Analysis of Multivariate Hash Functions

Jean-Philippe Aumasson* and Willi Meier**

FHNW, 5210 Windisch, Switzerland

Abstract. We analyse the security of new hash functions whose compression function is explicitly defined as a sequence of multivariate equations. First we prove non-universality of certain proposals with sparse equations, and deduce trivial collisions holding with high probability. Then we introduce a method inspired from coding theory for solving underdefined systems with a low density of non-linear monomials, and apply it to find collisions in certain functions. We also study the security of message authentication codes HMAC and NMAC built on multivariate hash functions, and demonstrate that families of low-degree functions over GF(2) are neither pseudo-random nor unpredictable.

1 Introduction

A fundamental idea of *multivariate cryptography* was stated by Shannon in 1949 [35]: *"if we could show that solving a certain system requires at least as much work as solving a system of simultaneous equations in a large number of unknowns, of a complex type, then we would have a lower bound of sorts for the work characteristic"*. Multivariate primitives are indeed directly described in terms of multivariate polynomial functions, in order to reduce certain security problems to the presumably hard problem of solving the system, and/or to problems like Polynomial Isomorphism, Minrank, *etc.* At the opposite, primitives without explicit multivariate equations might be attacked by first finding a full or partial description as a system of equations, then exploiting the latter system (ideally, solving it) – this is the principle of *algebraic attacks*. A number of multivariate primitives appeared since the early years of modern cryptology, mainly asymmetric schemes (Matsumoto-Imai [28], Ong-Schnorr-Shamir [33], HFE [34], *etc.*), and more recently, the stream cipher QUAD [7]. As the advent of RSA led to a multitude of works on integer factorisation, researchers designed new algorithms for solving multivariate systems of equations, to tackle multivariate primitives [14, 18, 24]. Note that, although every cipher possesses a characterisation as a system of Boolean equations, this latter is generally at least as hard to compute as breaking the cipher with a brute force attack.

This paper analyses *multivariate hash functions*, that is, iterated hash schemes built upon a function $\mathbb{K}^{m+n} \mapsto \mathbb{K}^n$ explicitly defined as a sequence of multivariate equations. More precisely, we focus our study on the later *compression function*, and relate its parameters to the security of the overall primitive. It is well-known that hash functions are essential in numerous real-life cryptographic schemes and protocols, be it digital signatures (in DSA and ECDSA), authentication codes (through HMAC), or for building pseudo-random function and derive keys. After several breakthroughs in the analysis of the previous standards MD5,

* Supported by the Swiss National Science Foundation under project number 113329.

** Supported by Hasler Foundation `http://www.haslerfoundation.ch/` under project number 2005

SHA-0, and SHA-1, the community is particularly attentive to new designs (*cf.* NIST's competition for a new standard [32]). Surprisingly, multivariate hash functions only appeared in 2007, in works by Billet, Peyrin and Robshaw [8], introducing the constructions MQ-HASH and RMQ-HASH, and simultaneously by Ding and Yang [20]. By reducing the problem of finding a preimage of a given digest to the problem of solving a multivariate system, a security guarantee is given for multivariate hash functions. Other "provably secure" hash functions exist, whose resistance to preimage and/or collision relies on problems as different as syndrome decoding [2, 25], (approximate) shortest vector in a lattice [6, 29], and non-trivial square root of very smooth numbers [12]. We notice that the term "multivariate hash functions" has already been employed in a non-cryptographic context [36] to denote functions hashing several objects at the same times, and that multivariate hash functions over GF(2) have recently been employed in the context of interactive hashing [27].

Our Work. Section 2 gives security definitions and presents our model of multivariate hash functions, along with the description of the constructions MQ-HASH, RMQ-HASH, and SCC. Section 3 then proves the non-universality of functions based on sparse equations, like SCC, and illustrates this with several trivial collisions holding with high probability. Section 4 introduces a new method for solving underdefined semi-sparse multivariate systems, that we apply to certain kinds of multivariate hash functions. Section 5 studies the security of message authentication codes NMAC and HMAC built on multivariate hash functions, showing a critical attack on NMAC-SCC. Section 6 demonstrates that all multivariate hash functions over GF(2) with small degree are efficiently predictable and distinguishable from random functions. As an aside, we identify weak instances of QUAD in Section 7, and eventually draw some conclusions in Section 8.

2 Preliminaries

Let $n \in \mathbb{N}^\star, m \in \mathbb{Z}$. A *multivariate system* (of equations) over a finite field $\mathbb{K}$ with n equations and $(m+n)$ unknowns is denoted as $\{h_i(x) = 0\}_{0 \leq i < n-1}$, or simply $h(x) = 0$, with $x = (x_0, \ldots, x_{m+n-1}) \in \mathbb{K}^{m+n}$. The system is called *underdefined* (respectively *overdefined*) when $m > 0$ (resp. $m < 0$). The degree $\deg(h)$ of the system is $\max_i \deg(h_i)$. We identify Boolean functions with their representative polynomial over GF(2), and the *weight* of a polynomial is defined as the number of non-null coefficients in its algebraic normal form. The number of square-free monomials in n variables (that is, considering $x^2 \equiv x$ as over GF(2)) of degree in $[0, d]$ is

$$\mathcal{N}(n, d) = \sum_{i=0}^{d} \binom{n}{i} . \tag{1}$$

The *density* of a polynomial of degree d in n variables is the ratio between its weight and $\mathcal{N}(n, d)$, so that a *random system* of density $\delta \in [0, 1]$ has its equations with expected weight $\lfloor \delta \mathcal{N}(n, d) \rfloor$ (for convenience, we omit from now the flooring symbols $\lfloor . \rfloor$), such that each monomial has probability δ to appear in an arbitrary component. We call a random system

sparse when it has density $\delta \ll 50\%$, and *semi-sparse* when only monomials of certains degrees have a density $\ll 50\%$ (for example, imagine a cubic system where δ only applies to the cubic monomials). Eventually, when we mention "random" objects, it implicitly means with respect to a uniform distribution in the appropriate sample space, unless precised differently.

2.1 Security Definitions

We give the security definitions for *families* of hash functions[1], that is, subsets of the set of all functions $\mathbb{K}^{m+n} \mapsto \mathbb{K}^n$ for fixed $\mathbb{K}$, m, and n. These families can also be seen as *distributions* over this superset.

Preimage	
Input	a random hash function $h \in \mathcal{F}$, a random digest $y \in \{0,1\}^n$.
Output	$x \in \{0,1\}^{m+n}$ such that $h(x) = y$.

Second Preimage	
Input	a random hash function $h \in \mathcal{F}$, a random input $x \in \{0,1\}^{m+n}$.
Output	$x' \in \{0,1\}^{m+n}$ such that $h(x) = h(x')$ and $x \neq x'$.

Collision	
Input	a random hash function $h \in \mathcal{F}$.
Output	$x, x' \in \{0,1\}^{m+n}$ such that $h(x) = h(x')$ and $x \neq x'$.

In addition, the term *near-collision* designates a collision over only certain bits of the digest. For an ideal hash function h, the problems above have complexity of about 2^n, 2^n and $2^{n/2}$ evaluations of h respectively.

Another 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. Since we actually consider distributions of functions rather than single instances, the following definitions from [31] are relevant.

Definition 1. *A distribution of functions $\mathcal{F}$ is* pseudo-random *if*

1. *this distribution is efficient (i.e., it is easy to sample functions according to the distribution and to compute their value), and*
2. *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.*

Definition 2. *A distribution of functions $\mathcal{F}$ is* unpredictable *if*

1. *this distribution is efficient, and*

[1] We further call "hash functions" mappings $\mathbb{K}^{m+n} \mapsto \mathbb{K}^n$, independently of their role of compression function in iterated hash functions.

2. *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.*

(See [31] for more formal definitions 3.1 and 3.2.) Finally, we recall the definition of ε-universality.

Definition 3. *A family of functions $\mathcal{F}$ is* ε-universal *if for any distinct inputs x and x' and a random $h \in \mathcal{F}$, the probability (over the choice of h) that $h(x) = h(x')$ is at most ε.*

We rather consider the computational version, denoted ε-cAU (computational almost universality, see [4]), such that for a family not ε-cAU, one can efficiently compute such a pair (x, x').

2.2 Multivariate Hash Functions

A *multivariate hash function* $h : \mathbb{K}^{m+n} \mapsto \mathbb{K}^n$ is explicitly defined as a sequence of n polynomial functions $h_i : \mathbb{K}^{m+n} \mapsto \mathbb{K}$ for some finite field $\mathbb{K}$, its *components*. A *family* of multivariate hash functions is characterised by a *construction* scheme, along with a choice of *parameters* for this scheme, thereby defining a distribution over the set of all functions $\mathbb{K}^{m+n} \mapsto \mathbb{K}^n$, where $\mathbb{K}$, m, and n are fixed either by the construction itself of by the parameters. An *instance* is then randomly picked with respect to that distribution, casting into the framework of *probabilistic hash functions* [9].

Given an arbitrary family of multivariate hash functions $\mathcal{F}$, solving Preimage reduces (in the Turing sense) to solving the system $h(x) = y$ for random $h \in \mathcal{F}$ and $y \in \mathbb{K}^n$. When $\mathcal{F}$ corresponds to the set of all quadratic systems over $\mathbb{K}$ with $(m + n)$ unknowns and n equations, Preimage reduces to the problem MQ, known to be NP-hard for any finite field $\mathbb{K}$ if m is small [26]:

Multivariate Quadratic (MQ)	
Input	a finite field $\mathbb{K}$, a system $f = \{f_i\}_{0 \leq i < n}$ of n random quadratic equations in $n + m$ variables over $\mathbb{K}$, $n \in \mathbb{N}^\star$, $m \in \mathbb{Z}$.
Output	$x \in \mathbb{K}^{m+n}$ such that $f(x) = 0$.

The problem of solving a multivariate system is also assumed hard for higher degrees (state-of-the art methods are briefly surveyed in Section 2.3). Furthermore, no efficient quantum algorithm is known yet to solve multivariate systems, hence multivariate hash functions have chances to survive in a world with efficient quantum computers.

On the other hand, Collision reduces to solving the equation $h(x) - h(x') = 0$ with the constraint $x \neq x'$, which will not be an instance of MQ. Another technique to find collisions is to assume that there exists a colliding pair (x, x') with difference $\Delta = x - x' = (x_i - x'_i)_{0 \leq i < m+n}$, then computing that pair by solving the system

$$h(x) - h(x - \Delta) = 0 \ , \tag{2}$$

for a fixed and *known* difference $\Delta \neq 0$. This system has degree at most $\deg(h) - 1$, and is expected to have at least one solution for a sufficiently large m. We shall further refer to this technique as the *generic attack*.

Composed Quadratics Construction. The construction MQ-HASH by Billet, Peyrin and Robshaw [8], and an unnamed construction by Ding and Yang [20], propose to define a quartic (degree 4) system h using two composed quadratic systems f and g, such that $h = g \circ f$. Following the ideas of [1], the first box $f : \mathbb{K}^{m+n} \mapsto \mathbb{K}^r$, for $r > (m+n)$, expands the input, while a second box $g : \mathbb{K}^r \mapsto \mathbb{K}^n$ compresses the intermediate value. Security aspects are much more developed in [8] than in [20], and we will only consider that former reference for composed quadratics. Hereafter we present a succinct overview of the security arguments for MQ-HASH.

First, the main result of [8] is the reduction of PREIMAGE to the problem of inverting f or g, which proves PREIMAGE-resistance, assuming the hardness of MQ for the parameters chosen. Although no reduction is given for COLLISION, several arguments are presented: indeed, a necessary property for h to resist COLLISION is the COLLISION-resistance of the expanding box f; this is expected to hold since f will actually be *collision-free* with high probability for well chosen parameters, as stated by Proposition 1 of [8]. In the worst case, when there exists a pair (x, x') such that $f(x) = f(x')$, this can be recovered by solving a linear system only if the difference $(x - x')$ is known. However, since the expected number of colliding pairs is very low, only very few differentials would lead to a collision. Choosing $2(m+n) - r < s$ ensures that a random instance will possess a collision with probability $< 2^{-s}$, let alone the hardness of finding the corresponding difference. Another strategy to find a colliding pair consists in

1. finding a collision (y, y') for g, and
2. computing preimages of y and y' by f,

but this again is not efficient since f is assumed hard to invert.

The iteration mode for MQ-HASH is a basic Merkle-Damgård mode, with standard padding and no output filter. In order the function to meet the 80-bit security level (meaning here a minimum of 2^{80} trials in average to find a collision), the authors of MQ-HASH propose to use the family over $\mathbb{K} = \mathrm{GF}(2)$ with message blocks of $m = 32$ bits, a chaining value of $n = 160$ bits, and an intermediate value of $r = 464$ bits. We will refer to this family along the paper.

An alternative constrution to MQ-HASH called RMQ-HASH [8] proposes to define

$$h(x) = h(x_1||x_2) = f(x_1, g(x_2)),$$

with the message block represented by x_1, and the previous chaining value by x_2. The functions f and g are quadratic, and defined as $f : \mathbb{K}^{m+r} \mapsto \mathbb{K}^n$, and $g : \mathbb{K}^n \mapsto \mathbb{K}^r$. This construction is just presented as a possible variant of MQ-HASH, and no security analysis is provided. In the remainder of the paper, we will rather concentrate our study on MQ-HASH, which remains the main proposal of [8], and whose analysis partially overlaps RMQ-HASH's.

However we can already observe that when $m > n$, one may simply set a random value for x_2, such that PREIMAGE reduces to solving a quadratic system of n equations in m unknown, but COLLISION with a given difference can be computed by solving a linear system with as many equations and unknowns.

Sparse Cubic Construction. This construction introduced by Ding and Yang [20] merely consists in a cubic system $h : \mathbb{K}^{2n} \mapsto \mathbb{K}^n$ (thus $m = n$) of density δ. In other words, every component h_i has exactly $\delta\mathcal{N}(2n, 3)$ monomials. We further use the shortcut "SCC" to refer this construction.

Clearly, PREIMAGE reduces to solving a sparse cubic system, assumed hard for well chosen parameters by the designers. The generic attack against COLLISION directly reduces here to solving a *sparse* quadratic system. Although assumed hard, the problem of solving sparse systems is not as hard as the general case (*cf.* Section 2.3). On the other hand, sparse systems provide a considerable speed-up, as well as reduced storage requirements. Several families are suggested, characterised by their parameters hereafter (recall $m = n$).

1. For 160-bit digests:
   - $\mathbb{K} = \mathrm{GF}(2)$, $n = 160$, $\delta = 0.1\%$
   - $\mathbb{K} = \mathrm{GF}(2^4)$, $n = 40$, $\delta = 0.1\%$
   - $\mathbb{K} = \mathrm{GF}(2^8)$, $n = 20$, $\delta = 0.2\%$
2. For 256-bit digests:
   - $\mathbb{K} = \mathrm{GF}(2)$, $n = 256$, $\delta = 0.1\%$
   - $\mathbb{K} = \mathrm{GF}(2^4)$, $n = 64$, $\delta = 0.1\%$
   - $\mathbb{K} = \mathrm{GF}(2^8)$, $n = 32$, $\delta = 0.1\%$
   - $\mathbb{K} = \mathrm{GF}(2^{16})$, $n = 16$, $\delta = 0.2\%$

2.3 Solving Multivariate Systems

To solve a system of multivariate equations, cryptanalysts mainly employ methods derived from Buchberger's algorithm to compute a Gröbner basis of a polynomial ideal. The most efficient ones are Faugère's F_4 and F_5 [22,23], and the algorithms of the XL family [13,16,17,19]. Those algorithms perform better on overdefined systems, and F_4 and F_5 are know to take advantage of sparse systems. Some work concentrates on the particularities of underdefined [15], overdefined [16], or sparse systems [37,38]. More recently, sparse systems derived from cryptographic primitives were solved by converting the system into a SAT instance, then solving this instance using an efficient SAT-solver (*e.g.* MINISAT [21]), and finally converting the solution to a solution of the system [3, 30]. Unfortunately, the complexity of multivariate solvers is often hard to estimate. Empirical results are here probably more significant for cryptanalysts. For instance, the algorithm XL-Wiedemann was demonstrated [40] to solve MQ over $\mathrm{GF}(2^8)$ with $n = 40$ equations and $n + m = 20$ unknowns in less than 2^{45} cycles of a 64-bit AMD Opteron processor (a few hours of computation).

3 Non-Universality of Sparse Function Families

In this section we give a simple result on the universality of sparse multivariate hash functions independently of the degree of the components, and deduce collisions holding with high probability.

3.1 General Case

Consider a family $\mathcal{F}$ of multivariate hash functions $\mathbb{K}^{m+n} \mapsto \mathbb{K}^n$ of density δ. Then for a random $h \in \mathcal{F}$, 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 (let's call such a x_i an *isolated variable*) with probability $(1-\delta)^n$. When this event occurs, it is easy to see that

$$h(0,\ldots,0,x_i=0,0,\ldots,0) = h(0,\ldots,0,x_i=1,0,\ldots,0) \ . \tag{3}$$

Consequently, for any such pair of inputs, a collision occurs in a random $h \in \mathcal{F}$ with probability $(1-\delta)^n$. In other words, $\mathcal{F}$ is not $(1-\delta)^n$-cAU. Moreover, by trying all possible such pairs of input, one gets at least one collision with probability

$$\rho = 1 - (1-(1-\delta)^n)^{n+m} \ . \tag{4}$$

For all the parameters of SCC proposed in [20], $\rho \approx 1$, hence with high probability at least one such collision exists.

When no isolated variable exists in the original system, one might be found in a derived system, obtained by suitably fixing values of a subset of the variables, such that there exists an isolated variable in the new system. To find a derived system with x_j isolated, one can rewrite all components as

$$h_i(x) = x_j \cdot d_i(x) + e_i(x) \ , \tag{5}$$

for polynomial functions d_i and e_i, such that $e_i(x)$ does not contain the variable x_j in any monomial. Consequently, $\deg(d_i) \leq (\deg(h)-1)$ and $\deg(e_i) \leq \deg(h)$. Consider now the system $\{d_i(x)=0\}_{0\leq i<n}$, with $(m+n-1)$ unknowns: a solution gives a valuation such that the output is independent of x_j. This is an alternative manner to find collisions by solving a system of reduced degree, equivalent to the generic collision attack.

Finally, observe that when the monomial x_i appears in exactly k equations, then $h(0,\ldots,0,x_i=0,0,\ldots,0)$ and $h(0,\ldots,0,x_i=1,0,\ldots,0)$ collide over exactly $(n-k)$ bits, thus bringing near-collisions when k is small.

3.2 Case of Even Components over GF(2)

Now consider a family of multivariate hash functions $\mathrm{GF}(2)^{m+n} \mapsto \mathrm{GF}(2)^n$ of density δ, such that δ imposes an *even* number of monomials in each component h_i. Since the constant monomial 1 appears with probability δ in a given h_i, the collision

$$h(0,\ldots,0) = h(1,\ldots,1) \tag{6}$$

will hold for a random h with probability $(1-\delta)^n$. It is a different method to see that such families are not $(1-\delta)^n$-cAU. For a random instance of SCC with 160-bit digest, the collision in Eq. (6) holds with probability 0.73, and for 256-bit digests, with probability 0.60.

When the system does not have only "even" equations, one might look for a suitable derived system where all components have an even number of non-constant monomials. One can observe that finding such a system is equivalent to finding a preimage of $h(0,\ldots,0)$.

Analogously to the previous observations, a near-collision over $(n-k)$ bit occurs when exactly k equations have an even number of non-constant monomials.

4 Solving Underdefined Semi-Sparse Systems

In a multivariate hash function, replacing a sparse system of equations by a semi-sparse one, where the density $\delta \ll 50\%$ only applies to monomials of degree > 1, avoids the weaknesses of Section 3. However collisions might be found in semi-sparse systems, as shown in the present section: we introduce a method for solving underdefined quadratic systems with density of quadratic terms $\delta \lll 50\%$, then apply it to find collisions in semi-sparse cubic systems, based on the generic attack.

4.1 Description of the Method

Consider a *random* quadratic system $h(x) = 0$ in $(m+n)$ variables with n equations, $m > 0$, such that each equation contains each degree 1 monomial with probability $1/2$, but with a density of quadratic terms δ, and null constant terms (homogeneous system). The linear system $h'(x) = 0$ obtained by removing the quadratic monomials then describes the parity-check matrix of a random linear code C of length $(m+n)$, dimension m (assuming linear independency of the equations), and unknown minimal distance $d_{\min}$. Each solution of the system $h'(x) = 0$ then corresponds to a codeword of C. The key observation is that a low-weight solution of this system will be a solution of $h(x) = 0$ if in each component the sum of all quadratic monomials happens to vanish. For a random word of weight w this latter event has probability (*cf.* piling-up lemma in Appendix)

$$\left(\frac{1}{2} + 2^{\delta\binom{m+n}{2}-1}\left|\frac{1}{2} - \frac{w}{m+n}\right|^{\delta\binom{m+n}{2}}\right)^n , \tag{7}$$

where $\delta\binom{m+n}{2}$ is the expected number of quadratic monomials in an equation.

The best algorithm known so far for finding a low-weight codeword in a random linear code [10] requires a "work factor" estimated to

$$\exp_2\left\{0.12 \times (m+n-1) H\left(\frac{m}{m+n-1} + 2^{-5}\right) + 10\right\} , \tag{8}$$

where H is the binary entropy function,

$$H(\varepsilon) = -\varepsilon\log(\varepsilon) - (1-\varepsilon)\log(1-\varepsilon) . \tag{9}$$

Before applying this algorithm, we need to compute the generating matrix of the linear code C from the parity-check matrix derived from the system $h'(x) = 0$, which adds a cost in $\mathcal{O}((m+n)^3)$.

The expected efficiency of this technique cannot be precisely established, since the distance $d_{\min}$ of the code is *a priori* unknown, as well as its expected value. Useful results are the Gilbert-Varshamov bound

$$\sum_{i=0}^{d_{\min}-2} \binom{m+n-1}{i} < 2^n \ , \tag{10}$$

and an upper bound on the number of codewords of weight $\leq \varepsilon(m+n)$, equal to $2^{(m+n)H(\varepsilon)}$.

Note that this method can also be applied to systems of degree larger than two, in which case it succeeds as soon as all the sums of monomials of degree at least two happen to vanish in each equation of the system.

4.2 Application to Multivariate Hash Functions

Consider a variant of SCC over GF(2) with $n = 160$, where the density $\delta \ll 50\%$ only applies to the *cubic* monomials. Using the generic collision attack with a differential of weight 1, a colliding pair of inputs can be computed by solving a quadratic system with in average $\delta\binom{2n-1}{2}$ quadratic monomials, inherited from the cubic monomials of the original system (since there are exactly $\binom{2n-1}{2}$ cubic monomials containing a given x_j). We then consider the system built by removing those quadratic terms, in order to apply the method of the previous subsection. The expected work factor to find a minimal weight word in the associated linear code is about 2^{48} (*cf.* Eq. (8)), and there are at most $\approx 2^{14}$ codewords of weight ≤ 40. Assume that a word with weight ≤ 40 is found. Then a collision is found for $\delta = 0.2\%$ with probability ≥ 0.0017, for $\delta = 0.1\%$ with probability ≥ 0.0402, and for $\delta = 0.05\%$ with probability ≥ 0.1988. The ratios "success probability over complexity" are then clearly higher than for a birthday attack. Nonetheless, one should be careful by mixing asymptotic estimates and assertions on concrete instances; for instance, the effective computation time of the word-finding algorithm of "work factor" of 2^{48} is probably much higher than the cost of computing 2^{48} digests.

Finally, note that we considered a homogeneous system, whereas the one we need in SCC is not necessarily; we may easily convert this system to a homogeneous one, by introducing a dummy variable X as soon as the constant 1 appears. Then the words obtained will have $X = 1$ with probability $w/(m+n)$, hence the attack has to be repeated about $(m+n)/w$ times (that is, with as many different codewords), to succeed – assuming a uniform distribution of the non-null offsets in those words. An alternative solution is to directly modify the algorithm of [10] to suit non-homogeneous systems.

5 Key Recovery for NMAC and HMAC

In this section we consider a concrete application of hash functions: we show that the message authentication codes NMAC and HMAC [5] built on multivariate hash functions can be

attacked by solving large overdefined systems. This alternative to exhaustive search directly follows from the explicit structure of such hash functions.

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

$$\mathrm{NMAC}_{k_1,k_2}(x) = h_{k_1}^\star(h_{k_2}^\star(x)) \ . \tag{11}$$

Let an attacker have access to NMAC_{k_1,k_2} as a black box. With N queries of $\mathrm{NMAC}_{k_1,k_2}(x)$ for N distinct x's long of one message block (thus for $x \in \mathbb{K}^m$), she gets nN equations in $2n$ unknowns, of degree d^{b+1}, where b is the number of message blocks of $h_{k_2}^\star(x)$. That is, $b = \lceil n/m \rceil$. If $m \geq n$, then $b = 1$, thus the key (k_1, k_2) can be recovered by solving a system of nN degree d^2 equations in $2n$ unknowns. If k_2 is known, the same observations apply to recover k_1 except that the system has now only n unknowns and degree d^b.

The HMAC construction with key k is defined by

$$\mathrm{HMAC}_k(x) = h_{\mathsf{iv}}^\star(k \oplus \mathrm{OPAD}||h_{\mathsf{iv}}^\star(k \oplus \mathrm{IPAD}||x)) \ , \tag{12}$$

with constant OPAD and IPAD long of one message block, k at most as long as the constants, and a fixed initial value $\mathsf{iv} \in \mathbb{K}^n$. The input of the outer $h_{\mathsf{iv}}^\star$ is then an element of $\mathbb{K}^{m+n}$, and the input of the inner function is an element of $\mathbb{K}^{m+|x|}$, where $|x|$ is the number of field elements of x. Hence both the inner and the outer $h_{\mathsf{iv}}^\star$ run the compression function at least twice. In this best-case scenario (when $n \leq m$), with N queries with a m-element x, one gets nN equations of degree d^3 in $|k|$ unknowns.

Are those attacks faster than exhaustive search of the key(s) ? This depends on the construction, and on the parameters chosen. For instance, for the MQ-HASH proposal we have $\mathbb{K} = \mathrm{GF}(2)$, $n/m = 160/32 = 5$, so the attack on NMAC requires to solve a system of degree 320 with 2^{32} equations in 320 unknowns, certainly hard. For NMAC-SCC with $\mathbb{K} = \mathrm{GF}(2)$ and $m = n = 160$, with $N \leq 2^{160}$ queries one gets $320N$ equations of degree 9 in 320 unknowns (k_1 and k_2). Thus with about 2^{48} queries, one obtains enough equations to solve the system by linearisation (about 2^{56} variables): this gives a complexity about $(2^{56})^3 = 2^{168}$ (higher than exhaustive search's cost 2^{160}). If $\mathbb{K} = \mathrm{GF}(2^8)$ and $m = n = 20$, with $N \leq 2^{160}$ queries one gets $40N$ equations of degree 9 in 40 unknowns. Thus with about 2^{23} queries, one obtains enough equations to solve the system by linearisation (about 2^{28} variables): complexity is then about $(2^{28})^3 \approx 2^{74}$ (smaller than exhaustive search's cost 2^{160}). However, in both cases, memory requirements for a practical implementation seem unrealistic.

Note that the complexity evaluations above are independent of the density of the system. For sparse systems, the cost of linearisation can be reduced (since certain monomials might not appear in the system), as well as other methods as F_5 or SAT-solvers can take advantage of low-density systems.

6 Pseudo-Randomness and Unpredictability

We show here that all families of low-degree multivariate hash functions over GF(2) are neither pseudo-random nor unpredictable. This is a consequence of Fact 1, holding for an arbitrary family $\mathcal{F}$ of degree d multivariate hash functions.

Fact 1 *If one is given a random $h \in \mathcal{F}$ as a black box, computing the algebraic normal form of h can be achieved in $\mathcal{N}(m+n,d)$ queries to the box.*

This obviously holds for any function family, not only multivariate ones. However for low-degree multivariate hash functions $\mathcal{N}(m+n,d)$ is much lower than for a random function, whose degree, though unknown, is maximal ($= m+n$) with almost certainty. In this case $\mathcal{N}(m+n,d) = 2^{m+n}$.

We now briefly justify Fact 1. Let us call B the challenge box with components $\{B_i\}_{0\leq i<n}$ (*a priori* unknown), 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 then recovers all the linear terms of the algebraic normal forms of the B_i's, using the knowledge of the constant terms. Once all the linear terms are known, all weight 2 queries give the quadratic monomials, which are used to deduce the list of cubic monomials, and so on, until degree d. As a consequence, for a family $\mathcal{F}$ of degree $d < m+n$, we have the following facts.

Fact 2 *Given a black box either a random $h \in \mathcal{F}$ or a random function $GF(2)^{m+n} \mapsto GF(2)^n$, one can identify the box with probability $\geq (1-2^{-n})$, $\mathcal{N}(m+n,d)$ queries to the box, and a negligible amount of computation.*

Fact 3 *Given a random $h \in \mathcal{F}$ as a black box, one can find $h(x)$, for any x of weight $> d$ without querying the box with x, with $\mathcal{N}(m+n,d)$ queries to the box, and a negligible amount of computation.*

In Fact 2, 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 $> d$. Since a random function will have an output distinct from the degree d system's with probability $\geq (1-2^{-n})$, one identifies the box with high probability. The result of Fact 3 is also quite simple: in order to find $h(x)$, one simply has to compute the algebraic normal form of the function using black box queries, then evaluates the digest of any input without an explicit query.

Consequently, all the families of MQ-HASH and SCC over GF(2) with reasonable parameters fail to be pseudo-random and unpredictable, since $\mathcal{N}(m+n,d)$ shall be much lower than 2^n. For the parameters proposed, one distinguishes a random instance of MQ-HASH and SCC from a random function within respectively $2^{25.74}$ and $2^{22.38}$ black box queries. Note that in the iterated version, the padding rule makes those techniques not applicable.

Another noteworthy property of multivariate hash functions (over an arbitrary $\mathbb{K}$) is that, given a random $x = (x_1,\ldots,x_{m+n-1})$ and a random $h \in \mathcal{F}$, one can easily find a distinct h' in $\mathcal{F}$ such that $h'(x) = h(x)$, by adding and/or removing monomials in one or several equations. Although we see no impact on security *a priori*, this property must be kept in mind when designing protocols involving multivariate hash functions.

7 Weak Instances of the Stream Ciphers QUAD

QUAD [7] is a construction of multivariate stream ciphers, based on two random quadratic systems $P : \mathbb{K}^n \mapsto \mathbb{K}^r$ (output function) and $Q : \mathbb{K}^n \mapsto \mathbb{K}^n$ (update function). Given an initial state $x_0 \in \mathbb{K}^n$ derived from a key and a nonce, the i-th internal state is $x_i = Q^i(x)$, and the i-th r-bit output is $y_i = P(x_i)$, so that $\{y_i\}_{0 \leq i}$ is the keystream of the cipher.

From the observations of Section 3, we can see that if Q contains an isolated variable, then two distinct initial states producing identical keystreams can be found, and if $P \circ Q^i$ contains an isolated variable, we can find distinct states whose keystreams collide on the i-th output block. Analogously, a trivial collision with all-zero and all-one inputs holds if $\mathbb{K} = \mathrm{GF}(2)$ when all components of Q or $P \circ Q^i$ have an even number of non-constant monomials. When $\mathbb{K} = \mathrm{GF}(2)$, the techniques of Section 6 apply as well to distinguish a random instance of QUAD from a random oracle when given as a black box fed with an initial state. Note that this is not a distinguisher in the usual sense for stream ciphers, in which the instance is known, but not the initial state.

Finally, those observations are *not threatening* for the security of QUAD, since weak instances appear with very low probability, and the distinguisher assumes as known the secret information of the cipher.

8 Conclusion

We have studied several security aspects of multivariate hash functions, both in the general case and for the specific constructions MQ-HASH and SCC. Our main results are summarised below.

- Multivariate hash functions over GF(2) of degree $\ll (m + n)$ are neither pseudo-random nor unpredictable.
- NMAC message authentication codes built on certain cubic multivariate hash functions allow key recovery faster than by exhaustive search.
- Families of multivariate hash functions of given density are not ρ-universal, for ρ given by Eq. (4).
- Constructions based on sparse systems are vulnerable to trivial collisions and near-collisions.
- Collisions in certain semi-sparse hash functions can be solved by using an efficient algorithm to find a low-weight word in a random linear code.

The first result applies to both MQ-HASH and SCC, while the second to the fourth results only apply to SCC, and the last one to a semi-sparse variant of SCC.

Further work seems necessary to establish whether multivariate hash functions can be competitive with conservative designs in terms of performances, as well as to design more elaborate constructions improving on security and/or on efficiency.

References

1. William Aiello, Stuart Haber, and Ramarathnam Venkatesan. New constructions for secure hash functions. In Serge Vaudenay, editor, *Fast Software Encryption*, volume 1372 of *Lecture Notes in Computer Science*, pages 150–167. Springer, 1998.
2. Daniel Augot, Matthieu Finiasz, and Nicolas Sendrier. A family of fast syndrome based cryptographic hash functions. In Ed Dawson and Serge Vaudenay, editors, *Mycrypt*, volume 3715 of *Lecture Notes in Computer Science*, pages 64–83. Springer, 2005.
3. Gregory V. Bard, Nicolas T. Courtois, and Chris Jefferson. Efficient methods for conversion and solution of sparse systems of low-degree multivariate polynomials over GF(2) via SAT-solvers. Cryptology ePrint Archive, Report 2007/024, 2007.
4. Mihir Bellare. New proofs for NMAC and HMAC: Security without collision-resistance. In Cynthia Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 602–619. Springer, 2006.
5. Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In Neal Koblitz, editor, *CRYPTO*, volume 1109 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 1996.
6. Kamel Bentahar, Dan Page, Markku-Juhani O. Saarinen, Joseph H. Silverman, and Nigel Smart. LASH. Second NIST Cryptographic Hash Function Workshop, 2006.
7. Côme Berbain, Henri Gilbert, and Jacques Patarin. QUAD: A practical stream cipher with provable security. In Vaudenay [39], pages 109–128.
8. Olivier Billet, Matthew J. B. Robshaw, and Thomas Peyrin. On building hash functions from multivariate quadratic equations. In Josef Pieprzyk, Hossein Ghodosi, and Ed Dawson, editors, *ACISP*, volume 4586 of *Lecture Notes in Computer Science*, pages 82–95. Springer, 2007.
9. Ran Canetti, Daniele Micciancio, and Omer Reingold. Perfectly one-way probabilistic hash functions (preliminary version). In *STOC*, pages 131–140, 1998.
10. Anne Canteaut and Florent Chabaud. A new algorithm for finding minimum-weight words in a linear code: application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511. *IEEE Transactions on Information Theory*, 1(44):367–378, 1998.
11. Scott Contini, Arjen K. Lenstra, and Ron Steinfeld. VSH, an efficient and provable collision-resistant hash function. In Vaudenay [39], pages 165–182.
12. Scott Contini, Arjen K. Lenstra, and Ron Steinfield. VSH, an efficient and provable collision-resistant hash function. Cryptology ePrint Archive, Report 2006/193. Extended version of [11].
13. Nicolas Courtois. Higher order correlation attacks, XL algorithm and cryptanalysis of Toyocrypt. In Pil Joong Lee and Chae Hoon Lim, editors, *ICISC*, volume 2587 of *Lecture Notes in Computer Science*, pages 182–199. Springer, 2002.
14. Nicolas Courtois. Algebraic attacks over GF(2^k), application to HFE challenge 2 and Sflash-v2. In Feng Bao, Robert H. Deng, and Jianying Zhou, editors, *Public Key Cryptography*, volume 2947 of *Lecture Notes in Computer Science*, pages 201–217. Springer, 2004.
15. Nicolas Courtois, Louis Goubin, Willi Meier, and Jean-Daniel Tacier. Solving underdefined systems of multivariate quadratic equations. In David Naccache and Pascal Paillier, editors, *Public Key Cryptography*, volume 2274 of *Lecture Notes in Computer Science*, pages 211–227. Springer, 2002.
16. Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In Bart Preneel, editor, *EUROCRYPT*, volume 1807 of *Lecture Notes in Computer Science*, pages 392–407. Springer, 2000.
17. Nicolas Courtois and Jacques Patarin. About the XL algorithm over GF(2). In Marc Joye, editor, *CT-RSA*, volume 2612 of *Lecture Notes in Computer Science*, pages 141–157. Springer, 2003.
18. Nicolas Courtois and Josef Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In Yuliang Zheng, editor, *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287. Springer, 2002.
19. Claus Diem. The XL-algorithm and a conjecture from commutative algebra. In Pil Joong Lee, editor, *ASIACRYPT*, volume 3329 of *Lecture Notes in Computer Science*, pages 323–337. Springer, 2004.
20. Jintai Ding and Bo-Yin Yang. Multivariates polynomials for hashing. Cryptology ePrint Archive, Report 2007/137, 2007.
21. Niklas Eén and Niklas Sörensson. MINISAT. `http://www.cs.chalmers.se/Cs/Research/FormalMethods/MiniSat/`.
22. Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases (F_4). *Journal of Pure and Applied Algebra*, 139:61, 88 1999.

23. Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases without reductions to zero (F_5). In *ISSAC*, pages 75–83. ACM Press, 2002.
24. Jean-Charles Faugère and Antoine Joux. Algebraic cryptanalysis of hidden field equation (HFE) cryptosystems using Gröbner bases. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 44–60. Springer, 2003.
25. Matthieu Finiasz, Philippe Gaborit, and Nicolas Sendrier. Improved fast syndrome based cryptographic hash functions. ECRYPT Workshop on hash functions, 2007.
26. Michael Garey and David Johnson. *Computers and Intractability, a guide to the theory of NP-completeness*, page 251. Freeman, 1979.
27. Iftach Haitner and Omer Reingold. A new interactive hashing theorem. In *IEEE Conference on Computational Complexity*, 2007.
28. Hideki Imai and Tsutomu Matsumoto. A class of asymmetric crypto-systems based on polynomials over finite rings. *IEEE International Symposium on Information Theory*, pages 131–132, 1983.
29. Vadim Lyubashevsky, Daniele Micciancio, Chris Peikert, and Alon Rosen. Provably secure FFT hashing. 2nd NIST Cryptographic Hash Function Workshop, 2006.
30. Cameron McDonald, Chris Charnes, and Josef Pieprzyk. Attacking Bivium with MiniSat. Cryptology ePrint Archive, Report 2007/129.
31. Moni Naor and Omer Reingold. From unpredictability to indistinguishability: A simple construction of pseudo-random functions from MACs (extended abstract). In Hugo Krawczyk, editor, *CRYPTO*, volume 1462 of *Lecture Notes in Computer Science*, pages 267–282. Springer, 1998.
32. NIST. Plan for new cryptographic hash functions. `http://www.nist.gov/hash-function/`.
33. H. Ong, Claus-Peter Schnorr, and Adi Shamir. Efficient signature schemes based on polynomial equations. In G. R. Blakley and David Chaum, editors, *CRYPTO*, volume 196 of *Lecture Notes in Computer Science*, pages 37–46. Springer, 1984.
34. Jacques Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): Two new families of asymmetric algorithms. In Ueli M. Maurer, editor, *EUROCRYPT*, volume 1070 of *Lecture Notes in Computer Science*, pages 33–48. Springer, 1996.
35. Claude E. Shannon. Communication theory of secrecy systems. *Bell systems technical journal*, 28:646–714, 1949.
36. Roberto Tamassia and Nikos Triandopoulos. Computational bounds on hierarchical data processing with applications to information security. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *ICALP*, volume 3580 of *Lecture Notes in Computer Science*, pages 153–165. Springer, 2005.
37. Xijin Tang and Yong Feng. A new efficient algorithm for solving systems of multivariate polynomials equations. Cryptology ePrint Archive, Report 2005/312, 2005.
38. Håvard Raddum and Igor Semaev. New technique for solving sparse equation systems. Cryptology ePrint Archive, Report 2006/475, 2006.
39. Serge Vaudenay, editor. *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*. Springer, 2006.
40. Bo-Yin Yang, Owen Chia-Hsin Chen, Daniel J. Bernstein, and Jimmy Chen. Analysis of QUAD. In Alex Biryukov, editor, *Fast Software Encryption*, Lecture Notes in Computer Science, 2007. To appear.

Appendix

Lemma 4 (Piling-up). *If $\{X_i\}_{0\leq i<n}$ is a sequence of independent binary random variables with bias respectively $\varepsilon_i = |\frac{1}{2} - P(X_i = 0)|$, $0 \leq i < n$, then*

$$P(X_0 \oplus \cdots \oplus X_{n-1} = 0) = \frac{1}{2} + 2^{n-1} \prod_{0\leq i<n} \varepsilon_i \ .$$