to as an anomalous subset product problem, shortly ASPP [3].

Property 6: The ASPP $\underline{d} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} (\% M)$, where $\underline{b}_i = \underline{b}_i 2^{\partial_i}$ with $\partial_i = b_{i+(-1)^{\lfloor 2(i-1)/n \rfloor}(n/2)}$ and $\underline{b}_i$ being a bit shadow is computationally at least equivalent to the DLP in the same prime field.

See Section 4.3 for its proof.

4 Security Analysis of the New Non-MDS Hash Function

Because a hash function must be one-way, weakly collision-free, and sometimes required to be strongly collision-free, the new non-MDS hash function should also be at least one-way and weakly collision-free.

It is should be noted that $\lceil \lg M \rceil = m$, but not n, is the security dominant parameter of the new non-MDS hash function.

Definition 7: Let A and B be two computational problems. A is said to reduce to B in polynomial time, written as $A \le_{\mathrm{T}}^{\mathrm{P}} B$, if there is an algorithm for solving A which calls, as a subroutine, a hypothetical algorithm for solving B, and runs in polynomial time, excluding the time of the algorithm for solving B [9][17].

The hypothetical algorithm for solving B is called an oracle. It is easy to understand that no matter what the time complexity of the oracle is, it does not influence the result of the comparison.

$A \le_{\mathrm{T}}^{\mathrm{P}} B$ means that the difficulty of A is not greater than that of B, namely the time complexity of the fastest algorithm for solving A is not greater than that of the fastest algorithm for solving B when all polynomial times are treated as the identical magnitude. Concretely speaking, if A cannot be solved in polynomial or subexponential time, correspondingly B cannot also be solved in polynomial or subexponential time; and if B can be solved in polynomial or subexponential time, correspondingly A can also be solved in polynomial or subexponential time.

Definition 8: Let A and B be two computational problems. If $A \le_{\mathrm{T}}^{\mathrm{P}} B$ and $B \le_{\mathrm{T}}^{\mathrm{P}} A$, then A and B are said to be computationally equivalent, written as $A =_{\mathrm{T}}^{\mathrm{P}} B$ [9][17].

$A =_{\mathrm{T}}^{\mathrm{P}} B$ means that either if A is a intractability with a certain complexity on a condition that its dominant variable approaches a large number, B is also a intractability with the same complexity on the identical condition; or both A and B can be solved in linear or polynomial time.

Obviously, Definition 7 and 8 gives a partial order relation among the complexities or difficulties of computational problems [18], and suggest a reductive proof method called polynomial time Turing

reduction (PTR) [17].

In addition, for convenience sake, let $\hat{H}(y = f(x))$ represent the complexity or difficulty of the problem of solving $y = f(x)$ for x [19].

4.1 Proof of Property 5

In Section 3.1, the MPP is defined as $C_i \equiv (A_i W^{\ell(i)})^{\delta}$ (% M) with $A_i \in \Lambda = \{2, 3, \ldots, \tilde{P} \mid 10 \le \lceil \lg \tilde{P} \rceil \le 32\}$ and $\ell(i) \in \Omega = \{+/-5, +/-7, \ldots, +/-(2\tilde{n} + 3) \mid n \le \tilde{n} \le 2^{32}\}$ for $i = 1, \ldots, n$. What follows is the proof of Property 5, a property of the MPP.

Proof:

Firstly, systematically consider $C_i \equiv (A_i W^{\ell(i)})^{\delta}$ (% M) for $i = 1, \ldots, n$.

Assume that each $g_i \equiv A_i W^{\ell(i)}$ (% M) with $\ell(i) \in \{+/-5, +/-7, \ldots, +/-(2\tilde{n} + 3) \mid n \le \tilde{n} \le 2^{32}\}$ is a constant.

Let

$$g_i \equiv g^{x_i} \text{ (\% } M\text{), and } z_i \equiv \delta x_i \text{ (\% } \bar{M}\text{),}$$

where $g \in \mathbb{Z}_M^*$ be a generator.

Then, there is

$$C_i \equiv g_i^{\delta} \equiv g^{\delta x_i} \text{ (\% } M\text{) for } i = 1, \ldots, n.$$

Again let $\delta x_i \equiv z_i$ (% $\bar{M}$). Then

$$C_i \equiv g^{z_i} \text{ (\% } M\text{) for } i = 1, \ldots, n.$$

The above expression corresponds to the fact that in the ElGamal cryptosystem where many users share the modulus and a key generator, User 1 acquires a private key z_1 and a public key C_1, …, and User n acquires a private key z_n and a public key C_n. It is well known that in this case, the attack of an adversary is still faced with the DLP, namely seeking z_i from the simultaneous equation $C_i \equiv g^{z_i}$ (% M) for $i = 1, \ldots, n$ is computationally equivalent to the DLP [9].

Thus, when every g_i is weakened to a constant, seeking δ from $C_i \equiv g_i^{\delta}$ (% M) for $i = 1, \ldots, n$ is computationally equivalent to the DLP, which indicates that when every g_i is not a constant, seeking g_i and δ from $C_i \equiv g_i^{\delta}$ (% M) for $i = 1, \ldots, n$ is computationally at least equivalent to the DLP.

Secondly, singly consider a certain C_i, where the subscript i is designated.

Assume that $\tilde{O}_{\mathrm{m}}(C_i, M, \underline{R})$ is an oracle on solving $C_i \equiv g_i^{\delta}$ (% M) for g_i and δ, where i is in $\{1, \ldots, n\}$, and $\underline{R}$ is a constraint on g_i such that the original g_i and δ can be found.

Let $y \equiv g^x$ (% M) be of the DLP. Then, by calling $\tilde{O}_{\mathrm{m}}(y, M, g)$, x can be obtained.

According to Definition 7, there is

$$\hat{H}(y \equiv g^x \text{ (\% } M\text{)}) \le_{\mathrm{T}}^{\mathrm{P}} \hat{H}(C_i \equiv g_i^{\delta} \text{ (\% } M\text{)}),$$

which indicates that when only a certain g_i is known, seeking g_i and δ from $C_i \equiv g_i^{\delta}$ (% M) is computationally at least equivalent to the DLP.

Integrally, seeking the original $\{A_i\}$, $\{\ell(i)\}$, W, and δ from $C_i \equiv (A_i W^{\ell(i)})^{\delta}$ (% M) for $i = 1, \ldots, n$ is computationally at least equivalent to the DLP in the same prime field. □

4.2 Security of the Initialization Algorithm

Clearly, the security of the initialization algorithm depends on the security of the MPP $C_i \equiv (A_i W^{\ell(i)})^{\delta}$ (% M) with $A_i \in \Lambda = \{2, 3, \ldots, \tilde{P} \mid 10 \le \lceil \lg \tilde{P} \rceil \le 32\}$ and $\ell(i) \in \Omega = \{+/-5, +/-7, \ldots, +/-(2\tilde{n} + 3) \mid n \le \tilde{n} \le 2^{32}\}$ for $i = 1, \ldots, n$.

In [3], we analyze the security of the MPP $C_i \equiv (A_i W^{\ell(i)})^{\delta}$ (% M) with $A_i \in \{2, 3, \ldots, \tilde{P} \mid 863 \le \tilde{P} \le 1201\}$ and $\ell(i) \in \{5, 7, \ldots, (2n + 3)\}$ for $i = 1, \ldots, n$ from the three aspects, discover no subexponential time solution to it, and contrarily, find some evidence which inclines people to believe that the MPP is computationally harder than the DLP.

Considering that the set Ω is different from the old in [3], and the range of $\tilde{P}$ is larger than the old in [3], we will analyze the security of the MPP with different restrictions additionally.

4.2.1 Ineffectualness of Presupposing $\ell(x_1) + \ell(x_2) = \ell(y_1) + \ell(y_2)$

Because of $\Omega = \{+/-5, +/-7, \ldots, +/-(2\tilde{n} + 3)\}$, when the absolute values $|\ell(x_1)|$, $|\ell(x_2)|$, $|\ell(y_1)|$, $|\ell(y_2)|$ are determined, the value $\ell(x_1) + \ell(x_2) - (\ell(y_1) + \ell(y_2))$ has $2^4 = 16$ possible cases, which enhances the indeterminacy of the lever function, and increases the complexity of an attack task for cracking the MPP to some extent.

Adversaries may try to eliminate W through judging $\ell(x_1)+\ell(x_2)=\ell(y_1)+\ell(y_2)$.

$\forall\, x_1, x_2, y_1, y_2 \in [1, n]$, presuppose that $\ell(x_1)+\ell(x_2)=\ell(y_1)+\ell(y_2)$ holds.

Let $G_z \equiv C_{x_1}C_{x_2}(C_{y_1}C_{y_2})^{-1}$ (% M), namely

$$G_z \equiv (A_{x_1}A_{x_2}(A_{y_1}A_{y_2})^{-1})^{\delta} \text{ (\% } M\text{)}.$$

If the adversaries divine the values of A_{x_1}, A_{x_2}, A_{y_1}, A_{y_2}, and compute u, v_{x_1}, v_{x_2}, v_{y_1}, v_{y_2} in at least $L_M[1/3, 1.923]$ time such that

$$G_z \equiv g^{u},\ A_{x_1} \equiv g^{v_{x_1}},\ A_{x_2} \equiv g^{v_{x_2}},\ A_{y_1} \equiv g^{v_{y_1}},\ A_{y_2} \equiv g^{v_{y_2}} \text{ (\% } M\text{)},$$

where g is a generator of $(\mathbb{Z}_M^*, \cdot)$, then

$$u \equiv (v_{x_1}+v_{x_2}-v_{y_1}-v_{y_2})\delta \text{ (\% } \bar{M}\text{)}.$$

If $\gcd(v_{x_1}+v_{x_2}-v_{y_1}-v_{y_2}, \bar{M}) \mid u$, the congruence in δ has solutions. Because each of A_{x_1}, A_{x_2}, A_{y_1}, A_{y_2} may traverse the interval Λ, and the subscripts x_1, x_2, y_1, y_2 are unfixed, the number of potential values of δ is about $n^4|\Lambda|^4$. Notice that the number of non-repeated values of δ will be less than 2^m.

In succession, need to seek W. Now, the most effectual approach to seeking W is that for every i, the adversaries fix a value of δ, divine A_i and $\ell(i)$, and find the set $\bar{V}_i$ according to $C_i \equiv (A_i W^{\ell(i)})^{\delta}$ (% M), where $\bar{V}_i$ is the set of possible values of W meeting $C_i \equiv (A_i W^{\ell(i)})^{\delta}$ (% M) for $i = 1, \ldots, n$. If there exist $W_1 \in \bar{V}_1, \ldots, W_n \in \bar{V}_n$ being pairwise equal, the divination of δ, $\{A_i\}$, and $\{\ell(i)\}$ is thought right; else fix another value of δ, repeat the above process.

Notice that due to $\bar{M}/2 =$ a prime or the least prime factor of $\bar{M}/2 > 4n(2\tilde{n}+3)$, $W^{\ell(i)} \equiv C_i^{\delta^{-1}} A_i^{-1}$ (% M) can be solved in polynomial time, and besides letting $W = g^{\mu}$ % M is unnecessary.

It is not difficulty to understand that the size of every $\bar{V}_i$ is about $(2|\Omega|)|\Lambda|$.

In summary, the time complexity of the above attack task is

$$\begin{aligned} T &= (n+|\Lambda|)L_M[1/3, 1.923] + (n^4|\Lambda|^4) + (n^4|\Lambda|^4)(2|\Omega||\Lambda|)n \\ &\approx 2n^5|\Omega||\Lambda|^5. \end{aligned}$$

Concretely speaking,

For $m = n = 80$ with $|\Lambda| = 2^{10}$ & $|\Omega| = 80$, $T > 2(2^{6.3})^5(2^{6.3})(2^{10})^5 = 2^{88} > 2^m$.

For $m = n = 96$ with $|\Lambda| = 2^{12}$ & $|\Omega| = 96$, $T > 2(2^{6.5})^5(2^{6.5})(2^{12})^5 = 2^{100} > 2^m$.

For $m = n = 112$ with $|\Lambda| = 2^{14}$ & $|\Omega| = 112$, $T > 2(2^{6.8})^5(2^{6.8})(2^{14})^5 = 2^{112} = 2^m$.

For $m = n = 128$ with $|\Lambda| = 2^{16}$ & $|\Omega| = 2^{12}$, $T > 2(2^{7})^5(2^{12})(2^{16})^5 = 2^{128} = 2^m$.

For $m = n = 232$ with $|\Lambda| = 2^{32}$ & $|\Omega| = 2^{32}$, $T > 2(2^{7.8})^5(2^{32})(2^{32})^5 = 2^{232} = 2^m$.

Thus, the time complexity of the attack by presupposing $\ell(x_1)+\ell(x_2)=\ell(y_1)+\ell(y_2)$ is not less than $O(2^m)$ when $|\Lambda|$ and $|\Omega|$ are chosen suitably.

4.2.2 Ineffectualness of Guessing $\|W\|$

Owing to $80 \le \lceil \lg M \rceil \le 232$, $\bar{M}$ can be factorized in tolerable subexponential time, and further a value of $\|W\|$ can be guessed.

Adversaries may try to eliminate W through $W^{\|W\|} \equiv 1$ (% M).

Raising either side of every equation $C_i \equiv (A_i W^{\ell(i)})^{\delta}$ (% M) to the $\|W\|$-th power yields

$$C_i^{\|W\|} \equiv (A_i)^{\delta\|W\|} \text{ \% } M.$$

Suppose that the value of every $A_i \in \Lambda = \{2, 3, \ldots, \boldsymbol{P} \mid 10 \le \lceil \lg \boldsymbol{P} \rceil \le 32\}$ is guessed, or the possible values of every A_i are traversed.

Let $C_i \equiv g^{u_i}$ (% M), and $A_i \equiv g^{v_i}$ (% M), where g is a generator of $(\mathbb{Z}_M^*, \cdot)$. Then

$$u_i\|W\| \equiv v_i\|W\|\delta \text{ (\% } \bar{M}\text{)} \ (i = 1, \ldots, n).$$

Notice that $u_i \ne v_i\delta$ (% $\bar{M}$), and $\{v_1, \ldots, v_n\}$ is not a super increasing sequence.

The above congruence is seemingly the MH transform [20]. Actually, $\{v_1\|W\|, \ldots, v_n\|W\|\}$ is not a super increasing sequence, and moreover there is not necessarily $\lceil \lg(u_i\|W\|) \rceil = \lceil \lg \bar{M} \rceil$.

Because $v_i\|W\| \in [1, \bar{M}]$ is stochastic, the inverse δ^{-1} % $\bar{M}$ not need be close to the minimum $\bar{M}/(u_i\|W\|)$, $2\bar{M}/(u_i\|W\|)$, …, or $(u_i\|W\|-1)\bar{M}/(u_i\|W\|)$. Namely δ^{-1} may lie at any integral position of the interval $[k\bar{M}/(u_i\|W\|), (k+1)\bar{M}/(u_i\|W\|)]$, where $k = 0, 1, \ldots, u_i\|W\|-1$, which illustrates that the accumulation points of minima do not exist. Further observing, in this case, when i traverses the interval $[2, n]$, the number of intersections of the intervals containing δ^{-1} is likely the max of $\{u_1\|W\|, \ldots, u_n\|W\|\}$ which is promisingly close to $\bar{M}$. Therefore, the Shamir attack by the accumulation point of minima is fully ineffectual [21].

Even if find out δ^{-1} through the Shamir attack method, because each of $\{v_1, \ldots, v_n\}$ has $\|W\|$ solutions, the number of potential sequences $\{g^{v_1}, \ldots, g^{v_n}\}$ is up to $\|W\|^n$. Because of needing to verify whether

$\{g^{v_1}, \ldots, g^{v_n}\}$ is a coprime sequence for each different sequence $\{v_1, \ldots, v_n\}$, the number of possible coprime sequences is in proportion to $\|W\|^n$. Hence, the initial $\{A_1, \ldots, A_n\}$ cannot be determined in subexponential time. Further, the value of W cannot be computed, and the values of $\|W\|$ and δ^{-1} cannot be verified, which indicates that the MPP can also be resistant to the Shamir attack by the accumulation point of minima.

Additionally, the adversaries may divine the value of A_i in about $O(|A_i|)$ time with $i \in [1, n]$, and compute δ by $v_i\|W\| \equiv u_i\|W\|\delta \ (\% \ \bar{M})$. However, because of $\|W\| \mid \bar{M}$, the equation will have $\|W\|$ solutions. Therefore, the time complexity of finding the original δ is at least

$$\begin{aligned} T &= (n + |A_i|)L_M[1/3, 1.923] + |A_i|\|W\| \\ &\geq (n + |A_i|)L_M[1/3, 1.923] + 2^{\lceil \lg \underline{b} \rceil} 2^{m - \lceil \lg \underline{b} \rceil} \\ &> 2^m. \end{aligned}$$

It is also not less than $O(2^m)$.

4.3 Proof of Property 6

In Section 3.2, the ASPP is defined as $\underline{g} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% M)$, where $\underline{b}_i = \underline{b}_i 2^{\partial_i}$ with $\partial_i = b_{i + (-1)^{\lfloor 2(i-1)/n \rfloor}(n/2)}$ and $\underline{b}_i$ being a bit shadow. What follows is the proof of Property 6, a property of the ASPP.

Proof:

Assume that $\bar{O}_a(\underline{g}, C_1, \ldots, C_n, M)$ is an oracle on solving $\underline{g} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% M)$ for $\underline{b}_1 \ldots \underline{b}_n$, where $\underline{b}_1 \ldots \underline{b}_n$ is the bit long-shadow string of $b_1 \ldots b_n$.

Particularly, when $C_1 = \ldots = C_n = C$, define

$$\underline{g} \equiv \prod_{i=1}^{n} C^{(n+1)^{n-i} \underline{b}_i} \equiv \prod_{i=1}^{n} (C^{(n+1)^{n-i}})^{\underline{b}_i} \ (\% M)$$

with $0 \leq \underline{b}_i \leq n$, and define the corresponding oracle as $\bar{O}_a(\underline{g}, C^{(n+1)^{n-1}}, \ldots, C^{(n+1)^0}, M)$.

Let $\bar{G}_1 \equiv \prod_{i=1}^{n} C_i^{b_i} \ (\% M)$ be of the subset product problem (SPP) [3][22][23].

Since there is $0 \leq b_i \leq \underline{b}_i$, and the mapping from $\underline{b}_1 \ldots \underline{b}_n$ to $b_1 \ldots b_n$ is one-to-one, by calling $\bar{O}_a(\bar{G}_1, C_1, \ldots, C_n, M)$, we can find $b_1 \ldots b_n$.

By Definition 7, there is

$$\hat{H}(\bar{G}_1 \equiv \prod_{i=1}^{n} C_i^{b_i} \ (\% M)) \leq_T^P \hat{H}(\underline{g} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% M)).$$

By Property 5 in [3], there is

$$\hat{H}(y \equiv g^x \ (\% M)) \leq_T^P \hat{H}(\bar{G}_1 \equiv \prod_{i=1}^{n} C_i^{b_i} \ (\% M)).$$

Further, by transitivity, there is

$$\hat{H}(y \equiv g^x \ (\% M)) \leq_T^P \hat{H}(\underline{g} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% M)).$$

Therefore, solving $\underline{g} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% M)$ for $\underline{b}_1 \ldots \underline{b}_n$ is at least equivalent to the DLP in the same prime field in computational complexity. □

4.4 Security of the Compression Algorithm

The compression algorithm of which the input message is treated as only a block is the main body of the new non-MDS hash function, and thus, through it the four natural properties of the new hash function are embodied dominantly.

Clearly, the security of the compression algorithm depends on the security of the ASPP $\underline{g} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% M)$, where $\underline{b}_i = \underline{b}_i 2^{\partial_i}$ with $\partial_i = b_{i + (-1)^{\lfloor 2(i-1)/n \rfloor}(n/2)}$ and $\underline{b}_i$ being a bit shadow.

In [3], we analyze the security of the ASPP $\bar{G} \equiv \prod_{i=1}^{n} C_i^{b_i} \ (\% M)$ from the three aspects, discover no subexponential time solution to it, and contrarily, find some evidence which inclines people to believe that $\bar{G} \equiv \prod_{i=1}^{n} C_i^{b_i} \ (\% M)$ is computationally harder than the DLP. Due to $\underline{b}_i = \underline{b}_i 2^{\partial_i} \geq \underline{b}_i$, the security conclusion about $\bar{G} \equiv \prod_{i=1}^{n} C_i^{b_i} \ (\% M)$ is also suitable for $\underline{g} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% M)$ which is just another form of the ASPP. Hence $\underline{g} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% M)$ has no subexponential time solution at present.

In what follows, we will analyze whether the compression formula $\underline{g} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% M)$ satisfies the four natural properties of a hash function, and especially resists the three classical attacks or not.

In terms of Section 3.2, given the initial value $(\{C_i\}, M)$ and a short message $b_1 \ldots b_n$, it is transparently easy to calculate the digest $\underline{g} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% M)$.

4.4.1 Compression Algorithm Is Computationally One-way

Let $C_1 \equiv g^{u_1} \ (\% \ M)$, …, $C_n \equiv g^{u_n} \ (\% \ M)$, $\underline{d} \equiv g^{v} \ (\% \ M)$, where g is a generator of the group $(\mathbb{Z}_M^*, \cdot)$, and easily found when $\lceil \lg M \rceil < 1024$.

Then, solving $\underline{d} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% \ M)$ for $\underline{b}_1 \ldots \underline{b}_n$, namely $b_1 \ldots b_n$, is equivalent to solving

$$\underline{b}_1 u_1 + \ldots + \underline{b}_n u_n \equiv v \ (\% \ \bar{M}),$$

which is called an anomalous subset sum problem, shortly ASSP [3], and computationally at least equivalent to a subset sum problem (SSP) due to $\underline{b}_i = \underline{b}_i 2^{\partial_i} \geq \underline{b}_i \geq b_i \in [0, 1]$.

The SSP has been proved to be NP-complete in its feasibility recognition form, and its computational version, especially the high-density or big-length version, is NP-hard [9][24]. Hence, solving ASSP is at least NP-hard.

Moreover in the non-MDS hash function, there is $n \geq m = \lceil \lg M \rceil$ and $n \geq \underline{b}_i \geq b_i \in [0, 1]$. The knapsack density relevant to the ASSP $\underline{b}_1 u_1 + \ldots + \underline{b}_n u_n \equiv v \ (\% \ \bar{M})$ roughly equals

$$D = \sum_{i=1}^{n} \lceil \lg n \rceil / \lceil \lg M \rceil = n \lceil \lg n \rceil / m > \lceil \lg n \rceil > 1,$$

which means that there exists many solutions to $\underline{b}_1 u_1 + \ldots + \underline{b}_n u_n \equiv v \ (\% \ \bar{M})$, namely the original solution cannot be determined, or will not occur in a reduced lattice base defined by LLL [25]. Notice that only such a $\langle \underline{b}_1, \ldots, \underline{b}_n \rangle$ from which a right bit string can be deduced will be a reasonable solution vector. Experiments show that when $D > 1$, the probability that the original solution or a reasonable solution is found through LLL lattice base reduction is almost zero [26].

Hence, LLL lattice base reduction attack on ASSP [25][27] is utterly ineffectual, which illustrates that even although a DLP with the modulus bit-length less than 1024 can be solved, the original or a reasonable $\underline{b}_1 \ldots \underline{b}_n$ cannot be found yet in DLP subexponential time, namely $\underline{d} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% \ M)$ is computationally one-way.

4.4.2 Compression Algorithm Is Weakly Collision-free

Assume that $b_1 \ldots b_n \neq 0$ is a short message or a message digest from a classical hash function. By Definition 3, we easily understand that $\underline{b}_i = \underline{b}_i 2^{\partial_i} \leq n \ \forall i \in [1, n]$.

Given a short message $b_1 \ldots b_n \neq 0$, and let $b'_1 \ldots b'_n \neq 0$ be another short message to need to be found.

Let $\underline{b}_1 \ldots \underline{b}_n$ be the bit long-shadow string of $b_1 \ldots b_n$, and $\underline{b}'_1 \ldots \underline{b}'_n$ be the bit long-shadow string of $b'_1 \ldots b'_n$.

Let $\hat{lh}$ be the compression algorithm of the new non-MDS hash function described in Section 3.2. Hence, we have

$$\underline{d} = \hat{lh}(b_1 \ldots b_n) = \prod_{i=1}^{n} C_i^{\underline{b}_i} \% M,$$

and

$$\underline{d}' = \hat{lh}(b'_1 \ldots b'_n) = \prod_{i=1}^{n} C_i^{\underline{b}'_i} \% M,$$

where $\underline{b}_i = \underline{b}_i 2^{\partial_i}$ with $\partial_i = b_{i + (-1)^{\lfloor 2(i-1)/n \rfloor}(n/2)}$, and $\underline{b}'_i = \underline{b}'_i 2^{\partial'_i}$ with $\partial'_i = b'_{i + (-1)^{\lfloor 2(i-1)/n \rfloor}(n/2)}$.

If $\underline{d} = \underline{d}'$, there is $\prod_{i=1}^{n} C_i^{\underline{b}_i} \equiv \prod_{i=1}^{n} C_i^{\underline{b}'_i} \ (\% \ M)$.

Observe an extreme case.

Assume that $C_1 = \ldots = C_n = C$.

Owing to the max of $0 \leq \underline{b}_i \leq n$, we define logically

$$\prod_{i=1}^{n} C^{\underline{b}_i} \equiv \prod_{i=1}^{n} C^{(n+1)^{n-i} \underline{b}_i} \ (\% \ M).$$

Under the circumstances, if $\underline{d} = \underline{d}'$, then there is

$$\prod_{i=1}^{n} C^{(n+1)^{n-i} \underline{b}_i} \equiv \prod_{i=1}^{n} C^{(n+1)^{n-i} \underline{b}'_i} \ (\% \ M),$$

namely

$$C^{\sum_{i=1}^{n} (n+1)^{n-i} \underline{b}_i} \equiv C^{\sum_{i=1}^{n} (n+1)^{n-i} \underline{b}'_i} \ (\% \ M).$$

Let $z \equiv \sum_{i=1}^{n} \underline{b}_i (n+1)^{n-i} \ (\% \ \bar{M})$, and $z' \equiv \sum_{i=1}^{n} \underline{b}'_i (n+1)^{n-i} \ (\% \ \bar{M})$.

Correspondingly,

$$C^{z} \equiv C^{z'} \ (\% \ M).$$

We need to solve the above equation for z'.

If the order $\|C\|$ is known, let $z' = z + k\|C\|$, where $k \geq 1$ is an integer. Once a fit k is found, there will be $C^{z} \equiv C^{z'} \ (\% \ M)$, and a bit string can be inferred from $\underline{b}'_1 \ldots \underline{b}'_n$. However, seeking $\|C\|$ is of the integer factorization problem (IFP) at present because the prime factors of $\bar{M}$ must be known.

In practice, C_1, …, C_n that are produced through the algorithm in Section 3.1 are pairwise unequal, which implies that for any given short message $b_1 \ldots b_n$, seeking another short message $b'_1 \ldots b'_n$ such that

$\prod_{i=1}^{n} C_i^{\underline{b}_i} \equiv \prod_{i=1}^{n} C_i^{\underline{b}'_i}$ (% M) is harder than the IFP in computational complexity, namely $b'_1 \ldots b'_n$ for $l\hat{h}(b_1 \ldots b_n) = l\hat{h}(b'_1 \ldots b'_n)$ cannot be found in IFP subexponential time.

Therefore, we say that the new non-MDS hash function is weakly collision-free.

Again because the new hash function is non-MDS, and based on the intractabilities, like the Chaum-Heijst-Pfitzmann hash function, it is resistant to single-block differential attack [28].

4.4.3 Compression Algorithm Is Resistant to Birthday Attack

First, observe an example of whether any two students in a class have the same birthday.

Suppose that the class has 23 students. If a teacher specifies a day (say February 12), then the chance that at least one student is born on that day is $(1 - (364 / 365)^{23}) \approx 6.11$ %. However, the probability that at least one student has the same birthday as any other student is around $(1 - (365 \times \ldots \times 343 / 365^{23})) \approx 50.73$ %, which prompts birthday attack on hash functions.

Birthday attack is widely exploited for finding any two messages $\underline{m}$ and $\underline{m}'$ such that $\hat{h}(\underline{m}) = \hat{h}(\underline{m}')$, namely $(\underline{m}, \underline{m}')$ is a collision, where $\hat{h}$ is a hash function [29]. If the bit-length of a message digest is m, an adversary can find a collision $(\underline{m}, \underline{m}')$ such that $\hat{h}(\underline{m}) = \hat{h}(\underline{m}')$ with probability 50% in roughly $1.1774 \times 2^{m/2}$ time, namely with input of $1.1774 \times 2^{m/2}$ random messages [30].

However, to the new non-MDS hash, a collision is transformed into a mapping.

Theorem 1: The new non-MDS hash function is resistant to birthday attack on the assumption that the MPP and ASPP have only exponential time solutions.

Proof:

Let $b_1 \ldots b_n$ and $b'_1 \ldots b'_n$ be two arbitrary different short messages, and $\underline{b}_1 \ldots \underline{b}_n$ and $\underline{b}'_1 \ldots \underline{b}'_n$ be two related bit long-shadow strings.

Suppose that $\underline{d} = \underline{d}'$, namely $\prod_{i=1}^{n} C_i^{\underline{b}_i} \equiv \prod_{i=1}^{n} C_i^{\underline{b}'_i}$ (% M).

Because the ASPP has only exponential time solutions, we cannot directly solve $\underline{d} \equiv \prod_{i=1}^{n} C_i^{\underline{b}'_i}$ (% M) for $\underline{b}'_1 \ldots \underline{b}'_n$.

Then, there is

$$\prod_{i=1}^{n} (A_i W^{\ell(i)})^{\delta \underline{b}_i} \equiv \prod_{i=1}^{n} (A_i W^{\ell(i)})^{\delta \underline{b}'_i} \ (\% \ M).$$

Further,

$$W^{\underline{k}\delta} \prod_{i=1}^{n} (A_i)^{\delta \underline{b}_i} \equiv W^{\underline{k}'\delta} \prod_{i=1}^{n} (A_i)^{\delta \underline{b}'_i} \ (\% \ M),$$

where $\underline{k} = \sum_{i=1}^{n} \underline{b}_i \ell(i)$, $\underline{k}' = \sum_{i=1}^{n} \underline{b}'_i \ell(i)$ % $\overline{M}$, and $\underline{k} - \underline{k}' < 4n(2\tilde{n} + 3)$.

Raising either side of the above congruence to the δ^{-1}-th power yields

$$W^{\underline{k}} \prod_{i=1}^{n} A_i^{\underline{b}_i} \equiv W^{\underline{k}'} \prod_{i=1}^{n} A_i^{\underline{b}'_i} \ (\% \ M).$$

Without loss of generality, let $\underline{k} \geq \underline{k}'$. Because $(\mathbb{Z}_M^*, \cdot)$ is an Abelian group, we have

$$W^{\underline{k} - \underline{k}'} \equiv \prod_{i=1}^{n} A_i^{\underline{b}'_i} (\prod_{i=1}^{n} A_i^{\underline{b}_i})^{-1} \ (\% \ M).$$

Due to $\overline{M} / 2$ = a prime or the least prime factor of $\overline{M} / 2 > 4n(2\tilde{n} + 3)$, there is

$$W^{2^k} \equiv (\prod_{i=1}^{n} A_i^{\underline{b}'_i - \underline{b}_i})^{((\underline{k} - \underline{k}') / 2^k)^{-1}} \ (\% \ M), \tag{1}$$

where $k \in [0, 46)$ is a small integer, $(\underline{k} - \underline{k}') / 2^k$ is a prime, and $W \in (1, \overline{M})$ as a component of a private key is determinate, which manifests that if $\underline{b}_1 \ldots \underline{b}_n$ and $\underline{b}'_1 \ldots \underline{b}'_n$ satisfy (1), there will be $\underline{d} = \underline{d}'$.

For clear explanation, (1) is written as the form of a function:

$$x^{2^k} \equiv (\prod_{i=1}^{n} A_i^{\underline{b}'_i - \underline{b}_i})^{((\underline{k} - \underline{k}') / 2^k)^{-1}} \ (\% \ M). \tag{2}$$

Since $\overline{M}$ contains only one 2-factor, (2) has only two solutions when $k \neq 0$.

In other words, we may define a mapping from $\{0, 1\}^n \times \{0, 1\}^n$ to $\{1, \ldots, \overline{M}\}$:

$$\Psi(b_1 \ldots b_n, b'_1 \ldots b'_n) \equiv (\prod_{i=1}^{n} A_i^{\underline{b}'_i - \underline{b}_i})^{((\underline{k} - \underline{k}') / 2^k)^{-1}} \ (\% \ M),$$

where $\underline{b}_i = b_i 2^{\partial_i}$, $\underline{b}'_i = b'_i 2^{\partial_i}$, $\underline{k} = \sum_{i=1}^{n} \underline{b}_i \ell(i)$, $\underline{k}' = \sum_{i=1}^{n} \underline{b}'_i \ell(i)$ % $\overline{M}$, $k \in [0, 46)$ is a integer, and $(\underline{k} - \underline{k}') / 2^k$ is a prime.

Therefore, only if $\Psi(b_1 \ldots b_n, b'_1 \ldots b'_n) = W^{2^k}$ with $k \in [0, 46)$, can there exists $\underline{d} = \underline{d}'$. Obviously, $\forall (b_1 \ldots b_n, b'_1 \ldots b'_n) \in \{0, 1\}^n \times \{0, 1\}^n$, the probability that $\Psi(b_1 \ldots b_n, b'_1 \ldots b'_n) = W^{2^k}$ is nearly $1/2^m$.

Further, let $\underline{n}$ be the number of needed inputs $(b_1 \ldots b_n, b'_1 \ldots b'_n)$'s to find at least a $(b_1 \ldots b_n, b'_1 \ldots b'_n)$ such that $\Psi(b_1 \ldots b_n, b'_1 \ldots b'_n) = W^{2^k}$ with probability 50%, which is equivalent to finding any two messages $b_1 \ldots b_n$ and $b'_1 \ldots b'_n$ such that $l\hat{h}(b_1 \ldots b_n) = l\hat{h}(b'_1 \ldots b'_n)$ with probability 50%. Then $\underline{n}$ satisfies $1 - ((2^m - k)/2^m)^{\underline{n}} = 50\%$. Through computation, find that $\underline{n}$ is nearly 2^{m-1} with $k \in [0, 46)$.

The 2^{m-1} is far larger than the threshold $1.1774 \times 2^{m/2}$ for the effective birthday attack. The reason is

that a hidden restriction is imposed on the input ($b_1 \ldots b_n$, $b'_1 \ldots b'_n$), which is easily understood as the number of students of the class needs to be increased for finding with probability 50% any two students who have both the same birthday and the same *gender*.

Additionally, because a private key ($\{A_i\}$, $\{\ell(i)\}$, W, δ) is unknown for the adversary, and the MPP is intractable, it is also infeasible that the adversary finds specific $b_1 \ldots b_n$ and $b'_1 \ldots b'_n$ such that (1) holds by utilizing the private key.

Therefore, the new non-MDS hash can be resistant to the birthday attack, and at present, its security is nearly the $O(2^m)$ magnitude, but not $O(2^{m/2})$. □

4.4.4 Compression Algorithm Is Resistant to Meet-in-the-middle Attack

Meet-in-the-middle dichotomy used for attack on an intended expansion of a block cipher was first developed by Diffie and Hellman in 1977 [31]. Section 3.10 of [9] brings forth a meet-in-the-middle attack algorithm for solving a subset sum problem.

INPUT: a set of positive integers $\{c_1, c_2, \ldots, c_n\}$ and a positive integer s.

S1: Set $t \leftarrow \lfloor n / 2 \rfloor$.

S2: Construct a table with entries $(\sum_{i=1}^{t} c_i b_i, (b_1, b_2, \ldots, b_t))$ for $(b_1, b_2, \ldots, b_t) \in (\mathbb{Z}_2)^t$.
Sort this table by the first component.

S3: For each $(b_{t+1}, b_{t+2}, \ldots, b_n) \in (\mathbb{Z}_2)^{n-t}$, do the following:

S3.1: Compute $r \leftarrow s - \sum_{i=t+1}^{n} c_i b_i$ and check, using a binary search, whether r is the first component of some entry in the table;

S3.2: If $r = \sum_{i=1}^{t} c_i b_i$, then return (a solution is $(b_1, b_2, \ldots, b_n)$).

S4: Return (no solution exists).

OUTPUT: $b_i \in \{0, 1\}$, $1 \le i \le n$, such that $\sum_{i=1}^{n} c_i b_i = s$, provided such b_i exist.

It is not difficult to understand that the time complexity of the above algorithm is $O(n2^{n/2})$.

Let $b_1 \ldots b_n$ be a short message, and its digest be $\underline{d} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% M)$.

If $b_{n/2} = b_n = 1$ (thus, any bit *shadow* on the left of the middle point has no relation with bits on the right), an adversary may attempt to attack the ASPP $\underline{d} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% M)$ by the meet-in-the-middle method.

However, owing to $\underline{b}_i = b_i 2^{\partial_i}$ with $\partial_i = b_{i+(-1)^{\lfloor 2(i-1)/n \rfloor}(n/2)}$ for every $i \in [1, n]$, when i is from 1 to $n / 2$, there exists

$$\underline{b}_1 \ldots \underline{b}_{n/2} = (\underline{b}_1 2^{b_{1+n/2}}) \ldots (\underline{b}_{n/2} 2^{b_n}),$$

which involves all the bits of the short message, namely a reasonable middle point does not exist.

If a fork is selected in proportion to $(n / 3 : 2n / 3)$ or $(n / 4 : 3n / 4)$, the right of the fork substantially still involves all the bits $b_1, \ldots, b_n$.

For instance, let $n = 12$, a short message (a bit string) $= b_1 \ldots b_{12}$, and a fork be to (4 : 8), then

$$\underline{b}_5 \ldots \underline{b}_{12} = (\underline{b}_5 2^{b_{11}})(\underline{b}_6 2^{b_{12}})(\underline{b}_7 2^{b_1})(\underline{b}_8 2^{b_2})(\underline{b}_9 2^{b_3})(\underline{b}_{10} 2^{b_4})(\underline{b}_{11} 2^{b_5})(\underline{b}_{12} 2^{b_6})$$

involves all the bits $b_1, \ldots, b_{12}$.

The above dissection manifests that the meet-in-the-middle attack is essentially ineffectual on the new non-MDS hash function. Therefore, even if $n = m$, namely the input length = the output length of the function, the time complexity of the attack task is still $O(2^m)$ at present, but not $O(m2^{m/2})$.

Besides, unlike $\sum_{i=1}^{n} c_i = \sum_{i=1}^{n} b_i c_i + \sum_{i=1}^{n} \neg b_i c_i$ in the SSP, there is not

$$\prod_{i=1}^{n} C_i = \prod_{i=1}^{n} C_i^{\underline{b}_i} \prod_{i=1}^{n} C_i^{\neg \underline{b}_i} \ (\% M)$$

in the ASPP, where $\neg \underline{b}_i$ is the bit long-shadow of $\neg b_i$, which implies there does not exist an easy relation between the ASPP $\underline{d} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \ (\% M)$ and the dichotomy.

4.4.5 Compression Algorithm Is Resistant to Multi-block Differential Attack

The [32] and [33] show that multi-block near differential attack is effective on the iterative hash functions MD5, SHA-0, SHA-1, and SHA-256 which have multiple block-inputs and the Merkle-Damgård-Iteration structure [7][8].

It is well known that MD5, SHA-0, or SHA-1 will execute a number of rounds of inner iteration for each input block, and each round of the inner iteration consists of linear arithmetics and logic operators such as addition, shift, exclusive or etc.

The input of the new non-MDS hash function is a short message which may be treated as only one block. Its inner iteration consists of at most $2n$ modular multiplications which is nonlinear and intricate,

which indicates that the differential analysis of $g' \equiv \prod_{i=1}^{n} C_i^{b_i} (\% M)$ loses a basis.

Furthermore, in the new non-MDS hash, the inner nonlinear iteration leads to the fierce snowslide effect and strong noninvertibility (see Section 4.4.1), and makes it impossible to derive a set of sufficient conditions which ensure that the collision differential characteristics hold for two short messages which are expected to produce a collision.

Therefore, the new non-MDS hash is substantially distinct from the classical hashes MD5, SHA-0, SHA-1 etc, and the multi-block near differential attack suitable for the classical hashes will be utterly ineffective on the new non-MDS hash function.

4.4.6 Compression Algorithm Is Strongly Collision-free

Firstly, it is known from Section 4.4.2 that the new non-MDS hash function $l\hat{h}$ is weakly collision-free.

Secondly, for any arbitrary short message $b_1 \ldots b_n$, if want to find another short message $b'_1 \ldots b'_n$ such that $l\hat{h}(b_1 \ldots b_n) = l\hat{h}(b'_1 \ldots b'_n)$, adversaries must take $b'_1 \ldots b'_n$ from

$$\prod_{i=1}^{n} C_i^{b_i} \equiv \prod_{i=1}^{n} C_i^{b'_i} (\% M),$$

and further acquire the bit string $b'_1 \ldots b'_n$. It is known from Section 4.4.2 that such a collision problem is computationally harder than IFP at present.

Thirdly, the new non-MDS hash is resistant to classical or efficient attacks in common use — the birthday attack, meet-in-the-middle attack, and multi-block differential attack for example.

Lastly, any subexponential time algorithm for solving the ASPP $g' \equiv \prod_{i=1}^{n} C_i^{b_i} (\% M)$ is not found yet [34], and the most efficient method of solving $g' \equiv \prod_{i=1}^{n} C_i^{b_i} (\% M)$ is brute force attack so far. The analysis manifests that the security of the new non-MDS hash gets the $O(2^m)$ magnitude at present.

In sum, the new hash function is strongly collision-free. Further, we give a related theorem.

Theorem 2: If any arbitrary collision of the new non-MDS hash function can be found in subexponential time, the ASPP $\prod_{i=1}^{n} C_i^{\bar{y}_i} \equiv 1 (\% M)$ can be solved efficiently, where $\bar{y}_i \in [-n, n]$ is the difference of two bit long-shadows at the same position.

Proof:

According to Definition 3，it is easy to understand that for every b_i, there is $0 \le b_i \le n$.

Let $b_1 \ldots b_n \ne b'_1 \ldots b'_n \ne 0$ be two arbitrary bit strings, $b_1 \ldots b_n$ and $b'_1 \ldots b'_n$ be respectively two corresponding bit long-shadow strings.

Again let $\bar{y}_i = b_i - b'_i$, and then there is $\bar{y}_i \in [-n, n]$.

Since the interval $[-n, n]$ is wider than $[0, n]$, similar to $g' \equiv \prod_{i=1}^{n} C_i^{b_i} (\% M)$, the ASPP $\prod_{i=1}^{n} C_i^{\bar{y}_i} \equiv 1 (\% M)$ with $\bar{y}_i \in [-n, n]$ has no subexponential time solution [34], and is only faced with brute force attack.

Assume that $\prod_{i=1}^{n} C_i^{b_i} \equiv \prod_{i=1}^{n} C_i^{b'_i} (\% M)$ is a found collision between two arbitrary bit strings $b_1 \ldots b_n$ and $b'_1 \ldots b'_n$ in subexponential time.

From $\prod_{i=1}^{n} C_i^{b_i} \equiv \prod_{i=1}^{n} C_i^{b'_i} (\% M)$, we have

$$\prod_{i=1}^{n} C_i^{b_i - b'_i} \equiv 1 (\% M).$$

Let $\bar{y}_i \equiv b_i - b'_i \in [-n, n]$, and then

$$\prod_{i=1}^{n} C_i^{\bar{y}_i} \equiv 1 (\% M),$$

which means that the ASPP $\prod_{i=1}^{n} C_i^{\bar{y}_i} \equiv 1 (\% M)$ can be solved efficiently in subexponential time. It is in direct contradiction to the fact.

Therefore, the new non-MDS hash function is strongly collision-free. □

5 Applicability of the New Non-MDS Hash Function

The new non-MDS hash function may be applied in practice, which can be seen from three aspects.

5.1 Running Time of the Compression Algorithm

Suppose that running time is measured in the number of bit operations. Then it is easy to understand that the running time of a modular multiplication is $O(2 \lg^2 M)$ bit operations.

The initialization algorithm in Section 3.1 is one-shot, and not real-time, and thus it is unnecessary to care about its running time.

In what follows, we consider the running time of the compression algorithm in Section 3.2.

Because of $n \le \sum_{i=1}^{n} \underline{b}_i \le 2n$ for a nonzero bit string $b_1 \ldots b_n$, the compression algorithm takes at most $2n$ modular multiplications, namely the running time of the compression algorithm is $O((2n)2\lg^2 M) = O(4nm^2)$ bit operations which is relatively small.

5.2 Comparison with the Chaum-Heijst-Pfitzmann Hash

The Chaum-Heijst-Pfitzmann hash function is provably secure, and defined as follows [16]:

$$\hat{h}: w_1, w_2 \mapsto \hat{h}(w_1, w_2) = \alpha^{w_1}\beta^{w_2} \% p \quad (\{0, ..., q-1\}^2 \to \mathbb{Z}_p - \{0\}),$$

where w_1 and w_2 are the two complementary parts of a short message, p and q $(= (p-1)/2)$ are two big primes, and α and β are two generators of the group $(\mathbb{Z}_p^*, \cdot)$. Hence, the Chaum-Heijst-Pfitzmann function based on the difficulty of the DLP $\beta = \alpha^x \% p$ compresses a short message of $2(\lceil \lg p \rceil - 1)$ bits into a digest of $\lceil \lg p \rceil$ bits.

Let $\lceil \lg p \rceil = 1024$, and then the time complexity of computing $\log_\alpha \beta \% p$ is 2^{80} according to the subexponential time $L_p[1/3, 1.923]$ [9], which means that the security of the Chaum-Heijst-Pfitzmann hash is the 2^{80} magnitude when $\lceil \lg p \rceil = 1024$.

Let $\lceil \lg M \rceil = 80$, and then the time complexity of solving the ASPP $\underline{d} = \prod_{i=1}^{n} C_i^{\underline{b}_i} \% M$ for $\underline{b}_1, \ldots, \underline{b}_n$ is also 2^{80} since the ASPP only has an exponential time solution at present [34], which means that the security of the new non-MDS hash is also the 2^{80} magnitude when $\lceil \lg M \rceil = 80$. Besides, let the bit-length $n = 2046$ of a short message $(w_1, w_2) = (b_1 \ldots b_{1023}, b_{1024} \ldots b_{2046}) = b_1 \ldots b_n \ne 0$.

Under the same security, may draw a comparison between the new non-MDS hash and the Chaum-Heijst-Pfitzmann hash.

① *Running time of the compression algorithm*

The Chaum-Heijst-Pfitzmann hash: $2(4\lceil \lg p \rceil^3) = 2(4(1024)^3) = 8\ 589\ 934\ 592$ bit operations.

The new non-MDS hash: $4nm^2 = 4(2048)80^2 = 52\ 428\ 800$ bit operations.

② *Compression rate*

The Chaum-Heijst-Pfitzmann hash: $1024 / 2046 \approx 50.05\%$.

The new non-MDS hash: $80 / 2046 \approx 3.91\%$.

③ *Resisting birthday attack*

The number of inputs (w_1, w_2)'s needed by birthday attack on $\hat{h}(w_1, w_2) \equiv \alpha^{w_1}\beta^{w_2} (\% p)$ is about $2^{\lceil \lg p \rceil / 2} = 2^{512}$, larger than 2^{80} which is the security magnitude of the DLP $\beta = \alpha^x \% p$, and thus the Chaum-Heijst-Pfitzmann hash function cannot resist the birthday attack.

The number of inputs $b_1 \ldots b_n$'s needed by birthday attack on $l\hat{h}(b_1 \ldots b_n) = \prod_{i=1}^{n} C_i^{\underline{b}_i} \% M$ is about $2^{\lceil \lg M \rceil / 2} = 2^{40}$, smaller than 2^{80} which is the security magnitude of the ASPP $\underline{d} = \prod_{i=1}^{n} C_i^{\underline{b}_i} \% M$, and thus the new non-MDS hash function can resist the birthday attack.

④ *Provable security*

On the assumption that the DLP has a subexponential time solution, the Chaum-Heijst-Pfitzmann hash function is proved to be strongly collision-free in subexponential time.

Likewise, on the assumption that the ASPP has an exponential time solution, the new non-MDS hash function is also proved to be strongly collision-free in exponential time.

In summary, the new non-MDS hash has some advantages over the Chaum-Heijst-Pfitzmann one, and relatively the former may be regarded as lightweight.

5.3 Reformation of a Classical Hash Function

Because the new non-MDS hash function is resistant to birthday attack and meet-in-the-middle attack, a classical hash function of which the output is m bits, and the security is intended to be the $O(2^{m/2})$ magnitude may be reformed into a compact hash function of which the output is $m/2$ bits, and the security is still equivalent to the $O(2^{m/2})$ magnitude [35].

For example, let $b_1 \ldots b_{128}$ be the output of MD5 [36], $\underline{b}_1 \ldots \underline{b}_{128}$ be its bit long-shadow string, and $\lceil \lg M \rceil = 64$. Then, regard $\underline{d} = \prod_{i=1}^{128} C_i^{\underline{b}_i} \% M$ as the 64-bit output of the reformed MD5 with the equivalent security, where $C_i = (A_i W^{\ell(i)})^{\delta} \% M$ which is produced by the algorithm in Section 3.1.

Again for example, let $b_1 \ldots b_{160}$ be the output of SHA-1, $\underline{b}_1 \ldots \underline{b}_{160}$ be its bit long-shadow string, and $\lceil \lg M \rceil = 80$. Then, regard $\underline{d} = \prod_{i=1}^{160} C_i^{\underline{b}_i} \% M$ as the 80-bit output of the reformed SHA-1 with the equivalent security.

The above two examples indicate that we may exchange time for space when the related security remains unchanged.

6 Conclusion

In the paper, the authors propose a new non-MDS hash function which contains the initialization algorithm and the compression algorithm, and converts a short message or a message digest of n bits into a string of m bits, where $80 \le m \le 232$ and $80 \le m \le n \le 4096$.

The authors prove that both the MPP and the ASPP are computationally at least equivalent to the DLP in the same prime field, and analyze the security of the new non-MDS hash function. The analysis shows that the new non-MDS hash is computationally one-way, weakly collision-free, and strongly collision-free. Moreover, at present, any subexponential time algorithm for attacking the new non-MDS hash is not found, and its security gets be the $O(2^m)$ magnitude.

Especially, the analysis illustrates that the new non-MDS hash function is resistant to birthday attack and meet-in-the-middle attack. By utilizing this characteristic, one can reform a classical hash function with an m-bit output and an $O(2^{m/2})$ magnitude security into a compact hash function with an $m / 2$ bit output and the equivalent security.

Simultaneously, the authors dissect the running time of compression algorithm of the new non-MDS hash function, and it is $O(n m^2)$ bit operations.

The new non-MDS hash function opens a door to convenience for the utilization of a lightweight digital signing scheme of which the modulus length is not greater than 160 bits.

Acknowledgment

The authors would like to thank the Academicians Jiren Cai, Zhongyi Zhou, Jianhua Zheng, Changxiang Shen, Zhengyao Wei, Binxing Fang, Guangnan Ni, Andrew C. Yao, and Xicheng Lu for their important guidance, suggestions, and helps.

The authors also would like to thank the Professors Dingyi Pei, Dengguo Feng, Jie Wang, Ronald L. Rivest, Moti Yung, Adi Shamir, Dingzhu Du, Mulan Liu, Huanguo Zhang, Yixian Yang, Maozhi Xu, Xuejia Lai, Yongfei Han, Yupu Hu, Dongdai Lin, Chuankun Wu, Rongquan Feng, Ping Luo, Jianfeng Ma, Zhenfu Cao, Lusheng Chen, Chao Li, Wenbao Han, Bogang Lin, Qibin Zhai, Hong Zhu, Renji Tao, Bingru Yang, Zhiying Wang, Quanyuan Wu, and Zhichang Qi for their important advice, suggestions, and corrections.

References

[1] I. F. Blake, G. Seroussi, and N. P. Smart, *Elliptic Curves in Cryptography*, Cambridge University Press, Cambridge, UK, 1999, ch. 3-5.

[2] T. ElGamal, A Public-key Cryptosystem and a Signature Scheme Based on Discrete Logarithms, *IEEE Transactions on Information Theory*, vol. 31(4), 1985, pp. 469-472.

[3] S. Su and S. Lü, A Public Key Cryptosystem Based on Three New Provable Problems, *Theoretical Computer Science*, vol. 426-427, Apr. 2012, pp. 91-117.

[4] D. C. Ranasinghe, Lightweight Cryptography for Low Cost RFID, *Networked RFID Systems and Lightweight Cryptography*, Springer-Verlag, 2007, pp. 311-346.

[5] H.-Y. Chien, SASI: A new ultralightweight rfid authentication protocol providing strong authentication and strong integrity, *IEEE Transactions on Dependable and Secure Computing*, vol. 4(4), 2007, pp. 337-340.

[6] A. Shamir, SQUASH - A New MAC with Provable Security Properties for Highly Constrained Devices Such as RFID Tags, *Proc. of FSE' 08*, 2008.

[7] R. Merkle, One way hash functions and DES, *Proc. of Advances in Cryptology: CRYPTO 89*, Springer-Verlag, 1989, pp. 428-446.

[8] I. Damgard, A design principle for hash functions, *Proc. of Advances in Cryptology: CRYPTO 89*, Springer-Verlag, 1989, pp. 416-427.

[9] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, London, UK, 1997, ch. 2, 3, 5.

[10] W. Stallings, *Cryptography and Network Security: Principles and Practice* (2nd ed.), Prentice-Hall, New Jersey, 1999, ch. 8, 9.

[11] B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C* (2nd ed.), John Wiley & Sons, New York, 1996, ch. 18.

[12] S. Y. Yan, *Number Theory for Computing* (2nd ed.), Springer-Verlag, New York, 2002, ch. 1.

[13] T. W. Hungerford, *Algebra*, Springer-Verlag, New York, 1998, ch. 1-3.

[14] K. H. Rosen, *Elementary Number Theory and Its Applications* (5th ed.), Addison-Wesley, Boston, 2005, ch. 12.

[15] M.J. Wiener, *Cryptanalysis of Short RSA Secret Exponents*, IEEE Transactions on Information Theory, vol. 36(3), 1990, pp. 553-558.

[16] D. Chaum, E. Van Heijst, and B. Pfitzmann, Cryptographically strong undeniable signatures, unconditionally secure for the signer, *Proc. of Advances in Cryptology: CRYPTO '91* (LNCS 576), Springer-Verlag, 1992, pp. 470-484.

[17] D. Z. Du and K. Ko, *Theory of Computational Complexity*, John Wiley & Sons, New York, 2000, ch. 3-4.

[18] B. Schröder, *Ordered Sets: An Introduction*, Birkhäuser, Boston, 2003, ch. 3-4.

[19] M. Davis, *The Undecidable: Basic Papers on Undecidable Propositions, Unsolvable Problems and Computable Functions*, Dover Publications, Mineola, 2004, ch. 2-4.
[20] R. C. Merkle and M. E. Hellman, Hiding information and Signatures in Trapdoor Knapsacks, *IEEE Transactions on Information Theory*, vol. 24(5), 1978, pp. 525-530.
[21] A. Shamir, A Polynomial Time Algorithm for Breaking the Basic Merkle-Hellman Cryptosystem, *Proc. of the 23th IEEE Symposium on the Foundations of Computer Science*, IEEE, 1982, pp. 145-152.
[22] D. Naccache and J. Stern, A new public key cryptosystem, *Proc. of Advances in Cryptology: EUROCRYPT '97*, Springer-Verlag, 1997, pp. 27-36.
[23] S. Su, S. Lü, and X. Fan, Asymptotic Granularity Reduction and Its Application, *Theoretical Computer Science*, vol. 412(39), Sep. 2011, pp. 5374-5386.
[24] O. Goldreich, *Foundations of Cryptography: Basic Tools*, Cambridge University Press, Cambridge, UK, 2001, ch. 1-2.
[25] E. F. Brickell, Solving Low Density Knapsacks, *Proc. of Advance in Cryptology: CRYPTO '83*, Plenum Press, New York, 1984, pp. 25-37.
[26] T. Li and S. Su, Analysis of Success Rate of Attacking Knapsacks from JUNA Cryptosystem by LLL Lattice Basis Reduction, *Proc. of 2013 Int. Conf. on Comput. Intelligence and Security*, IEEE Computer, Dec. 2013, pp. 454-458.
[27] M. J. Coster, A. Joux, B. A. LaMacchia etc, Improved Low-Density Subset Sum Algorithms, *Computational Complexity*, vol. 2(2), 1992, pp. 111-128.
[28] T. Xie and D. Feng, Construct MD5 Collisions Using Just A Single Block Of Message, *Cryptology ePrint Archive*, http://eprint.iacr.org/2010/643, Dec. 2010.
[29] M. Bellare and T. Kohno, Hash Function Balance and Its Impact on Birthday Attacks, *Proc. of Advances in Cryptology: EUROCRYPT '04*, Springer-Verlag, 2004, pp. 401-418.
[30] M. Girault, R. Cohen, and M. Campana, A Generalized Birthday Attack, *Proc. of Advances in Cryptology: EUROCRYPT '88* (LNCS 330), Springer-Verlag, 1988, pp. 129-156.
[31] W. Diffie and M. E. Hellman, Exhaustive Cryptanalysis of the NBS Data Encryption Standard, *Computer*, vol. 10 (6), 1977, pp. 74-84.
[32] E. Biham, R. Chen, A. Joux etc, Collisions of SHA-0 and Reduced SHA-1, *Proc. of Advances in Cryptology: EUROCRYPT '05*, Springer-Verlag, 2005, pp. 36-57.
[33] X. Wang, Y. L. Yin, and H. Yu, Finding collisions in the full SHA-1, *Proc. of Advances in Cryptology: CRYPTO '05*, Springer-Verlag, 2005, pp. 17-36.
[34] S. Su and S. Lü, REESSE1+ · Reward · Proof by Experiment · A New Approach to Proof of P != NP, *Cornell University Library*, http://arxiv.org/pdf/0908.0482, Aug. 2009 (revised Aug. 2014).
[35] M. Bellare and D. Micciancio, A New Paradigm for Collision-free Hashing: Incrementality at Reduced Cost, *Proc.of Advances in Cryptology: EUROCRYPT '97*, Springer-Verlag, 1997, pp. 163-192.
[36] R. L. Rivest, The MD5 Message Digest Algorithm, *RFC 1321*, Apr. 1992.