On the security of multivariate hash functions

Yiyuan Luo[1], Xuejia Lai[1]

[1] *Department of Computer Science and Engineering, Shanghai Jiaotong University, Shanghai, China*

E-mail: luoyiyuan@sjtu.edu.cn;Received MONTH DATE, YEAR.

Abstract Multivariate hash functions are a type of hash functions whose compression function is explicitly defined as a sequence of multivariate equations. Olivier Billet etc. have designed the hash function MQ-HASH and Jintai Ding etc. also propose a similar construction, which the security depends on the difficulty of solving randomly drawn systems of multivariate equations over a finite field. Finding preimage and collision can be reduced to solve the multivariate equations, which is a well known NP-hard problem. To prove the security of MQ-HASH, the designer assume that a multivariate hash function is a pseudo-random number generator. In this paper, we analyze the security of multivariate hash functions and conclude that low degree multivariate functions such as MQ-HASH are neither pseudo-random nor unpredictable. There may be trivial collisions and fixed point attacks if the parameter of the compression function has been chosen. And they are also not computation-resistance, which makes MAC forgery easily.

Keywords Hash functions, MACs, Multivariate, MQ-HASH

1 Introduction

Hash functions are easy-to-compute compression functions that a variable length input and covert it to a fixed-length output. It is used in digital signature and message authentication. A good hash function is assume to be preimage resistance, second pre-image resistance and collision resistance. But when a hash function is used as message authentication code, the property of computation-resistance is also required. If a hash function f is computation-resistance, then given zero or more text-hash pairs $(x_i, h(x_i))$, it is computationally infeasible to compute any text-hash pair $(x, h(x))$ for any new input $x \neq x_i$ (including possibly for $h(x) = h(x_i)$ for some i. This property is also called the random oracle property[1]. A good hash function should behave as a random oracle.

The most popular hash functions in use today are MD5 and SHA-1, but they all are dedicated construction and the security of them are hard to analysis. Recently there are some proposals to build a hash function on a hard mathematical problem[2][3]. Multivariate hash functions are one of them the security based on the hardness of solving a system of multivariate functions over a finite field $\mathbb{F}$. Billet, Robshaw and Peyrin introduced the multivariate hash function MQ-HASH[2], and simultaneously Ding and Yang propose a similar construction[3]. To prove the pre-image resistant

of MQ-HASH, the author assume that a multivariate quadratic function is a pseudo-random generator. But we will show this is not right. After the proposal of multivariate hash functions, Aumasson and Meier conclude that multivariate hash functions over $GF(2)$ of low degree are neither pseudo-random nor unpredictable[5]. And NMAC message authentication codes built on certain cubic multivariate hash function (which is a proposal of Ding and Yang)[3] allow key recovery faster than by exhaustive search. There are also some trivial collisions and near collisions if the polynomials are sparse.

If a multivariate quadratic equation is used then it is easy to find collision, since the first order differentials of any quadratic polynomial is affine. This fact leads to the designer to increase the order of the polynomials, at the same time not to decrease the efficiency very much. So the degree of MQ-HASH is four and the order of Cubic Construction by Ding and Yang is three. But the designer ignore a fact that for polynomial equations of degree d, the dth derivative is a constant. This implies that low degree multivariate hash functions are neither pseudo-random nor unpredictable[4], since we can distinguish it between a random function by compute the dth derivative. If the result is different from previous, then we distinguish the hash because it is different from previous hash functions. This result is regardless of the finite field $\mathbb{F}$ and only needs negligible computation. For the MQ-HASH, we needs 16 times of computation and for Cubic Construction 8 times of computation is enough.

Our Work. In the next section we describe the construction. In section 3 we describe the higher order differential of multivariate polynomials. Section 4 describes the trivial collisions and fixed point attacks. Section 5 describe two methods to distinguish a multivariate hash function from random functions. Section 6 studies the security of MACs built on multivariate hash functions, Eventually we give the conclusion in section 7.

2 MQ-HASH and Cubic Construction

The MQ-HASH is designed by Billet, Robshaw and Peyrin. It is a Merkle-Damgard construction with the compression function built on multivariate hash functions. The input message M is appended a single bit '1' followed by as many '0' as required to leave the message 64 bits short of a multiple of the block length. The remaining 64 bits are then used for a representation of the length of the input message M in bits. Assume that the message M requires t blocks after padding and so $M = B_1 \parallel \cdots \parallel B_t$.

At iteration i, for $1 \leqslant i \leqslant t$, the compression function is used to update the value v_{i-1} of an v-bit chaining variable to v_i and v_0 is specified and fixed. Thus we have $v_i = h(v_{i-1}, B_i)$. The last chaining variable is used as the output of the hash function. The compression function is $h(v_{i-1}, B_i) = g \circ f(v_{i-1}, B_i)$ while the first function $f : \mathbb{F}^{m+n} \mapsto \mathbb{F}^r$, for $r > (m+n)$, expands the input, while a second function $g : \mathbb{F}^r \mapsto \mathbb{F}^n$ compresses the intermediate value and both f and g are quadratic.

The Cubic Construction is proposed by Ding and Yang. It is similar to the MQ-HASH except

mials. The compression function is $h(v_{i-1}, B_i) = f(v_{i-1}, B_i)$, while $f : \mathbb{F}^{2n} \mapsto \mathbb{F}^n$ is consisting of n cubic multivariate polynomials. Since the degree of MQ-HASH is four and higher than the Cubic Construction, hereafter we mainly concentrate on the MQ-HASH.

In [**?**], finding preimage of MQ-HASH is reduced to inverting f and g, since the NP-hard of MQ, it is very hard. To find a collision in MQ-HASH, one method is to finding collisions of f and the other is first finding collisions of g, then computing preimages of the two intermediates. This two method are both proven to be very hard.

3 Higher Order Derivatives of Multivariate Polynomials

Definition 2.1 (Higher order derivatives [6][7]**)** *Let $(S, +)$ and $(T, +)$ be Abelian groups. For a function $f : S \mapsto T$, the derivative of f at the point $a \in S$ is defined as*

$$\Delta_a f(x) = f(x + a) - f(x).$$

The i'th derivative of f at the point $a_1, \cdots, a_i$ is defined as

$$\Delta^{(i)}_{a_1,\cdots,a_i} f(x) = \Delta_{a_i}(\Delta^{(i-1)}_{a1,\cdots,a_{i-1}} f(x)).$$

Theorem 2.2 [6][7] *From the definition of derivative of multivariate functions, one can get the following result.*

$$f(x+a_1+a_2+\ldots+a_n) = \sum_{i=0}^{n} \sum_{1 \leqslant j_1 \ldots \leqslant j_i \leqslant n} \Delta^{(i)}_{a_{j_1},\ldots,a_{j_i}} f(x).$$

Theorem 2.3 *For any function $f : S \mapsto T$ with degree d, the d-th derivative of f is a constant.* [6]. When considering multivariate functions over $GF(2)$ the points $a_1, \ldots, a_i$ must be linearly independent for the i'th derivative not to be trivial zero.

The next step is to use higher order derivative of multivariate function to attack the multivariate hash functions. In fact, the attack is based on the property that the d-th derivative of a multivariate polynomials f with degree d is a constant. Suppose we compute the d-th derivative of multivariate function $f(x)$ with degree d at point $(a_1, \cdots, a_d)$, we get a constant $\mathcal{C}$, which is independent of the input x and only depends $(a_1, \cdots, a_d)$. One crucial notion for the security of hash functions is their pseudo-randomness, necessary for building secure key-derivation schemes, and, obviously, to instantiate pseudo-random functions. In [4] the definition of pseudo-random and unpredictable are given. A distribution of function is pseudo-random if it is easy to sample functions according to the distribution and to compute their value and it is hard to tell apart a function sampled according to this distribution from a uniformly distributed function given an adaptive access to the function as a black box. To show the distribution of multivariate functions is not pseudo-random, for a multivariate hash function f, we don't know the algebraic normal form of f and access it as a black box. Since we know the degree d of the multivariate function before hand, we compute the dth derivative for 2^d inputs, if the derivative is a constant, we can distinguish the function sampled from a uniformly distributed function.

In the other side, we can see the multivariate hash function is not unpredictable. A distribution

functions according to the distribution and to compute their value and for any efficient adversary that is given an adaptive black-box access to a function (sampled according to the distribution) it is hard to compute the value of the function at any point that was not queried explicitly. If we compute the hash value of some points of multivariate hash functions, it is easy to compute the value of a new point without access the black box. This can be see in the following corollary:

Corollary 2.4 *For a multivariate hash function F with degree d, the d-th derivative of F at point $(a_1, \cdots, a_d)$ is a constant $\mathcal{C}$ satisfies:*

$$\mathcal{C} = \sum_{\varepsilon_i \in \{0,1\}, 1 \leqslant i \leqslant d} (-1)^{\varepsilon_1 + \cdots + \varepsilon_d + 1} \cdot F(x + \varepsilon_1 a_1 + \cdots + \varepsilon_d a_d). \quad (1)$$

Thus, if $\varepsilon_1 a_1 + \cdots + \varepsilon_d a_d \neq \mathbf{0}$ when $\varepsilon_1 + \cdots + \varepsilon_d > 0$, we can get the F value of a input x by the following equation without access F:

$$F(x) = \sum_{\substack{\varepsilon_i \in \{0,1\}, 1 \leqslant i \leqslant d \\ \varepsilon_1 + \cdots + \varepsilon_d > 0}} (-1)^{\varepsilon_1 + \cdots + \varepsilon_d + 1} \cdot F(x + \varepsilon_1 a_1 + \cdots + \varepsilon_d a_d) - \mathcal{C}. \quad (2)$$

Proof. We prove the result by induction on the degree of F. For $d = 1$ the derivative of F at point a_1 is $\mathcal{C} = F(x + a_1) - F(x)$ and satisfies equation (1). Suppose (1) holds for $d - 1$. Then

$$\begin{aligned}
\mathcal{C} &= \Delta^{(d)}_{a_1, \cdots, a_d} F(x) \\
&= \Delta^{d-1}_{a_1, \cdots, a_{d-1}} (\Delta_{a_d} F(x)) \\
&= \Delta^{d-1}_{a_1, \cdots, a_{d-1}} (F(x + a_d) - F(x)) \\
&= \sum_{\varepsilon_i \in \{0,1\}, 1 \leqslant i \leqslant d-1} (-1)^{\varepsilon_1 + \cdots + \varepsilon_{d-1} + 1} \cdot (F(x + a_d + \varepsilon_1 a_1 + \cdots + \varepsilon_{d-1} a_{d-1}) \\
&= \sum_{\varepsilon_i \in \{0,1\}, 1 \leqslant i \leqslant d} (-1)^{\varepsilon_1 + \cdots + \varepsilon_d + 1} \cdot F(x + \varepsilon_1 a_1 + \cdots + \varepsilon_d a_d).
\end{aligned}$$

Equation 2 can be got directly by equation 1. Note that when $\varepsilon_1 + \cdots + \varepsilon_d > 0$, $\varepsilon_1 a_1 + \cdots + \varepsilon_d a_d \neq 0$ is required, since if it doesn't follows this condition, then we have already known $F(x)$ and needn't to do more. □

4 Trivial Collisions and Fixed Point Attack

4.1 Trivial Collisions

In [5] the definition of density of a polynomial is given. If we identify boolean functions with their representative polynomial over $GF(2)$, the weight of a polynomial is defined as the number of non-null coefficients in its algebraic normal form(ANF). The number of square-free monomials in n variables of degree in $[0, d]$ is $N(n, d) = \sum_{i=0}^{d} \binom{n}{i}$. The density of a polynomial of degree d in n variables is the ratio between its weight and $N(n, d)$, a random system of density $\delta \in [0, 1]$ has its equations with expected weight $\delta N(n, d)$.

Consider a family F of multivariate hash functions $\mathbb{F}^{m+n} \mapsto \mathbb{F}^n$ of density δ. Then for a random $h \in F$, any given monomial appears in an arbitrary component h_i with probability δ. In particular, a given degree 1 monomial x_i appears in no single component with probability $(1-\delta)^n$, when this happens , it is easy to see that $h(0, \ldots, x_i = 0, \ldots, 0)$ is the same as $h(0, \ldots, x_i = 1, \ldots, 0)$. Consequently, for any such pair of inputs, a collision like this occurs in probability $(1-\delta)^n$. Moreover, by trying all

collision with probability $p = 1-(1-(1-\delta)^n)^{n+m}$. Aumasson and Meier observe that for all the parameters of the Cubic Construction proposed in [3], $p \simeq 1$, hence with high probability at least one collision can be found, while it only needs $m+n$ times of access the hash function.

When the field is $GF(2)$, if each component h_i contains an even number of monomials, since the constant monomial 1 appears with probability δ in a given h_i, the collision $h(0,\ldots,0) = h(1,\ldots,1)$ will hold with probability $(1-\delta)^n$. For $n = 160$ and $m = 160$ in the Cubic Construction, this collision holds with probability 0.73.

In order to reduce the time cost of multivariate hash function, Ding and Yang use the sparse polynomials. They give a instance of multivariate polynomials with density less of 0.2%. That is to say, less than 0.2% of the coefficients are non-zero. If this system is used in practice, there will exist some input bits done't effect the output bits. Thus many trivial collisions will be found.

4.2 Fixed Point Attacks

In the origin construction of MQ-HASH, Billet, Robshaw and Peyrin give an instance which the chaining variable is 160 bits in length, the message block at each iteration is 32 bits in length, the compression function is $H_i = F(H_{i-1}, M)$. Since the M has only 32 bits, if we fixed $H_i = H_{i-1} = x$, then exhaustive search on the M, it is possible to find an M satisfies:

$$x = F(x, M) \tag{3}$$

The attack is called fixed point attack[8][9], if the ble to insert an arbitrary number of blocks equal to M without modifying the hash code. And it is also possible to producing collisions or a second preimage with this attack. In the above instance of MQ-HASH, to find an M satisfies equation **??**, we have 160 equations with 32 variables, so the probability of success to find such a fixed point is $2^{-(160-32)} = 2^{-128}$. Though this is impractical, it implies it must be careful of choose a good parameter for the security of multivariate hash functions.

5 Two Distinguish Methods for Multivariate Hash Functions

In this section we describes two methods that can distinguish the multivariate hash function from the random functions. And we analysis the efficiency of them. The first algorithm is given in [5] by computing the algebraic normal form of the multivariate hash functions. The second method is to use higher order differentials, which is more general, because it works over any finite field while the first method only work in $GF(2)$.

Theorem 5.1 (Aumasson and Meier) *For a multivariate hash function $F : GF(2)^{m+n} \mapsto GF(2)^n$ with low degree d, if we seem a random $h \in F$ as a black box, computing the algebraic normal form of h can be achieved in $\sum_{i=0}^{d} \binom{n}{i}$ queries to the box.*

Proof. If we seem B as the challenge box with components $\{B_i\}_{0 \leqslant i \leqslant n}$, then $B_i(0,\ldots,0)$ is equal to the constant term of the algebraic normal form of B_i, By querying B with all inputs of weight 1, one can recovers all the linear terms of the alge-

edge of the constant terms. If we know the all the weight 1 terms, we can queries weight 2 then get the quadratic monomials, continue doing this, we eventually get the algebraic normal form of h in $\sum_{i=0}^{d} \binom{n}{i}$ times. □

Theorem 5.2 (Higher Order Differential) *For a multivariate hash function $F : \mathbb{F}^{m+n} \mapsto \mathbb{F}^n$ with low degree d, if we seem F as a black box, we can compute the dth derivative of F by 2^d queries to the box. If we get the constant derivative, we can distinguish it between a random function.*

Proof. This can be directly got from corollary 2.4.□

In theorem 5.1, the box is identified by computing its algebraic normal form up to degree d, then evaluating the system obtained, and querying the box with a same input of degree $> d$. A random function will have an output distinct from the degree d system's with probability $\leqslant (1-2^{-n})$, one identifies the box with high probability. With this method, when the finite field is $GF(2)$ one can distinguish a random instance of MQ-HASH and Cubic Construction from a random function with respectively $2^{25.74}$ and $2^{22.38}$ black box queries.

Theorem 5.2 gives a more generic method to distinguish the black box. And its efficiency depends the degree of the multivariate polynomials which is 2^d. For the MQ-HASH, whose degree is 4, it needs 2^4 queries, while in the Cubic Construction, it only need 2^3 queries. Note theorem 5.1 only works in $GF(2)$, while theorem 5.2 works for any

6 The Security of MACs Built on Multivariate Hash Functions

Usually a message authentication code algorithm is constructed on MDC algorithms, by simply including a secret key k as part of the MDC input. A concern with this approach is that implicit but unverified assumptions are often made about the properties that MDCs have; in particular, while most MDCs are designed to provide one-wayness or collision resistance, the requirement of MAC algorithm is different. The MAC algorithm must be computation-resistance, that is, given zero or more text-MAC pairs, it is computationally infeasible to compute any new text-MAC without knowing the key. The most popular MAC constructed from MDC are Nested MACs(NMAC) and keyed-Hash MACs(HMAC). Given a multivariate hash function $F : \mathbb{F}^{m+n} \mapsto \mathbb{F}^n$ with degree d, the NMAC construction with a key $(k_1, k_2), k_i \in \mathbb{F}^n$ is:

$$\mathrm{NMAC}_{k_1,k_2}(x) = F_{k1}(F_{k_2}(x)).$$

We assume that the iterated hash function has no padding rule and the length of x is equal to one message block. For the MQ-HASH and Cubic Construction, the degree of the NMAC is $4^2 = 16$ and $3^2 = 9$ respectively. With the higher order differential method, let an attacker have access to NMAC_{k_1,k_2} as a black box, with 2^{16} and 2^9 queries for the message of his selection respectively he can compute the dth derivative, which is a constant, then he queries $2^{16} - 1 = 65535$ and $2^9 - 1 = 511$ times respectively again, he can make a new text-

The HMAC construction with a secret k is:

$$\mathrm{HMAC}_k(x) = F((k \oplus opad) \parallel F((x \oplus ipad) \parallel x)).$$

Hence it needs to call the compression function at least three times, the degree of HMAC is at least d^3. So in the MQ-HASH and Cubic Construction, to make a success selective forgery, it needs 2^{64} and 2^{27} queries.

7 Conclusions

In this paper we have analyzed the weakness of low degree multivariate hash functions, it shows that it must be careful when a parameter of multivariate hash function is chosen. We suggest that in order to improve the security of multivariate hash function, the degree cannot be too low, and the field $GF(2)$ is not a good choice. But when the degree is high and other fields are used, the efficiency will be decreased. The other question is there may be many weak instance about the random multivariate polynomials. To deploy a good random system is still an open problem.

References

[1] M. Bellare , P. Rogaway, Random oracles are practical: a paradigm for designing efficient protocols, In *Proceedings of the 1st ACM conference on Computer and communications security*, November 1993,03-05, 62-73.

[2] O. Billet, M.J. B. Robshaw, and T. Peyrin. On building hash functions from multivariate quadratic equations. In Josef Pieprzyk, Hossein Ghodosi, and Ed Dawson, editors, *ACISP*, LNCS 4586, Springer,2007, 82-95. .

[3] J. Ding and B. Yang. Multivariates polynomials for hashing. In Pei. D, Yung. M, Lin. D and Wu. C, editors , *Inscrypt 2007*, LNCS4990,Springer, 2008, 358-371.

[4] M. Naor and O. Reingold. From unpredictability to indistinguishability: A simple construction of pseudo-random functions from MACs (extended abstract). In Hugo Krawczyk, editor, *CRYPTO*, LNCS1462,Springer,1998, 267-282,

[5] J. Aumasson and W. Meier. Analysis of Multivariate Hash Functions. *Information Security and Cryptology - ICISC 2007*,LNCS 4817, Springer,2007, 309-323.

[6] X. Lai. Higher order derivatives and differential cryptanalysis. In *Communications and Cryptography: Two Sides of One Tapestry*, R.E. Blahut et al., eds., Kluwer Adademic Publishers, 1994, 227-233.

[7] L. R. Knudsen. Truncated and higher order differentials. In B.Preneel, editor, *FSE*, LNCS 1008, Springer,1995, 196-211.

[8] B.Preneel, Analysis and design of cryptographic hash functions, PhD thesis, Katholieke Universiteit Leuven (Belgium), Jan. 1993.

[9] B. Preneel, The state of cryptographic hash functions, In *Lectures on Data Security: Modern Cryptology in Theory and Practice*, LNCS 1561, Springer,1999, 158 C 182.