Higher Order Differential Cryptanalysis of Multivariate Hash Functions

Abstract

In this paper we propose an attack against multivariate hash functions, which is based on higher order differential cryptanalysis. As a result, this attack can be successful in finding the preimage of the compression function better than brute force and it is easy to make selective forgeries when a MAC is constructed by multivariate polynomials with low degree. It gives evidence that families of multivariate hash functions with low degree are neither pseudo-random nor unpredictable and one can distinguish a function from random functions, regardless of the finite field.

Key words: Cryptanalysis; Hash functions; Multivariate polynomials; Higher order; MAC;

1 Introduction

It is well known that hash functions play a fundamental role in data integrity and message authentication. Hash functions are used for data integrity in conjunction with digital signature, where a message is usually hashed first, and then the hash value is signed in place of the original message. Hash functions typically can be split into two classes, unkeyed hash functions(MDCs) and Keyed hash functions (MACs). MACs may be viewed as hash functions which take two inputs, a message and a secret key, and produce a output, with the design requirement that it be computationally infeasible in practice to produce the same output without knowledge of the key.

In many cryptographic applications of hash functions, it is desirable that the only way to produce a valid pair (x, y) is to first choose x, and then compute $y = h(x)$ by applying the function h to x. Other security requirements of hash functions are motivated by their applications in particular protocols, such as signature schemes. The three basic security requirements of MDCs are well known:

1. *Preimage resistance*: it is computationally infeasible to find an input which hashes to a specified output;
2. *2nd-preimage resistance*: it is computationally infeasible to find any second input which has the same output as any specified input.
3. *Collision resistance*: it is computationally infeasible to find any two distinct inputs x, x' which hash to the same output.

MACs usually requires higher security than MDCs. The following security property will be required when a MAC algorithm is used.

4. *Computation resistance*: given zero or more text-MAC pairs $(x_i, h_k(x_i))$, it is computationally infeasible to compute any text-MAC pair $(x, h_k(x))$ for any new input $x \neq x_i$ (including possibly for $h_k(x) = h_k(x_i)$ for some i).

The reason for why this property is required is that if computation-resistance does not hold, the MAC algorithm is subject to *MAC forgery*[12, 13]. The opponent can predict the value of $MAC_K(x)$ for a message x without initial knowledge of K. If the adversary can do this for a single message, he is said to be capable of *existential forgery*. If the adversary is able to determine the MAC for a message of his choice, he is said to be capable of *selective forgery*. Ideally, *existential forgery* is computationally infeasible; a less demanding requirement is that only *existential forgery* is so. Practical attacks often require that a forgery is *verifiable*, i.e., that the forged MAC is surely known to be correct on before hand. In fact, computation resistance implies preimage-resistance and collision resistant.

Since the cryptanalytic advances[8, 9] have shown weaknesses of MD5 and SHA-1 that allow collisions to be computed much faster than brute force, some new hash schemes are proposed to meet the security requirement. Multivariate hash functions are one of which based on hard problems that solving a high degree multivariate equation is hard. Some multivariate schemes appeared in [2, 3].

This paper exploits higher order differential cryptanalysis [10] to attack *multivariate hash functions*. *multivariate hash functions* is a iterated hash schemes built upon a function which takes $m + n$ bits input and n bits output defined as a sequence of multivariate functions. Exactly speaking, the *compression function* are constructed by multivariate polynomials.

Our Contribution. We propose a new attack on multivariate hash functions and prove that this type of hash functions are not random oracles. The attack method is to find relationships between distinct inputs and their respective outputs using higher order differential. This attack can be succeed in find preimage of the compression function better than brute force. It can make *selective forgery* on MACs build on multivariate hash functions. Though we focus on the compression function, this attacks can be extended to iterated hash functions.

Organization. The rest of the paper is organized as follows. Section 2 describes the generic model of multivariate hash functions and the high order derivatives of multivariate polynomials. In section 3, we describe the attack and shows multivariate hash functions are neither pseudo-random nor unpredictable. Section 4 analyze the security of MDCs and MACs built on multivariate hash functions. In section 5 we make some conclusions.

2 Preliminaries

2.1 Generic Model of Multivariate Hash Functions

Multivariate hash functions was introduced in[2, 3], it is based on a hard mathematical problem that solving a system of multivariate equations. Before multivariate polynomials have been used in construct hash functions, a number of asymmetric schemes based on multivariate functions have been proposed[4, 5]. Here we give a generic definition of multivariate hash functions.

Definition (Multivariate Hash Functions) A multivariate hash function is an iterated hash which the compression function $F : \mathbb{K}^{m+n} \mapsto \mathbb{K}^n$ is given by

$$\begin{aligned} F(x_1, \cdots, x_m, y_1, \cdots, y_n) &= (f_1(x_1, \cdots, x_m; y_1, \cdots, y_n), \\ & \quad \cdots, \\ & \quad f_n(x_1, \cdots, x_m; y_1, \cdots, y_n)) \end{aligned}$$

where each f_i is a randomly chosen higher order polynomial over the finite field $\mathbb{K}$ and all the coefficients are chosen randomly.
The current chaining state is $\mathbf{x} = (x_1, \cdots, x_m)$, and each input message block is $\mathbf{y} = (y_1, \cdots, y_n)$.

The construction MA-HASH by Billet, Robshaw and Peyrin[2] and a similar construction by Ding and Yang[3] are special case of multivariate hash functions defined above. MQ-HASH is a quartic (degree 4) system h using two composed quadratic systems f and g,such that $h = g \circ f$. Ding and Yang also propose a cubic construction where the system has a degree 3. In order to improve the efficiency of the system, they use sparse polynomials. Aumasson and Meier [1] find that multivariate hash functions over $GF(2)$ of low-degree are neither pseudorandom nor unpredictable and constructions based on sparse polynomials are vulnerable to trivial collisions and near collisions. They also claim that NMAC message authentication codes built on certain cubic multivariate hash functions allow key recovery faster than exhaustive search.

2.2 Higher Order Derivatives

In [10] a definition of higher order derivatives of discrete functions was given and the concept of higher order differentials was introduced. Later higher order differentials were used to attack block ciphers presumably secure against conventional differential attacks [11].

Definition (Higher order derivatives) 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

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

Proposition 2.1 *For any function $f : S \mapsto T$ with degree d, the d-th derivative of f is a constant.*

Example For $f(x_1, x_2) = x_1^2 + x_1x_2$, $x_i \in GF(3)$. We compute the 2-nd derivative at $(01, 12)$.

$$\begin{aligned} \Delta_{01} f(x_1, x_2) &= f(x_1 + 0, x_2 + 1) - f(x_1, x_2) \\ &= x_1 \\ \Delta^{(2)}_{(01,12)} f(x_1, x_2) &= \Delta_{01} f(x_1 + 1, x_2 + 2) - \Delta_{01} f(x_1, x_2) \\ &= (x_1 + 1) - x_1 \\ &= 1 \end{aligned}$$

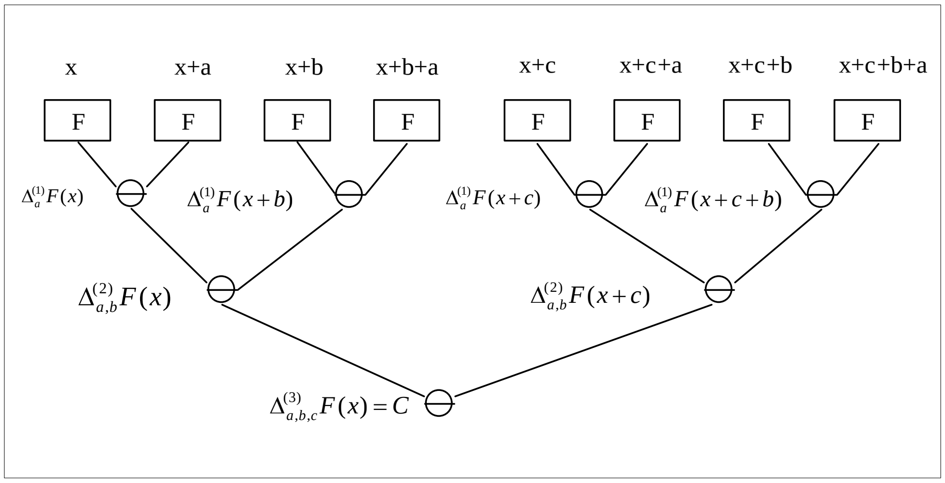
3 Attacks Using Higher Order Differentials

In this section we describe how to use higher order derivatives of multivariate functions to attack the hash function. 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 $f(\mathbf{x})$ with degree d at point $(a_1, \cdots, a_d)$, we get a constant $\mathcal{C}$, which is independent of the input $\mathbf{x}$ and only depends $(a_1, \cdots, a_d)$.

For example, for a multivariate hash function F with degree 3, it is easy to compute the 3rd derivative of F at point $(\mathbf{a}, \mathbf{b}, \mathbf{c})$. If we don't know the coefficients of polynomials, then we choose a random input $\mathbf{x}$, then we get

$$\begin{aligned} \mathcal{C} &= \Delta^{(3)}_{\mathbf{a},\mathbf{b},\mathbf{c}} F(\mathbf{x}) \\ &= \{[F(\mathbf{x}+\mathbf{c}+\mathbf{b}+\mathbf{a}) - F(\mathbf{x}+\mathbf{c}+\mathbf{b})] - [F(\mathbf{x}+\mathbf{c}+\mathbf{a}) - F(\mathbf{x}+\mathbf{c})]\} \\ &\quad -\{[F(\mathbf{x}+\mathbf{b}+\mathbf{a}) - F(\mathbf{x}+\mathbf{b})] - [F(\mathbf{x}+\mathbf{a}) - F(\mathbf{x})]\} \end{aligned}$$

Since every input $\mathbf{x}$ satisfies this equation, and if we have known $\mathcal{C}$ and the F value of 7 variables $\mathbf{x}+\mathbf{c}+\mathbf{b}+\mathbf{a}, \mathbf{x}+\mathbf{c}+\mathbf{b}, \cdots$, where each variable doesn't equal to $\mathbf{x}$, then we can compute $F(\mathbf{x})$ according to the equation above without access F. The following figure describe how it works.



x
x+a
x+b
x+b+a
x+c
x+c+a
x+c+b
x+c+b+a
F
$\Delta^{(1)}_a F(x)$
$\Delta^{(1)}_a F(x+b)$
$\Delta^{(1)}_a F(x+c)$
$\Delta^{(1)}_a F(x+c+b)$
$\Delta^{(2)}_{a,b} F(x)$
$\Delta^{(2)}_{a,b} F(x+c)$
$\Delta^{(3)}_{a,b,c} F(x) = C$

Figure 1: The 3rd derivative of F, while $\ominus$ means the difference

We have considered the case for degree three, then we will extend to higher degrees. Based on the above analysis, we easily get the following theorem.

Theorem 3.1 *For a multivariate hash function F with degree d, the d-th derivative of F at point $(\mathbf{a}_1, \cdots, \mathbf{a}_d)$ is a constant $\mathcal{C}$ satisfies:*

$$\mathcal{C} = \sum_{\varepsilon_i \in \{0,1\}, 1 \le i \le d} (-1)^{\varepsilon_1 + \cdots + \varepsilon_d + 1} \cdot F(\mathbf{x} + \varepsilon_1 \mathbf{a}_1 + \cdots + \varepsilon_d \mathbf{a}_d). \tag{1}$$

Thus, if $\varepsilon_1 \mathbf{a}_1 + \cdots + \varepsilon_d \mathbf{a}_d \neq \mathbf{0}$ when $\varepsilon_1 + \cdots + \varepsilon_d > 0$, we can get the F value of a input x by the following equation without access F:

$$F(x) = \sum_{\substack{\varepsilon_i \in \{0,1\}, 1 \le i \le d \\ \varepsilon_1 + \cdots + \varepsilon_d > 0}} (-1)^{\varepsilon_1 + \cdots + \varepsilon_d + 1} \cdot F(\mathbf{x} + \varepsilon_1 \mathbf{a}_1 + \cdots + \varepsilon_d \mathbf{a}_d) - \mathcal{C}. \tag{2}$$

Proof. We prove the result by induction on the degree of F. For $d = 1$ the derivative of F at point $\mathbf{a}_1$ is $\mathcal{C} = F(\mathbf{x}+\mathbf{a}_1) - F(\mathbf{x})$ and satisfies equation (1). Suppose (1) holds for $d-1$. Then

$$\begin{aligned}
\mathcal{C} &= \Delta^{(d)}_{a_1,\cdots,a_d} F(\mathbf{x}) \\
&= \Delta^{d-1}_{a_1,\cdots,a_{d-1}} (\Delta_{a_d} F(\mathbf{x})) \\
&= \Delta^{d-1}_{a_1,\cdots,a_{d-1}} (F(\mathbf{x}+\mathbf{a}_d) - F(\mathbf{x})) \\
&= \sum_{\varepsilon_i \in \{0,1\}, 1 \le i \le d-1} (-1)^{\varepsilon_1+\cdots+\varepsilon_{d-1}+1} \cdot \\
&\quad (F(\mathbf{x}+\mathbf{a}_d+\varepsilon_1\mathbf{a}_1+\cdots+\varepsilon_{d-1}\mathbf{a}_{d-1}) - F(\mathbf{x}+\varepsilon_1\mathbf{a}_1+\cdots+\varepsilon_{d-1}\mathbf{a}_{d-1})) \\
&= \sum_{\varepsilon_i \in \{0,1\}, 1 \le i \le d} (-1)^{\varepsilon_1+\cdots+\varepsilon_d+1} \cdot F(\mathbf{x}+\varepsilon_1\mathbf{a}_1+\cdots+\varepsilon_d\mathbf{a}_d).
\end{aligned}$$

Equation 2 can be got directly by equation 1. Note that when $\varepsilon_1+\cdots+\varepsilon_d > 0$, $\varepsilon_1\mathbf{a}_1+\cdots+\varepsilon_d\mathbf{a}_d \neq \mathbf{0}$ is required, since if it doesn't follows this condition, then we have already known $F(\mathbf{x})$ and we needn't to do more.

Since a well designed hash function should behavior like a random oracle[7] and the only efficient way to determine the hash value of a input $\mathbf{x}$ is to actually evaluate the hash function at the value $\mathbf{x}$, it is obvious that multivariate hash functions can not suffice to this condition. We can compute $F(\mathbf{x})$ when we get the hash value of input set $\{\mathbf{x}+\varepsilon_1\mathbf{a}_1+\cdots+\varepsilon_d\mathbf{a}_d \mid \varepsilon_i \in \{0,1\}, \varepsilon_1+\cdots+\varepsilon_d > 0, 1 \le i \le d\}$ which has $2^d - 1$ elements. This property implies multivariate hash functions are neither pseudo-random nor unpredictable [6].If we take the multivariate hash function as a black box and only know the maximum degree d. We don't know the coefficients of the polynomials, but can compute the d-th derivative $\mathcal{C}$ of function F at point $(\mathbf{a}_1, \cdots, \mathbf{a}_d)$. Then we can distinguish the function from random according the constant $\mathcal{C}$, since for a random function F, if the value that we get in the end is not $\mathcal{C}$, then the function is the random one, not the multivariate one. This implies F is not pseudo-random and obviously it is not unpredictable. (See [6] for the definition of pseudo-random and unpredictable).

4 On the Security of MDCs and MACs Based on Multivariate Hash function

In this section we consider the security of MDC and MAC which are constructed by multivariate hash functions. We show that finding preimages in this type of MDCs can be attacked better than brute force. And MACs of this type are not computation-resistance, which make selective forgery easily.

4.1 Preimage Attack on Compression Functions

For a multivariate compression function F, preimage attack is to find any preimage $\mathbf{x}'$ such that $F(\mathbf{x}') = \mathbf{y}$ when given any y for which a corresponding input is not known. For a ideal compression function yielding n-bit hash-values the strength is 2^n and for a random $\mathbf{x}$ the probability of it is the preimage of $\mathbf{y}$ is 2^{-n}. For a multivariate hash function with degree d, the attack is as follows:

1. For a multivariate compression function $F : \mathbb{K}^{m+n} \mapsto \mathbb{K}^n$ with degree d, given a $\mathbf{y}' = (y_1', \cdots, y_n')$ to find $\mathbf{x}' = (x_1, \cdots, x_m; y_1, \cdots, y_n)$ that satisfies $F(\mathbf{x}') = \mathbf{y}'$.

2. Fixed the first part $(x_1, \cdots, x_m)$ of $\mathbf{x}'$, choose a set of elements $(\mathbf{a}_1, \cdots, \mathbf{a}_d)$ where $\mathbf{a_i} = (a_1, \cdots, a_n)$ satisfies the condition that when $\varepsilon_1 + \cdots + \varepsilon_d > 0$, $\varepsilon_1\mathbf{a}_1 + \cdots + \varepsilon_d\mathbf{a}_d \neq \mathbf{0}$ where $\varepsilon_i \in \{0, 1\}$. Compute the d-th derivative constant $\mathcal{C}$ by equation (1).

3. Choose an $\mathbf{x}$ while the first part $(x_1, \cdots, x_m)$ is fixed and the remaining are random. Then $F(\mathbf{x})$ can be computed through equation (2) without access F. Thus we query the F function $2^d - 1$ times and usually get 2^d distinct hash-values.

4. So the probability of query $2^d - 1$ times to find a preimage is approximately $\frac{2^d}{k^n}$, where k is the order of the field $\mathbb{K}$. And this probability is higher than the ideal probability $\frac{2^d-1}{k^n}$.

For the MQ-HASH [2] with degree 4 while $\mathbb{K} = GF(2)$. The probability of the above attack is $\frac{16}{2^n}$ while the ideal probability is $\frac{15}{2^n}$. Though this attack improve the succeed probability such a little that can be ignored, it exposes that multivariate hash functions are not ideal and has some weakness.

4.2 MAC Forgeries

It is obviously that the adversary can make selective forgery when multivariate hash function has been used. This attack can be extended when the message authentication codes NMAC and HMAC are built on multivariate hash functions.

Let F be a multivariate hash function $\mathbb{K}^{m+n} \mapsto \mathbb{K}^n$ of degree d. For an arbitrary known $h \in F$, we consider a iterated hash function $H_k^* : \mathbb{K}^* \mapsto \mathbb{K}^n$ with initial value $k \in \mathbb{K}^n$, no padding rule, and no output filter. For $x \in \mathbb{K}^*$, the NMAC construction with secret key $(k_1, k_2), k_i \in \mathbb{K}^n$, is describe as follows:

$$\mathrm{NMAC}_{k_1,k_2}(x) = h_{k_1}(H_{k_2}^*(x))$$

and the HMAC construction with key k is defined by

$$\mathrm{HMAC}_k(x) = H_{iv}^*(k \oplus \mathrm{OPAD} \parallel H_{iv}^*(k \oplus \mathrm{IPAD} \parallel x)),$$

with constant OPAD and IPAD long of one message block.

In fact, both the NMAC and HMAC can be viewed a multivariate function $F_k(x)$ with higher degree that depends with the message block length of x. This is because the composite of $F(x)$ is also a multivariate polynomial function with higher degree. In the NMAC described above, the degree is d^2, while the degree of HMAC is d^3.

5 Conclusion

In this paper, we describe a new attack on the multivariate hash functions ,which is based on higher order differential cryptanalysis. As a result, this attack can be succeed in find preimage of the compression function better than brute force and it is easy to make selective forgeries when a MAC is constructed by multivariate polynomials. Since every dedicated hash function may be write in multivariate boolean functions, it seems the attack can be succeed when the degree of the function is low.

References

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

[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, pp.82-95. Springer, 2007.

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

[4] H. Imai and T. Matsumoto. A class of asymmetric crypto-systems based on polynomials over finite rings. *IEEE International Symposium on Information Theory*, pages 131-132, 1983.

[5] J. Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): Two new families of asymmetric algorithms. In Ueli M. Maurer, editor, *EUROCRYPT*, LNCS1070, pp.33-48. Springer,1996.

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

[7] 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*, pp.62-73, November 03-05, 1993.

[8] X. Wang, D. Feng, X. Lai and H. Yu. Collisions for hash functions MD4, MD5, HAVAL-128 and RIPEMD. *Cryptology ePrint Archive*, Report 2004/199, http://eprint.iacr.org/.

[9] X. Wang, Y. L. Yin and H. Yu. Finding collisions in the full sha-1. In *CRYPTO*, Victor Shoup, editor, LNCS3621, pp.17-36, Springer,2005.

[10] 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, pp.227-233, 1994.

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

[12] B. Preneel, The state of cryptographic hash functions, In *Lectures on Data Security: Modern Cryptology in Theory and Practice*, LNCS 1561, pp. 158C182, Springer,1999.

[13] A. J. Menezes, P. C. Oorschot and S. Vanstone. Handbook of Applied Cryptography. CRC Press, 1996.