Security Analysis of Even-Mansour Structure Hash Functions

Shiwei Chen(✉), Ting Cui, and Chenhui Jin

The PLA SSF Information Engineering University, Zhengzhou, China

Abstract. In this paper, we mainly focus on the security of Even-Mansour structure hash functions, including preimage attack resistance and multi-block collision attack resistance.

Firstly, we focus on the Even-Mansour structure hash function with two iterations. Basing on the permutation used in the Even-Mansour structure hash function we construct two new functions f_1 and f_2, and find the partial invariables of input-output in one function f_1. Then using the partial invariables of input-output and the meet-in-the-middle techniques, we present a preimage attack on the Even-Mansour structure hash function with two iterations, with the time complexity of $2^{a(2^a-1)}+2^a+2^{n-2a}$ functional operations of f_1 or f_2 and the memory is 2^a a-bit values, where $a2^a \leq n$ and n is the size of hash value.

Secondly, we extend the Even-Mansour structure hash function to the one with arbitrary iterations. Utilizing the property that the beginning and the ending of every iteration in the Even-Mansour structure both need XOR the message or the transform result of the message, we construct many chaining values with relations in each iteration, which makes that the number of the final chaining values is equal to the product of the number of output chaining values in each iteration, and thereby propose our multi-block collision attack on the Even-Mansour structure hash functions with the time complexity of $t2^{\frac{s}{2t}}$ queries of F permutation and memory complexity of $O(2^{s/2})$, where t is the block number of collision message and s is the size of truncated hash value.

Keywords: Hash function · Even-Mansour structure hash function · Preimage attack · Multi-block collision attack · Partial invariables of input-output · Meet-in-the-middle technique

1 Introduction

Hash functions are an important class of primitive in modern cryptography, mainly used in many cryptographic protocols, message authentication, etc. A hash function H transfers a message M with arbitrary length into a fixed-length message digest h *called hash value.* If the length of the hash value is n bits, we call the hash function a *n-bit hash function.* To guarantee the security of the applications, a hash function H needs to satisfy the following three basic security principles, that is, preimage resistance, second preimage resistance and collision resistance. For a n-bit hash function, if the computational complexity of finding a second preimage or a preimage is less than 2^n,

D. Gao et al. (Eds.): ICICS 2021, LNCS 12919, pp. 163–173, 2021.
https://doi.org/10.1007/978-3-030-88052-1_10

then the hash function is considered not to be second preimage resistance or preimage resistance, and if the computational complexity of finding a collision is less than $2^{n/2}$, then the hash function is considered not to be collision resistance.

In one hash function, the inputs include one message and one fixed initial chaining value, and the output is the fixed-length hash value. Correspondingly, in one block cipher, its inputs are one plaintext and one key, and output is the ciphertext. Since these two structures are similar, cryptographers usually regard the key in the block cipher as the message in the hash function, and thereby construct hash functions based on block ciphers. Preneel [1] proposed 11 kinds of methods of constructing hash functions based on block cipher, which include Davies-Meyer (DM) construction used in the MD family hash functions.

Furthermore, there is one important construction in block cipher, that is, Even-Mansour (EM for short) structure. The EM structure was proposed by Even and Mansour [2] in 1997, which allows to construct block cipher from a permutation F. In this construction, the plaintext firstly XORs the key K_1, then bypass F, and finally XORs the key K_2, i.e., the ciphertext is computed by $c = F(p \oplus K_1) \oplus K_2$. In 2012, Dunkelaman and Shamir [3] proved that the EM construction remains the same security level even if $K_1 = K_2$. Meanwhile, Bogdanov et al. [4] generalized the EM structure into ones with more than one rounds, where different permutations are utilized in separated round, and they pointed out the security bound of the distinguishing attack. In 2017, Isobe [5] used the meet-in-the-middle technique to present the key recovery attack on the two variants, one of which is $E_K^{(2)}(x)$, that is,

$$E_K^{(2)}(x) = P_2(P_1(x \oplus K) \oplus \pi(K)) \oplus K$$

where P_1, P_2, P are n-bit permutation, K is n-bit key and π is an simple key schedule. In 2019, Leurent and Sibleyras [6] presented Low-Memory attacks against two-round Even-Mansour using the 3-XOR problem.

Though the Even-Mansour block cipher and hash function have the similar structure, the messages in Even-Mansour structure hash function could be chosen and different message blocks are used in different iterations. In 2013, Yiyuan Luo and Xuejia Lai [7] proposed the EM structure hash functions, and proposed one two-block collision attack with the time complexity of $2^{s/4+1}$(s is the size of hash value). In 2015, the Kupyna hash function was approved as the new Ukrainian standard DSTU 7564:2014 [12], which uses the Davies-Meyer compression function based on the Even-Mansour scheme. Furthermore, in the design of some new hash functions, the compression functions are based on permutations, such as SHA-3 [8], light-weight hash function PHOTON [9], JH hash function [10], etc. In this paper, we will research on the security of the EM hash function, including preimage attack resistance and multi-block collision attack resistance.

Our Contributions. For the Even-Mansour structure hash function with two iterations, we firstly construct two new functions f_1 and f_2 based on the permutation F, and then find the partial invariables of input-output in one function f_1. Then, using the partial invariables of input-output and the meet-in-the-middle techniques, we present a preimage attack on Even-Mansour structure hash functions with two iterations, and analyse the computational complexity of our preimage attack.

Then, for the EM structure hash functions with more iterations, we utilize the property that the beginning and the end of every iteration in Even-Mansour structure both need XOR the message or the transform result of message, to construct many chaining values with relations in each iteration, which makes that the number of the final chaining values is equal to the product of the number of output chaining values in each iteration, and thereby proposes our multi-block collision attack on the Even-Mansour structure hash functions and analyse the computational complexity of our multi-block collision attack.

Outline. The remainder of this paper is organized as follows. In Sect. 2, we describe the EM structure hash functions and some notations. In Sect. 3, we present our preimage attack on Even-Mansour structure hash functions with two iterations and analyse the computational complexity. And then we propose our Multi-block collision attack on Even-Mansour structure hash function with arbitrary iterations and analyse the computational complexity in Sect. 4. In Sect. 5, we conclude our work and propose the future work.

2 Preliminaries

2.1 Description of the Even-Mansour Structure Hash Functions

Even-Mansour structure was proposed by Even and Mansour [2] in 1997 to construct a scheme for a block cipher, which uses a fixed n-bit permutation. The n-bit plaintext is firstly XORed with n-bit K_1, and the result is the input of the permutation. The output of the permutation is XORed with n-bit K_2, and the result is the ciphertext. In [3], Dunkelman and Shamir showed that the original two-key Even-Mansour structure is not minimal since it can be simplified into a single key structure with $K_1 = K_2 = K$.

Since the structure of a hash function is quite similar to that of a block cipher, we can learn the designing of hash functions from mature designed block ciphers. In 2017, Isobe et al. [5] presented one variants of the two-round Even-Mansour structure block cipher, that is,

$$E_K^{(2)}(x) = P_2(P_1(x \oplus K) \oplus K) \oplus K$$

If the K and x are replaced by message and chaining value respectively, then we obtain the EM structure hash function with two iterations.

Let F be a n-bit random permutation, m_1, m_2 be two n-bit input messages, and $h_i (1 \leq i \leq 2)$ be the output chaining value after the i^{th} iteration, $h_0 = IV$ be a fixed constant, n be the size of the hash value. Then the Even-Mansour structure hash function with two iterations $EM_n^{(2)}(IV, m_1, m_2)$ is described as follows:

Step1. Set $h_0 = IV$. For i from 1 to 2, compute

$$h_i = F(h_{i-1} \oplus m_i) \oplus m_i$$

Step2. Output the h_2 as the hash value.

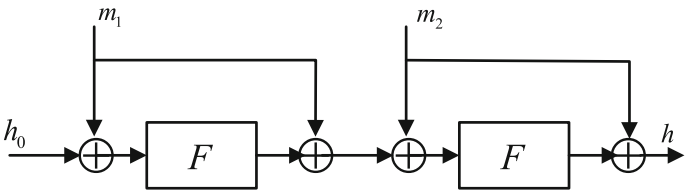

Fig. 1. The workflow of Even-Mansour structure hash function with two iterations

2.2 Notations

In this paper, we mainly focus on the security of the Even-Mansour structure hash function with s-bit hash value, t block n-bit input message $m_1, m_2, \cdots, m_t$ and initial chaining value IV, which is noted as $EM_s^{(t)}(IV, m_1, m_2, \cdots, m_t)$. If not specified, the notations used in this paper is described as follows:

F: the n-bit permutation used in EM structure hash function;

m_i: the n-bit message used in the i^{th} iteration;

$Tur_a(x)$: the most a bits of the variable x;

$x||y$: the concatenation of x and y, that is, for $x \in \{0,1\}^a, y \in \{0,1\}^{n-a}$, $x||y = 2^{n-a}x \oplus y$;

x_a, x_{n-a}: the most a bits and the least $(n\text{-}a)$ bits of variable x respectively, that is $x = x_a||x_{n-a}$;

u_i, v_i: the input and output respectively of the permutation F in the i-th iteration;

3 Our Preimage Attack on Even-Mansour Structure Hash Functions

In [5], Isobe et al. proposed the key recovery attack on the $E_K^{(2)}(x)$, in which the single key is used, so they need to guarantee the consistency of the two rounds. However, in the EM structure hash functions, different message blocks are used in different rounds and the message block could be chosen, which increases the degree of freedom for searching. In this section, utilizing the permutation F, we construct two new functions f_1 and f_2, and then outline the method to find the partial invariables of input-output in the f_1 and f_2 functions. Using the partial invariables of input-output in f_1 and the meet-in-the-middle techniques, we present a new preimage attack on the EM structure hash functions with two iterations.

3.1 Construction of the Partial Invariables of Input-Output in the Functions

Let $f : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^n$ be a function from $M \times X$ to Y. Then the partial invariables and target partial invariables of inputs and outputs in f are defined as follows.

Definition 1 [11] **.** Let $x \in X, y \in Y$. If for any $m \in M$, there is $f(m,x) = y$. Then (x, y) is called *the invariable pair of input-output*. If there exists one $y' \in \{0,1\}^a (a < n)$ such that for any $m \in M$, we have $Tur_a(f(m,x)) = y'$, then (x, y') is called *the partial invariable pair of a-bit input-output*. If the y' of (x, y') is fixed, then (x, y') is called *target invariable pair of a-bit input-output*.

Let $f : \{0,1\}^a \times \{0,1\}^n \to \{0,1\}^n$ from $M' \times X$ to Y. Then finding one partial invariable pair of a-bit input-output according to the following procedure:

Step1. Randomly choose one $x \in \{0,1\}^n$;

Step2. Choose one $m_a \in \{0,1\}^a$, compute $y_1 = f(m_a, x)$, and store $y' = Tur_a(y_1)$;

Step3. Choose another new $m'_a \in \{0,1\}^a$, such that $m'_a \neq m_a$, and compute

$$y_2 = f(m'_a, x);$$

Step4. Check whether $y' = Tur_a(y_2)$ or not. If yes, then return to Step 3. If all m'_a in $\{0,1\}^a$ are passed, then output (x, y'); or else, return to step1.

Then we analyse the time complexity of finding one partial invariable pair of a-bit input-output (x, y')。

In Step 4, the probability of $y' = Tr_a(y_2)$ is 2^{-a}, so the probability of $2^a - 1$ m'_a all passing is $(2^{-a})^{2^a-1}$. Therefore the time complexity of finding one partial invariable pair of a-bit input-output (x, y') is $2^{a(2^a-1)}$ functional operations of f.

3.2 Our Preimage Attack on Even-Mansour Structure Hash Functions

In every compression function of EM structure hash function, the message block is different. Each message block is firstly XORed to the chaining variable, and then is XORed to the result of the random n-bit permutation F. The output is the input chaining variable of the next compression function. So, to obtain the preimage of EM structure hash function with two iterations, the key point is finding one two-block message, which could keep consistency in the two-round compression function.

3.2.1 Our Preimage Attack on Even-Mansour Structure Hash Functions with Two Iterations

Let m_{1a} and $m_{1(n-a)}$ be respectively the most a bits and the least $n-a$ bits of m_1, and $m_1 \oplus IV$ be the XOR of the first message block m_1 and the initial chaining variable IV. Then we have

$$m_1 \oplus IV = (m_{1a} \oplus IV_a)||(m_{1(n-a)} \oplus IV_{n-a})$$

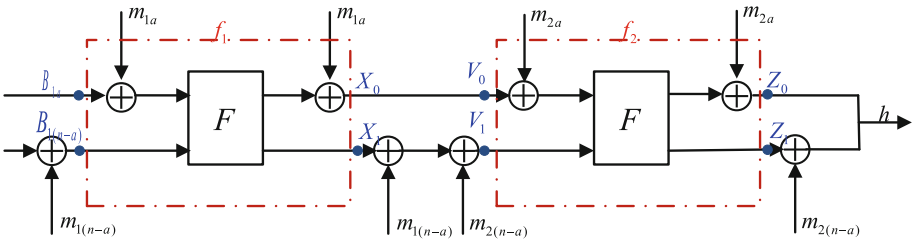

Fig. 2. The workflow of our preimage attack

The function f_1 is constructed as described in Fig. 2.

Let $B_1 = B_{1a}||B_{1(n-a)} = IV_a||(IV_{n-a} \oplus m_{1(n-a)})$ be the input chaining variable of f_1. Then we have

$$f_1(m_{1a}, B_1) = F(B_1 \oplus (m_{1a}||0_{n-a})) \oplus (m_{1a}||0_{n-a}) \stackrel{\Delta}{=} X_0||X_1$$

where $X_0 \in \{0,1\}^a, X_1 \in \{0,1\}^{n-a}$. From the above expression, we know that it only depends on the message m_{1a}, which is helpful to find the partial invariable pair of input-output in the function f_1.

Let $V = V_0||V_1$ be the input chaining variable of the function f_2 as described in Fig. 3. Then we have.

$$V_0 = X_0,\ f_2(m_{2a}, V) = F(V \oplus (m_{2a}||0_{n-a})) \oplus (m_{2a}||0_{n-a}) \stackrel{\Delta}{=} Z_0||Z_1.$$

Next we firstly seek for the partial input-output fixed-point of the function f_1, and use the different values of the remaining output of f_1 to obtain more values to be chosen. Then compute backward from the given hash value and utilizing the meet-in-the-middle technique and the property of the EM structure hash function, we obtain one preimage M of the given hash value h *for* $EM_n^{(2)}(IV, m_1, m_2)$. *The whole process is divided into two phases, that is, online phase and offline phase.*

Offline Phase:

Step1 Utilizing the algorithm described in Sect. 3.1, we could obtain one pair of partial *invariable pair* input-output fixed-point (B_1, X_0) of the function f_1, that is, for any $m_{1a} \in \{0,1\}^a$, the pair (B_1, X_0) satisfies:

$$Tur_a(f_1(m_{1a}, B_1)) = X_0;$$

Step2 For all the $m_{1a} \in \{0,1\}^a$, compute the remaining bits of the function f_1, that is,

$$f_1(m_{1a}^{(i)}, B_1) = X_0||X_1^{(i)}, i = 0, 1, \cdots, 2^a - 1$$

Compute $m_{1(n-a)} = IV_{n-a} \oplus B_{1(n-a)}$ and then $m_{1(n-a)} \oplus X_1^{(i)}$. Store $(m_{1a}^{(i)}, m_{1(n-a)} \oplus X_1^{(i)})$, $i = 0, 1, \cdots, 2^a - 1$ in table T_1.

Online Phase:

Step1. Randomly choose one $V_1 \in \{0,1\}^{n-a}$, and search for one $m_{2a} \in \{0,1\}^a$, such that

$$Tr_a(f_2(m_{2a}, V_0||V_1)) = h_a = Z_0$$

where $V_0 = X_0$;

Step2. For the m_{2a} obtained in Step1, compute the remaining bits of the function f_2, that is, $f_2(m_{2a}, V_0||V_1) = h_a||Z_1$;

Step3. Compute $m_{2(n-a)} = Z_1 \oplus h_{n-a}$, and judge whether $m_{2(n-a)} \oplus V_1$ is equal to one element $m_{1(n-a)} \oplus X_1^{(i)}$ in table T_1 or not. If yes, output m_{2a}, $m_{2(n-a)}$, $m_{1(n-a)}$ and $m_{1a}^{(i)}$ corresponding to $m_{1(n-a)} \oplus X_1^{(i)}$ in table T_1; Or else, return to Step1.

3.2.2 The Complexity of Our Preimage Attack

In this section, we analyze the time complexity and memory of our preimage attack proposed in Sect. 3.2.1.

The Complexity of the Offline Phase. In Step1, according to the algorithm described in Sect. 3.1, we know that the time complexity of finding one pair of partial a-bit input-output fixed-point (B_1, X_0) of the function f_1 is $2^{a(2^a-1)}$ functional operations of f_1. Since $B_{1a} = IV_a$ is fixed, there are only 2^{n-a} values of B_1 to choose. Hence, the condition $2^{a(2^a-1)} \leq 2^{n-a}$ needs to be satisfied, that is $a2^a \leq n$; In Step2, the time complexity of computing the values of f_1 for all the $m_{1a} \in \{0,1\}^a$ is 2^a functional operations of f_1. Hence, the time complexity of the offline phase is $2^{a(2^a-1)}+2^a$ functional operations of f_1 and the memory is $2^a a$-bit values.

The Complexity of the Online Phase. In Step1, the time complexity of searching for $m_{2a} \in \{0,1\}^a$ such that $Tr_a(f_2(m_{2a}, V_0||V_1)) = h_a = Z_0$ is 2^a functional operations of f_2; In Step2, we need to compute the remaining bits of the function f_2 for one m_{2a}, so the time complexity is one functional operations of f_2, which can be ignored; In Step3, since there are 2^a different values of $m_{1(n-a)} \oplus X_1^{(i)}$ in table T_1, the probability that $m_{2(n-a)} \oplus V_1$ is equal to one element $m_{1(n-a)} \oplus X_1^{(i)}$ in table T_1 *is* $2^{-(n-2a)}$, and, so we need to process the Step1–3 2^{n-2a} times to obtain one pair of $(V_1, X_1^{(i)})$ satisfying $m_{2(n-a)} \oplus V_1 = m_{1(n-a)} \oplus X_1^{(i)}$. Hence, the time complexity is about 2^{n-2a} functional operations of f_2.

In a word, the time complexity of our preimage attack is $2^{a(2^a-1)}+2^a + 2^{n-2a}$ functional operations of f_1 or f_2 and the memory is $2^a a$-bit values, where $a2^a \leq n$.

4 Multi-block Collision Attack on Even-Mansour Structure Hash Function

In this section, we extend the EM structure hash function with two iterations to the one with arbitrary iterations, that is $EM_s^{(t)}(IV, m_1, m_2, \cdots, m_t)(t \geq 2)$.

Let F be a n-bit random permutation, $m_1, m_2, \cdots, m_t (t \geq 2)$ be n-bit input messages, and $h_i (1 \leq i \leq t)$ be the output chaining value after the i^{th} iteration, $h_0 = IV$ be a fixed constant, L_1 and L_2 be two transformations from $\{0,1\}^n$ to $\{0,1\}^n$, $s (s \leq n)$ be the size of the hash value. Then the Even-Mansour structure hash function $EM_s^{(t)}(IV, M)$ is described as follows (See Fig. 3):

Step1. Set $h_0 = IV$. For i from 1 to t, compute

$$h_i = F(h_{i-1} \oplus m_i) \oplus L_1(m_i) \oplus L_2(h_{i-1})$$

Step2. Output the most s bits of h_t as the hash value.

In 2013, Yiyuan Luo and Xuejia Lai [7] proposed two-block collision attack on the Even-Mausour structure hash functions, with the time complexity of $2^{s/4+1}$ of F, and they did not analyse the memory. Next we present one multi-block collision attack on the Even-Mausour structure hash function, and analyse the time complexity and memory.

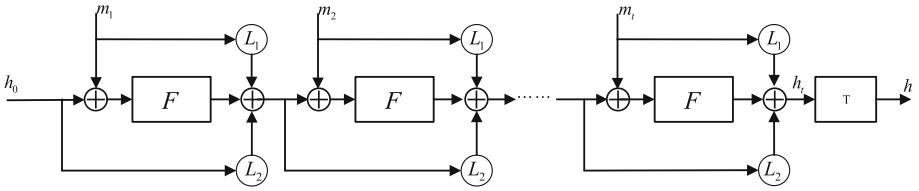

Fig. 3. The workflow of EM structure hash function with t-iteration

In every iteration of the Even-Mausour structure hash function, the input of the permutation F is related to the message, and the output of the permutation F is Xored into the message, which is the input of the second iteration. Hence, we could firstly have access to the permutation F, and use the result to compute the messages and the output chaining values. For each chaining value, use the same way to compute the message and the output chaining values in the next iteration. In this section, we present our multi-block collision attack on $EM_s^{(t)}(IV, M)$, which is an Even-Mansour structure hash function with s-bit hash value. We are to find two different t-block ($t \geq 2$) message $M = m_1||m_2||\cdots||m_t$ and $M' = m'_1||m'_2||\cdots||m'_t$, such that

$$EM_s^{(t)}(IV, M) = EM_s^{(t)}(IV, M')$$

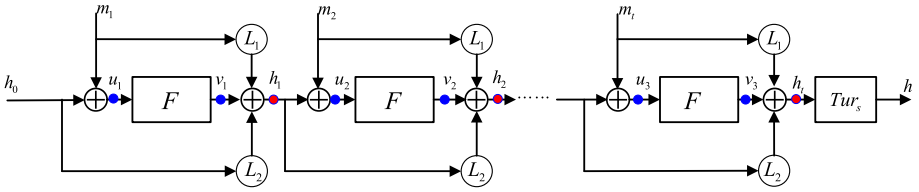

Fig. 4. The workflow of our multi-block collision attack

4.1 Our Multi-block Collision Attack on the Even-Mansour Structure Hash Function

Let F be one random permutation, $(m_1, m_2, \cdots, m_t)(t \geq 2)$ be t-block input message, h_i be n-bit chaining variable, and $h_0 = IV$, $(u_i, v_i)(1 \leq i \leq t)$ be the pair of input-output of the F in the *i-th* iteration. Then our multi-block collision attack on the Even-Mansour structure hash function is described as follows:

Step1. In the first iteration, randomly choose r_1 different values $u_1^{(1)}, u_1^{(2)}, \cdots, u_1^{(r_1)}$, and respectively visit the permutation F, and thereby obtain r_1 pairs of input-output $(u_1^{(1)}, v_1^{(1)}), (u_1^{(2)}, v_1^{(2)}), \cdots, (u_1^{(r_1)}, v_1^{(r_1)})$。Since $h_1 = v_1 \oplus L_1(m_1) \oplus L_2(IV)$, we could obtain r_1 random values of h_1。Store the r_1 quads $(\Phi, u_1^{(j)}, v_1^{(j)}, h_1^{(j)})(1 \leq j \leq r_1)$;

Step2. In the second iteration, randomly choose r_2 different values $u_2^{(1)}, u_2^{(2)}, \cdots, u_2^{(r_2)}$, and respectively visit the permutation F, and thereby obtain r_2 pairs of input-output $(u_2^{(1)}, v_2^{(1)}), (u_2^{(2)}, v_2^{(2)}), \cdots, (u_2^{(r_2)}, v_2^{(r_2)})$;

Step3. For every $(u_2^{(j)}, v_2^{(j)})(1 \le j \le r_2)$, compute $m_2 = u_2^{(j)} \oplus h_1$ and $h_2 = v_2^{(j)} \oplus L_1(m_2) \oplus L_2(h_1)$.

Since we obtain r_1 random values of h_1 in Step1, for each $(u_2^{(j)}, v_2^{(j)})(1 \le j \le r_2)$, we could obtain r_1 random (m_2, h_2)。And since we have r_2 $(u_2^{(j)}, v_2^{(j)})$ in Step2, we could obtain $r_1 r_2$ random (m_2, h_2)。Store $r_1 r_2$ quads

$$(h_1^{(k)}, u_2^{(j)}, v_2^{(j)}, h_2^{(j)})(1 \le k \le r_1, 1 \le j \le r_2);$$

Step4. In the i-th iteration, randomly choose r_i different values $u_i^{(1)}, u_i^{(2)}, \cdots, u_i^{(r_i)}$ and respectively visit the random permutation, and thereby obtain r_i pairs of input-output $(u_i^{(1)}, v_i^{(1)}), (u_i^{(2)}, v_i^{(2)}), \cdots, (u_i^{(r_2)}, v_i^{(r_2)})$。For every $(u_i^{(j)}, v_i^{(j)})(1 \le j \le r_i)$, compute

$$m_i = u_i^{(j)} \oplus h_{i-1} \quad \text{and} \quad h_i = v_i^{(j)} \oplus L_1(m_i) \oplus L_2(h_{i-1}).$$

Since in the $(i-1)$-st iteration, we obtain $r_1 r_2 \cdots r_{i-1}$ random values of h_{i-1}, for every $(u_i^{(j)}, v_i^{(j)})(1 \le j \le r_i)$, we obtain $r_1 r_2 \cdots r_{i-1}$ random (m_2, h_2)。Store $r_1 r_2 \cdots r_{i-1} r_i$ quads $(h_{i-1}^{(k)}, u_i^{(j)}, v_i^{(j)}, h_i^{(j)})(1 \le k \le r_1 \cdots r_{i-1}, 1 \le j \le r_i)$;

Step5. Do as above up to the t-th iteration, and we could obtain $r_1 r_2 \cdots r_{t-1} r_t$ random (m_t, h_t)。Store $r_1 r_2 \cdots r_t$ quads $(h_{t-1}^{(k)}, u_t^{(j)}, v_t^{(j)}, h_t^{(j)})(1 \le k \le r_1 \cdots r_{t-1}, 1 \le j \le r_t)$;

Step6. If the size of the hash value is s, then according to the birthday attack, we have that if $r_1 r_2 \cdots r_{t-1} r_t = 2^{s/2}$, then the pair $h_t^{(p)}$ and $h_t^{(q)}$ satisfying $h_t^{(p)} = h_t^{(q)}$ could be found with the probability of 0.39. Then searching for the table of the t-th iteration, we could obtain the quads corresponding to the $h_t^{(p)}$ and $h_t^{(q)}$, that is,$(h_{t-1}^{(k_1)}, u_t^{(p)}, v_t^{(p)}, h_t^{(p)})$ and $(h_{t-1}^{(k_2)}, u_t^{(q)}, v_t^{(q)}, h_t^{(q)})$。Then compute

$$m_t^{(p)} = h_{t-1}^{(k_1)} \oplus u_t^{(p)} \quad \text{and} \quad m_t^{(q)} = h_{t-1}^{(k_2)} \oplus u_t^{(q)}.$$

Continue to search for the table of the (t-1)-*st* iteration to obtain the quads corresponding to $h_{t-1}^{(k_1)}$ and $h_{t-1}^{(k_2)}$, and then compute the pair of message used in the (t-1)-*st* iteration. Look forward for the two t-block collision messages.

4.2 Analysis of the Computational Complexity

The computational complexity of our algorithm includes time complexity and memory. Firstly, we analyze the time complexity of our multi-block collision attack.

In Step1 and Step2, we need to visit the permutation F r_1 times and r_2 times respectively; In Step3, the computation of (m_2, h_2) could be ignored, and we need to visit the permutation F r_i times. Hence, the time complexity of our multi-block collision attack is $\sum_{i=1}^{t} r_i$ times of visiting permutation F, where $r_1 r_2 \cdots r_{t-1} r_t = 2^{s/2}$.

Due to the Geometric inequality

$$\frac{1}{t} \sum_{i=1}^{t} r_i \ge \sqrt[t]{r_1 r_2 \cdots r_t}$$

if and only if $r_1 = r_2 = \cdots = r_t$, the equality is guaranteed. Therefore, when $r_1 = r_2 = \cdots = r_t = 2^{\frac{s}{2t}}$, the time complexity reaches the minimal, that is, $t2^{\frac{s}{2t}}$.

Next we analyze the memory of our algorithm.

Since we need to look forward to obtain the two t-block collision messages, the results in Step1–5 need to be stored. In Step1, we need store the r_1 pairs of input-output $(u_1^{(1)}, v_1^{(1)}), (u_1^{(2)}, v_1^{(2)}), \cdots, (u_1^{(r_1)}, v_1^{(r_1)})$ and the corresponding $h_1^{(1)}, h_1^{(2)}, \cdots, h_1^{(r_1)}$, that is, r_1 triples $(u_1^{(j)}, v_1^{(j)}, h_1^{(j)})(1 \le j \le r_1)$; In Step2–3, for every $h_1^{(j)}(1 \le j \le r_1)$, we need store r_2 triples $(u_2^{(j)}, v_2^{(j)}, h_2^{(j)})(1 \le j \le r_2)$, hence we need store $r_1 r_2$ triples $(u_2^{(j)}, v_2^{(j)}, h_2^{(j)})(1 \le j \le r_2)$; In Step4, for every $h_{i-1}^{(j)}(1 \le j \le r_1 r_2 \cdots r_{i-1})$, we need store r_i triples $(u_i^{(j)}, v_i^{(j)}, h_i^{(j)})(1 \le j \le r_i)$. Hence, the memory complexity of our multi-block collision attack is $r_1 + r_1 r_2 + r_1 r_2 r_3 + \cdots + r_1 r_2 \cdots r_t$. When the time complexity reaches the minimal, that is,$r_1 = r_2 = \cdots = r_t = 2^{\frac{s}{2t}}$, the memory complexity is

$$r_1 + r_1 r_2 + r_1 r_2 r_3 + \cdots + r_1 r_2 \cdots r_t = (2^{\frac{s}{2t}})^{t+1} - 1 = O(2^{s/2})$$

In a word, the time complexity of our t-block collision attack is $t2^{\frac{s}{2t}}$ times of visiting the permutation F, and the memory complexity is about $O(2^{s/2})$.

5 Conclusion

In this paper, we analyse the security of the Even-Mansour structure hash function, mainly using the property that the message both needs to be XORed to the beginning and the end of every iteration. Firstly, utilizing the partial invariables of input-output of one function f_1 based on the permutation F and the meet-in-the-middle techniques, we firstly present a preimage attack on Even-Mansour structure hash functions with two iterations, of which the time complexity is $2^{a(2^a-1)}+2^a+2^{n-2a}$ functional operations of f_1 or f_2 and the memory is $2^a a$-bit values, where n is the size of hash value and $a2^a \le n$. Secondly, utilizing the property that the beginning and the end of every iteration in Even-Mansour structure both need XOR the message or the transform result of message, we propose our multi-block collision attack on Even-Mansour structure hash functions with the time complexity of $t2^{\frac{s}{2t}}$ queries of F permutation and memory complexity of $O(2^{s/2})$, where t is the block number of collision message and s is the size of hash value. In the future work, we would like to apply our attack on the hash functions with EM structure.

Acknowledgment. We are grateful to the anonymous referees for their valuable comments. The work in this paper is supported by the National Natural Science Foundation of China (Grant No: 61802438 and 61772547).

References

1. Preneel, B., Govaerts, R., Vandewalle, J.: Hash functions based on block ciphers: a synthetic approach. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 368–378. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48329-2_31

2. Even, S., Mansour, Y.: A construction of a cipher from a single pseudorandom permutation. J. Cryptol. **10**(3), 151–161 (1997). https://doi.org/10.1007/s001459900025
3. Dunkelman, O., Keller, N., Shamir, A.: Minimalism in cryptography: the even-Mansour scheme revisited. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 336–354. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_21
4. Bogdanov, A., Knudsen, L.R., Leander, G., Standaert, F.-X., Steinberger, J., Tischhauser, E.: Key-alternating ciphers in a provable setting: encryption using a small number of public permutations. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 45–62. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_5
5. Isobe, T., Shibutani, K.: New key recovery attacks on minimal two-round even-Mansour ciphers. Asiacrypt 2017, Part I, LNCS 10624, pp. 244–263 (2017)
6. Leurent, G., Sibleyras, F.: Low-memory attacks against two-round even-mansour using the 3-XOR problem. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 210–235. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_8
7. Luo, Y.Y., Lai, X.J.: Attacks on JH, Grøstl and SMASH Hash Functions. http://eprint.iacr.org/2013/233.pdf
8. Dworkin, M.: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions, Federal Inf. Process. Stds. (NIST FIPS), National Institute of Standards and Technology, Gaithersburg, MD. https://doi.org/10.6028/NIST.FIPS.202
9. Guo, J., Peyrin, T., Poschmann, A.: The PHOTON family of lightweight hash function. CRYPTO 2011, LNCS 6841, pp. 222–239 (2011)
10. Wu, H.J.: The hash function JH (2011). http://www3.ntu.edu.sg/home/wuhj/research/jh/jhround3.pdf
11. Isobe, T.: A single-key attack on the full GOST block cipher. J. Cryptol. **26**(1), 172–189 (2013)
12. Oliynykov, R., et al.: A new standard of Ukraine: The Kupyna hash function. Cryptology ePrint Archive, Report 2015/885 (2015). http://eprint.iacr.org/2015/885References