Analysis of multivariate hash functions

Jean-Philippe Aumasson, Willi Meier

n|w University of Applied Sciences Northwestern Switzerland
School of Engineering

$$\begin{cases} 3xy^2 + zt &= 0 \\ x^2z + 5xyt &= 0 \\ y^3 + 7z + 11t &= 0 \\ x^2t + 13yz &= 0 \end{cases}$$

Characteristics of **multivariate systems**:

- Base field: typically an extension of GF(2) for crypto.
- Nb. of unknowns n, nb. of equations m, ratio n/m.

For any field, when $n \approx m$, solving a random quadratic system is **NP-hard** (problem $\mathcal{MQ}$).

Easier for **sparse systems**

SOLVING MULTIVARIATE SYSTEMS

- **Linearization**: needs #equations $\geq$ #monomials.
- Variants of Buchberger's algorithm for Groebner bases:
  - $\mathbf{F_4}$ and $\mathbf{F_5}$ [Faugère 99, 02],
  - **XL** & co [Lazard 83, Courtois-Klimov-Patarin-Shamir 99],
- **SAT-solvers** with ANF$\leftrightarrow$SAT conversion [Massaci-Marraro 00, Courtois-Bard 06],
- Dedicated methods for under-/over-defined or sparse systems.

Ex: GF(256) system with 40 eq. and 20 unknowns, solved by **XL-Wiedemann** within $< 2^{45}$ Opteron cycles ("a few hours") [Yang-Chen-Bernstein-Chen 07].

MULTIVARIATE CRYPTOGRAPHY

Mainly **asymmetric** schemes (signature, encryption).

Pioneering works with C* [Matsumoto-Imai 88]
and HFE [Patarin 96].
Subsequent variants (PMI, QUARTZ, SFLASH, TTS, etc.), and a stream cipher construction (QUAD).

Advantages:

- **Fast** in cheap hardware and smart-cards, short signatures.
- **Reduction** to a hard problem ($\mathcal{MQ}$, IP, Minrank, etc.).

But many designs and/or instances broken with differential attacks, rank attacks, system solvers, etc.

MULTIVARIATE HASH FUNCTIONS

Merkle-Damgård construction with m-field-element message blocks and n-field-element chaining value.

Compression function

$$h : \mathbb{K}^{m+n} \mapsto \mathbb{K}^n, \; m \in \mathbb{Z}$$

explicitly defined as n algebraic equations

$$\{h_i : \mathbb{K}^{m+n} \mapsto \mathbb{K}\}_{0 \leq i < n}.$$

For a given set of parameters (m, n, degree, density, etc.) we consider **families** indexed by the equation system.

Security reduction for **preimage** only, for a **random** instance h.

(We'll also call h a "hash function".)

SECURITY DEFINITIONS

For hash function families $\mathcal{F} = \{h_{(i)}\}_i$.

Preimage

- *Input* a random function $h \in \mathcal{F}$, a random image y
- *Output* x such that $h(x) = y$

Collision

- *Input* a random function $h \in \mathcal{F}$
- *Output* x, x' such that $h(x) = h(x')$.

Family ε-**universal** if $\forall (x, x')$,

$$\Pr_{h \in \mathcal{F}}[h(x) = h(x')] \leq \varepsilon.$$

QUADRATIC HASH (DEGREE 2)

Quadratic components ($\deg(h_i) = 2$, $0 \le i < n$).

Can find **collisions efficiently** by solving the linear system

$$h(x) - h(x - \Delta) = 0$$

for an arbitrary fixed and known difference $\Delta \neq 0$.

Time cost in $\mathcal{O}(m^3)$.

Generally, finding collisions in a degree-d system essentially reduces to solving a degree-$(d-1)$ system.

SPARSE CUBIC HASH (DEGREE 3)

[Ding-Yang 07]

Cubic components ($\deg(h_i) = 3$, $0 \leq i < n$), with

$$h : \mathbb{K}^{2n} \mapsto \mathbb{K}^n$$

of fixed density $\delta = 0.1\%$ (*vs.* expected density 50% for a random system).

Low density $\Rightarrow$ less storage requirements, faster, etc.
but **no longer reduction** to a NP-hard problem.

QUARTIC HASH (DEGREE 4)

[Billet-Robshaw-Peyrin 07]

Two composed quadratic systems:

$$h = g \circ f$$

with

$$f : \mathbb{K}^{m+n} \mapsto \mathbb{K}^r,\ g : \mathbb{K}^r \mapsto \mathbb{K}^n, r > m + n.$$

Security reduction to $\mathcal{MQ}$ for preimage.

Large memory requirements,
e.g. ≈ 3 Mb for SHA-1 parameters over GF(2)

HOW SECURE IS IT ?

1. Universality and collisions for sparse systems
2. Collisions for semi-sparse systems
3. Pseudo-randomness and unpredictability
4. HMAC and NMAC

COLLISIONS IN SPARSE SYSTEMS

Key fact: for a random h of low density, there exists with high probability a collision of the form

$$h(0,\ldots,0) = h(0,\ldots,0,x_i \neq 0,0,\ldots,0).$$

Ex:

$$h(x,y,z) : \begin{cases} \mathbf{x}yz + \mathbf{x}y + z &= 0 \\ \mathbf{x}z + yz + y &= 0 \\ \mathbf{x}yz + y + z &= 0 \end{cases} \Rightarrow h(0,0,0) = h(1,0,0)$$

$\Rightarrow$ **universality** and **collision resistance** broken for sparse systems. (degree-independent.)

Solution: don't choose a low density for linear terms (**semi-sparse** systems).

COLLISIONS IN SEMI-SPARSE SYSTEMS

Consider **cubic hash** over GF(2), low density for **cubic** monomials only.

Idea: find a collision for the system **without cubic monomials**, such that the collision holds for the complete system with non-negligible probability.

COLLISIONS IN SEMI-SPARSE SYSTEMS

Algorithm for **collision search**, given a semi-sparse cubic system $h(x) = 0$:

1. Compute the (quadratic) **differential system**

$$h'(x) = h(x) - h(x - \Delta)$$

2. Remove quadratic monomials in $h'(x)$, get $h''(x)$
3. Compute the **generating matrix** of the corresponding linear code
4. Find a **low-weight word** of this code (a solution of $h''(x) = 0$)

The low-weight word will be a solution of $h'(x) = 0$ iff all sums of quadratic monomials vanish.
(A solution of $h'(x) = 0$ gives a **collision** for h)

COLLISIONS IN SEMI-SPARSE SYSTEMS

Bottleneck: find **low-weight words** in a random linear code; fastest algorithm in [Canteaut-Chabaud 98].

For realistic parameters: GF(2) system with 160 equations and 320 unknowns, density 0.1% for cubic monomials only:

$$\text{Ratio time/success } \approx 2^{52},$$

against $\approx 2^{80}$ for a birthday attack.

$\Rightarrow$ semi-sparse better than sparse systems, but still insecure.

DISTRIBUTIONS QUALITY

Definitions for function families [Naor-Reingold 98], for a **black-box** random instance h over GF(2):

- **Pseudo-randomness**: hard to distinguish from a random function.
- **Unpredictability**: for all x, hard to compute $h(x)$ without querying the box with x.

DISTRIBUTIONS QUALITY

Key fact: given h as a black box, one can **reconstruct the ANF** within

$$\sum_{i=0}^{d} \binom{m+n}{i} \text{ queries to the box,}$$

with queries of increasing weight.

$\Rightarrow$ **breaks pseudo-randomness and unpredictability** for low-degree functions

For parameters proposed of cubic and quartic functions, $< 2^{26}$ queries for both schemes.

Can fix this with some padding rule and/or output filter ?

KEY RECOVERY IN HMAC AND NMAC

$\text{HMAC}_k(x) = h(k \oplus \text{OPAD} \| h(k \oplus \text{IPAD} \| x))$
$\Rightarrow$ can get equations of **degree $\mathbf{d^3}$** ($d = \deg(h)$).

$\text{NMAC}_{k_1,k_2}(x) = h_{k_1}(h_{k_2}(x))$
$\Rightarrow$ can get equations of **degree $\mathbf{d^2}$**.

Depending on parameters, linearization and/or system solvers may outperform brute force...

Ex: **NMAC** with sparse cubics over GF(256) with 20 equations and 40 variables. 2^{23} queries are sufficient to run linearization (time cost $C \cdot 2^{74}$ vs. 2^{160} by brute force).

FIXES ?

We studied **compression functions**... can iterated hash be secured with convenient

- padding rule ?
- output filter ?
- operating mode ?
- high degree system ?

SUMMARY

Multivariate hash provide

- **speed** in HW (presumably, need benchmarks),
- **security reduction** for preimage,

but

- give no argument for **collision resistance**,
- do not provide **pseudo-random** function families,
- **sparse** equations can lead to trivial collisions,
- NMAC potentially weaker than HMAC,

Analysis of multivariate hash functions

Jean-Philippe Aumasson, Willi Meier

University of Applied Sciences Northwestern Switzerland
School of Engineering