Supposedly Hard Problems In Multivariate Cryptography

Charles Bouillaguet

Université de Versailles Saint-Quentin
Versailles, France

Séminaire CARAMEL
20 janvier 2012

The Hard Problem Underlying Multivariate Cryptography

- **RSA Encryption**:

$$y = x^e \bmod N, \qquad \text{with } x, y \in \mathbb{Z}/N\mathbb{Z}$$

- **Multivariate Quadratic Encryption**:

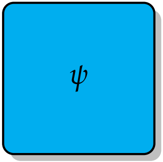
$$\begin{aligned}
\mathbf{y}_1 &= \mathbf{x}_1{}^2 + \mathbf{x}_1\mathbf{x}_3 + \mathbf{x}_2\mathbf{x}_3 + \mathbf{x}_2\mathbf{x}_4 + \mathbf{x}_3{}^2 + \mathbf{x}_3\mathbf{x}_4 + 1 \\
\mathbf{y}_2 &= \mathbf{x}_1{}^2 + \mathbf{x}_1\mathbf{x}_2 + \mathbf{x}_1\mathbf{x}_3 + \mathbf{x}_2{}^2 + \mathbf{x}_2\mathbf{x}_4 + \mathbf{x}_3{}^2 + \mathbf{x}_4{}^2 + 1 \\
\mathbf{y}_3 &= \mathbf{x}_1\mathbf{x}_2 + \mathbf{x}_1\mathbf{x}_4 + \mathbf{x}_2\mathbf{x}_3 + \mathbf{x}_2\mathbf{x}_4 + \mathbf{x}_3{}^2 + \mathbf{x}_3\mathbf{x}_4 + \mathbf{x}_4{}^2 \\
\mathbf{y}_4 &= \mathbf{x}_1\mathbf{x}_2 + \mathbf{x}_1\mathbf{x}_3 + \mathbf{x}_2{}^2 + \mathbf{x}_2\mathbf{x}_3 + \mathbf{x}_3\mathbf{x}_4 \\
&\quad \text{with } \mathbf{x}, \mathbf{y} \in \left(\mathbb{F}_q\right)^n
\end{aligned}$$

Rationale

Solving **MQ** Polynomial Systems is **NP**-hard over any field

Multivariate Quadratic Trapdoor One-Way Functions

A trapdoor must be embedded in the equations



A Common Construction: Obfuscation

1. Non-linear function $\psi : (\mathbb{F}_q)^n \to (\mathbb{F}_q)^n$
   - easily invertible, sometimes public (as in SFLASH)
2. Express it as multivariate polynomials over $(\mathbb{F}_q)^n$
3. **Obfuscate** ψ: compose with secret matrices S and T
4. **PK** $= T \circ \psi \circ S$ (the obfuscated representation of ψ)

Multivariate Quadratic Trapdoor One-Way Functions

A trapdoor must be embedded in the equations

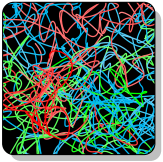
$$T \circ \psi \circ S$$

A Common Construction: **Obfuscation**

1. Non-linear function $\psi : (\mathbb{F}_q)^n \to (\mathbb{F}_q)^n$
   - easily invertible, sometimes public (as in SFLASH)
2. Express it as multivariate polynomials over $(\mathbb{F}_q)^n$
3. **Obfuscate** ψ: compose with secret matrices S and T
4. **PK** $= T \circ \psi \circ S$ (the obfuscated representation of ψ)

Multivariate Quadratic Trapdoor One-Way Functions

A trapdoor must be embedded in the equations

PK $=$ 

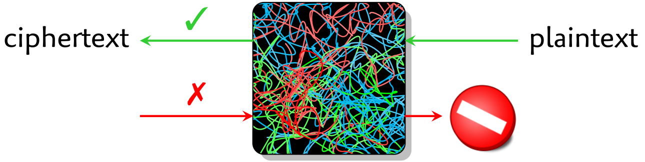
A Common Construction: **Obfuscation**

1. Non-linear function $\psi : (\mathbb{F}_q)^n \to (\mathbb{F}_q)^n$
   - easily invertible, sometimes public (as in SFLASH)
2. Express it as multivariate polynomials over $(\mathbb{F}_q)^n$
3. **Obfuscate** ψ: compose with secret matrices S and T
4. **PK** $= T \circ \psi \circ S$ (the obfuscated representation of ψ)

Multivariate Quadratic Trapdoor One-Way Functions

Is it Secure?

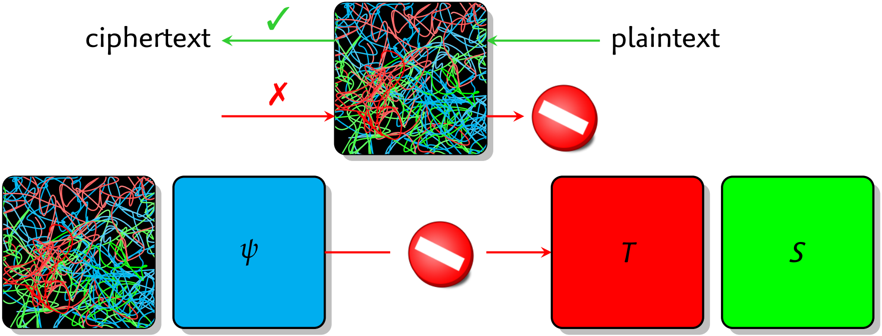
1. Public-key must be one-way
   - Even though ψ is not
   - Hardness of (a special case of) **MQ**
2. Retrieving S and T must be (very) hard
   - Hardness of **Polynomial Linear Equivalence**



Multivariate Quadratic Trapdoor One-Way Functions

Is it Secure?

1. Public-key must be one-way
   - Even though ψ is not
   - Hardness of (a special case of) **MQ**
2. Retrieving S and T must be (very) hard
   - Hardness of **Polynomial Linear Equivalence**



Examples of Constructions

1. **C***

$$\psi(X) = X^{1+q^\theta} \text{ over } \mathbb{F}_{q^n}, \text{ but quadratic over } \left(\mathbb{F}_q\right)^n$$

2. **SFLASH** (truncated **C***)
3. **H**idden **M**atrix

$$\psi(M) = M^2, \qquad M = \begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{pmatrix}$$

4. **T**ractable **R**ational **M**aps **S**ignatures
5. **M**ultivariate **Q**uadratic **Q**uasigroups
6. ℓ-**IC** signatures
7. ...

Examples of Constructions

1. **C***

$$\psi(X) = X^{1+q^\theta} \text{ over } \mathbb{F}_{q^n}, \text{ but quadratic over } \left(\mathbb{F}_q\right)^n$$

2. **SFLASH** (truncated **C***)
3. **H**idden **M**atrix

$$\psi(M) = M^2, \qquad M = \begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{pmatrix}$$

4. **T**ractable **R**ational **M**aps **S**ignatures
5. **M**ultivariate **Q**uadratic **Q**uasigroups
6. ℓ-**IC** signatures
7. ...

Examples of Constructions

1. **C***

$$\psi(X) = X^{1+q^\theta} \text{ over } \mathbb{F}_{q^n}, \text{ but quadratic over } \left(\mathbb{F}_q\right)^n$$

2. **SFLASH** (truncated **C***)
3. **H**idden **M**atrix

$$\psi(M) = M^2, \qquad M = \begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{pmatrix}$$

4. **T**ractable **R**ational **M**aps **S**ignatures
5. **M**ultivariate **Q**uadratic **Q**uasigroups
6. ℓ-**IC** signatures
7. ...

Examples of Constructions

1. **C***

$$\psi(X) = X^{1+q^\theta} \text{ over } \mathbb{F}_{q^n}, \text{ but quadratic over } \left(\mathbb{F}_q\right)^n$$

2. **SFLASH** (truncated **C***)
3. **H**idden **M**atrix

$$\psi(M) = M^2, \qquad M = \begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{pmatrix}$$

4. **T**ractable **R**ational **M**aps **S**ignatures
5. **M**ultivariate **Q**uadratic **Q**uasigroups
6. ℓ-**IC** signatures
7. ...

Examples of Constructions

1. **C***

$$\psi(X) = X^{1+q^\theta} \text{ over } \mathbb{F}_{q^n}, \text{ but quadratic over } (\mathbb{F}_q)^n$$

2. **SFLASH** (truncated **C***)
3. **H**idden **M**atrix

$$\psi(M) = M^2, \qquad M = \begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{pmatrix}$$

4. **T**ractable **R**ational **M**aps **S**ignatures
5. **M**ultivariate **Q**uadratic **Q**uasigroups
6. ℓ-**IC** signatures
7. ...

Examples of Constructions

1. **C***

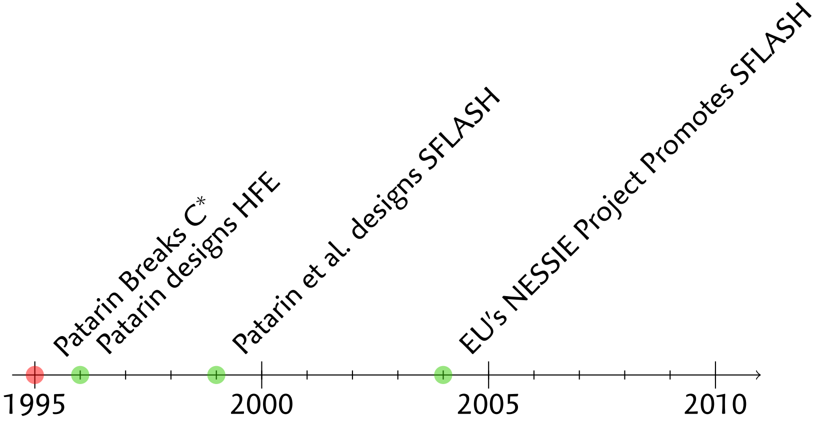
$$\psi(X) = X^{1+q^\theta} \text{ over } \mathbb{F}_{q^n}, \text{ but quadratic over } \left(\mathbb{F}_q\right)^n$$

2. **SFLASH** (truncated **C***)
3. **H**idden **M**atrix

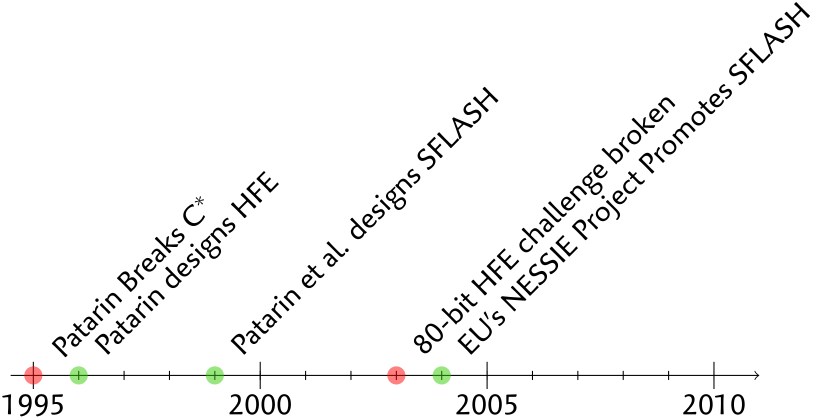
$$\psi(M) = M^2, \qquad M = \begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{pmatrix}$$

4. **T**ractable **R**ational **M**aps **S**ignatures
5. **M**ultivariate **Q**uadratic **Q**uasigroups
6. ℓ-**IC** signatures
7. ...

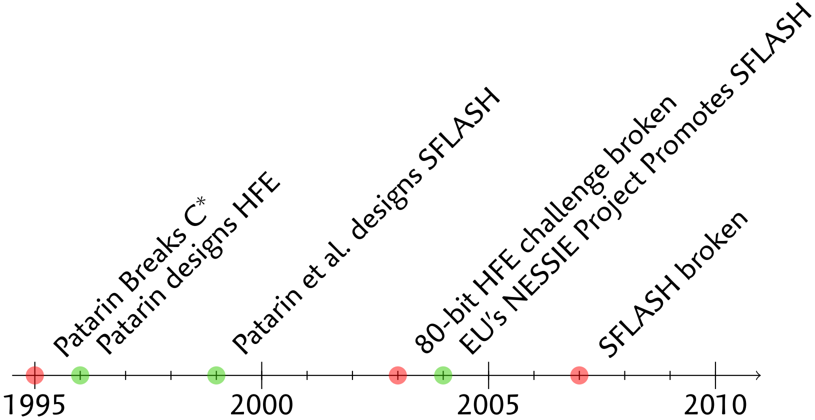
The Golden Age of Multivariate Cryptography : 1996–2007



The Golden Age of Multivariate Cryptography : 1996–2007



The Golden Age of Multivariate Cryptography : 1996–2007



Examples of Constructions

1. **C*** [Broken in 1995 !]

$$\psi(X) = X^{1+q^\theta} \text{ over } \mathbb{F}_{q^n}, \text{ but quadratic over } \left(\mathbb{F}_q\right)^n$$

2. **SFLASH** (truncated **C***) [Broken in 2007 !]
3. **H**idden **M**atrix [Broken in 2010!]

$$\psi(M) = M^2, \qquad M = \begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{pmatrix}$$

4. **T**ractable **R**ational **M**aps **S**ignatures [Broken in 2004 !]
5. **M**ultivariate **Q**uadratic **Q**uasigroups [Broken in 2009]
6. ℓ-**IC** signatures [Broken in 2009]
7. ... [They are all broken]

Why this Fiasco ?

Problems with **MQ** : the case of HFE

- MQ equations **much easier** to solve than random ones w/ Gröbner Basis algorithms (subexponential)
- **Problem** : non-random MQ instances
  - consequence of the structure of the trapdoor
- Secure parameters exist though.

Problems with **PLE** : the case of SFLASH

- non-linear function $\psi(X) = X^{1+q^\theta}$ is special
- Ad Hoc algo. solve these particular PLE instances in PTIME
- **Problem** : non-random PLE instances
  - consequence of the structure of the trapdoor

Two Options

Option A

1. Pick Your favorite multivariate scheme
2. Study the particular MQ and PLE instances it defines
3. Design special algorithms for the scheme

$\rightarrow$ If you break schemes, you're a dangerous cryptanalyst !

Option B

1. Study MQ and PLE *in general* (random instances)
2. Design generic algorithms that always work
3. Necessarily less efficient than their specialized counterparts

$\rightarrow$ Are you a harmless computer scientist ?

Two Options

Option A

1. Pick Your favorite multivariate scheme
2. Study the particular MQ and PLE instances it defines
3. Design special algorithms for the scheme

→ If you break schemes, you're a dangerous cryptanalyst !

Option B

1. Study MQ and PLE *in general* (random instances)
2. Design generic algorithms that always work
3. Necessarily less efficient than their specialized counterparts

→ Are you a harmless computer scientist ?

I'm not completely harmless

Solving Multivariate Quadratic Equations

Problem: Find $(\mathbf{x}_1, \ldots, \mathbf{x}_n) \in (\mathbb{F}_q)^n$ such that

$$\begin{cases} 1 & = \mathbf{x}_1{}^2 + \mathbf{x}_1\mathbf{x}_3 + \mathbf{x}_2\mathbf{x}_3 + \mathbf{x}_2\mathbf{x}_4 + \mathbf{x}_3{}^2 + \mathbf{x}_3\mathbf{x}_4 \\ 0 & = \mathbf{x}_1{}^2 + \mathbf{x}_1\mathbf{x}_2 + \mathbf{x}_1\mathbf{x}_3 + \mathbf{x}_2{}^2 + \mathbf{x}_2\mathbf{x}_4 + \mathbf{x}_3{}^2 + \mathbf{x}_4{}^2 \\ 0 & = \mathbf{x}_1\mathbf{x}_2 + \mathbf{x}_1\mathbf{x}_4 + \mathbf{x}_2\mathbf{x}_3 + \mathbf{x}_2\mathbf{x}_4 + \mathbf{x}_3{}^2 + \mathbf{x}_3\mathbf{x}_4 + \mathbf{x}_4{}^2 \\ 1 & = \mathbf{x}_1\mathbf{x}_2 + \mathbf{x}_1\mathbf{x}_3 + \mathbf{x}_2{}^2 + \mathbf{x}_2\mathbf{x}_3 + \mathbf{x}_3\mathbf{x}_4 \end{cases}$$

- Exhaustive search costs $\to \mathcal{O}(q^n)$
- Gröbner basis $\to \mathcal{O}(\alpha^n)$

Conclusion

- Gröbner bases should be faster on large fields (not $\mathbb{F}_2$)

Complexity of Gröbner Basis Computation

How slow are Gröbner basis computation anyway ?

$\rightarrow$ difficult to say anything sensible on the subject

- Complexity $\mathcal{O}(\alpha^n)$ over any field $\mathbb{F}_q$
- $\alpha = 16$ in simplified versions of the F_5 algorithm
- suggests that $q = 16$ is the cutoff point

Improving GB's with exhaustive search

- Combinations of GB and exhaustive search are claimed to run in time $\mathcal{O}\left(2^{0.8n}\right)$ over $\mathbb{F}_2$
- But constant factors are large...
- ...and it is slower than exhaustive search until $n \geq 200$
- **Conclusion** : over $\mathbb{F}_2$, **exhaustive search** is the way to go!

Exhaustive Search for MQ over $\mathbb{F}_2$

Let $V = (\mathbb{F}_2)^n$, and $f : V \to V$ be a quadratic map.

$$f(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=i}^{n} a_{ij} \cdot \mathbf{x}_i \mathbf{x}_j + \sum_{i=1}^{n} b_i \cdot \mathbf{x}_i + c$$

Naive Exhaustive Search

1: **for** i **from** 1 **to** 2^n **do**
2: $\quad \mathbf{x} \leftarrow V[i]$
3: $\quad \mathbf{y} \leftarrow f(\mathbf{x})$
4: $\quad$ **if** $\mathbf{y} = 0$ **then** Report $\mathbf{x}$ as solution
5: **end for**

- Evaluating f costs $\frac{n(n+3)}{2}$ XORs
- Full exhaustive search $= \mathcal{O}\left(n^2 \cdot 2^n\right)$

Exhaustive Search for MQ over $\mathbb{F}_2$: Improvement #1

Idea

Suppose I know $\mathbf{y} = f(\mathbf{x})$

$$\begin{cases} \mathbf{y}_1 & = \mathbf{x}_1{}^2 + \mathbf{x}_1\mathbf{x}_3 + \mathbf{x}_2\mathbf{x}_3 + \mathbf{x}_2\mathbf{x}_4 + \mathbf{x}_3{}^2 + \mathbf{x}_3\mathbf{x}_4 \\ \mathbf{y}_2 & = \mathbf{x}_1{}^2 + \mathbf{x}_1\mathbf{x}_2 + \mathbf{x}_1\mathbf{x}_3 + \mathbf{x}_2{}^2 + \mathbf{x}_2\mathbf{x}_4 + \mathbf{x}_3{}^2 + \mathbf{x}_4{}^2 \\ \mathbf{y}_3 & = \mathbf{x}_1\mathbf{x}_2 + \mathbf{x}_1\mathbf{x}_4 + \mathbf{x}_2\mathbf{x}_3 + \mathbf{x}_2\mathbf{x}_4 + \mathbf{x}_3{}^2 + \mathbf{x}_3\mathbf{x}_4 + \mathbf{x}_4{}^2 \\ \mathbf{y}_4 & = \mathbf{x}_1\mathbf{x}_2 + \mathbf{x}_1\mathbf{x}_3 + \mathbf{x}_2{}^2 + \mathbf{x}_2\mathbf{x}_3 + \mathbf{x}_3\mathbf{x}_4 \end{cases}$$

To “flip” $\mathbf{x}_2$, only recompute $\leq n$ terms per polynomial

$$\frac{\partial f}{\partial \mathbf{x}_2}(\mathbf{y}) = f(\mathbf{y}) + f(\mathbf{y} + \mathbf{x}_2) \text{ is } \textbf{affine} \rightarrow \text{evaluates in } \mathcal{O}(n) \text{ ops.}$$

A (Folklore) More Efficient Exaustive Search

i	$\textsc{Gray}(i)$	$b_1(i)$
0	0000	0
1	0001	1
2	0011	0
3	0010	2
4	0110	0
5	0111	1
6	0101	0
7	0100	3
8	1100	0
9	1101	1
10	1111	0
11	1110	2
12	1010	0
13	1011	1
14	1001	0

Improved Exhaustive Search

$\mathbf{x} \leftarrow 0$
$\mathbf{y} \leftarrow f(0)$
for i **from** 0 **to** $2^n - 1$ **do**
  $k \leftarrow b_1(i+1)$
  $\mathbf{z} \leftarrow \textsc{DotProduct}(\mathbf{x}, D_k)$
  $\mathbf{y} \leftarrow \mathbf{y} \oplus \mathbf{z}$
  if $\mathbf{y} = 0$ **then** Report $\mathbf{x}$ as solution
  $\mathbf{x} \leftarrow \mathbf{x} \oplus e_k$
end for

- $\textsc{DotProduct}$ costs n XORs
- Full exhaustive search $= \mathcal{O}(n \cdot 2^n)$

Exhaustive Search for MQ over $\mathbb{F}_2$: Improvement #2

i	$\textsc{Gray}(i)$	$b_1(i)$
0	0000	0
1	0001	1
2	0011	0
3	0010	2
4	0110	0
5	0111	1
6	0101	0
7	0100	3
8	1100	0
9	1101	1
10	1111	0
11	1110	2
12	1010	0
13	1011	1
14	1001	0

Theorem

If i and j are consecutive integers s.t. $b_1(i) = b_1(j)$, then $\textsc{Gray}(i)$ and $\textsc{Gray}(j)$ differ in two bits.

$$\mathbf{z} \leftarrow \textsc{DotProduct}(\mathbf{x}, D_k)$$

differ in two bits

equal

$$\mathbf{z} \leftarrow \textsc{DotProduct}(\mathbf{x}, D_k)$$

Exhaustive Search for MQ over $\mathbb{F}_2$: Improvement #2

i	$\text{GRAY}(i)$	$b_1(i)$
0	0000	0
1	0001	1
2	0011	0
3	0010	2
4	0110	0
5	0111	1
6	0101	0
7	0100	3
8	1100	0
9	1101	1
10	1111	0
11	1110	2
12	1010	0
13	1011	1
14	1001	0

Theorem

If i and j are consecutive integers s.t. $b_1(i) = b_1(j)$, then $\text{GRAY}(i)$ and $\text{GRAY}(j)$ differ in two bits.

$$\mathbf{z} \leftarrow \text{DOTPRODUCT}(\mathbf{x}, D_k)$$

$$\vdots$$

$$\mathbf{z} \leftarrow \text{DOTPRODUCT}(\mathbf{x} + 2\text{ bits}, D_k)$$

Exhaustive Search for MQ over $\mathbb{F}_2$: Improvement #2

i	Gray(i)	$b_1(i)$
0	0000	0
1	0001	1
2	0011	0
3	0010	2
4	0110	0
5	0111	1
6	0101	0
7	0100	3
8	1100	0
9	1101	1
10	1111	0
11	1110	2
12	1010	0
13	1011	1
14	1001	0

Theorem

If i and j are consecutive integers s.t. $b_1(i) = b_1(j)$, then Gray(i) and Gray(j) differ in two bits.

$$\mathbf{z}_k \leftarrow \textsc{DotProduct}(\mathbf{x}, D_k)$$

$$\vdots$$

$$\mathbf{z}_k \leftarrow \mathbf{z}_k + \textsc{DotProduct}(2 \text{ bits}, D_k)$$

A New, Even More Efficient Exaustive Search

Even More Improved Exhaustive Search

$\mathbf{x} \leftarrow 0$
$\mathbf{y} \leftarrow f(0)$
initialize the $\mathbf{z}[i]$
for i **from** 0 **to** $2^n - 1$ **do**
    $k_1 \leftarrow b_1(i+1)$
    $k_2 \leftarrow b_2(i+1)$
    $\mathbf{z}[k_1] \leftarrow \mathbf{z}[k_1] \oplus D_{k_1}[k_2]$
    $\mathbf{y} \leftarrow \mathbf{y} \oplus \mathbf{z}[k_1]$
    if $\mathbf{y} = 0$ **then** Report $\textsc{Gray}(i)$ as solution
end for

- Each iteration costs 2 XORs
- Full exhaustive search $= \mathcal{O}(2^n)$

Other Improvements

This generalizes to **degree d**

- Evaluating each polynomial required **d** XORs

This generalizes to several polynomials

- Just enumerate them all in an SIMD fashion (very efficient)

→ In fact, enumerate 32 of them (good for registers)

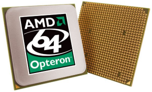
→ Then test the others against their common zeroes

This is easily parallelizable

- optimization: Synchronize the parallel process

→ they fetch the same data at the same time

Efficient Implementation(s)

			
# core	2×4	2×4	480
GHz	2.3	2.26	1.25
degree 2 cycles/iteration $n = 48$?	0.37 1h35	0.52 2h22	2.69 21 min
degree 3 cycles/iteration $n = 48$?	0.62 2h35	0.98 4h00	4.57 36 min
degree 4 cycles/iteration $n = 48$?	0.89 3h45	1.32 5h35	15.97 2h06min

What About 80-bit Security?

80-bit Security

- Not so long ago, it was considered a “decent” level
- 80 quadratic eq. in 80 $\mathbb{F}_2$-variables offer 80 bits of security

- world 3rd fastest computer
- Nat. Center for Comp. Sciences
- 224 256 $\times$  @ 2.6GHz
- Solves the problem in $\approx$ 18 years

Better results possible with more ad hoc hardware

Summer Project

Outrageous Claim

As of today, my code is the fastest way to solve arbitrary systems of boolean equations over $\mathbb{F}_2$, when this can be done in practice.

...but only I have it.

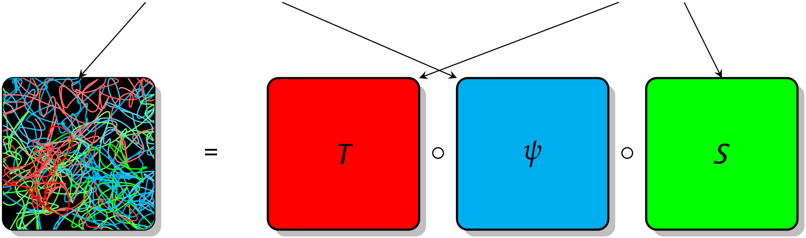
Intern Wanted

- Having it in SAGE would be great
- It's probably not so complicated
- but I can't find the time...

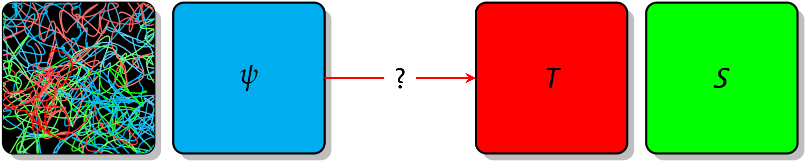
Polynomial Equivalence Problems

vectors of n multivariate quadratic polynomials in n variables

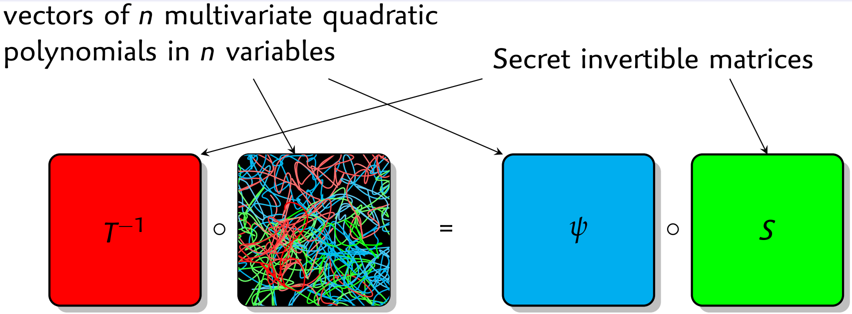
Secret invertible matrices



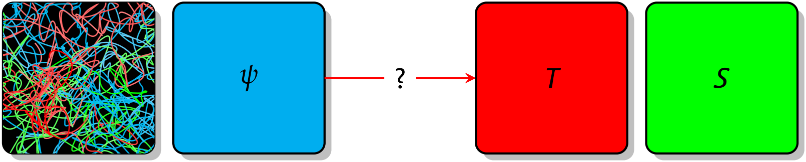
The Problem:



Polynomial Equivalence Problems



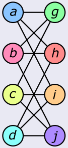
The Problem:



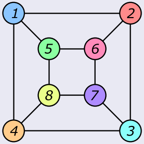
Complexity-Theoretic Status of PLE

Could **PLE** be Solvable in Deterministic Polynomial Time ?

Courtois-Goubin-Patarin, 1998 : **Graph Isomorphism** $\leq$ **PLE**



- Transform instances of **GI** into **PLE**
- 99.999999% sure that **PLE** $\notin$ **P**



Is it **NP**-hard?

Courtois-Goubin-Patarin, 1998 + Faugère-Perret, 2006 : No !

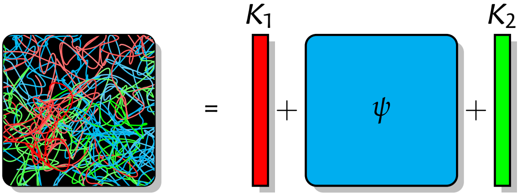
$\longrightarrow$ This does not mean that all instances are hard

Similarity With the Even-Mansour Cipher

PLE looks a lot like the **Even-Mansour Cipher**

- turn a single random permutation ψ into a block cipher

$\rightarrow$ XOR two secret keys before and after ψ

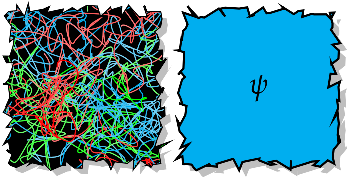


Provable Security

- Adversary queries the EM cipher (resp. psi) X times
- And queries ψ Y times
- Cannot tell EM apart from an ideal cipher if $XY < 2^n$

Easy and Hard Cases

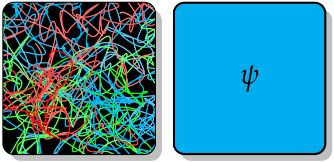
Inhomogeneous Case



$$f(\mathbf{x}) = \sum_{i=1}^{n}\sum_{j=i}^{n} a_{ij}\cdot \mathbf{x}_i\mathbf{x}_j + \sum_{i=1}^{n} b_i \cdot \mathbf{x}_i + c$$

- Gröbner-based = $\mathcal{O}\left(n^9\right)$
- "Differential" = $\mathcal{O}\left(n^6\right)$
- Inversion-free To-n-Fro = $\mathcal{O}\left(n^3\right)$

Homogeneous Case



$$f(\mathbf{x}) = \sum_{i=1}^{n}\sum_{j=i}^{n} a_{ij}\cdot \mathbf{x}_i\mathbf{x}_j$$

The Inhomogeneous Case

Strategy

build a **matrix pencil equivalence problem**:

$$T \times (\lambda \cdot A + \mu \cdot B) = (\lambda \cdot C + \mu \cdot D) \times S$$

Why is inhomogeneousness helpful ?

1. Slice ζ and ψ in homogeneous components

$$\zeta = \underbrace{\zeta^{(2)}}_{\textit{quadratic}} + \underbrace{\zeta^{(1)}}_{\textit{linear}} + \underbrace{\zeta^{(0)}}_{\textit{constant}}$$

2. S and T act **separately** on the homogeneous components

$$T \circ \zeta^{(2)} = \psi^{(2)} \circ S \quad \underbrace{T \cdot \zeta^{(1)} = \psi^{(1)} \cdot S}_{\text{linear equations}} \quad \underbrace{T \cdot \zeta^{(0)} = \psi^{(0)}}_{T \text{ known on a point}}$$

The Inhomogeneous Case

Strategy

build a **matrix pencil equivalence problem**:

$$T \times (\lambda \cdot A + \mu \cdot B) = (\lambda \cdot C + \mu \cdot D) \times S$$

Why is inhomogeneousness helpful ?

1. Slice ζ and ψ in homogeneous components

$$\zeta = \underbrace{\zeta^{(2)}}_{\textit{quadratic}} + \underbrace{\zeta^{(1)}}_{\textit{linear}} + \underbrace{\zeta^{(0)}}_{\textit{constant}}$$

2. S and T act **separately** on the homogeneous components

$$T \circ \zeta^{(2)} = \psi^{(2)} \circ S \quad \underbrace{T \cdot \zeta^{(1)} = \psi^{(1)} \cdot S}_{\text{linear equations}} \quad \underbrace{T \cdot \zeta^{(0)} = \psi^{(0)}}_{T \text{ known on a point}}$$

A Nice Tool for Multivariate Cryptanalysis

Switching to the Differential

1. Define the "**Differential**" (bilinear symmetric map):

$$\begin{array}{rccl} \mathrm{D}\psi: & (\mathbb{F}_q)^n \times (\mathbb{F}_q)^n & \to & (\mathbb{F}_q)^n \\ & (\mathbf{x},\mathbf{y}) & \mapsto & \psi(\mathbf{x}+\mathbf{y}) - \psi(\mathbf{x}) - \psi(\mathbf{y}) + \psi(0) \end{array}$$

2. Define the "Diffential in $\mathbf{x}_0$" : $\mathrm{D}_{\mathbf{x}_0}\psi(\mathbf{y}) = \mathrm{D}\psi(\mathbf{x}_0,\mathbf{y})$.
3. $\mathrm{D}_{\mathbf{x}_0}\psi$ is an endomorphism of $(\mathbb{F}_q)^n$ (i.e. a matrix).

$$T \circ \zeta = \psi \circ S \xrightarrow{\text{Differential}} T \times \mathrm{D}_{\mathbf{x}_0}\zeta = \mathrm{D}_{S\cdot\mathbf{x}_0}\psi \times S$$

Problem

We need to know the image of S on a point...

A Nice Tool for Multivariate Cryptanalysis

Switching to the Differential

1. Define the "**Differential**" (bilinear symmetric map):

$$\begin{array}{rccl} \mathrm{D}\psi : & (\mathbb{F}_q)^n \times (\mathbb{F}_q)^n & \rightarrow & (\mathbb{F}_q)^n \\ & (\mathbf{x}, \mathbf{y}) & \mapsto & \psi(\mathbf{x}+\mathbf{y}) - \psi(\mathbf{x}) - \psi(\mathbf{y}) + \psi(0) \end{array}$$

2. Define the "Diffential in $\mathbf{x}_0$" : $\mathrm{D}_{\mathbf{x}_0}\psi(\mathbf{y}) = \mathrm{D}\psi(\mathbf{x}_0, \mathbf{y})$.
3. $\mathrm{D}_{\mathbf{x}_0}\psi$ is an endomorphism of $(\mathbb{F}_q)^n$ (i.e. a matrix).

$$T \circ \zeta = \psi \circ S \xrightarrow{\text{Differential}} T \times \mathrm{D}_{\mathbf{x}_0}\zeta = \mathrm{D}_{S\cdot\mathbf{x}_0}\psi \times S$$

Problem

We need to know the image of S on a point...

Combining our Forces

$$\underbrace{T \cdot \zeta^{(1)} = \psi^{(1)} \cdot S}_{\text{linear equations}} \qquad \underbrace{T \cdot \zeta^{(0)} = \psi^{(0)}}_{T \text{ known on a point}}$$

Transfer relation from T to S

1. Assume that there are $\mathbf{x}_0$ and $\mathbf{y}_0$ such that

$$\zeta^{(1)} \cdot \mathbf{x}_0 = \zeta^{(0)} \qquad \psi^{(1)} \cdot \mathbf{y}_0 = \psi^{(0)}$$

2. Then:

$$\begin{aligned} T \cdot \zeta^{(0)} &= \psi^{(0)} && T \text{ known on a point} \\ \left[T \times \zeta^{(1)}\right] \cdot \mathbf{x}_0 &= \psi^{(0)} && \\ \left[\psi^{(1)} \times S\right] \cdot \mathbf{x}_0 &= \psi^{(0)} && \text{linear equations} \\ S \cdot \mathbf{x}_0 &= \mathbf{y}_0 && \end{aligned}$$

And the Pencil is Here

$$T \times \left(\lambda \cdot \zeta^{(1)} + \mu \cdot \mathrm{D}_{\mathbf{x}_0}\zeta\right) = \left(\lambda \cdot \psi^{(1)} + \mu \cdot \mathrm{D}_{\mathbf{y}_0}\psi\right) \times S$$

Necessary Conditions

1. $\zeta^{(0)} \neq 0$
2. $\exists \mathbf{x}_0$ s.t. $\zeta^{(1)} \cdot \mathbf{x}_0 = \zeta^{(0)}$

Random instances meet them with macroscopic prob. ($\geq 1/4$)

Why go through this hassle?

Pencil $\rightarrow$ S and T live in a subspace of dimension $\approx n$

And the Pencil is Here

$$T \times \left(\lambda \cdot \zeta^{(1)} + \mu \cdot \mathrm{D}_{\mathbf{x}_0}\zeta\right) = \left(\lambda \cdot \psi^{(1)} + \mu \cdot \mathrm{D}_{\mathbf{y}_0}\psi\right) \times S$$

Necessary Conditions

1. $\zeta^{(0)} \neq 0$
2. $\exists \mathbf{x}_0$ s.t. $\zeta^{(1)} \cdot \mathbf{x}_0 = \zeta^{(0)}$

Random instances meet them with macroscopic prob. ($\geq 1/4$)

Why go through this hassle?

Pencil $\rightarrow$ S and T live in a subspace of dimension $\approx n$

Concluding step

$$T = \sum_{i=1}^{n} T_i \cdot X_i \qquad S = \sum_{i=1}^{n} S_i \cdot X_i$$

Identify coefficient-wise

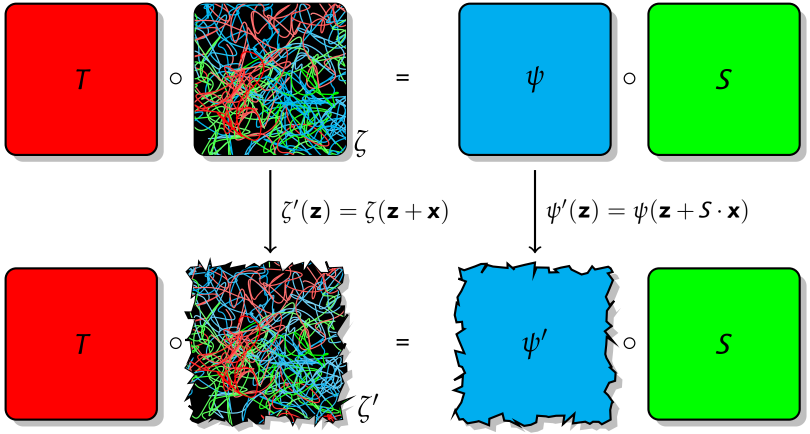
$$T \circ \zeta = \psi \circ S$$

- n equalities between quadratic polynomials
- $\approx n^2$ monomials in each polynomial

$\rightarrow$ $\approx n^3$ quadratic equations in $X_1, \ldots, X_n$

- Gauss-reduce the quadratic equations in time $\mathcal{O}\left(n^6\right)$
- Find the values of all the monomials, including the X_i

Dehomogenization

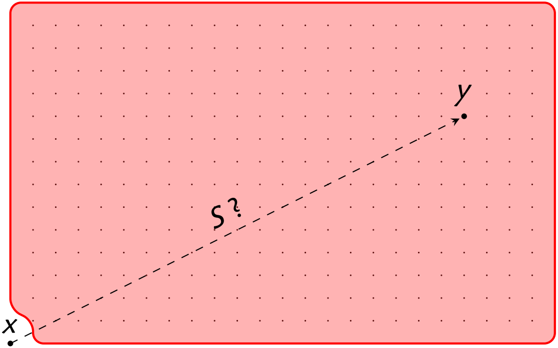


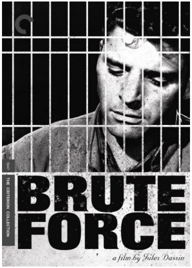
Finding the Image of S on One Point

Efficient Algorithms available...

... Once the image of S is known on one point

- Exhaustive Search $\rightarrow q^n$ trials...
- Natural approach: birthday paradox





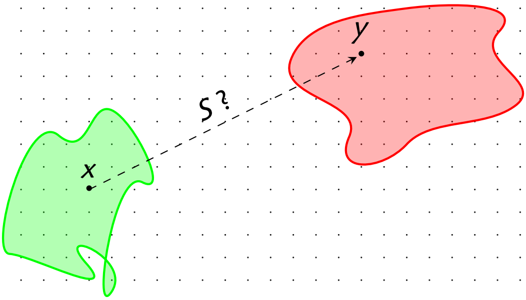
- Try pairs (x, y)
- Assume $y = S \cdot x$
- Dehomogenize
- Solution found?

Finding the Image of S on One Point

Efficient Algorithms available...

... Once the image of S is known on one point

- Exhaustive Search $\rightarrow q^n$ trials...
- Natural approach: birthday paradox



- Try pairs (x, y)
- Assume $y = S \cdot x$
- Dehomogenize
- Solution found?

Machinery

A Key Tool for Multivariate Cryptanalysis

Given a quadratic map $\phi : (\mathbb{F}_q)^n \to (\mathbb{F}_q)^n$, its **differential** is:

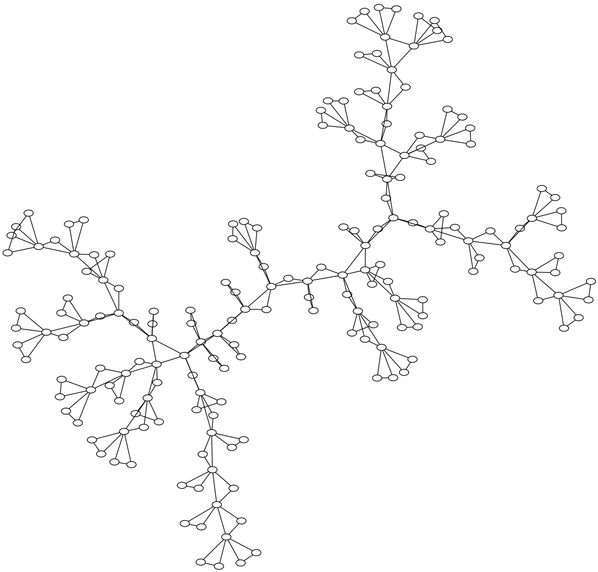
$$\begin{array}{rccc} \mathrm{D}\phi : & (\mathbb{F}_q)^n \times (\mathbb{F}_q)^n & \to & (\mathbb{F}_q)^n \\ & (\mathbf{x}, \mathbf{y}) & \mapsto & \phi(\mathbf{x}+\mathbf{y}) - \phi(\mathbf{x}) - \phi(\mathbf{y}) + \phi(0) \end{array}$$

$\mathrm{D}\phi$ is a symmetric bilinear map.

From any Quadratic Map ϕ We Define a **Undirected Graph** G_ψ:

- Vertices: $(\mathbb{F}_q)^n - \{0\}$
- Edges: $\{\mathbf{x} \leftrightarrow \mathbf{y} \mid \mathrm{D}\phi(\mathbf{x}, \mathbf{y}) = 0\}$

Machinery



Machinery

A Key Tool for Multivariate Cryptanalysis

Given a quadratic map $\phi : (\mathbb{F}_q)^n \to (\mathbb{F}_q)^n$, its **differential** is:

$$\begin{array}{rccl} \mathrm{D}\phi : & (\mathbb{F}_q)^n \times (\mathbb{F}_q)^n & \to & (\mathbb{F}_q)^n \\ & (\mathbf{x}, \mathbf{y}) & \mapsto & \phi(\mathbf{x}+\mathbf{y}) - \phi(\mathbf{x}) - \phi(\mathbf{y}) + \phi(0) \end{array}$$

$\mathrm{D}\phi$ is a symmetric bilinear map.

From any Quadratic Map ϕ We Define a Undirected Graph G_ψ:

- Vertices: $(\mathbb{F}_q)^n - \{0\}$
- Edges: $\{\mathbf{x} \leftrightarrow \mathbf{y} \mid \mathrm{D}\phi(\mathbf{x}, \mathbf{y}) = 0\}$

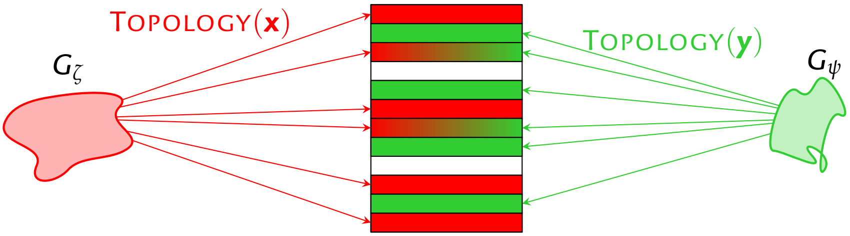
If $T \circ \zeta = \psi \circ S$, then...

S is a **Graph Isomorphism** that sends G_ζ to G_ψ.

Topological Hashing

S is a **Graph Isomorphism** that sends G_ζ to G_ψ

- **x and $(S \cdot \mathbf{x})$ have neighborhoods of the same "shape"**



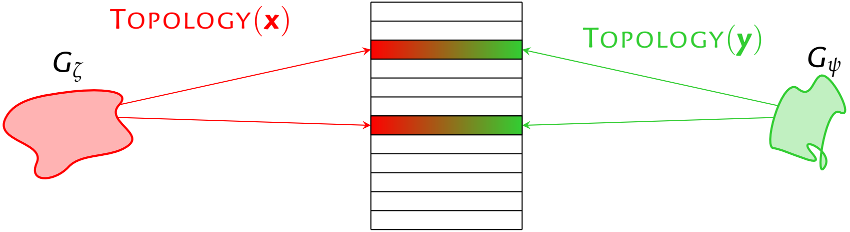
"Topological Meet-in-the middle" Algorithm

- **Sample** random points **x** in G_ζ, **store** $\textsc{Topology}(\mathbf{x}) \mapsto \mathbf{x}$
- **Sample** random points **y** in G_ψ, **store** $\textsc{Topology}(\mathbf{y}) \mapsto \mathbf{y}$
- **for all** colliding pairs, **assume** $\mathbf{y} = S \cdot \mathbf{x}$, **dehomogenize**, etc.

Topological Hashing

S is a **Graph Isomorphism** that sends G_ζ to G_ψ

- **x and $(S \cdot \mathbf{x})$ have neighborhoods of the same "shape"**



"Topological Meet-in-the middle" Algorithm

- **Sample** random points $\mathbf{x}$ in G_ζ, **store** $\textsc{Topology}(\mathbf{x}) \mapsto \mathbf{x}$
- **Sample** random points $\mathbf{y}$ in G_ψ, **store** $\textsc{Topology}(\mathbf{y}) \mapsto \mathbf{y}$
- **for all** colliding pairs, **assume** $\mathbf{y} = S \cdot \mathbf{x}$, **dehomogenize**, etc.

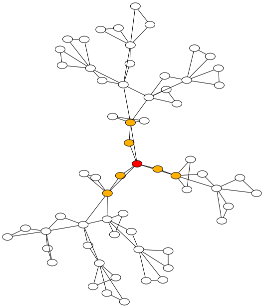
Topological Hashing: Extracting Little Information

Problem

Deterministically extract topological information?

Simple Solution

$$\textsc{Topology}(\mathbf{x}) \;\approx\; \#\text{adjacent vertices}$$

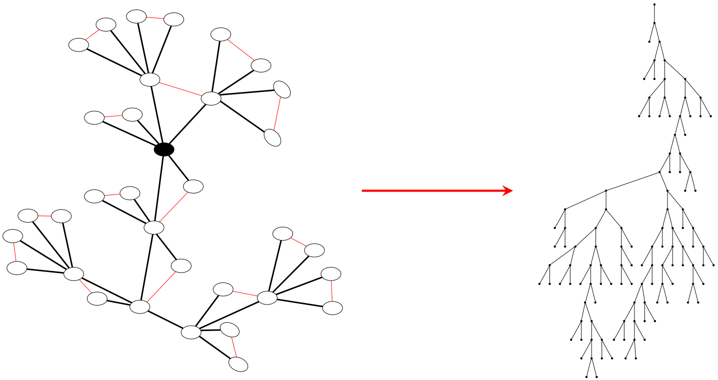


- Sample $q^{n/3}$ points in both G_ζ and G_ϕ
- Running time $\mathcal{O}\left(q^{2n/3}\right)$, success probability close to 1

Topological Hashing: Extracting Much More Information

Graphs are very sparse

- Tree-like (besides the small triangles)
- Kill the triangles $\rightarrow$ actual tree (BFS, no backwards edges)
- The topology of trees is easy to encode



Topological Hashing: Extracting Much More Information

Complicated Solution

$$\textsc{Topology}(\mathbf{x}) \approx \text{Tree-encoding (depth } n \log n\text{)}$$

- Sample $q^{n/2}$ points with "deep" neighborhoods

Theorem

If the trees are random and independent, then $\mathcal{O}(1)$ collisions
(prob. of "accidental" collision negligible, even with exponentially many trees)

- Running time $\mathcal{O}(q^{n/2})$, success probability close to 1

Conclusion

1. The **MQ** problem
   - Faster exhaustive search over $\mathbb{F}_2$
     - $\mathcal{O}\left(n^2 \cdot 2^n\right) \rightarrow \mathcal{O}\left(n \cdot 2^n\right) \rightarrow \mathcal{O}\left(2^n\right)$
   - **80-bit challenge not strictly out of reach**
2. The **PLE** problem
   - Faster polynomial algorithms for the inhomogeneous case
     - $\mathcal{O}\left(n^9\right) \rightarrow \mathcal{O}\left(n^6\right) \rightarrow \mathcal{O}\left(n^3\right)$
   - First working birthday algorithm for the homogeneous case
     - $\mathcal{O}\left(q^{3n}\right) \rightarrow \mathcal{O}\left(q^n\right) \rightarrow \mathcal{O}\left(q^{2n/3}\right) \rightarrow \mathcal{O}\left(q^{n/2}\right)$
     - Currently known to work over $\mathbb{F}_2$, extension seems easy
   - **The "obfuscation" technique is probably a bad idea**

And...

Thank You