Multivariate Cryptography

Postquantum Crypto Minischool

Ruben Niederhagen

July 12, 2022

Quantum Computing

Introduction

History:

1981	Feynman introduces “quantum simulation”.

Introduction

History:

1981	Feynman introduces “quantum simulation”.
1985	Universal “quantum computer” proposed by Deutsch.

Introduction

History:

1981	Feynman introduces "quantum simulation".
1985	Universal "quantum computer" proposed by Deutsch.
1994/96	First practically relevant algorithms by Shor and Grover.
1997	First practical experiments.

Introduction

History:

1981	Feynman introduces "quantum simulation".
1985	Universal "quantum computer" proposed by Deutsch.
1994/96	First practically relevant algorithms by Shor and Grover.
1997	First practical experiments.
1998	First two-qubit quantum computer.

Introduction

History:

1981	Feynman introduces “quantum simulation”.
1985	Universal “quantum computer” proposed by Deutsch.
1994/96	First practically relevant algorithms by Shor and Grover.
1997	First practical experiments.
1998	First two-qubit quantum computer.
2001	First seven-qubit quantum computer.

Introduction

History:

1981	Feynman introduces "quantum simulation".
1985	Universal "quantum computer" proposed by Deutsch.
1994/96	First practically relevant algorithms by Shor and Grover.
1997	First practical experiments.
1998	First two-qubit quantum computer.
2001	First seven-qubit quantum computer.
|	Many technical and theoretical improvements.

Introduction

History:

1981	Feynman introduces “quantum simulation”.
1985	Universal “quantum computer” proposed by Deutsch.
1994/96	First practically relevant algorithms by Shor and Grover.
1997	First practical experiments.
1998	First two-qubit quantum computer.
2001	First seven-qubit quantum computer.
|	Many technical and theoretical improvements.
2012	D-Waves 84 qubit quantum annealer (non-universal).

Introduction

History:

1981	Feynman introduces “quantum simulation”.
1985	Universal “quantum computer” proposed by Deutsch.
1994/96	First practically relevant algorithms by Shor and Grover.
1997	First practical experiments.
1998	First two-qubit quantum computer.
2001	First seven-qubit quantum computer.
|	Many technical and theoretical improvements.
2012	D-Waves 84 qubit quantum annealer (non-universal).
|	Further technical improvements.

Introduction

History:

1981	Feynman introduces “quantum simulation”.
1985	Universal “quantum computer” proposed by Deutsch.
1994/96	First practically relevant algorithms by Shor and Grover.
1997	First practical experiments.
1998	First two-qubit quantum computer.
2001	First seven-qubit quantum computer.
|	Many technical and theoretical improvements.
2012	D-Waves 84 qubit quantum annealer (non-universal).
|	Further technical improvements.
May 2017	IBM introduces 17-qubit quantum computer.

Introduction

History:

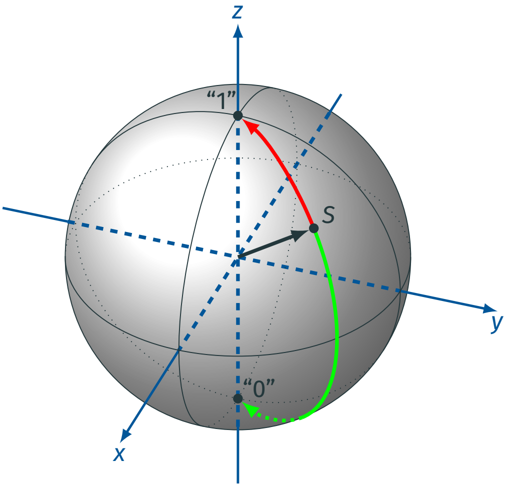
1981	Feynman introduces “quantum simulation”.
1985	Universal “quantum computer” proposed by Deutsch.
1994/96	First practically relevant algorithms by Shor and Grover.
1997	First practical experiments.
1998	First two-qubit quantum computer.
2001	First seven-qubit quantum computer.
|	Many technical and theoretical improvements.
2012	D-Waves 84 qubit quantum annealer (non-universal).
|	Further technical improvements.
May 2017	IBM introduces 17-qubit quantum computer.
March 2018	Google announces 72-qubit quantum computer.

Introduction

History:

1981	Feynman introduces "quantum simulation".
1985	Universal "quantum computer" proposed by Deutsch.
1994/96	First practically relevant algorithms by Shor and Grover.
1997	First practical experiments.
1998	First two-qubit quantum computer.
2001	First seven-qubit quantum computer.
|	Many technical and theoretical improvements.
2012	D-Waves 84 qubit quantum annealer (non-universal).
|	Further technical improvements.
May 2017	IBM introduces 17-qubit quantum computer.
March 2018	Google announces 72-qubit quantum computer.
Jan. 2019	First commercial quantum computer "IBM Q System One".

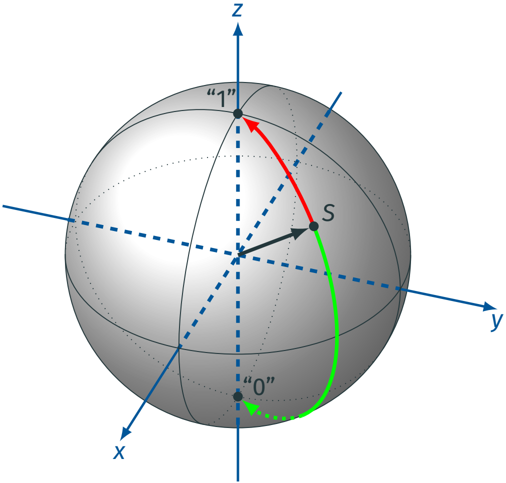
Qubits: Superposition



Visualization:

- Point on the surface of a sphere.
- At measurement (in regard to some base), the qubit “snaps” into position “0” or “1”.

Qubits: Superposition



Visualization:

- Point on the surface of a sphere.
- At measurement (in regard to some base), the qubit "snaps" into position "0" or "1".

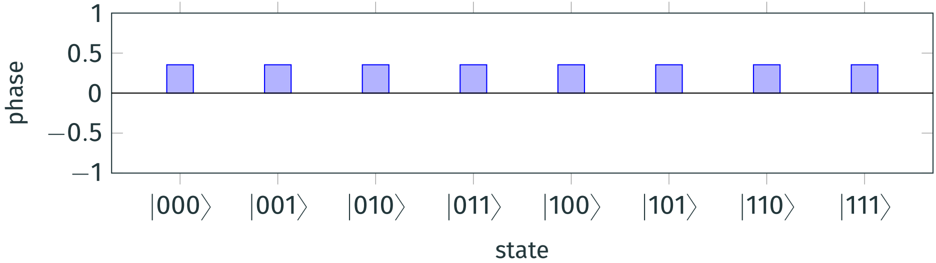
Mathematical:

- Two-dimensional complex vector space,
- written in Braket-Notation, e.g, $|1\rangle$, $|0\rangle$, $\sqrt{\frac{1}{2}}(|0\rangle + |1\rangle)$.

Qubits: Entanglement

Example – System of 3 qubits:

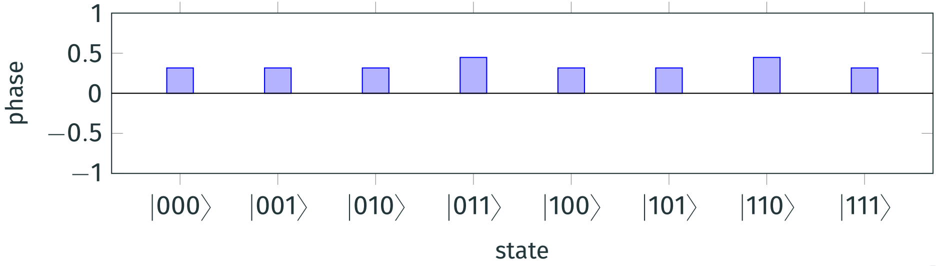
	$\|000\rangle$	$\|001\rangle$	$\|010\rangle$	$\|011\rangle$	$\|100\rangle$	$\|101\rangle$	$\|110\rangle$	$\|111\rangle$
probability	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125
	0.1	0.1	0.1	0.2	0.1	0.1	0.2	0.1
	0.1	0.1	0.1	0.1	0.1	0.1	0.3	0.1



Qubits: Entanglement

Example – System of 3 qubits:

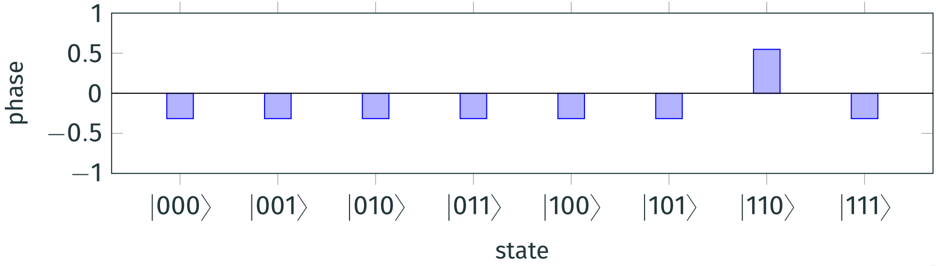
	$\|000\rangle$	$\|001\rangle$	$\|010\rangle$	$\|011\rangle$	$\|100\rangle$	$\|101\rangle$	$\|110\rangle$	$\|111\rangle$
probability	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125
	0.1	0.1	0.1	0.2	0.1	0.1	0.2	0.1
	0.1	0.1	0.1	0.1	0.1	0.1	0.3	0.1



Qubits: Entanglement

Example – System of 3 qubits:

	$\|000\rangle$	$\|001\rangle$	$\|010\rangle$	$\|011\rangle$	$\|100\rangle$	$\|101\rangle$	$\|110\rangle$	$\|111\rangle$
probability	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125
	0.1	0.1	0.1	0.2	0.1	0.1	0.2	0.1
	0.1	0.1	0.1	0.1	0.1	0.1	0.3	0.1



Quantum Algorithms: Grover

Grover's Algorithm:

- Search in "unsorted database" of N entries in $O(\sqrt{N})$ steps.

Quantum Algorithms: Grover

Grover's Algorithm:

- Search in "unsorted database" of N entries in $O(\sqrt{N})$ steps.
- Find n-bit key using $O(\sqrt{2^n}) = O(2^{n/2})$ instead of $O(2^n)$ operations.

Quantum Algorithms: Grover

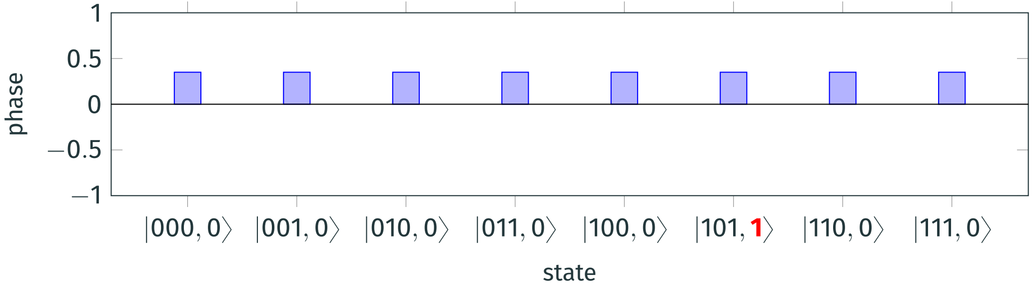
Grover's Algorithm:

- Search in "unsorted database" of N entries in $O(\sqrt{N})$ steps.
- Find n-bit key using $O(\sqrt{2^n}) = O(2^{n/2})$ instead of $O(2^n)$ operations.
- Quadratic speedup of brute-force attacks.

Quantum Algorithms: Grover

Main steps:

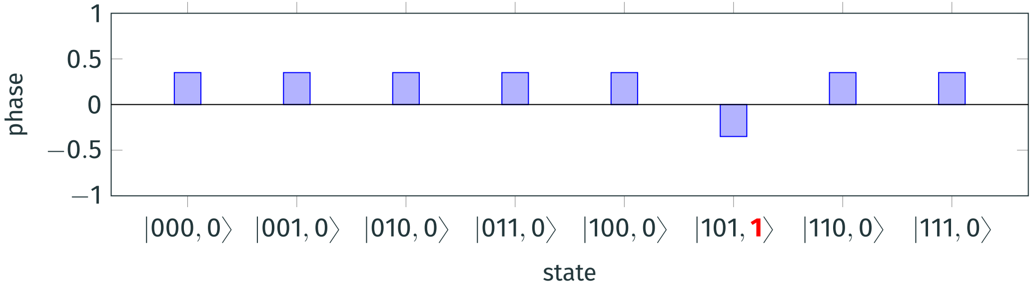
- **Phase Inversion:**
  Invert the phase of states based on a control bit.



Quantum Algorithms: Grover

Main steps:

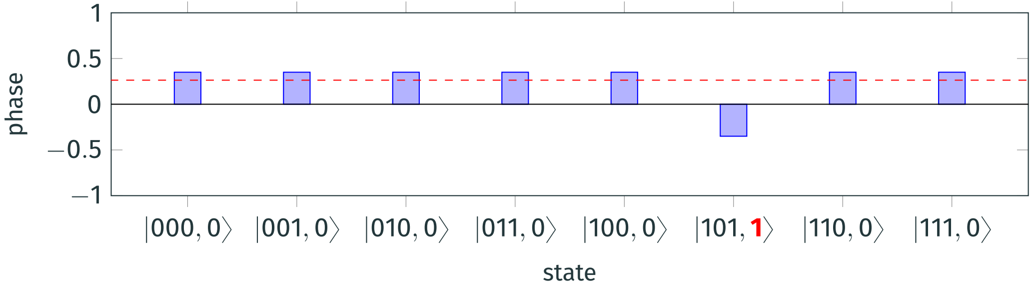
- **Phase Inversion:**
  Invert the phase of states based on a control bit.



Quantum Algorithms: Grover

Main steps:

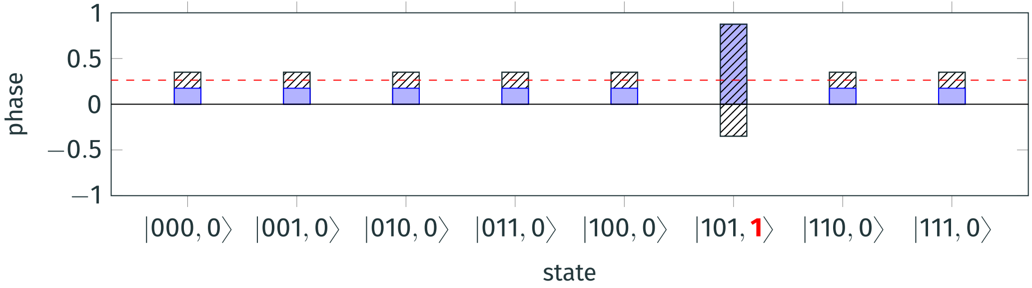
- **Phase Inversion:**
  Invert the phase of states based on a control bit.
- **Inversion about the Average:**
  Invert the complex phase around the average.



Quantum Algorithms: Grover

Main steps:

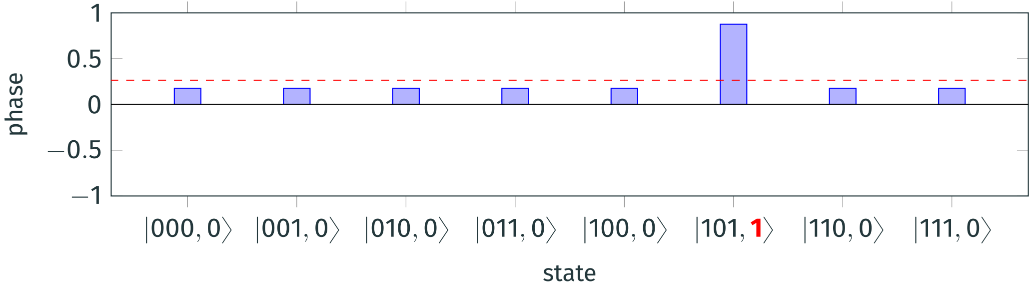
- **Phase Inversion:**
  Invert the phase of states based on a control bit.
- **Inversion about the Average:**
  Invert the complex phase around the average.



Quantum Algorithms: Grover

Main steps:

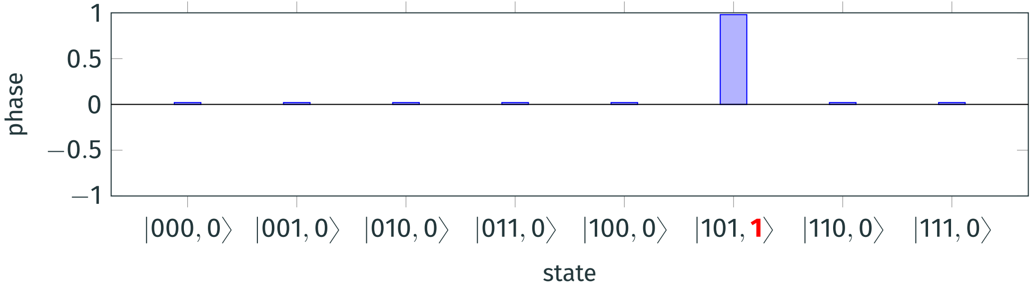
- **Phase Inversion:** Invert the phase of states based on a control bit.
- **Inversion about the Average:** Invert the complex phase around the average.



Quantum Algorithms: Grover

Main steps:

- **Phase Inversion:** Invert the phase of states based on a control bit.
- **Inversion about the Average:** Invert the complex phase around the average.
- **Repeat $\sqrt{2^n}$ times!** This gives the quadratic speedup.



Quantum Algorithms: Grover

Approach:

Implement the problem as function $f : (x_1, \ldots, x_n) \mapsto y$
with $f(\vec{x_l}) = 1$ for the unknown "correct" input $\vec{x_l}$
and $f(\vec{x}) = 0$ for all other inputs $\vec{x}$.

Quantum Algorithms: Grover

Approach:

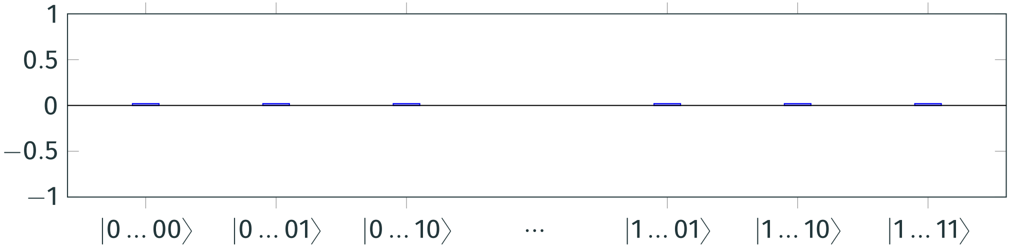
Implement the problem as function $f : (x_1, \ldots, x_n) \mapsto y$
with $f(\vec{x_l}) = 1$ for the unknown "correct" input $\vec{x_l}$
and $f(\vec{x}) = 0$ for all other inputs $\vec{x}$.

Grover uses f as sub-function; f is called $\sqrt{2^n}$ times.
At the beginning all n qubits are in equally distributed superposition,
at the end the correct solution $\vec{x_l}$ is measured with high probability.

Quantum Algorithms: Grover

Example: $f(x) \mapsto$ AES128("<DOCTYPE html>", x) = 0x45 0x59 ... 0xA1

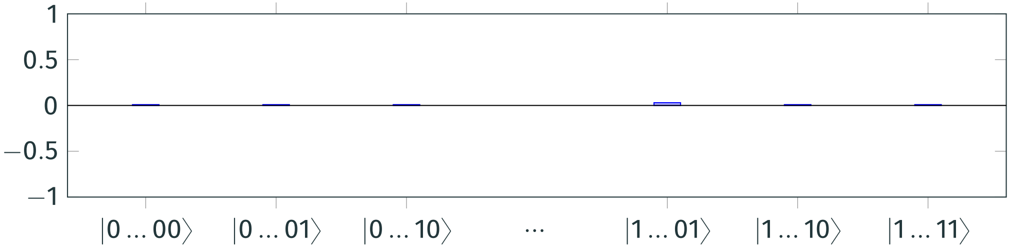
State space of entangled qubits:



Quantum Algorithms: Grover

Example: $f(x) \mapsto$ AES128(“<DOCTYPE html>”, x) = 0x45 0x59 ... 0xA1

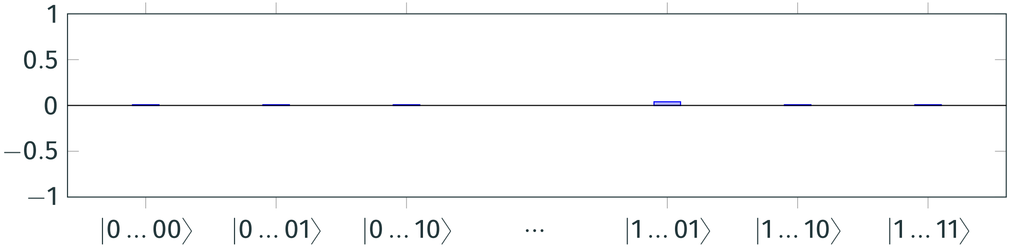
State space of entangled qubits:



Quantum Algorithms: Grover

Example: $f(x) \mapsto$ AES128(“<DOCTYPE html>”, x) = 0x45 0x59 … 0xA1

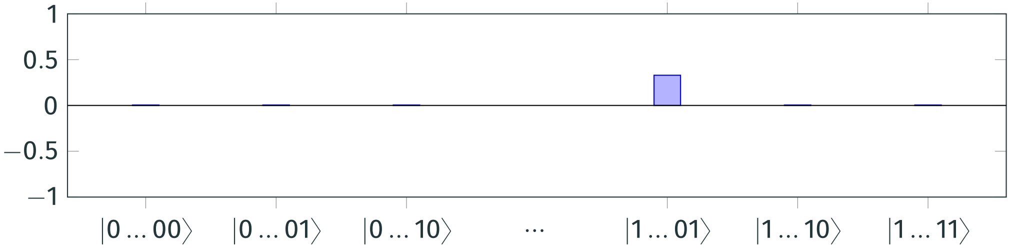
State space of entangled qubits:



Quantum Algorithms: Grover

Example: $f(x) \mapsto$ AES128("<DOCTYPE html>", x) = 0x45 0x59 ... 0xA1

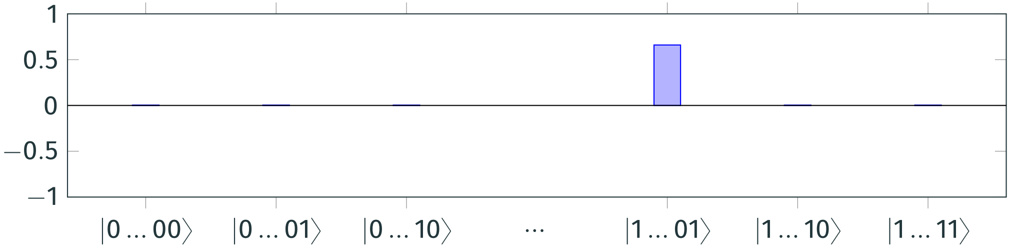
State space of entangled qubits:



Quantum Algorithms: Grover

Example: $f(x) \mapsto$ AES128(“<DOCTYPE html>”, x) = 0x45 0x59 ... 0xA1

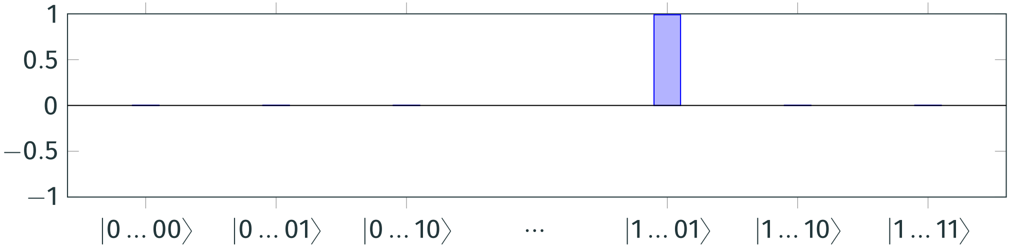
State space of entangled qubits:



Quantum Algorithms: Grover

Example: $f(x) \mapsto$ AES128("<DOCTYPE html>", x) = 0x45 0x59 ... 0xA1

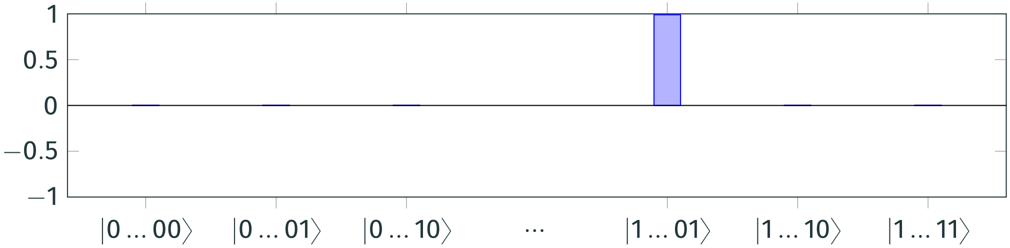
State space of entangled qubits:



Quantum Algorithms: Grover

Example: $f(x) \mapsto$ AES128(“<DOCTYPE html>”, x) = 0x45 0x59 ... 0xA1

State space of entangled qubits:



About $\sqrt{2^{128}} = 2^{64}$ iterations are necessary.

Shor's Algorithm:

Solves the "hidden-subgroup problem" in finite abelian groups.

Quantum Algorithms: Shor

Shor's Algorithm:

Solves the "hidden-subgroup problem" in finite abelian groups.

A **very efficient** algorithm for **very specific** problems:

Solves the *integer factorisation* and *discrete logarithm* problem in polynomial time.

Quantum Algorithms: Shor

Goal: Factor *N*.

1. Pick a random number $a \leqslant N$.

Quantum Algorithms: Shor

Goal: Factor N.

1. Pick a random number $a \leqslant N$.
2. If $\gcd(a, N) \neq 1$, return $\gcd(a, N)$, else continue.

Quantum Algorithms: Shor

Goal: Factor N**.**

1. Pick a random number $a \leqslant N$.
2. If $\gcd(a, N) \neq 1$, return $\gcd(a, N)$, else continue.
3. Find the period r of $a \bmod N$, i.e., the smallest integer r such that

$$a^r \equiv 1 \mod N.$$

Quantum Algorithms: Shor

Goal: Factor N**.**

1. Pick a random number $a \leqslant N$.
2. If $\gcd(a, N) \neq 1$, return $\gcd(a, N)$, else continue.
3. Find the period r of $a \bmod N$, i.e., the smallest integer r such that

$$a^r \equiv 1 \mod N.$$

4. If r is odd or if $a^{r/2} + 1 \equiv 0 \pmod{N}$, go back to step 1.

Quantum Algorithms: Shor

Goal: Factor N.

1. Pick a random number $a \leqslant N$.
2. If $\gcd(a, N) \neq 1$, return $\gcd(a, N)$, else continue.
3. Find the period r of $a \bmod N$, i.e., the smallest integer r such that

$$a^r \equiv 1 \mod N.$$

4. If r is odd or if $a^{r/2} + 1 \equiv 0 \pmod N$, go back to step 1.
5. We have

$$a^r - 1 = kN$$

Quantum Algorithms: Shor

Goal: Factor *N*.

1. Pick a random number $a \leqslant N$.
2. If $\gcd(a, N) \neq 1$, return $\gcd(a, N)$, else continue.
3. Find the period r of $a \bmod N$, i.e., the smallest integer r such that
$$a^r \equiv 1 \mod N.$$
4. If r is odd or if $a^{r/2} + 1 \equiv 0 \pmod{N}$, go back to step 1.
5. We have
$$\begin{aligned} a^r - 1 &= kN \\ (a^{r/2} + 1)(a^{r/2} - 1) &= kN. \end{aligned}$$

Quantum Algorithms: Shor

Goal: Factor N.

1. Pick a random number $a \leqslant N$.
2. If $\gcd(a, N) \neq 1$, return $\gcd(a, N)$, else continue.
3. Find the period r of $a \bmod N$, i.e., the smallest integer r such that
$$a^r \equiv 1 \mod N.$$
4. If r is odd or if $a^{r/2} + 1 \equiv 0 \pmod N$, go back to step 1.
5. We have
$$a^r - 1 = kN$$
$$(a^{r/2} + 1)(a^{r/2} - 1) = kN.$$
Compute the non-trivial factor $\gcd(a^{r/2} \pm 1, N)$ of N.

Quantum Algorithms: Shor

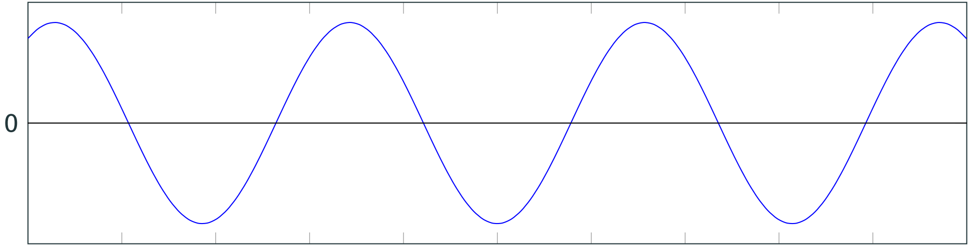
Goal: Factor N**.**

1. Pick a random number $a \leqslant N$.
2. If $\gcd(a, N) \neq 1$, return $\gcd(a, N)$, else continue.
3. Find the period r of $a \bmod N$, i.e., the smallest integer r such that
$$a^r \equiv 1 \mod N.$$
4. If r is odd or if $a^{r/2} + 1 \equiv 0 \pmod N$, go back to step 1.
5. We have
$$a^r - 1 = kN$$
$$(a^{r/2} + 1)(a^{r/2} - 1) = kN.$$

   Compute the non-trivial factor $\gcd(a^{r/2} \pm 1, N)$ of N.

Quantum algorithm for finding periods:

Idea: perform a Quantum Fourier Transform (QFT) to measure the period.



Quantum algorithm for finding periods:

Idea: perform a Quantum Fourier Transform (QFT) to measure the period.

Shor's Algorithm:

Unfortunately, current **asymmetric cryptography** is based on the integer factorisation and the discrete logarithm problem.

Quantum Algorithms: Shor

Shor's Algorithm:

Unfortunately, current **asymmetric cryptography** is based on the integer factorisation and the discrete logarithm problem.

- Integer factorisation in polynomial time:
  $\Rightarrow$ breaks RSA, ...

Quantum Algorithms: Shor

Shor's Algorithm:

Unfortunately, current **asymmetric cryptography** is based on the integer factorisation and the discrete logarithm problem.

- Integer factorisation in polynomial time:
  $\Rightarrow$ breaks RSA, ...
- Discrete logarithm in polynomial time:
  $\Rightarrow$ breaks DH, DSA; ECC: ECDH, ECDSA, ...

Quantum Algorithms: Shor

Shor's Algorithm:

Unfortunately, current **asymmetric cryptography** is based on the integer factorisation and the discrete logarithm problem.

- Integer factorisation in polynomial time:
  $\Rightarrow$ breaks RSA, ...
- Discrete logarithm in polynomial time:
  $\Rightarrow$ breaks DH, DSA; ECC: ECDH, ECDSA, ...

$\Rightarrow$ **We need new crypto to defend against quantum computers!**

Myths, Facts, Challenges, and Questions

Myths:

Quantum computers...

- do **not** compute all solution paths in parallel
  and do **not** instantly deliver the correct solution!

Myths, Facts, Challenges, and Questions

Myths:

Quantum computers...

- do **not** compute all solution paths in parallel
  and do **not** instantly deliver the correct solution!
- do **not** help much at NP-hard problems!

Myths, Facts, Challenges, and Questions

Myths:

Quantum computers...

- do **not** compute all solution paths in parallel
  and do **not** instantly deliver the correct solution!
- do **not** help much at NP-hard problems!
- do **not** solve the "traveling salesmen problem"!
  complexity: $n!$ for n cities
  solvable today: 20 cities; $20! \approx 2^{61}$ operations
  using Grover: 33 cities; $\sqrt{33!} \approx 2^{61}$ quantum operations

Myths, Facts, Challenges, and Questions

Facts:

Quantum computers...

- can accelerate certain computations.

Myths, Facts, Challenges, and Questions

Facts:

Quantum computers...

- can accelerate certain computations.
- threaten symmetric cryptography (Grover):
  $\Rightarrow$ Double key length!
  $\Rightarrow$ 256-bit keys for AES.

Myths, Facts, Challenges, and Questions

Facts:

Quantum computers...

- can accelerate certain computations.
- threaten symmetric cryptography (Grover):
  $\Rightarrow$ Double key length!
  $\Rightarrow$ 256-bit keys for AES.
- break wide-spread asymmetric cryptography (Shor):
  $\Rightarrow$ The end for RSA, ECC, DH, ECDH, DSA, ECDSA..!
  $\Rightarrow$ Alternatives are in preparation.

Myths, Facts, Challenges, and Questions

Technological Challenges:

- Keep entanglement and superposition stable;
  qubit state needs to be stable but easy to manipulate.

Myths, Facts, Challenges, and Questions

Technological Challenges:

- Keep entanglement and superposition stable;
  qubit state needs to be stable but easy to manipulate.
  $\Rightarrow$ Error correction on qubits.

Myths, Facts, Challenges, and Questions

Technological Challenges:

- Keep entanglement and superposition stable;
  qubit state needs to be stable but easy to manipulate.
  $\Rightarrow$ Error correction on qubits.
  $\Rightarrow$ Requires several (many?) physical qubits for one logical qubit.

Myths, Facts, Challenges, and Questions

Technological Challenges:

- Keep entanglement and superposition stable;
  qubit state needs to be stable but easy to manipulate.
  $\Rightarrow$ Error correction on qubits.
  $\Rightarrow$ Requires several (many?) physical qubits for one logical qubit.
- Map algorithms efficiently to hardware.

Technological Challenges:

- Keep entanglement and superposition stable;
  qubit state needs to be stable but easy to manipulate.
  $\Rightarrow$ Error correction on qubits.
  $\Rightarrow$ Requires several (many?) physical qubits for one logical qubit.
- Map algorithms efficiently to hardware.
  $\Rightarrow$ Match instruction sets.

Technological Challenges:

- Keep entanglement and superposition stable;
  qubit state needs to be stable but easy to manipulate.
  $\Rightarrow$ Error correction on qubits.
  $\Rightarrow$ Requires several (many?) physical qubits for one logical qubit.
- Map algorithms efficiently to hardware.
  $\Rightarrow$ Match instruction sets.
  $\Rightarrow$ Gates operate only on neighbouring qubits?

Myths, Facts, Challenges, and Questions

Technological Challenges:

- Keep entanglement and superposition stable;
  qubit state needs to be stable but easy to manipulate.
  $\Rightarrow$ Error correction on qubits.
  $\Rightarrow$ Requires several (many?) physical qubits for one logical qubit.
- Map algorithms efficiently to hardware.
  $\Rightarrow$ Match instruction sets.
  $\Rightarrow$ Gates operate only on neighbouring qubits?
- Scale quantum computer size.

Myths, Facts, Challenges, and Questions

Technological Challenges:

- Keep entanglement and superposition stable;
  qubit state needs to be stable but easy to manipulate.
  $\Rightarrow$ Error correction on qubits.
  $\Rightarrow$ Requires several (many?) physical qubits for one logical qubit.
- Map algorithms efficiently to hardware.
  $\Rightarrow$ Match instruction sets.
  $\Rightarrow$ Gates operate only on neighbouring qubits?
- Scale quantum computer size.
  $\Rightarrow$ Relevant algorithms require 1,500 to 6,000 logical qubits.

Myths, Facts, Challenges, and Questions

Questions:

- Are quantum computers really coming?

Response:

Myths, Facts, Challenges, and Questions

Questions:

- Are quantum computers really coming?

Response:

The majority of experts says: "Yes!"

Myths, Facts, Challenges, and Questions

Questions:

- Are quantum computers really coming?
- When do *large* quantum computers arrive?

Response:

Myths, Facts, Challenges, and Questions

Questions:

- Are quantum computers really coming?
- When do *large* quantum computers arrive?

Response:

Unclear...

Likelihood to break RSA-2048 in 24h*:

- 2026? ($< 1\%$)
- 2031? ($< 5\%$)
- 2036? ($\approx 50\%$)
- 2041? ($> 70\%$)
- 2051? ($> 95\%$)

Since 10 years: "In 15 years?"

* Mosca and Piani, Quantum Threat Timeline Report, 2021

Myths, Facts, Challenges, and Questions

Questions:

- Are quantum computers really coming?
- When do *large* quantum computers arrive?
- What schemes are going to fall first?

Response:

Myths, Facts, Challenges, and Questions

Questions:

- Are quantum computers really coming?
- When do *large* quantum computers arrive?
- What schemes are going to fall first?

Response:

- ECC?
- RSA?
- ...
- AES-128?

Myths, Facts, Challenges, and Questions

Questions:

- Are quantum computers really coming?
- When do *large* quantum computers arrive?
- What schemes are going to fall first?
- When do we need to start worry?

Response:

Myths, Facts, Challenges, and Questions

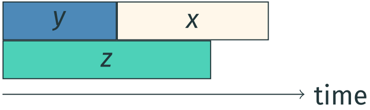
Questions:

- Are quantum computers really coming?
- When do *large* quantum computers arrive?
- What schemes are going to fall first?
- When do we need to start worry?

Response:

Mosca:

- Data must be protected for x years.
- We need y years to migrate to secure schemes.
- It takes z years before quantum computers break current crypto.



Myths, Facts, Challenges, and Questions

Questions:

- Are quantum computers really coming?
- When do *large* quantum computers arrive?
- What schemes are going to fall first?
- When do we need to start worry?
- What are the alternatives?

Response:

Myths, Facts, Challenges, and Questions

Questions:

- Are quantum computers really coming?
- When do *large* quantum computers arrive?
- What schemes are going to fall first?
- When do we need to start worry?
- What are the alternatives?

Response:

Post-Quantum Cryptography…

Post-Quantum Cryptography

Main PQC families:

- Lattice-based cryptography (e.g., NTRU, Kyber, Dilthium)
- Code-based cryptography (e.g., Classic McEliece, BIKE, HQC)
- Multivariate-quadratic-equations cryptography (e.g., Rainbow, UOV)
- Hash based cryptography (e.g., XMSS, LMS, SPHINCS+)
- Isogeny-based cryptography (e.g., SIDH, SIKE)

For these systems no efficient usage of Shor's algorithm is known.
Grover's algorithm has to be taken into account when choosing key sizes.

Post-Quantum Cryptography

Main PQC families:

- Lattice-based cryptography (e.g., NTRU, Kyber, Dilthium)
- Code-based cryptography (e.g., Classic McEliece, BIKE, HQC)
- Multivariate-quadratic-equations cryptography (e.g., Rainbow, UOV)
- Hash based cryptography (e.g., XMSS, LMS, SPHINCS+)
- Isogeny-based cryptography (e.g., SIDH, SIKE)

For these systems no efficient usage of Shor's algorithm is known.
Grover's algorithm has to be taken into account when choosing key sizes.

Multivariate Cryptography

Underlying problem:

Solving a system of m multivariate polynomial equations in n variables over $\mathbb{F}_q$ is called the MP problem.

Multivariate Cryptography

Underlying problem:
Solving a system of m multivariate polynomial equations in n variables over $\mathbb{F}_q$ is called the MP problem.

Example

$$5x_1^3x_2x_3^2 + 17x_2^4x_3 + 23x_1^2x_2^4 + 13x_1 + 12x_2 + 5 = 0$$
$$12x_1^2x_2^3x_3 + 15x_1x_3^3 + 25x_2x_3^3 + 5x_1 + 6x_3 + 12 = 0$$
$$28x_1x_2x_3^4 + 14x_2^3x_3^2 + 16x_1x_3 + 32x_2 + 7x_3 + 10 = 0$$

Multivariate Cryptography

Underlying problem:

Solving a system of m multivariate polynomial equations in n variables over $\mathbb{F}_q$ is called the MP problem.

Example

$$5x_1^3x_2x_3^2 + 17x_2^4x_3 + 23x_1^2x_2^4 + 13x_1 + 12x_2 + 5 = 0$$
$$12x_1^2x_2^3x_3 + 15x_1x_3^3 + 25x_2x_3^3 + 5x_1 + 6x_3 + 12 = 0$$
$$28x_1x_2x_3^4 + 14x_2^3x_3^2 + 16x_1x_3 + 32x_2 + 7x_3 + 10 = 0$$

Hardness:

The MP problem is an NP-complete problem even for multivariate *quadratic* systems and $q = 2$.

Multivariate Cryptography

Underlying problem:

Solving a system of m multivariate polynomial equations in n variables over $\mathbb{F}_q$ is called the MP problem.

Example

$$
\begin{aligned}
x_3x_2 + x_2x_1 + x_2 + x_1 + 1 &= 0 \\
x_3x_1 + x_3x_2 + x_3 + x_1 &= 0 \\
x_3x_2 + x_3x_1 + x_3 + x_2 &= 0
\end{aligned}
$$

Hardness:

The MP problem is an NP-complete problem even for multivariate *quadratic* systems and $q = 2$.

Multivariate Cryptography

Notation:

For a set $f = (f_1, \dots, f_m)$ of m quadratic polynomials in n variables over $\mathbb{F}_2$, let $f(x) = (f_1(x), \dots, f_m(x)) \in \mathbb{F}_2^m$ be the solution vector of the evaluation of f for a vector $x \in \mathbb{F}_2^n$.

Multivariate Cryptography

Notation:

For a set $f = (f_1, \dots, f_m)$ of m quadratic polynomials in n variables over $\mathbb{F}_2$, let $f(x) = (f_1(x), \dots, f_m(x)) \in \mathbb{F}_2^m$ be the solution vector of the evaluation of f for a vector $x \in \mathbb{F}_2^n$.

Definition ($\mathcal{MQ}$ over $\mathbb{F}_2$)

Let $\mathcal{MQ}(\mathbb{F}_2^n, \mathbb{F}_2^m)$ be the set of all systems of quadratic equations in n variables and m equations over $\mathbb{F}_2$.
We call one element $P \in \mathcal{MQ}(\mathbb{F}_2^n, \mathbb{F}_2^m)$ an instance of $\mathcal{MQ}$ over $\mathbb{F}_2$.

NP-Completeness of $\mathcal{MQ}$

Solvable in NP-time:

The following non-deterministic polynomial-time algorithm solves $\mathcal{MQ}$-$\mathbb{F}_2$ for a given system of equations:

1. Guess an assignment A for $(x_0, \dots, x_{n-1}) \in \{0, 1\}^n$.
2. Check if all m equations are satisfied by A.
3. Output A or go to an infinity loop, respectively.

NP-Completeness of $\mathcal{MQ}$

Solvable in NP-time:

The following non-deterministic polynomial-time algorithm solves $\mathcal{MQ}$-$\mathbb{F}_2$ for a given system of equations:

1. Guess an assignment A for $(x_0, \dots, x_{n-1}) \in \{0, 1\}^n$.
2. Check if all m equations are satisfied by A. $\Leftarrow$ **polynomial cost**
3. Output A or go to an infinity loop, respectively.

NP-Completeness of $\mathcal{MQ}$

NP-hardness:

Reduce 3-SAT to $\mathcal{MQ}$-$\mathbb{F}_2$.

$$(b_1 \vee \neg b_2 \vee b_3) \wedge (b_1 \vee b_2) \wedge (\neg b_4)$$

NP-Completeness of $\mathcal{MQ}$

NP-hardness:

Reduce 3-SAT to $\mathcal{MQ}$-$\mathbb{F}_2$.

$$(b_1 \vee \neg b_2 \vee b_3) \wedge (b_1 \vee b_2) \wedge (\neg b_4)$$

Replace all $(l_i \vee l_j)$ **by** $(l_i + l_j + l_i l_j)$**,**
replace all $(l_i \vee l_j \vee l_k)$ **by** $(l_i + l_j + l_k + l_i l_j + l_i l_k + l_j l_k + l_i l_j l_k)$**:**

$$(b_1 + \neg b_2 + b_3 + b_1 \neg b_2 + b_1 b_3 + \neg b_2 b_3 + b_1 \neg b_2 b_3) \wedge (b_1 + b_2 + b_1 b_2) \wedge (\neg b_4)$$

NP-Completeness of $\mathcal{MQ}$

NP-hardness:

Reduce 3-SAT to $\mathcal{MQ}$-$\mathbb{F}_2$.

$$(b_1 \vee \neg b_2 \vee b_3) \wedge (b_1 \vee b_2) \wedge (\neg b_4)$$

Replace all b_i by x_i and all $\neg b_i$ by $(1 - x_i)$:

$$\Big(x_1 + (1 - x_2) + x_3 + x_1(1 - x_2) + x_1x_3 + (1 - x_2)x_3 + x_1(1 - x_2)x_3\Big) \wedge (x_1 + x_2 + x_1x_2) \wedge (1 - x_4)$$

NP-Completeness of $\mathcal{MQ}$

NP-hardness:

Reduce 3-SAT to $\mathcal{MQ}$-$\mathbb{F}_2$.

$$(b_1 \vee \neg b_2 \vee b_3) \wedge (b_1 \vee b_2) \wedge (\neg b_4)$$

Construct an equation $e_i : c_i = 1$ for each clause c_i:

$$\begin{aligned} x_1 + (1 - x_2) + x_3 + x_1(1 - x_2) + x_1x_3 + (1 - x_2)x_3 + x_1(1 - x_2)x_3 &= 1 \\ x_1 + x_2 + x_1x_2 &= 1 \\ 1 - x_4 &= 1 \end{aligned}$$

NP-Completeness of $\mathcal{MQ}$

NP-hardness:

Reduce 3-SAT to $\mathcal{MQ}$-$\mathbb{F}_2$.

$$(b_1 \vee \neg b_2 \vee b_3) \wedge (b_1 \vee b_2) \wedge (\neg b_4)$$

Expand all terms:

$$\begin{aligned} x_1x_2 + x_1x_2x_3 + x_2x_3 + x_2 &= 0 \\ x_1x_2 + x_1 + x_2 + 1 &= 0 \\ x_4 &= 0 \end{aligned}$$

NP-Completeness of $\mathcal{MQ}$

NP-hardness:

Reduce 3-SAT to $\mathcal{MQ}$-$\mathbb{F}_2$.

$$(b_1 \vee \neg b_2 \vee b_3) \wedge (b_1 \vee b_2) \wedge (\neg b_4)$$

Iteratively add a new equation for each remaining cubic term:

$$\begin{aligned} x_1x_2 + x_5x_3 + x_2x_3 + x_2 &= 0 \\ x_1x_2 + x_1 + x_2 + 1 &= 0 \\ x_4 &= 0 \\ x_5 &= x_1x_2 \end{aligned}$$

NP-Completeness of $\mathcal{MQ}$

NP-hardness:

Reduce 3-SAT to $\mathcal{MQ}$-$\mathbb{F}_2$.

$$(b_1 \vee \neg b_2 \vee b_3) \wedge (b_1 \vee b_2) \wedge (\neg b_4)$$

Final equation system:

$$\begin{aligned} x_3x_5 + x_2x_3 + x_2 + x_5 &= 0 \\ x_1 + x_2 + x_5 + 1 &= 0 \\ x_4 &= 0 \\ x_1x_2 + x_5 &= 0 \end{aligned}$$

NP-Completeness of $\mathcal{MQ}$

NP-hardness:

Reduce 3-SAT to $\mathcal{MQ}$-$\mathbb{F}_2$.

$$(b_1 \vee \neg b_2 \vee b_3) \wedge (b_1 \vee b_2) \wedge (\neg b_4)$$

Final equation system:

$$\begin{aligned} x_3x_5 + x_2x_3 + x_2 + x_5 &= 0 \\ x_1 + x_2 + x_5 + 1 &= 0 \\ x_4 &= 0 \\ x_1x_2 + x_5 &= 0 \end{aligned}$$

3-SAT $\leqslant_{\text{poly}}$ $\mathcal{MQ}$-$\mathbb{F}_2$

NP-Completeness of $\mathcal{MQ}$

Theorem

$\mathcal{MQ}$-$\mathbb{F}_2$ is NP-complete.

Proof.

We showed that $\mathcal{MQ}$-$\mathbb{F}_2 \in$ NP and 3-SAT $\leqslant_{\text{poly}}$ $\mathcal{MQ}$-$\mathbb{F}_2$.

Thus, $\mathcal{MQ}$-$\mathbb{F}_2$ is NP-complete. $\square$

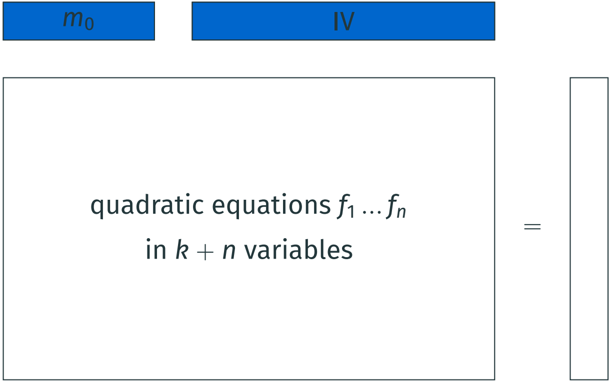
Cryptosystems

Hashing

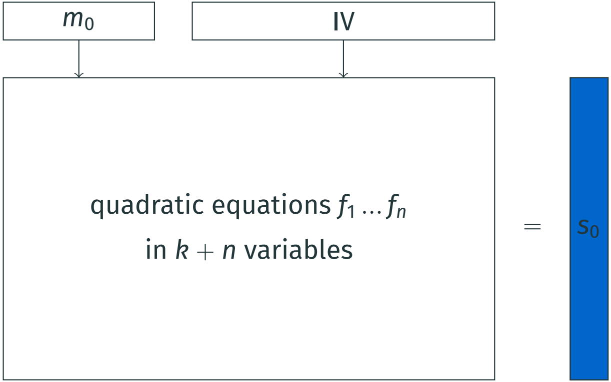
Cryptographic hash function:

- Pre-image resistance:
  Given a hash h it should be difficult to find any message m such that $h = \mathrm{H}(m)$.
- Second pre-image resistance:
  Given an input m_0 it should be difficult to find another input m_1 such that $m_0 \neq m_1$ and $\mathrm{H}(m_0) = \mathrm{H}(m_1)$.
- Collision resistance:
  It should be difficult to find two different messages m_0 and m_1 such that that $m_0 \neq m_1$ and $\mathrm{H}(m_0) = \mathrm{H}(m_1)$.

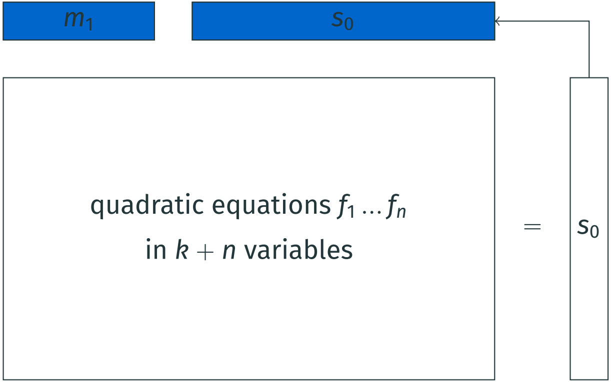
Hashing



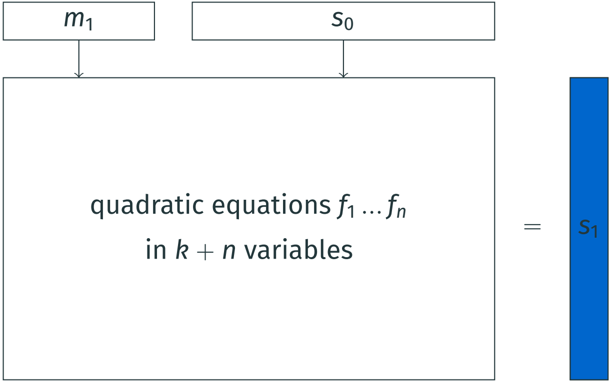
Hashing



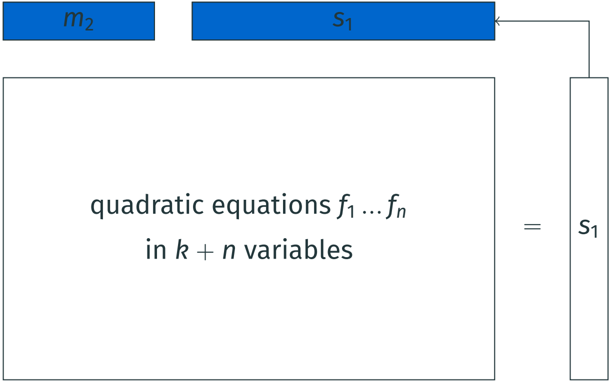
Hashing



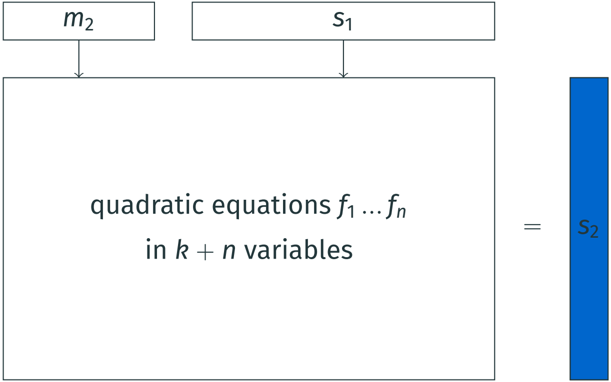
Hashing



Hashing



Hashing



Hashing

quadratic equations $f_1 \ldots f_n$ in $k + n$ variables $= s_2$

H

Hashing

Problem: Easy to find collisions!

$$f(m, \mathrm{IV}) = f(m', \mathrm{IV}')$$

$$f(m, \mathrm{IV}) = f(m + a, \mathrm{IV} + b)$$

$$f(m, \mathrm{IV}) - f(m + a, \mathrm{IV} + b) = 0$$

Hashing

Problem: Easy to find collisions!

$$\begin{aligned} \mathrm{f}(m, \mathrm{IV}) &= \mathrm{f}(m', \mathrm{IV}') \\ \mathrm{f}(m, \mathrm{IV}) &= \mathrm{f}(m + a, \mathrm{IV} + b) \\ \mathrm{f}(m, \mathrm{IV}) - \mathrm{f}(m + a, \mathrm{IV} + b) &= 0 \end{aligned}$$

$$f_0(x) = c_{2,1}x_2x_1 + c_{2,0}x_2x_0 + c_{1,0}x_1x_0 + c_2x_2 + c_1x_1 + c_0x_0 + c$$

Hashing

Problem: Easy to find collisions!

$$\mathrm{f}(m, \mathrm{IV}) = \mathrm{f}(m', \mathrm{IV}')$$
$$\mathrm{f}(m, \mathrm{IV}) = \mathrm{f}(m + a, \mathrm{IV} + b)$$
$$\mathrm{f}(m, \mathrm{IV}) - \mathrm{f}(m + a, \mathrm{IV} + b) = 0$$

$$f_0(x) = c_{2,1}x_2x_1 + c_{2,0}x_2x_0 + c_{1,0}x_1x_0 + c_2x_2 + c_1x_1 + c_0x_0 + c$$

$$f_0(x) - f_0(x + a) = c_{2,1}x_2x_1 + c_{2,0}x_2x_0 + c_{1,0}x_1x_0 + c_2x_2 + c_1x_1 + c_0x_0 + c$$
$$-\big(c_{2,1}(x_2 + a_2)(x_1 + a_1) + \ldots c_2(x_2 + a_2) + \cdots + c\big)$$

Hashing

Problem: Easy to find collisions!

$$\begin{aligned} \mathrm{f}(m, \mathrm{IV}) &= \mathrm{f}(m', \mathrm{IV}') \\ \mathrm{f}(m, \mathrm{IV}) &= \mathrm{f}(m + a, \mathrm{IV} + b) \\ \mathrm{f}(m, \mathrm{IV}) - \mathrm{f}(m + a, \mathrm{IV} + b) &= 0 \end{aligned}$$

$$f_0(x) = c_{2,1}x_2x_1 + c_{2,0}x_2x_0 + c_{1,0}x_1x_0 + c_2x_2 + c_1x_1 + c_0x_0 + c$$

$$f_0(x) - f_0(x + a) = c_{2,1}x_2x_1 + c_{2,0}x_2x_0 + c_{1,0}x_1x_0 + c_2x_2 + c_1x_1 + c_0x_0 + c$$
$$-(c_{2,1}(x_2 + a_2)(x_1 + a_1) + \ldots c_2(x_2 + a_2) + \cdots + c)$$

$$f_0(x) - f_0(x + a) = c_{2,1}x_2x_1 + c_{2,0}x_2x_0 + c_{1,0}x_1x_0 + c_2x_2 + c_1x_1 + c_0x_0 + c$$
$$-(c_{2,1}(x_2x_1 + a_1x_2 + a_2x_1 + a_1a_2) + \ldots c_2x_2 + c_2a_2 + \cdots + c)$$

Hashing

Problem: Easy to find collisions!

$$f(m, \mathrm{IV}) = f(m', \mathrm{IV}')$$
$$f(m, \mathrm{IV}) = f(m + a, \mathrm{IV} + b)$$
$$f(m, \mathrm{IV}) - f(m + a, \mathrm{IV} + b) = 0$$

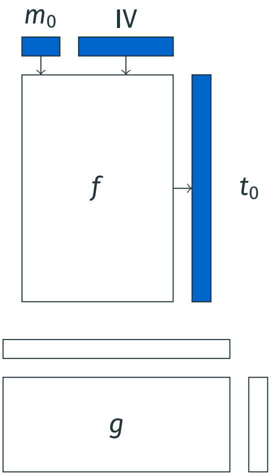
$$f_0(x) = c_{2,1}x_2x_1 + c_{2,0}x_2x_0 + c_{1,0}x_1x_0 + c_2x_2 + c_1x_1 + c_0x_0 + c$$

$$f_0(x) - f_0(x + a) = c_{2,1}x_2x_1 + c_{2,0}x_2x_0 + c_{1,0}x_1x_0 + c_2x_2 + c_1x_1 + c_0x_0 + c$$
$$-(c_{2,1}(x_2 + a_2)(x_1 + a_1) + \ldots c_2(x_2 + a_2) + \cdots + c)$$

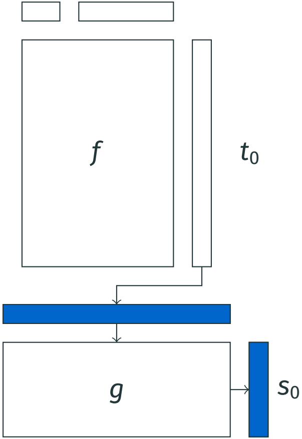
$$f_0(x) - f_0(x + a) = c_{2,1}x_2x_1 + c_{2,0}x_2x_0 + c_{1,0}x_1x_0 + c_2x_2 + c_1x_1 + c_0x_0 + c$$
$$-(c_{2,1}(x_2x_1 + a_1x_2 + a_2x_1 + a_1a_2) + \ldots c_2x_2 + c_2a_2 + \cdots + c)$$

$\Rightarrow$ Underdefined linear system of $k + n$ variables and n equations!

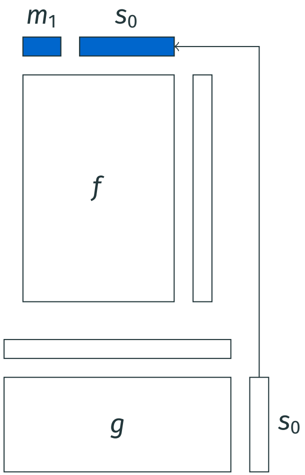
Hashing



Hashing



Hashing



Hashing

Example (MQ-HASH)

$f : \mathbb{F}_2^{n+k} \rightarrow \mathbb{F}_2^r$

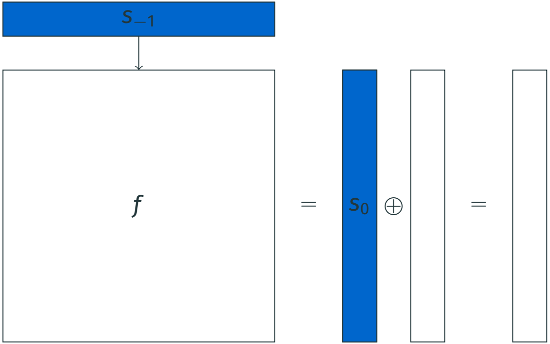
$g : \mathbb{F}_2^r \rightarrow \mathbb{F}_2^n$

$H : (g \circ f)(s_1, \ldots, s_n, m_1, \ldots, m_k)$

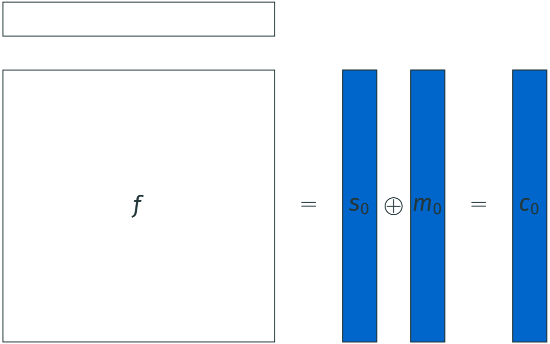
MQ-HASH: $k = 32$, $n = 160$ and $r = 464$.

Symmetric Encryption

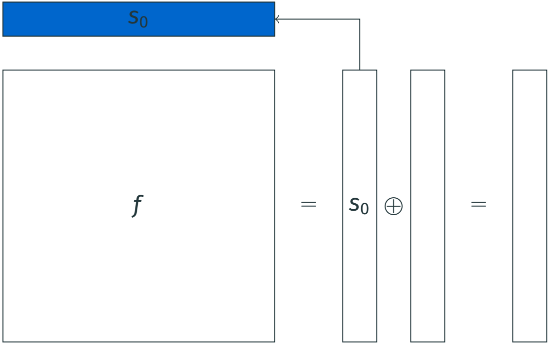
Pre-process symmetric key and IV to obtain initial state s_{-1}.



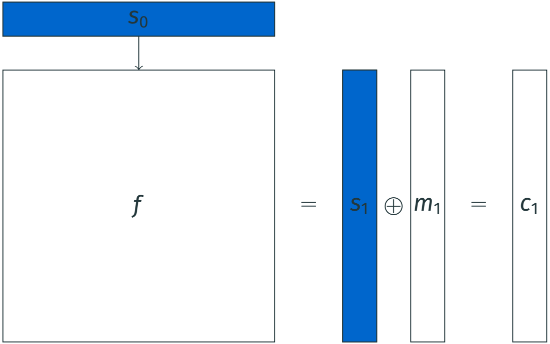
Symmetric Encryption



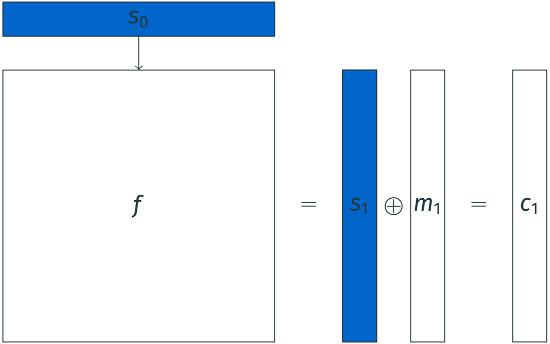
Symmetric Encryption



Symmetric Encryption



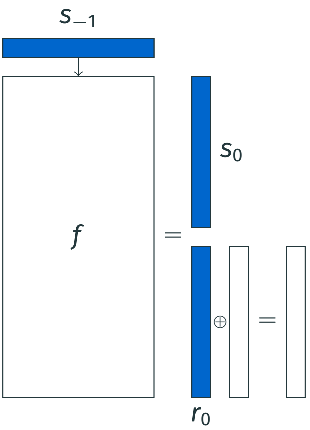
Symmetric Encryption



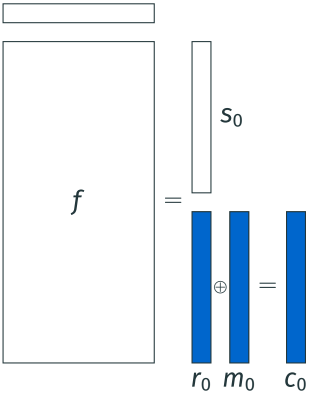
Easy to obtain key stream with a single known plain text block!

Symmetric Encryption

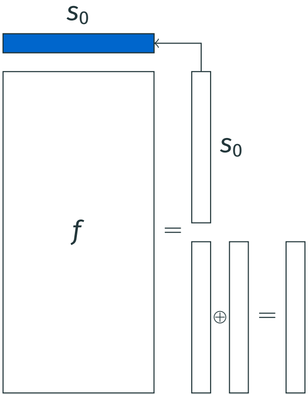
Pre-process symmetric key and IV to obtain initial state s_{-1}.



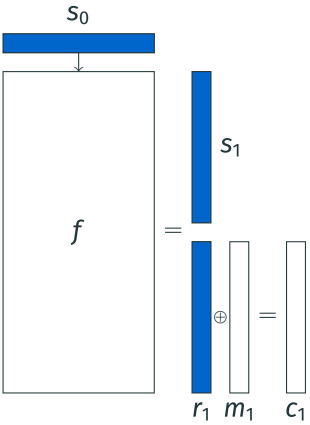
Symmetric Encryption



Symmetric Encryption



Symmetric Encryption



Symmetric Encryption

QUAD stream cipher

Provably secure!

Symmetric Encryption

QUAD stream cipher

Provably secure!

Initialy suggested parameters QUAD(256,20,20) have been broken!

Symmetric Encryption

QUAD stream cipher

Provably secure!

Initialy suggested parameters QUAD(256,20,20) have been broken!

Parameters that are still considered secure:
QUAD(2,160,160), QUAD(2,256,256), QUAD(2,350,350), ...

Public-Key Encryption

Composition of functions with known inverse:

Secretly choose f, g, h with known inverse functions f^{-1}, g^{-1}, h^{-1}.

Release $F = f \circ g \circ h$ as public key and h^{-1}, g^{-1}, f^{-1} as private key.

Public-Key Encryption

Composition of functions with known inverse:

Secretly choose f, g, h with known inverse functions f^{-1}, g^{-1}, h^{-1}.

Release $F = f \circ g \circ h$ as public key and h^{-1}, g^{-1}, f^{-1} as private key.

Example

Choose $f = (f_1, \dots, f_n), h = (h_1, \dots, h_n)$ as sets of independent linear equations and

$$g(g_1, \dots, g_n) = \begin{pmatrix} g_1 : & x_1, \\ g_2 : & x_2 + p_2(x_1), \\ g_3 : & x_3 + p_3(x_1, x_2), \\ & \dots \\ g_4 : & x_n + p_4(x_1, \dots, x_{n-1}) \end{pmatrix},$$

with p_i quadratic in $x_1, \dots, x_i$.

Public-Key Encryption

Example

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2 x_1 + x_2) \\ x_4 + (x_3 x_1 + x_3 x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

Public-Key Encryption

Example

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2x_1 + x_2) \\ x_4 + (x_3x_1 + x_3x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

$$F = f \circ g \circ h = \begin{pmatrix} x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 \\ x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 + 1 \\ x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 \\ x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 \end{pmatrix}$$

Public-Key Encryption

Example (Encryption)

$$F = \begin{pmatrix} x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 \\ x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 + 1 \\ x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 \\ x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 \end{pmatrix}$$

$$F(1,0,0,1) = \begin{pmatrix} 0 \cdot 0 + 0 \cdot 1 + 0 \cdot 1 + 1 + 0 + 0 + 1 \\ 1 \cdot 0 + 1 \cdot 1 + 0 \cdot 0 + 0 \cdot 1 + 1 + 1 \\ 1 \cdot 0 + 1 \cdot 1 + 0 \cdot 1 + 0 + 1 \\ 0 \cdot 0 + 0 \cdot 1 + 0 \cdot 1 + 1 + 1 \end{pmatrix} = (0,1,0,0)$$

Public-Key Encryption

Example (Decryption)

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2 x_1 + x_2) \\ x_4 + (x_3 x_1 + x_3 x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

$$f^{-1} = \begin{pmatrix} y_4 + y_3 + y_2 \\ y_3 + y_2 + y_1 + 1 \\ y_4 + y_3 + y_2 + y_1 + 1 \\ y_3 + y_1 + 1 \end{pmatrix}$$

$$f^{-1}(0, 1, 0, 0) = (1, 0, 0, 1)$$

Public-Key Encryption

Example (Decryption)

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2x_1 + x_2) \\ x_4 + (x_3x_1 + x_3x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

$$\begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2x_1 + x_2) \\ x_4 + (x_3x_1 + x_3x_2 + x_1) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Public-Key Encryption

Example (Decryption)

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2 x_1 + x_2) \\ x_4 + (x_3 x_1 + x_3 x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

$$\begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2 x_1 + x_2) \\ x_4 + (x_3 x_1 + x_3 x_2 + x_1) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Public-Key Encryption

Example (Decryption)

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2x_1 + x_2) \\ x_4 + (x_3x_1 + x_3x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

$$\begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2x_1 + x_2) \\ x_4 + (x_3x_1 + x_3x_2 + x_1) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Example (Decryption)

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2x_1 + x_2) \\ x_4 + (x_3x_1 + x_3x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

$$\begin{pmatrix} x_1 \\ x_2 + (1 + 1) \\ x_3 + (x_2x_1 + x_2) \\ x_4 + (x_3x_1 + x_3x_2 + x_1) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Public-Key Encryption

Example (Decryption)

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2 x_1 + x_2) \\ x_4 + (x_3 x_1 + x_3 x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 + (x_2 x_1 + x_2) \\ x_4 + (x_3 x_1 + x_3 x_2 + x_1) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Example (Decryption)

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2 x_1 + x_2) \\ x_4 + (x_3 x_1 + x_3 x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 + (0 \cdot 1 + 0) \\ x_4 + (x_3 x_1 + x_3 x_2 + x_1) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Public-Key Encryption

Example (Decryption)

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2x_1 + x_2) \\ x_4 + (x_3x_1 + x_3x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 + (x_3x_1 + x_3x_2 + x_1) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Public-Key Encryption

Example (Decryption)

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2 x_1 + x_2) \\ x_4 + (x_3 x_1 + x_3 x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 + (0 \cdot 1 + 0 \cdot 0 + 1) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Public-Key Encryption

Example (Decryption)

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2 x_1 + x_2) \\ x_4 + (x_3 x_1 + x_3 x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 + 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Public-Key Encryption

Example (Decryption)

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2 x_1 + x_2) \\ x_4 + (x_3 x_1 + x_3 x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Example (Decryption)

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2 x_1 + x_2) \\ x_4 + (x_3 x_1 + x_3 x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$g^{-1}(1, 0, 0, 1) = (1, 0, 0, 0)$$

Example (Decryption)

$$f = \begin{pmatrix} x_3 + x_1 + 1 \\ x_4 + x_2 \\ x_4 + x_3 + x_1 \\ x_3 + x_2 \end{pmatrix}, g = \begin{pmatrix} x_1 \\ x_2 + (x_1 + 1) \\ x_3 + (x_2 x_1 + x_2) \\ x_4 + (x_3 x_1 + x_3 x_2 + x_1) \end{pmatrix}, h = \begin{pmatrix} x_2 + x_1 \\ x_3 + x_2 \\ x_4 + x_2 + 1 \\ x_4 + x_2 + x_1 \end{pmatrix}.$$

$$h^{-1} = \begin{pmatrix} y_4 + y_3 + 1 \\ y_4 + y_3 + y_1 + 1 \\ y_4 + y_2 + y_3 + y_1 + 1 \\ y_4 + y_1 \end{pmatrix}$$

$$h^{-1}(1,0,0,0) = (1,0,0,1)$$

Public-Key Encryption

Attention!

$$g(g_1, \dots, g_n) = \begin{pmatrix} g_1 : & x_1, \\ g_2 : & x_2 + p_2(x_1), \\ g_3 : & x_3 + p_3(x_1, x_2), \\ & \dots \\ g_4 : & x_n + p_4(x_1, \dots, x_{n-1}) \end{pmatrix}$$

Public-Key Encryption

Attention!

$$g(g_1, \dots, g_n) = \begin{pmatrix} g_1: & x_1, \\ g_2: & x_2 + p_2(x_1), \\ g_3: & x_3 + p_3(x_1, x_2), \\ & \dots \\ g_4: & x_n + p_4(x_1, \dots, x_{n-1}) \end{pmatrix}$$

$f \circ g \circ h$ is **not** a hard instance of $\mathcal{MQ}$-$\mathbb{F}_2$ due to the linearity of g_1 (and g_2)!

Public-Key Encryption

Attention!

$$g(g_1, \ldots, g_n) = \begin{pmatrix} g_1 : & x_1, \\ g_2 : & x_2 + p_2(x_1), \\ g_3 : & x_3 + p_3(x_1, x_2), \\ & \ldots \\ g_4 : & x_n + p_4(x_1, \ldots, x_{n-1}) \end{pmatrix}$$

$f \circ g \circ h$ is **not** a hard instance of $\mathcal{MQ}$-$\mathbb{F}_2$
due to the linearity of g_1 (and g_2)!

Solution:

Make composition more complicated; this is ongoing research.

Public-Key Encryption

Attention!

$$g(g_1, \ldots, g_n) = \begin{pmatrix} g_1: & x_1, \\ g_2: & x_2 + p_2(x_1), \\ g_3: & x_3 + p_3(x_1, x_2), \\ & \ldots \\ g_4: & x_n + p_4(x_1, \ldots, x_{n-1}) \end{pmatrix}$$

$f \circ g \circ h$ is **not** a hard instance of $\mathcal{MQ}$-$\mathbb{F}_2$
due to the linearity of g_1 (and g_2)!

Solution:

Make composition more complicated; this is ongoing research.

Many asymmetric $\mathcal{MQ}$-$\mathbb{F}_2$ schemes that have been prosed so far have been broken!

Signatures

Basic scheme (known from RSA etc.):

- Signing: Encrypt message hash with private key.
- Verification: Decrypt signature with public key and compare to message hash.

Signatures

Basic scheme (known from RSA etc.):

- Signing: Encrypt message hash with private key.
- Verification: Decrypt signature with public key and compare to message hash.

No secure multivariate public key system → no secure signature scheme...

Signatures

Basic scheme (known from RSA etc.):

- Signing: Encrypt message hash with private key.
- Verification: Decrypt signature with public key and compare to message hash.

No secure multivariate public key system → no secure signature scheme...

Wrong!

There actually are secure multivariate signature schemes that are not based on public key encryption.

Example (Oil and Vinegar)

Private key:

$$f = \begin{pmatrix} x_6 + x_3 + 1 \\ x_6 + x_3 + x_1 \\ x_5 + x_3 + 1 \\ x_4 + x_2 + 1 \\ x_3 + x_2 + 1 \\ x_5 + x_1 \end{pmatrix}, g = \begin{pmatrix} x_6x_1 + x_5x_2 + x_4x_2 + x_2x_1 + x_4 + x_3 \\ x_4x_1 + x_3x_2 + x_4 + x_1 + 1 \\ x_6x_3 + x_5x_3 + x_3x_2 + x_6 + x_5 + x_1 + 1 \end{pmatrix}$$

Example (Oil and Vinegar)

Private key:

$$f = \begin{pmatrix} x_6 + x_3 + 1 \\ x_6 + x_3 + x_1 \\ x_5 + x_3 + 1 \\ x_4 + x_2 + 1 \\ x_3 + x_2 + 1 \\ x_5 + x_1 \end{pmatrix}, g = \begin{pmatrix} x_6x_1 + x_5x_2 + x_4x_2 + x_2x_1 + x_4 + x_3 \\ x_4x_1 + x_3x_2 + x_4 + x_1 + 1 \\ x_6x_3 + x_5x_3 + x_3x_2 + x_6 + x_5 + x_1 + 1 \end{pmatrix}$$

Public key: $g \circ f =$

$$\begin{pmatrix} x_6x_5 + x_6x_4 + x_6x_3 + x_5x_3 + x_4x_3 + x_4x_1 + x_3x_1 + x_4 + x_2 \\ x_6x_5 + x_6x_4 + x_6x_3 + x_6x_2 + x_5x_3 + x_5x_1 + x_4x_3 + x_3x_2 + x_3x_1 + x_6 + x_1 \\ x_6x_5 + x_6x_3 + x_5x_3 + x_5x_2 + x_3x_2 + x_3 + x_1 \end{pmatrix}$$

Signatures

Example (Signing)

$$f = \begin{pmatrix} x_6 + x_3 + 1 \\ x_6 + x_3 + x_1 \\ x_5 + x_3 + 1 \\ x_4 + x_2 + 1 \\ x_3 + x_2 + 1 \\ x_5 + x_1 \end{pmatrix}, g = \begin{pmatrix} x_6x_1 + x_5x_2 + x_4x_2 + x_2x_1 + x_4 + x_3 \\ x_4x_1 + x_3x_2 + x_4 + x_1 + 1 \\ x_6x_3 + x_5x_3 + x_3x_2 + x_6 + x_5 + x_1 + 1 \end{pmatrix}$$

Example (Signing)

Oil variables: x_6, x_5, x_4; Vinegar variables: x_3, x_2, x_1.

$$f = \begin{pmatrix} x_6 + x_3 + 1 \\ x_6 + x_3 + x_1 \\ x_5 + x_3 + 1 \\ x_4 + x_2 + 1 \\ x_3 + x_2 + 1 \\ x_5 + x_1 \end{pmatrix}, g = \begin{pmatrix} x_6x_1 + x_5x_2 + x_4x_2 + x_2x_1 + x_4 + x_3 \\ x_4x_1 + x_3x_2 + x_4 + x_1 + 1 \\ x_6x_3 + x_5x_3 + x_3x_2 + x_6 + x_5 + x_1 + 1 \end{pmatrix}$$

Signatures

Example (Signing)

Oil variables: x_6, x_5, x_4; Vinegar variables: x_3, x_2, x_1.

$$f = \begin{pmatrix} x_6 + x_3 + 1 \\ x_6 + x_3 + x_1 \\ x_5 + x_3 + 1 \\ x_4 + x_2 + 1 \\ x_3 + x_2 + 1 \\ x_5 + x_1 \end{pmatrix}, g = \begin{pmatrix} x_6x_1 + x_5x_2 + x_4x_2 + x_2x_1 + x_4 + x_3 \\ x_4x_1 + x_3x_2 + x_4 + x_1 + 1 \\ x_6x_3 + x_5x_3 + x_3x_2 + x_6 + x_5 + x_1 + 1 \end{pmatrix}$$

Randomly choose x_3, x_2, x_1, e.g., $x_3 = 0, x_2 = 1, x_1 = 0$:

$$g' = \begin{pmatrix} 0x_6 + 1x_5 + 1x_4 + 1 \cdot 0 + x_4 + 0 \\ 0x_4 + 0 \cdot 1 + x_4 + 0 + 1 \\ 0x_6 + 0x_5 + 0 \cdot 1 + x_6 + x_5 + 0 + 1 \end{pmatrix}$$

Signatures

Example (Signing)

Oil variables: x_6, x_5, x_4; Vinegar variables: x_3, x_2, x_1.

$$f = \begin{pmatrix} x_6 + x_3 + 1 \\ x_6 + x_3 + x_1 \\ x_5 + x_3 + 1 \\ x_4 + x_2 + 1 \\ x_3 + x_2 + 1 \\ x_5 + x_1 \end{pmatrix}, g = \begin{pmatrix} x_6x_1 + x_5x_2 + x_4x_2 + x_2x_1 + x_4 + x_3 \\ x_4x_1 + x_3x_2 + x_4 + x_1 + 1 \\ x_6x_3 + x_5x_3 + x_3x_2 + x_6 + x_5 + x_1 + 1 \end{pmatrix}$$

Randomly choose x_3, x_2, x_1, e.g., $x_3 = 0, x_2 = 1, x_1 = 0$:

$$g' = \begin{pmatrix} 0x_6 + 1x_5 + 1x_4 + 1 \cdot 0 + x_4 + 0 \\ 0x_4 + 0 \cdot 1 + x_4 + 0 + 1 \\ 0x_6 + 0x_5 + 0 \cdot 1 + x_6 + x_5 + 0 + 1 \end{pmatrix} = \begin{pmatrix} x_5 \\ x_4 + 1 \\ x_6 + x_5 + 1 \end{pmatrix}$$

Signatures

Example (Signing)

Oil variables: x_6, x_5, x_4; Vinegar variables: x_3, x_2, x_1.

$$f = \begin{pmatrix} x_6 + x_3 + 1 \\ x_6 + x_3 + x_1 \\ x_5 + x_3 + 1 \\ x_4 + x_2 + 1 \\ x_3 + x_2 + 1 \\ x_5 + x_1 \end{pmatrix}, g = \begin{pmatrix} x_6x_1 + x_5x_2 + x_4x_2 + x_2x_1 + x_4 + x_3 \\ x_4x_1 + x_3x_2 + x_4 + x_1 + 1 \\ x_6x_3 + x_5x_3 + x_3x_2 + x_6 + x_5 + x_1 + 1 \end{pmatrix}$$

Sign $h = (1, 1, 0)$:

$$\begin{aligned} x_5 &= 1 \\ x_4 + 1 &= 1 \\ x_6 + x_5 + 1 &= 0 \end{aligned}$$

Signatures

Example (Signing)

Oil variables: x_6, x_5, x_4; Vinegar variables: x_3, x_2, x_1.

$$f = \begin{pmatrix} x_6 + x_3 + 1 \\ x_6 + x_3 + x_1 \\ x_5 + x_3 + 1 \\ x_4 + x_2 + 1 \\ x_3 + x_2 + 1 \\ x_5 + x_1 \end{pmatrix}, g = \begin{pmatrix} x_6x_1 + x_5x_2 + x_4x_2 + x_2x_1 + x_4 + x_3 \\ x_4x_1 + x_3x_2 + x_4 + x_1 + 1 \\ x_6x_3 + x_5x_3 + x_3x_2 + x_6 + x_5 + x_1 + 1 \end{pmatrix}$$

Sign $h = (1, 1, 0)$:

$$x_5 = 1$$
$$x_4 = 0$$
$$x_6 = 0$$

Signatures

Example (Signing)

Oil variables: x_6, x_5, x_4; Vinegar variables: x_3, x_2, x_1.

$$f = \begin{pmatrix} x_6 + x_3 + 1 \\ x_6 + x_3 + x_1 \\ x_5 + x_3 + 1 \\ x_4 + x_2 + 1 \\ x_3 + x_2 + 1 \\ x_5 + x_1 \end{pmatrix}, g = \begin{pmatrix} x_6x_1 + x_5x_2 + x_4x_2 + x_2x_1 + x_4 + x_3 \\ x_4x_1 + x_3x_2 + x_4 + x_1 + 1 \\ x_6x_3 + x_5x_3 + x_3x_2 + x_6 + x_5 + x_1 + 1 \end{pmatrix}$$

Sign $h = (1, 1, 0)$:

$$x_5 = 1$$

$$x_4 = 0$$

$$x_6 = 0$$

$$g^{-1}(1, 1, 0) = (0, 1, 0, 0, 1, 0)$$

Signatures

Example (Signing)

$g^{-1}(1, 1, 0) = (0, 1, 0, 0, 1, 0)$

$$f = \begin{pmatrix} x_6 + x_3 + 1 \\ x_6 + x_3 + x_1 \\ x_5 + x_3 + 1 \\ x_4 + x_2 + 1 \\ x_3 + x_2 + 1 \\ x_5 + x_1 \end{pmatrix},$$

Signatures

Example (Signing)

$g^{-1}(1,1,0) = (0,1,0,0,1,0)$

$$f = \begin{pmatrix} x_6 + x_3 + 1 \\ x_6 + x_3 + x_1 \\ x_5 + x_3 + 1 \\ x_4 + x_2 + 1 \\ x_3 + x_2 + 1 \\ x_5 + x_1 \end{pmatrix}, f^{-1} = \begin{pmatrix} x_2 + x_1 + 1 \\ x_6 + x_5 + x_3 + x_2 + x_1 + 1 \\ x_6 + x_3 + x_2 + x_1 \\ x_6 + x_5 + x_4 + x_3 + x_2 + x_1 \\ x_6 + x_2 + x_1 + 1 \\ x_6 + x_3 + x_2 + 1 \end{pmatrix}$$

Signatures

Example (Signing)

$g^{-1}(1,1,0) = (0,1,0,0,1,0)$

$$f = \begin{pmatrix} x_6 + x_3 + 1 \\ x_6 + x_3 + x_1 \\ x_5 + x_3 + 1 \\ x_4 + x_2 + 1 \\ x_3 + x_2 + 1 \\ x_5 + x_1 \end{pmatrix}, f^{-1} = \begin{pmatrix} x_2 + x_1 + 1 \\ x_6 + x_5 + x_3 + x_2 + x_1 + 1 \\ x_6 + x_3 + x_2 + x_1 \\ x_6 + x_5 + x_4 + x_3 + x_2 + x_1 \\ x_6 + x_2 + x_1 + 1 \\ x_6 + x_3 + x_2 + 1 \end{pmatrix}$$

$f^{-1}(0,1,0,0,1,0) = (0,0,0,1,1,0)$

Signatures

Example (Signing)

$g^{-1}(1,1,0) = (0,1,0,0,1,0)$

$$f = \begin{pmatrix} x_6 + x_3 + 1 \\ x_6 + x_3 + x_1 \\ x_5 + x_3 + 1 \\ x_4 + x_2 + 1 \\ x_3 + x_2 + 1 \\ x_5 + x_1 \end{pmatrix}, f^{-1} = \begin{pmatrix} x_2 + x_1 + 1 \\ x_6 + x_5 + x_3 + x_2 + x_1 + 1 \\ x_6 + x_3 + x_2 + x_1 \\ x_6 + x_5 + x_4 + x_3 + x_2 + x_1 \\ x_6 + x_2 + x_1 + 1 \\ x_6 + x_3 + x_2 + 1 \end{pmatrix}$$

$f^{-1}(0,1,0,0,1,0) = (0,0,0,1,1,0)$

$s = f^{-1}g^{-1}(1,1,0) = (0,0,0,1,1,0)$

Signatures

Example (Verification)

$h = (1, 1, 0), s = (0, 0, 0, 1, 1, 0)$

Example (Verification)

$h = (1, 1, 0), s = (0, 0, 0, 1, 1, 0)$

$g \circ f =$

$$\begin{pmatrix} x_6x_5 + x_6x_4 + x_6x_3 + x_5x_3 + x_4x_3 + x_4x_1 + x_3x_1 + x_4 + x_2 \\ x_6x_5 + x_6x_4 + x_6x_3 + x_6x_2 + x_5x_3 + x_5x_1 + x_4x_3 + x_3x_2 + x_3x_1 + x_6 + x_1 \\ x_6x_5 + x_6x_3 + x_5x_3 + x_5x_2 + x_3x_2 + x_3 + x_1 \end{pmatrix}$$

Signatures

Example (Verification)

$h = (1, 1, 0), s = (0, 0, 0, 1, 1, 0)$

$g \circ f =$

$$\begin{pmatrix} x_6x_5 + x_6x_4 + x_6x_3 + x_5x_3 + x_4x_3 + x_4x_1 + x_3x_1 + x_4 + x_2 \\ x_6x_5 + x_6x_4 + x_6x_3 + x_6x_2 + x_5x_3 + x_5x_1 + x_4x_3 + x_3x_2 + x_3x_1 + x_6 + x_1 \\ x_6x_5 + x_6x_3 + x_5x_3 + x_5x_2 + x_3x_2 + x_3 + x_1 \end{pmatrix}$$

$h' = g \circ f(0, 0, 0, 1, 1, 0) = (1, 1, 0)$

Public key encryption scheme?

Oil and Vinegar can not be used as public key encryption scheme due to the randomness of the vinegar variables.

Signatures

Public key encryption scheme?

Oil and Vinegar can not be used as public key encryption scheme due to the randomness of the vinegar variables.

Oil and Vinegar is broken!

Signatures

Public key encryption scheme?

Oil and Vinegar can not be used as public key encryption scheme due to the randomness of the vinegar variables.

Oil and Vinegar is broken!

There are variations of Oil and Vinegar, e.g., Unbalanced Oil and Vinegar (UOV), that are considered secure.

From OV to UOV:

The attack on OV exploits the fact that there are as many oil variables as there are vinegar variables.

However, the attack is not applibale if there are (many) more vinegar than oil variables.

Signatures

From OV to UOV:

The attack on OV exploits the fact that there are as many oil variables as there are vinegar variables.

However, the attack is not applibale if there are (many) more vinegar than oil variables.

UOV parameter recommendations:

field	n	o (oil)	v (vinegar)	bit security
$\mathbb{F}_{2^4}$	160	64	96	128
$\mathbb{F}_{2^8}$	112	44	68	128
$\mathbb{F}_{2^8}$	184	72	112	192
$\mathbb{F}_{2^8}$	244	96	148	256

System Solving

System Solving

Algebraic Cryptanalysis:

Obtain a system of multivariate polynomial equations with the secret among the variables.

- Naturally breaks multivariate crypto schemes,
- does not break AES as first advertised,
- but does break, e.g., KeeLoq.

Gröbner Bases

Example

$$F = \begin{pmatrix} x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 \\ x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 + 1 \\ x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 \\ x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 \end{pmatrix}$$

Find x for $F(x) = (0, 1, 0, 0)$.

Gröbner Bases

Example

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \quad (1)$$

$$x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 + 1 = 1 \quad (2)$$

$$x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 = 0 \quad (3)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 = 0 \quad (4)$$

Gröbner Bases

Example

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \quad (1)$$

$$x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 + 1 = 1 \quad (2)$$

$$x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 = 0 \quad (3)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 = 0 \quad (4)$$

Gröbner Bases

Example

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \quad (1)$$

$$x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 = 0 \quad (2)$$

$$x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 = 0 \quad (3)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 = 0 \quad (4)$$

Gröbner Bases

Example

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \quad (1)$$

$$x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 = 0 \quad (2)$$

$$x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 = 0 \quad (3)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 = 0 \quad (4)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_2 + 1 = 0 \qquad (2) + (3) = \quad (5)$$

Gröbner Bases

Example

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \quad (1)$$

$$x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 = 0 \quad (2)$$

$$x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 = 0 \quad (3)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 = 0 \quad (4)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_2 + 1 = 0 \qquad (2) + (3) = \quad (5)$$

$$x_2 + x_1 + 1 = 0 \qquad (4) + (5) = \quad (6)$$

Gröbner Bases

Example

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \quad (1)$$

$$x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 = 0 \quad (2)$$

$$x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 = 0 \quad (3)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 = 0 \quad (4)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_2 + 1 = 0 \qquad (2) + (3) = \quad (5)$$

$$x_2 + x_1 + 1 = 0 \qquad (4) + (5) = \quad (6)$$

$$x_3 + x_2 = 0 \qquad (1) + (4) = \quad (7)$$

Gröbner Bases

Example

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \qquad (1)$$
$$x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 = 0 \qquad (2)$$
$$x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 = 0 \qquad (3)$$
$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 = 0 \qquad (4)$$
$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_2 + 1 = 0 \qquad (2) + (3) = \qquad (5)$$
$$x_2 + x_1 + 1 = 0 \qquad (4) + (5) = \qquad (6)$$
$$x_3 + x_2 = 0 \qquad (1) + (4) = \qquad (7)$$
$$x_3x_2x_1 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 + x_3 = 0 \qquad x_3(1) + (2) = \qquad (8)$$

Gröbner Bases

Example

$$
\begin{aligned}
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 &= 0 & & (1)\\
x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 &= 0 & & (2)\\
x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 &= 0 & & (3)\\
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 &= 0 & & (4)\\
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_2 + 1 &= 0 & (2) + (3) = & (5)\\
x_2 + x_1 + 1 &= 0 & (4) + (5) = & (6)\\
x_3 + x_2 &= 0 & (1) + (4) = & (7)\\
x_3x_2x_1 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 + x_3 &= 0 & x_3(1) + (2) = & (8)\\
x_3x_2x_1 + x_4x_1 + x_3x_2 + x_3x_1 + x_2 + 1 &= 0 & x_3(4) + (3) = & (9)
\end{aligned}
$$

Gröbner Bases

Example

$$
\begin{aligned}
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 &= 0 & & (1)\\
x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 &= 0 & & (2)\\
x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 &= 0 & & (3)\\
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 &= 0 & & (4)\\
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_2 + 1 &= 0 & (2) + (3) = & (5)\\
x_2 + x_1 + 1 &= 0 & (4) + (5) = & (6)\\
x_3 + x_2 &= 0 & (1) + (4) = & (7)\\
x_3x_2x_1 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 + x_3 &= 0 & x_3(1) + (2) = & (8)\\
x_3x_2x_1 + x_4x_1 + x_3x_2 + x_3x_1 + x_2 + 1 &= 0 & x_3(4) + (3) = & (9)\\
x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + 1 &= 0 & (8) + (9) = & (10)
\end{aligned}
$$

Gröbner Bases

Example

$$
\begin{aligned}
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 &= 0 & & (1)\\
x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 &= 0 & & (2)\\
x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 &= 0 & & (3)\\
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 &= 0 & & (4)\\
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_2 + 1 &= 0 & (2) + (3) = & (5)\\
x_2 + x_1 + 1 &= 0 & (4) + (5) = & (6)\\
x_3 + x_2 &= 0 & (1) + (4) = & (7)\\
x_3x_2x_1 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 + x_3 &= 0 & x_3(1) + (2) = & (8)\\
x_3x_2x_1 + x_4x_1 + x_3x_2 + x_3x_1 + x_2 + 1 &= 0 & x_3(4) + (3) = & (9)\\
x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + 1 &= 0 & (8) + (9) = & (10)\\
x_4 + x_3 + x_2 + 1 &= 0 & x_1(7) + (10) = & (11)
\end{aligned}
$$

Gröbner Bases

Example

$$
\begin{aligned}
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 &= 0 & & (1)\\
x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 &= 0 & & (2)\\
x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 &= 0 & & (3)\\
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 &= 0 & & (4)\\
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_2 + 1 &= 0 & (2) + (3) = & (5)\\
x_2 + x_1 + 1 &= 0 & (4) + (5) = & (6)\\
x_3 + x_2 &= 0 & (1) + (4) = & (7)\\
x_3x_2x_1 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 + x_3 &= 0 & x_3(1) + (2) = & (8)\\
x_3x_2x_1 + x_4x_1 + x_3x_2 + x_3x_1 + x_2 + 1 &= 0 & x_3(4) + (3) = & (9)\\
x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + 1 &= 0 & (8) + (9) = & (10)\\
x_4 + x_3 + x_2 + 1 &= 0 & x_1(7) + (10) = & (11)\\
x_4 + 1 &= 0 & (7) + (11) = & (12)
\end{aligned}
$$

Gröbner Bases

Example

$$
\begin{aligned}
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 &= 0 && && (1)\\
x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 &= 0 && && (2)\\
x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 &= 0 && && (3)\\
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 &= 0 && && (4)\\
x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_2 + 1 &= 0 && (2) + (3) = && (5)\\
x_2 + x_1 + 1 &= 0 && (4) + (5) = && (6)\\
x_3 + x_2 &= 0 && (1) + (4) = && (7)\\
x_3x_2x_1 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 + x_3 &= 0 && x_3(1) + (2) = && (8)\\
x_3x_2x_1 + x_4x_1 + x_3x_2 + x_3x_1 + x_2 + 1 &= 0 && x_3(4) + (3) = && (9)\\
x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + 1 &= 0 && (8) + (9) = && (10)\\
x_4 + x_3 + x_2 + 1 &= 0 && x_1(7) + (10) = && (11)\\
x_4 &= 1 && (7) + (11) = && (12)
\end{aligned}
$$

Gröbner Bases

Example

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \quad (1)$$

$$x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 = 0 \quad (2)$$

$$x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 = 0 \quad (3)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 = 0 \quad (4)$$

$$x_2 + x_1 + 1 = 0 \quad (6)$$

$$x_3 + x_2 = 0 \quad (7)$$

$$x_4 = 1 \quad (12)$$

Gröbner Bases

Example

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \quad (1)$$

$$x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 = 0 \quad (2)$$

$$x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 = 0 \quad (3)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 = 0 \quad (4)$$

$$x_2 + x_1 + 1 = 0 \quad (6)$$

$$x_3 + x_2 = 0 \quad (7)$$

$$x_4 = 1 \quad (12)$$

$$x_4x_3x_1 + x_4x_3 + x_2x_1 + x_4 + x_3 + x_1 = 0 \qquad x_3(3) + (4) = \quad (13)$$

Gröbner Bases

Example

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \quad (1)$$

$$x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 = 0 \quad (2)$$

$$x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 = 0 \quad (3)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 = 0 \quad (4)$$

$$x_2 + x_1 + 1 = 0 \quad (6)$$

$$x_3 + x_2 = 0 \quad (7)$$

$$x_4 = 1 \quad (12)$$

$$x_4x_3x_1 + x_4x_3 + x_2x_1 + x_4 + x_3 + x_1 = 0 \qquad x_3(3) + (4) = \quad (13)$$

$$x_4x_3x_1 + x_3x_2x_1 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \qquad (1) + x_3(2) = \quad (14)$$

Gröbner Bases

Example

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \qquad (1)$$

$$x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 = 0 \qquad (2)$$

$$x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 = 0 \qquad (3)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 = 0 \qquad (4)$$

$$x_2 + x_1 + 1 = 0 \qquad (6)$$

$$x_3 + x_2 = 0 \qquad (7)$$

$$x_4 = 1 \qquad (12)$$

$$x_4x_3x_1 + x_4x_3 + x_2x_1 + x_4 + x_3 + x_1 = 0 \qquad x_3(3) + (4) = \qquad (13)$$

$$x_4x_3x_1 + x_3x_2x_1 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \qquad (1) + x_3(2) = \qquad (14)$$

$$x_2 = 0 \qquad (14) + (13) + (9) + x_4(7) + x_4(6) + x_2(7) + (12) = \qquad (15)$$

Gröbner Bases

Example

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \qquad (1)$$

$$x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 = 0 \qquad (2)$$

$$x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 = 0 \qquad (3)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 = 0 \qquad (4)$$

$$x_2 + x_1 + 1 = 0 \qquad (6)$$

$$x_3 + x_2 = 0 \qquad (7)$$

$$x_4 = 1 \qquad (12)$$

$$x_4x_3x_1 + x_4x_3 + x_2x_1 + x_4 + x_3 + x_1 = 0 \qquad x_3(3) + (4) = \qquad (13)$$

$$x_4x_3x_1 + x_3x_2x_1 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \qquad (1) + x_3(2) = \qquad (14)$$

$$x_2 = 0 \qquad (14) + (13) + (9) + x_4(7) + x_4(6) + x_2(7) + (12) = \qquad (15)$$

$$x_3 = 0 \qquad (7) + (15) = \qquad (16)$$

Gröbner Bases

Example

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \quad (1)$$

$$x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 = 0 \quad (2)$$

$$x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 = 0 \quad (3)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 = 0 \quad (4)$$

$$x_2 + x_1 + 1 = 0 \quad (6)$$

$$x_3 + x_2 = 0 \quad (7)$$

$$x_4 = 1 \quad (12)$$

$$x_4x_3x_1 + x_4x_3 + x_2x_1 + x_4 + x_3 + x_1 = 0 \qquad x_3(3) + (4) = \quad (13)$$

$$x_4x_3x_1 + x_3x_2x_1 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \qquad (1) + x_3(2) = \quad (14)$$

$$x_2 = 0 \qquad (14) + (13) + (9) + x_4(7) + x_4(6) + x_2(7) + (12) = \quad (15)$$

$$x_3 = 0 \qquad (7) + (15) = \quad (16)$$

$$x_1 = 1 \qquad (6) + (15) = \quad (17)$$

Gröbner Bases

Example

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \qquad (1)$$

$$x_4x_3 + x_4x_1 + x_3x_2 + x_2x_1 + x_4 = 0 \qquad (2)$$

$$x_4x_3 + x_4x_1 + x_3x_1 + x_2 + 1 = 0 \qquad (3)$$

$$x_3x_2 + x_3x_1 + x_2x_1 + x_4 + x_1 = 0 \qquad (4)$$

$$x_2 + x_1 + 1 = 0 \qquad (6)$$

$$x_3 + x_2 = 0 \qquad (7)$$

$$x_4 = 1 \qquad (12)$$

$$x_4x_3x_1 + x_4x_3 + x_2x_1 + x_4 + x_3 + x_1 = 0 \qquad x_3(3) + (4) = \qquad (13)$$

$$x_4x_3x_1 + x_3x_2x_1 + x_3x_1 + x_2x_1 + x_4 + x_3 + x_2 + x_1 = 0 \qquad (1) + x_3(2) = \qquad (14)$$

$$x_2 = 0 \qquad (14) + (13) + (9) + x_4(7) + x_4(6) + x_2(7) + (12) = \qquad (15)$$

$$x_3 = 0 \qquad (7) + (15) = \qquad (16)$$

$$x_1 = 1 \qquad (6) + (15) = \qquad (17)$$

Gröbner Bases

Algorithm due to Buchberger:

- Transform set of equations to a Gröbner basis; obtain solution of the system from the final representation.
- During computation, the maximum degree increases to $D > 2$.
- There are several improvements of Buchbergers algorithm, e.g., Faugère's F_4 and F_5 (implemented, e.g., in Magma).

Extended Linearization

The XL algorithm

- *XL* is an acronym for *extended linearization*:
  - *extend* a quadratic system by multiplying with appropriate monomials,
  - *linearize* by treating each monomial as an independent variable,
  - solve the linearized system.
- Special case of Gröbner basis algorithms.
- First suggested by Lazard (1983).
- Reinvented by Courtois, Klimov, Patarin, and Shamir (2000).
- More “easy” to parallelize compared to Gröbner basis solvers.

Extended Linearization

Basic idea:

For $b \in \mathbb{N}^n$ denote by x^b the monomial $x_1^{b_1} x_2^{b_2} \dots x_n^{b_n}$ and by $|b| = b_1 + b_2 + \cdots + b_n$ the total degree of x^b.

given:	finite field $K = \mathbb{F}_q$
	system $\mathcal{A}$ of m multivariate quadratic equations:
	$\ell_1 = \ell_2 = \cdots = \ell_m = 0,\ \ell_i \in K[x_1, x_2, \dots, x_n]$
choose:	operational degree $D \in \mathbb{N}$
extend:	system $\mathcal{A}$ to the system
	$\mathcal{R}^{(D)} = \{x^b \ell_i = 0 : \lvert b \rvert \leqslant D - 2, \ell_i \in \mathcal{A}\}$
linearize:	consider $x^d, d \leqslant D$ a new variable, obtain linear system $\mathcal{M}$
solve:	linear system $\mathcal{M}$

Extended Linearization

Basic idea:

For $b \in \mathbb{N}^n$ denote by x^b the monomial $x_1^{b_1} x_2^{b_2} \dots x_n^{b_n}$ and by $|b| = b_1 + b_2 + \cdots + b_n$ the total degree of x^b.

given: finite field $K = \mathbb{F}_q$
system $\mathcal{A}$ of m multivariate quadratic equations:
$\ell_1 = \ell_2 = \cdots = \ell_m = 0,\ \ell_i \in K[x_1, x_2, \dots, x_n]$

choose: operational degree $D \in \mathbb{N}$ How?

extend: system $\mathcal{A}$ to the system
$\mathcal{R}^{(D)} = \{x^b \ell_i = 0 : |b| \leqslant D - 2, \ell_i \in \mathcal{A}\}$

linearize: consider $x^d, d \leqslant D$ a new variable, obtain linear system $\mathcal{M}$

solve: linear system $\mathcal{M}$

Extended Linearization

Basic idea:

For $b \in \mathbb{N}^n$ denote by x^b the monomial $x_1^{b_1} x_2^{b_2} \dots x_n^{b_n}$ and by $|b| = b_1 + b_2 + \cdots + b_n$ the total degree of x^b.

given: finite field $K = \mathbb{F}_q$

system $\mathcal{A}$ of m multivariate quadratic equations:

$\ell_1 = \ell_2 = \cdots = \ell_m = 0,\ \ell_i \in K[x_1, x_2, \dots, x_n]$

choose: operational degree $D \in \mathbb{N}$ How?

extend: system $\mathcal{A}$ to the system

$\mathcal{R}^{(D)} = \{x^b \ell_i = 0 : |b| \leqslant D - 2, \ell_i \in \mathcal{A}\}$

linearize: consider $x^d, d \leqslant D$ a new variable, obtain linear system $\mathcal{M}$

solve: linear system $\mathcal{M}$

minimum degree D_0 for reliable termination (Yang and Chen):

$$D_0 := \min\{D : ((1 - \lambda)^{m-n-1}(1 + \lambda)^m)[D] \leqslant 0\}$$

Extended Linearization

Basic idea:

For $b \in \mathbb{N}^n$ denote by x^b the monomial $x_1^{b_1} x_2^{b_2} \ldots x_n^{b_n}$ and by $|b| = b_1 + b_2 + \cdots + b_n$ the total degree of x^b.

given: finite field $K = \mathbb{F}_q$
system $\mathcal{A}$ of m multivariate quadratic equations:
$\ell_1 = \ell_2 = \cdots = \ell_m = 0,\ \ell_i \in K[x_1, x_2, \ldots, x_n]$

choose: operational degree $D \in \mathbb{N}$ How?

extend: system $\mathcal{A}$ to the system
$\mathcal{R}^{(D)} = \{x^b \ell_i = 0 : |b| \leqslant D - 2, \ell_i \in \mathcal{A}\}$

linearize: consider x^d, $d \leqslant D$ a new variable, obtain linear system $\mathcal{M}$

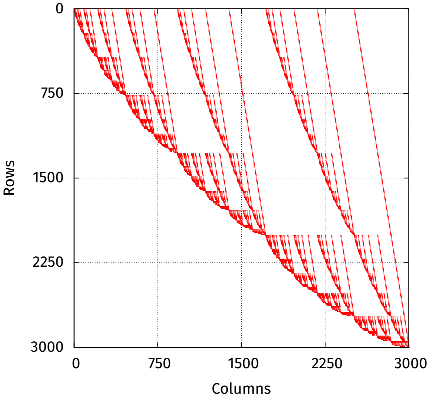
solve: linear system $\mathcal{M}$ How?

minimum degree D_0 for reliable termination (Yang and Chen):

$$D_0 := \min\{D : ((1-\lambda)^{m-n-1}(1+\lambda)^m)[D] \leqslant 0\}$$

Extended Linearization

Solve the sparse linear system $\mathcal{M}$:



Use, e.g., the (block) Lanczos or the (block) Wiedemann algorithm.

Brute Force

Efficiency:

Gröbner basis solvers and XL are efficient for solving multivariate polynomial systems over *large* finite fields.

Brute Force

Efficiency:

Gröbner basis solvers and XL are efficient for solving multivariate polynomial systems over *large* finite fields.

Most Efficient Algorithm for $\mathbb{F}_2$:

Brute-force search, testing all 2^n possible inputs.

Exhaustive Search — Approach

Full-Evaluation Approach

- Evaluate the whole equation for each possible input.
- Time Complexity: $O(2^n n^2)$
- Memory Complexity: $O(n)$

Exhaustive Search — Approach

$k = 01010_b;\ x_4 = 0,\ x_3 = 1,\ x_2 = 0,\ x_1 = 1,\ x_0 = 0$

$$\begin{aligned} f &= x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1 \\ f &= 0 \cdot 0 + 1 \cdot 0 + 0 \cdot 1 + 1 + 1 + 0 + 1 \end{aligned}$$

Exhaustive Search — Approach

$k = 01010_b;\ x_4 = 0,\ x_3 = 1,\ x_2 = 0,\ x_1 = 1,\ x_0 = 0$

$$\begin{aligned} f &= x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1 \\ f &= 0 \cdot 0 + 1 \cdot 0 + 0 \cdot 1 + 1 + 1 + 0 + 1 \end{aligned}$$

$k = 01011_b;\ x_4 = 0,\ x_3 = 1,\ x_2 = 0,\ x_1 = 1,\ x_0 = 1$

$$\begin{aligned} f &= x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1 \\ f &= 0 \cdot 0 + 1 \cdot 1 + 0 \cdot 1 + 1 + 1 + 1 + 1 \end{aligned}$$

Exhaustive Search — Approach

$k = 01010_b;\ x_4 = 0,\ x_3 = 1,\ x_2 = 0,\ x_1 = 1,\ x_0 = 0$

$$f = x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1$$
$$f = 0 \cdot 0 + 1 \cdot 0 + 0 \cdot 1 + 1 + 1 + 0 + 1$$

$k = 01011_b;\ x_4 = 0,\ x_3 = 1,\ x_2 = 0,\ x_1 = 1,\ x_0 = 1$

$$f = x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1$$
$$f = 0 \cdot 0 + 1 \cdot 1 + 0 \cdot 1 + 1 + 1 + 1 + 1$$

$k = 01100_b;\ x_4 = 0,\ x_3 = 1,\ x_2 = 1,\ x_1 = 0,\ x_0 = 0$

$$f = x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1$$
$$f = 0 \cdot 1 + 1 \cdot 0 + 1 \cdot 0 + 1 + 0 + 0 + 1$$

Exhaustive Search — Approach

$k = 01010_b;\ x_4 = 0,\ x_3 = 1,\ x_2 = 0,\ x_1 = 1,\ x_0 = 0$

$$\begin{aligned} f &= x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1 \\ f &= 0\cdot 0 + 1\cdot 0 + 0\cdot 1 + 1 + 1 + 0 + 1 \end{aligned}$$

$k = 01011_b;\ x_4 = 0,\ x_3 = 1,\ x_2 = 0,\ x_1 = 1,\ x_0 = 1$

$$\begin{aligned} f &= x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1 \\ f &= 0\cdot 0 + 1\cdot 1 + 0\cdot 1 + 1 + 1 + 1 + 1 \end{aligned}$$

$k = 01001_b$ **in *Gray-code* order**

$$\begin{aligned} f &= x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1 \\ f &= 0\cdot 0 + 1\cdot 1 + 0\cdot 0 + 1 + 0 + 1 + 1 \end{aligned}$$

Exhaustive Search — Approach

$k = 01010_b;\ x_4 = 0,\ x_3 = 1,\ x_2 = 0,\ x_1 = 1,\ x_0 = 0$

$$
\begin{aligned}
f &= x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1\\
f &= 0\cdot 0 + 1\cdot 0 + 0\cdot 1 + 1 + 1 + 0 + 1
\end{aligned}
$$

$k = 01011_b;\ x_4 = 0,\ x_3 = 1,\ x_2 = 0,\ x_1 = 1,\ x_0 = 1$

$$
\begin{aligned}
f &= x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1\\
f &= 0\cdot 0 + 1\cdot 1 + 0\cdot 1 + 1 + 1 + 1 + 1
\end{aligned}
$$

$k = 01001_b$ **in *Gray-code* order**

$$
\begin{aligned}
f &= x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1\\
f &= 0\cdot 0 + 1\cdot 1 + 0\cdot 0 + 1 + 0 + 1 + 1\\
f &= f(01011_b) - 0\cdot 1 - 1 + 0\cdot 0 + 0
\end{aligned}
$$

Exhaustive Search — Approach

$k = 01010_b;\ x_4 = 0,\ x_3 = 1,\ x_2 = 0,\ x_1 = 1,\ x_0 = 0$

$$
\begin{aligned}
f &= x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1 \\
f &= 0 \cdot 0 + 1 \cdot 0 + 0 \cdot 1 + 1 + 1 + 0 + 1
\end{aligned}
$$

$k = 01011_b;\ x_4 = 0,\ x_3 = 1,\ x_2 = 0,\ x_1 = 1,\ x_0 = 1$

$$
\begin{aligned}
f &= x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1 \\
f &= 0 \cdot 0 + 1 \cdot 1 + 0 \cdot 1 + 1 + 1 + 1 + 1
\end{aligned}
$$

$k = 01001_b$ **in *Gray-code* order**

$$
\begin{aligned}
f &= x_4x_2 + x_3x_0 + x_2x_1 + x_3 + x_1 + x_0 + 1 \\
f &= 0 \cdot 0 + 1 \cdot 1 + 0 \cdot 0 + 1 + 0 + 1 + 1 \\
f &= f(01011_b) - 0 \cdot 1 - 1 + 0 \cdot 0 + 0 \\
f &= f(01011_b) + \frac{\partial f}{\partial x_1}(01001_b)
\end{aligned}
$$

Exhaustive Search — Approach

Full-Evaluation Approach

- Evaluate the whole equation for each possible input.
- Time Complexity: $O(2^n n^2)$
- Memory Complexity: $O(n)$

Exhaustive Search — Approach

Full-Evaluation Approach

- Evaluate the whole equation for each possible input.
- Time Complexity: $O(2^n n^2)$
- Memory Complexity: $O(n)$

Gray-Code Approach

- Only re-compute those parts of the equation that have changed.
- Enumerate input vector in Gray-code order.
- Update solution using the derivatives of the involved variables.
- Time Complexity: $O(2^n m)$
- Memory Complexity: $O(n^2 m)$

Trade computation for memory.

Joux-Vitse's Crossbred Algorithm

Basic idea:

- Extend the original $\mathcal{MQ}$ system to a system with a degree D lower than the degree required for XL.
- Derive a sub-system that has at most degree d in the first k variables.
- Solve this sub-system by iterating over the remaining $n - k$ variables and solving the resulting degree-d system in k variables in each iteration.

For $d = 1$, this requires to only solve a linear system in k variables for each assignment of $n - k$ variables.

Joux-Vitse's Crossbred Algorithm

Example

By fixing the last two variables x_3 and x_4 to, e.g., $x_3 = 0$ and $x_4 = 0$, the sub-system

$$S = \begin{cases} x_1x_4 + x_2x_3 + x_1 + x_3 + x_4 = 0 \\ x_1x_3 + x_3x_4 + x_2 + 1 = 0 \\ x_2x_3 + x_2x_4 + x_3x_4 + x_1 + x_4 = 0 \end{cases}$$

becomes a linear system in x_1 and x_2.

Joux-Vitse's Crossbred Algorithm

Example

By fixing the last two variables x_3 and x_4 to, e.g., $x_3 = 0$ and $x_4 = 0$, the sub-system

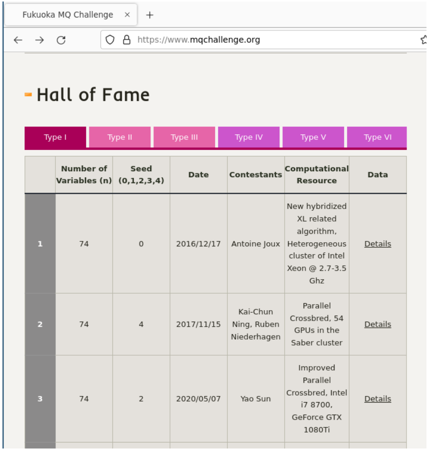
$$S = \begin{cases} x_1x_4 + x_2x_3 + x_1 + x_3 + x_4 = 0 \\ x_1x_3 + x_3x_4 + x_2 + 1 = 0 \\ x_2x_3 + x_2x_4 + x_3x_4 + x_1 + x_4 = 0 \end{cases}$$

becomes a linear system in x_1 and x_2.

Fast enumeration:

Enumerate all possible assignements for the fixed variables using Gray-code enumeration.

Joux-Vitse's Crossbred Algorithm



Fukuoka MQ Challenge ×

https://www.mqchallenge.org

Hall of Fame

Type I | Type II | Type III | Type IV | Type V | Type VI

	Number of Variables (n)	Seed (0,1,2,3,4)	Date	Contestants	Computational Resource	Data
1	74	0	2016/12/17	Antoine Joux	New hybridized XL related algorithm, Heterogeneous cluster of Intel Xeon @ 2.7-3.5 Ghz	Details
2	74	4	2017/11/15	Kai-Chun Ning, Ruben Niederhagen	Parallel Crossbred, 54 GPUs in the Saber cluster	Details
3	74	2	2020/05/07	Yao Sun	Improved Parallel Crossbred, Intel i7 8700, GeForce GTX 1080Ti	Details

For $n = 74$ variables:

- **Joux and Vitse 2016:**
  18 hours on 448 cores
  (expected: 180 hours)
- **Ning and Niederhagen 2017:**
  33 hours on 54 GPUs
  (expected: 76 hours)
- **Sun 2020:**
  82 hours on 10 GPUs
  (expected: 113 hours)

Thank you very much for your attention!