Improved Gröbner basis computation with applications in cryptography

Christian Eder
joint work with: John Perry, Justin Gash, Bjarke Roune Hammersholt and Jean-Charles Faugère

POLSYS Team, UPMC, Paris, France

June 25, 2013

- **Improvement 1: Signature-based Gröbner Basis algorithms**
- **Improvement 2: Specialized Gaussian Elimination**
- **Use GB algorithms in algebraic cryptanalysis**

Definition

$G = \{g_1, \ldots, g_r\}$ is a **Gröbner Basis** for $I = \langle f_1, \ldots, f_m \rangle$ if

1. $G \subset I$ and
2. $\langle \mathrm{lm}(g_1), \ldots, \mathrm{lm}(g_r) \rangle = \langle \mathrm{lm}(f) \mid f \in I \rangle$.

Definition

$G = \{g_1, \ldots, g_r\}$ is a **Gröbner Basis** for $I = \langle f_1, \ldots, f_m \rangle$ if

1. $G \subset I$ and
2. $\langle \mathrm{lm}(g_1), \ldots, \mathrm{lm}(g_r) \rangle = \langle \mathrm{lm}(f) \mid f \in I \rangle$.

Satz (Buchberger's Criterion)

The following are equivalent:

1. *G is a Gröbner Basis for $\langle G \rangle$.*
2. *For all $f, g \in G$ it holds that $\mathrm{spol}(f, g) \xrightarrow{G} 0$, where*

$$\mathrm{spol}(f, g) = \mathrm{lc}(g) \frac{\mathrm{lcm}(\mathrm{lm}(f), \mathrm{lm}(g))}{\mathrm{lm}(f)} f - \mathrm{lc}(f) \frac{\mathrm{lcm}(\mathrm{lm}(f), \mathrm{lm}(g))}{\mathrm{lm}(g)} g.$$

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle$
Output: Gröbner Basis G **for** I

1. $G \leftarrow \emptyset$
2. $G \leftarrow G \cup \{f_i\}$ **for all** $i \in \{1, \ldots, m\}$
3. $P \leftarrow \{(f_i, f_j) \mid f_i, f_j \in G, i > j\}$

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle$
Output: Gröbner Basis G **for** I

1. $G \leftarrow \emptyset$
2. $G \leftarrow G \cup \{f_i\}$ **for all** $i \in \{1, \ldots, m\}$
3. $P \leftarrow \{(f_i, f_j) \mid f_i, f_j \in G, i > j\}$
4. **While** $P \neq \emptyset$
   (a) **Choose** $(f, g) \in P$, $P \leftarrow P \setminus \{(f, g)\}$
   (b) $h \leftarrow \mathrm{spol}(f, g)$

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle$
Output: Gröbner Basis G **for** I

1. $G \leftarrow \emptyset$
2. $G \leftarrow G \cup \{f_i\}$ **for all** $i \in \{1, \ldots, m\}$
3. $P \leftarrow \{(f_i, f_j) \mid f_i, f_j \in G, i > j\}$
4. **While** $P \neq \emptyset$
   (a) **Choose** $(f, g) \in P$, $P \leftarrow P \setminus \{(f, g)\}$
   (b) $h \leftarrow \mathrm{spol}(f, g)$
      (i) **If** $h \xrightarrow{G} 0$

Input: Ideal $I = \langle f_1, \dots, f_m \rangle$
Output: Gröbner Basis G **for** I

1. $G \leftarrow \emptyset$
2. $G \leftarrow G \cup \{f_i\}$ **for all** $i \in \{1, \dots, m\}$
3. $P \leftarrow \{(f_i, f_j) \mid f_i, f_j \in G, i > j\}$
4. **While** $P \neq \emptyset$
   (a) **Choose** $(f, g) \in P$, $P \leftarrow P \setminus \{(f, g)\}$
   (b) $h \leftarrow \text{spol}(f, g)$
      (i) **If** $h \xrightarrow{G} 0$
      (ii) **If** $h \xrightarrow{G} r \neq 0$

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle$
Output: Gröbner Basis G **for** I

1. $G \leftarrow \emptyset$
2. $G \leftarrow G \cup \{f_i\}$ **for all** $i \in \{1, \ldots, m\}$
3. $P \leftarrow \{(f_i, f_j) \mid f_i, f_j \in G, i > j\}$
4. **While** $P \neq \emptyset$
   (a) **Choose** $(f, g) \in P$, $P \leftarrow P \setminus \{(f, g)\}$
   (b) $h \leftarrow \mathrm{spol}(f, g)$
      (i) **If** $h \xrightarrow{G} 0$
      (ii) **If** $h \xrightarrow{G} r \neq 0$
      $P \leftarrow P \cup \{(r, g) \mid g \in G\}$
      $G \leftarrow G \cup \{r\}$
5. **Return** G

Input: Ideal $I = \langle f_1, \ldots, f_m \rangle$
Output: Gröbner Basis G **for** I

1. $G \leftarrow \emptyset$
2. $G \leftarrow G \cup \{f_i\}$ **for all** $i \in \{1, \ldots, m\}$
3. $P \leftarrow \{(f_i, f_j) \mid f_i, f_j \in G, i > j\}$
4. **While** $P \neq \emptyset$
   (a) **Choose** $(f, g) \in P$**,** $P \leftarrow P \setminus \{(f, g)\}$
   (b) $h \leftarrow \text{spol}(f, g)$
      (i) **If** $h \xrightarrow{G} 0 \Rightarrow$ **no new information**
      (ii) **If** $h \xrightarrow{G} r \neq 0 \Rightarrow$ **new information**
      $P \leftarrow P \cup \{(r, g) \mid g \in G\}$
      $G \leftarrow G \cup \{r\}$
5. **Return** G

Example

Let $I = \langle g_1, g_2 \rangle \in \mathbb{Q}[x, y, z]$ be given where $\mathbf{g_1} = \mathbf{xy} - \mathbf{z^2}$, $\mathbf{g_2} = \mathbf{y^2} - \mathbf{z^2}$, and let $<$ be the graded reverse lexicographical ordering.

Example

Let $I = \langle g_1, g_2 \rangle \in \mathbb{Q}[x, y, z]$ be given where $\mathbf{g_1} = \mathbf{xy} - \mathbf{z^2}$, $\mathbf{g_2} = \mathbf{y^2} - \mathbf{z^2}$, and let $<$ be the graded reverse lexicographical ordering.

$$\begin{aligned} \operatorname{spol}(g_2, g_1) &= xg_2 - yg_1 = \mathbf{xy^2} - xz^2 - \mathbf{xy^2} + yz^2 \\ &= -xz^2 + yz^2, \end{aligned}$$

so it reduces w.r.t. G to $\mathbf{g_3} = \mathbf{xz^2} - \mathbf{yz^2}$.

Example

Let $I = \langle g_1, g_2 \rangle \in \mathbb{Q}[x, y, z]$ be given where $\mathbf{g_1} = \mathbf{xy} - \mathbf{z^2}$, $\mathbf{g_2} = \mathbf{y^2} - \mathbf{z^2}$, and let $<$ be the graded reverse lexicographical ordering.

$$\begin{aligned} \mathrm{spol}(g_2, g_1) &= xg_2 - yg_1 = \mathbf{xy^2} - xz^2 - \mathbf{xy^2} + yz^2 \\ &= -xz^2 + yz^2, \end{aligned}$$

so it reduces w.r.t. G to $\mathbf{g_3} = \mathbf{xz^2} - \mathbf{yz^2}$.

$$\mathrm{spol}(g_3, g_1) = \mathbf{xyz^2} - y^2z^2 - \mathbf{xyz^2} + z^4 = -y^2z^2 + z^4.$$

Example

Let $I = \langle g_1, g_2 \rangle \in \mathbb{Q}[x, y, z]$ be given where $\mathbf{g_1 = xy - z^2}$, $\mathbf{g_2 = y^2 - z^2}$, and let $<$ be the graded reverse lexicographical ordering.

$$\begin{aligned}\mathrm{spol}(g_2, g_1) &= xg_2 - yg_1 = \mathbf{xy^2} - xz^2 - \mathbf{xy^2} + yz^2 \\ &= -xz^2 + yz^2,\end{aligned}$$

so it reduces w.r.t. G to $\mathbf{g_3 = xz^2 - yz^2}$.

$$\mathrm{spol}(g_3, g_1) = \mathbf{xyz^2} - y^2z^2 - \mathbf{xyz^2} + z^4 = -y^2z^2 + z^4.$$

We can reduce even further with $z^2 g_2$:

$$-y^2z^2 + z^4 + y^2z^2 - z^4 = 0.$$

Example

Let $I = \langle g_1, g_2 \rangle \in \mathbb{Q}[x, y, z]$ be given where $\mathbf{g_1} = \mathbf{xy} - \mathbf{z^2}$, $\mathbf{g_2} = \mathbf{y^2} - \mathbf{z^2}$, and let $<$ be the graded reverse lexicographical ordering.

$$\begin{aligned} \text{spol}(g_2, g_1) &= xg_2 - yg_1 = \mathbf{xy^2} - xz^2 - \mathbf{xy^2} + yz^2 \\ &= -xz^2 + yz^2, \end{aligned}$$

so it reduces w.r.t. G to $\mathbf{g_3} = \mathbf{xz^2} - \mathbf{yz^2}$.

$$\text{spol}(g_3, g_1) = \mathbf{xyz^2} - y^2z^2 - \mathbf{xyz^2} + z^4 = -y^2z^2 + z^4.$$

We can reduce even further with $z^2 g_2$:

$$-y^2z^2 + z^4 + y^2z^2 - z^4 = 0.$$

$\Rightarrow$ **How can we discard such zero reductions in advance?**

Let $I = \langle f_1, \ldots, f_m \rangle$.
Idea: Give each $f \in I$ a bit more structure:

Let $I = \langle f_1, \ldots, f_m \rangle$.

Idea: Give each $f \in I$ a bit more structure:

1. Let R^m be generated by $e_1, \ldots, e_m$, $\prec$ a well-ordering on the monomials of R^m, and let $\pi : R^m \to R$ such that

$$\pi(e_i) = f_i \text{ for all } i.$$

Let $I = \langle f_1, \ldots, f_m \rangle$.

Idea: Give each $f \in I$ a bit more structure:

1. Let R^m be generated by $e_1, \ldots, e_m$, $\prec$ a well-ordering on the monomials of R^m, and let $\pi : R^m \to R$ such that
$$\pi(e_i) = f_i \text{ for all } i.$$
2. Each $p \in I$ can be represented by an
$$s = \sum_{i=1}^{m} h_i e_i \in R^m \text{ such that } p = \pi(s).$$

Let $I = \langle f_1, \dots, f_m \rangle$.

Idea: Give each $f \in I$ a bit more structure:

1. Let R^m be generated by $e_1, \dots, e_m$, $\prec$ a well-ordering on the monomials of R^m, and let $\pi : R^m \to R$ such that
$$\pi(e_i) = f_i \text{ for all } i.$$
2. Each $p \in I$ can be represented by an
$$s = \sum_{i=1}^{m} h_i e_i \in R^m \text{ such that } p = \pi(s).$$
3. **A signature** of p is given by
$$\mathrm{sig}(p) = \mathrm{lm}_{\prec}(s) \text{ with } p = \pi(s).$$

Let $I = \langle f_1, \ldots, f_m \rangle$.

Idea: Give each $f \in I$ a bit more structure:

1. Let R^m be generated by $e_1, \ldots, e_m$, $\prec$ a well-ordering on the monomials of R^m, and let $\pi : R^m \to R$ such that
$$\pi(e_i) = f_i \text{ for all } i.$$
2. Each $p \in I$ can be represented by an
$$s = \sum_{i=1}^{m} h_i e_i \in R^m \text{ such that } p = \pi(s).$$
3. **A signature** of p is given by
$$\mathrm{sig}(p) = \mathrm{lm}_{\prec}(s) \text{ with } p = \pi(s).$$
4. **A minimal signature** of p exists due to $\prec$.

Our example – now with signatures and $\prec_{\text{pot}}$

We have already computed the following data:

$$
\begin{aligned}
g_1 &= xy - z^2,\ \operatorname{sig}(g_1) = e_1,\\
g_2 &= y^2 - z^2,\ \operatorname{sig}(g_2) = e_2,\\
g_3 &= \operatorname{spol}(g_2, g_1) = xg_2 - yg_1\\
\Rightarrow \operatorname{sig}(g_3) &= x \operatorname{sig}(g_2) = xe_2.
\end{aligned}
$$

Our example – now with signatures and $\prec_{\text{pot}}$

We have already computed the following data:

$$\begin{aligned} g_1 &= xy - z^2,\ \mathrm{sig}(g_1) = e_1,\\ g_2 &= y^2 - z^2,\ \mathrm{sig}(g_2) = e_2,\\ g_3 &= \mathrm{spol}(g_2, g_1) = xg_2 - yg_1\\ \Rightarrow \mathrm{sig}(g_3) &= x\,\mathrm{sig}(g_2) = xe_2. \end{aligned}$$

$\mathrm{spol}(g_3, g_1) = yg_3 - z^2 g_1$:

$$\mathrm{sig}\left(\mathrm{spol}(g_3, g_1)\right) = y\,\mathrm{sig}(g_3) = xye_2.$$

Our example – now with signatures and $\prec_{pot}$

We have already computed the following data:

$$\begin{aligned} g_1 &= xy - z^2, \ \mathrm{sig}(g_1) = e_1, \\ g_2 &= y^2 - z^2, \ \mathrm{sig}(g_2) = e_2, \\ g_3 &= \mathrm{spol}(g_2, g_1) = xg_2 - yg_1 \\ \Rightarrow \mathrm{sig}(g_3) &= x\,\mathrm{sig}(g_2) = xe_2. \end{aligned}$$

$\mathrm{spol}(g_3, g_1) = yg_3 - z^2 g_1$:

$$\mathrm{sig}\left(\mathrm{spol}(g_3, g_1)\right) = y\,\mathrm{sig}(g_3) = xye_2.$$

Note that $\mathrm{sig}\left(\mathrm{spol}(g_3, g_1)\right) = xy\,e_2$ and $\mathrm{lm}(g_1) = xy$.

Our example – now with signatures and $\prec_{pot}$

We have already computed the following data:

$$\begin{aligned} g_1 &= xy - z^2,\ \mathrm{sig}(g_1) = e_1,\\ g_2 &= y^2 - z^2,\ \mathrm{sig}(g_2) = e_2,\\ g_3 &= \mathrm{spol}(g_2, g_1) = xg_2 - yg_1\\ \Rightarrow \mathrm{sig}(g_3) &= x\,\mathrm{sig}(g_2) = xe_2. \end{aligned}$$

$\mathrm{spol}(g_3, g_1) = yg_3 - z^2 g_1$:

$$\mathrm{sig}\left(\mathrm{spol}(g_3, g_1)\right) = y\,\mathrm{sig}(g_3) = xye_2.$$

Note that $\mathrm{sig}\left(\mathrm{spol}(g_3, g_1)\right) = xy\,e_2$ and $\mathrm{lm}(g_1) = xy$.

$\Rightarrow$ **We know that $\mathrm{spol}(g_3, g_1)$ will reduce to zero w.r.t. G.**

Why do we know this?

The general idea is to check the signatures of the generated s-polynomials.

Why do we know this?

The general idea is to check the signatures of the generated s-polynomials.

If $\mathrm{sig}\left(\mathrm{spol}(f, g)\right)$ is not minimal for $\mathrm{spol}(f, g)$ then
$\Rightarrow \mathrm{spol}(f, g)$ is discarded.

Why do we know this?

The general idea is to check the signatures of the generated s-polynomials.

If $\mathrm{sig}\left(\mathrm{spol}(f,g)\right)$ is not minimal for $\mathrm{spol}(f,g)$ then
$\Rightarrow$ $\mathrm{spol}(f,g)$ is discarded.

Our goal

Find and discard as many s-polynomials as possible for which the algorithm computes a non-minimal signature.

The general idea is to check the signatures of the generated s-polynomials.

If $\mathrm{sig}\left(\mathrm{spol}(f,g)\right)$ is not minimal for $\mathrm{spol}(f,g)$ then
$\Rightarrow \mathrm{spol}(f,g)$ is discarded.

Our goal

Find and discard as many s-polynomials as possible for which the algorithm computes a non-minimal signature.

Our task

We need to take care of the correctness of the signatures throughout the computations.

Why do we know this?

The general idea is to check the signatures of the generated s-polynomials.

If $\operatorname{sig}\left(\operatorname{spol}(f,g)\right)$ is not minimal for $\operatorname{spol}(f,g)$ then
$\Rightarrow \operatorname{spol}(f,g)$ is discarded.

Our goal

Find and discard as many s-polynomials as possible for which the algorithm computes a non-minimal signature.

Our task

We need to take care of the correctness of the signatures throughout the computations.

Note

We order P by increasing signatures, so we always take the s-polynomial of minimal signature.

Non-minimal signature (NM)

$\mathrm{sig}(h)$ not minimal for h? $\Rightarrow$ Remove h.

Non-minimal signature (NM)

$\mathrm{sig}(h)$ not minimal for h? $\Rightarrow$ Remove h.

Sketch of proof

1. There exists a syzygy $s \in R^m$ such that $\mathrm{lm}(s) = \mathrm{sig}(h)$.
   $\Rightarrow$ We can represent h with a lower signature.
2. Pairs are handled by increasing signatures.
   $\Rightarrow$ All relations of lower signature are already taken care of.

$\square$

Non-minimal signature (NM)

sig(h) not minimal for h? $\Rightarrow$ Remove h.

Sketch of proof

1. There exists a syzygy $s \in R^m$ such that $\operatorname{lm}(s) = \operatorname{sig}(h)$.
   $\Rightarrow$ We can represent h with a lower signature.
2. Pairs are handled by increasing signatures.
   $\Rightarrow$ All relations of lower signature are already taken care of.

$\square$

Our example with $\prec_{\text{pot}}$ revisited

$\operatorname{sig}(\operatorname{spol}(g_3, g_1)) = xye_2$

$$\left.\begin{array}{l} g_1 = xy - z^2 \\ g_2 = y^2 - z^2 \end{array}\right\} \Rightarrow \operatorname{psyz}(g_2, g_1) = g_1 e_2 - g_2 e_1 = xye_2 + \dots$$

Rewritable signature (RW)

$\mathrm{sig}(g) = \mathrm{sig}(h)$? $\Rightarrow$ Remove either g or h.

Rewritable signature (RW)

$\mathrm{sig}(g) = \mathrm{sig}(h)? \Rightarrow$ Remove either g or h.

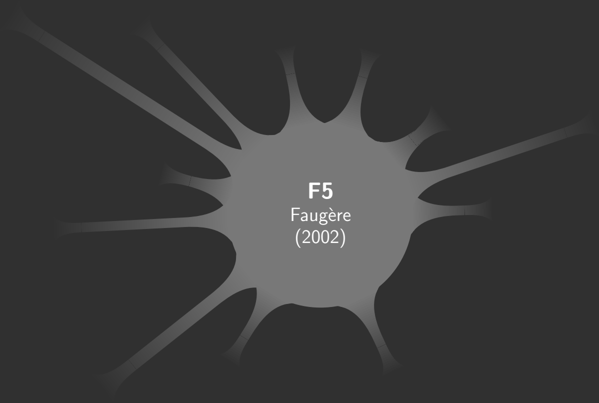
Sketch of proof

1. $\mathrm{sig}(g - h) \prec \mathrm{sig}(g), \mathrm{sig}(h)$.
2. Pairs are handled by increasing signatures.
   $\Rightarrow$ All necessary computations of lower signature have already taken place.
   $\Rightarrow$ We can represent h by

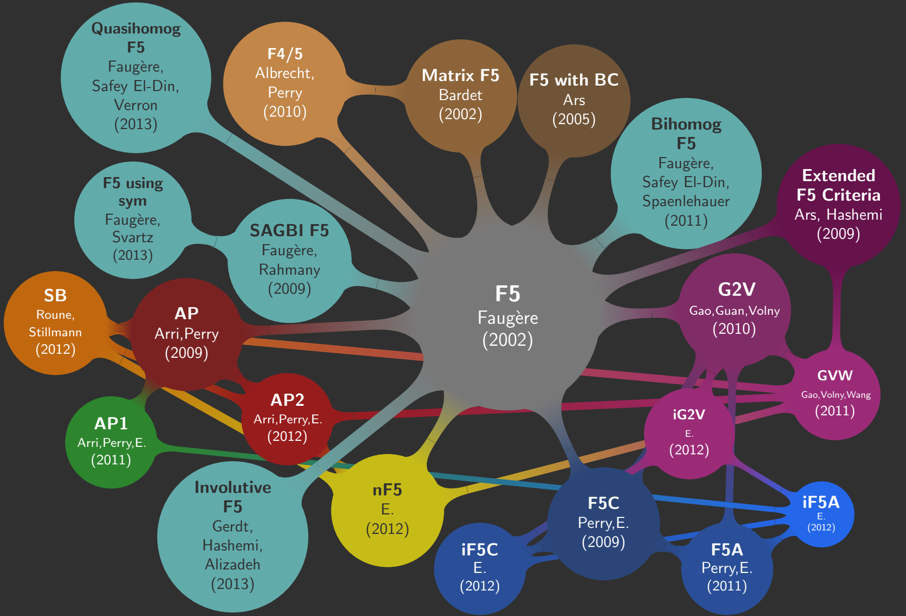
$$h = g + \text{ elements of lower signature.}$$

$\square$

A good decade on signature-based algorithms



A good decade on signature-based algorithms



- Improvement 1: Signature-based Gröbner Basis algorithms
- **Improvement 2: Specialized Gaussian Elimination**
- Use GB algorithms in algebraic cryptanalysis

Use **Linear Algebra** for reduction steps in GB computations.

Improve Gaussian Elimination

Use **Linear Algebra** for reduction steps in GB computations.

$$\begin{matrix} 1 & 3 & 0 & 0 & 7 & 1 & 0 \\ 1 & 0 & 4 & 1 & 0 & 0 & 5 \\ 0 & 1 & 6 & 0 & 8 & 0 & 1 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 3 & 1 \end{matrix}$$

Improve Gaussian Elimination

Use **Linear Algebra** for reduction steps in GB computations.

$$\begin{matrix} 1 & 3 & 0 & 0 & 7 & 1 & 0 \\ 1 & 0 & 4 & 1 & 0 & 0 & 5 \\ 0 & 1 & 6 & 0 & 8 & 0 & 1 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 3 & 1 \end{matrix}$$

Knowledge of underlying GB structure

Improve Gaussian Elimination

Use **Linear Algebra** for reduction steps in GB computations.

$$
\begin{array}{rl}
\text{s-polynomial} & \left\{ \begin{array}{ccccccc} 1 & 3 & 0 & 0 & 7 & 1 & 0 \\ 1 & 0 & 4 & 1 & 0 & 0 & 5 \end{array} \right. \\
\text{s-polynomial} & \left\{ \begin{array}{ccccccc} 0 & 1 & 6 & 0 & 8 & 0 & 1 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \end{array} \right. \\
\text{reducer} \quad \leftarrow & \begin{array}{ccccccc} 0 & 0 & 0 & 0 & 1 & 3 & 1 \end{array}
\end{array}
$$

Knowledge of underlying GB structure

Use **Linear Algebra** for reduction steps in GB computations.

$$
\begin{array}{rc}
\text{s-polynomial} & \left\{ \begin{array}{ccccccc} 1 & 3 & 0 & 0 & 7 & 1 & 0 \\ 1 & 0 & 4 & 1 & 0 & 0 & 5 \end{array} \right. \\
\text{s-polynomial} & \left\{ \begin{array}{ccccccc} 0 & 1 & 6 & 0 & 8 & 0 & 1 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \end{array} \right. \\
\text{reducer} \quad \leftarrow & \begin{array}{ccccccc} 0 & 0 & 0 & 0 & 1 & 3 & 1 \end{array}
\end{array}
$$

Knowledge of underlying GB structure

Use **Linear Algebra** for reduction steps in GB computations.

$$
\begin{array}{ll}
\text{s-polynomial} & \left\{\begin{array}{ccccccc}1&3&0&0&7&1&0\\1&0&4&1&0&0&5\end{array}\right. \\
\text{s-polynomial} & \left\{\begin{array}{ccccccc}0&1&6&0&8&0&1\\0&5&0&0&0&2&0\end{array}\right. \\
\text{reducer} & \leftarrow \begin{array}{ccccccc}0&0&0&0&1&3&1\end{array}
\end{array}
$$

Knowledge of underlying GB structure

Idea

Do a static **reordering before** the Gaussian Elimination to achieve a better initial shape. **Reorder afterwards**.

1st step: Sort pivot and non-pivot columns

$$\begin{matrix} 1 & 3 & 0 & 0 & 7 & 1 & 0 \\ 1 & 0 & 4 & 1 & 0 & 0 & 5 \\ 0 & 1 & 6 & 0 & 8 & 0 & 1 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 3 & 1 \end{matrix}$$

1st step: Sort pivot and non-pivot columns

1	3	0	0	7	1	0
1	0	4	1	0	0	5
0	1	6	0	8	0	1
0	5	0	0	0	2	0
0	0	0	0	1	3	1

Pivot column

1st step: Sort pivot and non-pivot columns

$$\begin{matrix} 1 & 3 & 0 & 0 & 7 & 1 & 0 \\ 1 & 0 & 4 & 1 & 0 & 0 & 5 \\ 0 & 1 & 6 & 0 & 8 & 0 & 1 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 3 & 1 \end{matrix}$$

Pivot column

1st step: Sort pivot and non-pivot columns

$$\begin{matrix} 1 & 3 & 0 & 0 & 7 & 1 & 0 \\ 1 & 0 & 4 & 1 & 0 & 0 & 5 \\ 0 & 1 & 6 & 0 & 8 & 0 & 1 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 3 & 1 \end{matrix}$$

Pivot column

Non-Pivot column

Faugère-Lachartre Idea

1st step: Sort pivot and non-pivot columns

$$\begin{array}{ccccccc}
1 & 3 & 0 & 0 & 7 & 1 & 0 \\
1 & 0 & 4 & 1 & 0 & 0 & 5 \\
0 & 1 & 6 & 0 & 8 & 0 & 1 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 0 & 0 & 1 & 3 & 1
\end{array}$$

Pivot column

Non-Pivot column

Faugère-Lachartre Idea

1st step: Sort pivot and non-pivot columns

$$
\begin{array}{ccccccc}
1 & 3 & 0 & 0 & 7 & 1 & 0 \\
1 & 0 & 4 & 1 & 0 & 0 & 5 \\
0 & 1 & 6 & 0 & 8 & 0 & 1 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 0 & 0 & 1 & 3 & 1
\end{array}
\longrightarrow
\begin{array}{ccc|cccc}
1 & 3 & 7 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 4 & 1 & 0 & 5 \\
0 & 1 & 8 & 6 & 0 & 0 & 9 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 0 & 0 & 3 & 1
\end{array}
$$

Pivot column

Non-Pivot column

2nd step: Sort pivot and non-pivot rows

$$\begin{array}{ccc:cccc}
1 & 3 & 7 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 4 & 1 & 0 & 5 \\
0 & 1 & 8 & 6 & 0 & 0 & 9 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 0 & 0 & 3 & 1
\end{array}$$

2nd step: Sort pivot and non-pivot rows

$$\begin{array}{ccc:cccc}
1 & 3 & 7 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 4 & 1 & 0 & 5 \\
0 & 1 & 8 & 6 & 0 & 0 & 9 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 0 & 0 & 3 & 1
\end{array}$$

Pivot row

2nd step: Sort pivot and non-pivot rows

$$\begin{array}{ccc:cccc}
1 & 3 & 7 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 4 & 1 & 0 & 5 \\
0 & 1 & 8 & 6 & 0 & 0 & 9 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 0 & 0 & 3 & 1
\end{array}$$

Pivot row

Non-Pivot row

Faugère-Lachartre Idea

2nd step: Sort pivot and non-pivot rows

$$
\begin{array}{ccc:cccc}
1 & 3 & 7 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 4 & 1 & 0 & 5 \\
0 & 1 & 8 & 6 & 0 & 0 & 9 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 0 & 0 & 3 & 1
\end{array}
$$

Pivot row

Non-Pivot row

Faugère-Lachartre Idea

2nd step: Sort pivot and non-pivot rows

$$
\begin{array}{ccc:cccc}
1 & 3 & 7 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 4 & 1 & 0 & 5 \\
0 & 1 & 8 & 6 & 0 & 0 & 9 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 0 & 0 & 3 & 1
\end{array}
\longrightarrow
\begin{array}{ccc:cccc}
1 & 0 & 0 & 4 & 1 & 0 & 5 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 0 & 0 & 3 & 1 \\
\hdashline
1 & 3 & 7 & 0 & 0 & 1 & 0 \\
0 & 1 & 8 & 6 & 0 & 0 & 9
\end{array}
$$

Pivot row

Non-Pivot row

3rd step: Reduce lower left part to zero

$$\left.\begin{array}{ccc|cccc}
1 & 0 & 0 & 4 & 1 & 0 & 5 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 0 & 0 & 3 & 1 \\
\hline
1 & 3 & 7 & 0 & 0 & 1 & 0 \\
0 & 1 & 8 & 6 & 0 & 0 & 9
\end{array}\right.$$

3rd step: Reduce lower left part to zero

$$
\left(\begin{array}{ccc:cccc}
1 & 0 & 0 & 4 & 1 & 0 & 5 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 0 & 0 & 3 & 1 \\
\hdashline
1 & 3 & 7 & 0 & 0 & 1 & 0 \\
0 & 1 & 8 & 6 & 0 & 0 & 9
\end{array}\right)
\longrightarrow
\left(\begin{array}{ccc:cccc}
1 & 0 & 0 & 4 & 1 & 0 & 5 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 0 & 0 & 3 & 1 \\
\hdashline
0 & 0 & 0 & 7 & 10 & 3 & 10 \\
0 & 0 & 0 & 6 & 0 & 2 & 1
\end{array}\right)
$$

4th step: Reduce lower right part

$$
\begin{array}{ccc|cccc}
1 & 0 & 0 & 4 & 1 & 0 & 5 \\
0 & 5 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 0 & 0 & 3 & 1 \\
\hline
0 & 0 & 0 & 7 & 10 & 3 & 10 \\
0 & 0 & 0 & 6 & 0 & 2 & 1
\end{array}
$$

4th step: Reduce lower right part

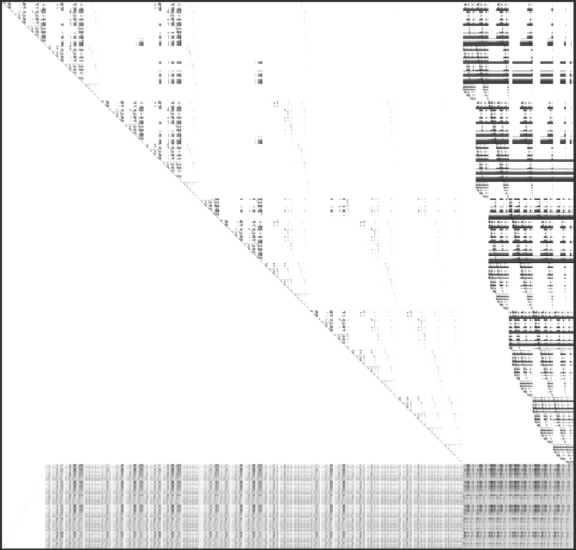
$$\left(\begin{array}{ccc|cccc} 1 & 0 & 0 & 4 & 1 & 0 & 5 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 3 & 1 \\ \hline 0 & 0 & 0 & 7 & 10 & 3 & 10 \\ 0 & 0 & 0 & 6 & 0 & 2 & 1 \end{array}\right) \longrightarrow \left(\begin{array}{ccc|cccc} 1 & 0 & 0 & 4 & 1 & 0 & 5 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 3 & 1 \\ \hline 0 & 0 & 0 & 7 & 10 & 3 & 10 \\ 0 & 0 & 0 & 0 & 4 & 1 & 5 \end{array}\right)$$

4th step: Reduce lower right part

$$\left(\begin{array}{ccc|cccc} 1 & 0 & 0 & 4 & 1 & 0 & 5 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 3 & 1 \\ \hline 0 & 0 & 0 & 7 & 10 & 3 & 10 \\ 0 & 0 & 0 & 6 & 0 & 2 & 1 \end{array}\right) \longrightarrow \left(\begin{array}{ccc|cccc} 1 & 0 & 0 & 4 & 1 & 0 & 5 \\ 0 & 5 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 3 & 1 \\ \hline 0 & 0 & 0 & 7 & 10 & 3 & 10 \\ 0 & 0 & 0 & 0 & 4 & 1 & 5 \end{array}\right)$$

5th step: Remap columns of lower right part

How our matrices look like



Improvements:

- Use knowledge of underlying GB structures
- Parallelization of Linear Algebra
- Divide sparse and dense data as much as possible

Faugère-Lachartre Idea

Improvements:

- Use knowledge of underlying GB structures
- Parallelization of Linear Algebra
- Divide sparse and dense data as much as possible

Recent research:

- Improve parallelization
- Better usage of cache:
  Use small blocks inside matrix per thread
- Use more of the polynomials structure
- Relax idea of signature-based GB algorithms

- Improvement 1: Signature-based Gröbner Basis algorithms

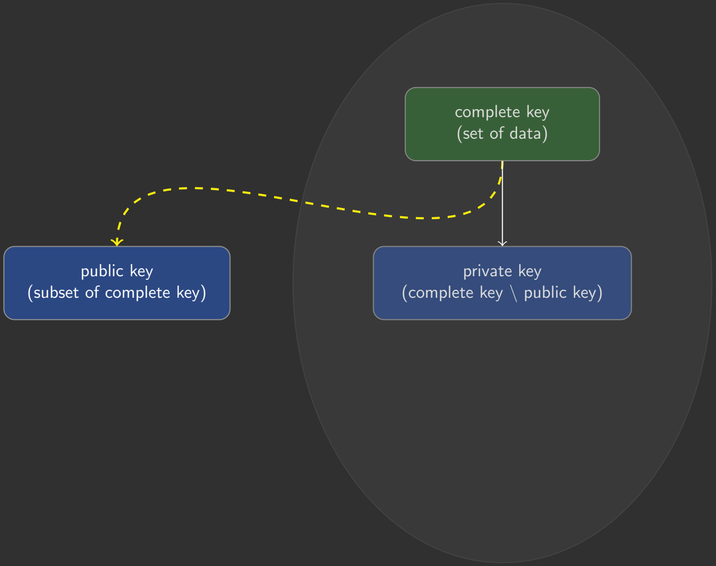
- Improvement 2: Specialized Gaussian Elimination

- **Use GB algorithms in algebraic cryptanalysis**

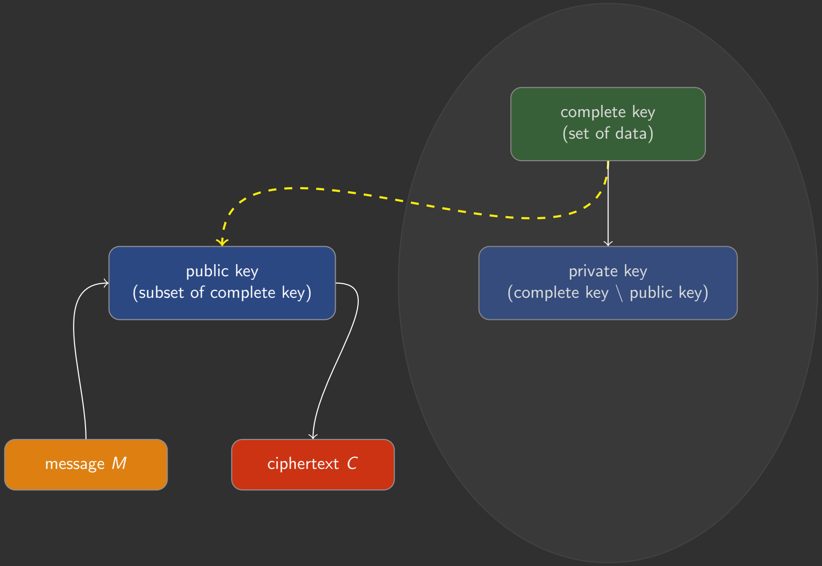
General idea of asymmetric cryptography

complete key
(set of data)

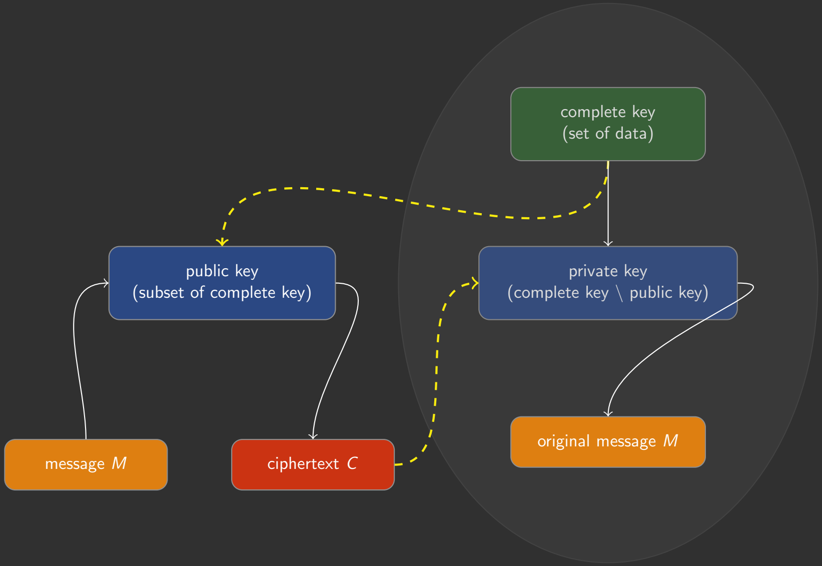
General idea of asymmetric cryptography



General idea of asymmetric cryptography



General idea of asymmetric cryptography



Choice of HFE Polynomial

Choose **private polynomial** p such that

- $p \in F_{q^n}(x)$ (mostly $q = 2$),
- $\deg(p) = d$,
- p is "easily" invertible over F_{q^n}, i.e. find any solution of $p(x) = y$.

Choose **private polynomial** p such that

- $p \in F_{q^n}(x)$ (mostly $q = 2$),
- $\deg(p) = d$,
- p is "easily" invertible over F_{q^n}, i.e. find any solution of $p(x) = y$.

Common choice:

$$p(x) = \sum_{i,j} \alpha_{i,j} x^{q^{u_{i,j}} + q^{v_{i,j}}} + \sum_k \beta_k x^{q^{w_k}} + \gamma.$$

Choose **private polynomial** p such that

- $p \in F_{q^n}(x)$ (mostly $q = 2$),
- $\deg(p) = d$,
- p is "easily" invertible over F_{q^n}, i.e. find any solution of $p(x) = y$.

Common choice:

$$p(x) = \sum_{i,j} \alpha_{i,j} x^{q^{u_{i,j}} + q^{v_{i,j}}} + \sum_k \beta_k x^{q^{w_k}} + \gamma.$$

Note

- Greater $d \implies$ greater security

Choose **private polynomial** p such that

- $p \in F_{q^n}(x)$ (mostly $q = 2$),
- $\deg(p) = d$,
- p is "easily" invertible over F_{q^n}, i.e. find any solution of $p(x) = y$.

Common choice:

$$p(x) = \sum_{i,j} \alpha_{i,j} x^{q^{u_{i,j}} + q^{v_{i,j}}} + \sum_{k} \beta_k x^{q^{w_k}} + \gamma.$$

Note

- Greater $d \implies$ greater security
- Complexity of computing p^{-1} depends quadratically on d.

Choice of HFE Polynomial

Choose **private polynomial** p such that

- $p \in F_{q^n}(x)$ (mostly $q = 2$),
- $\deg(p) = d$,
- p is "easily" invertible over F_{q^n}, i.e. find any solution of $p(x) = y$.

Common choice:

$$p(x) = \sum_{i,j} \alpha_{i,j} x^{q^{u_{i,j}} + q^{v_{i,j}}} + \sum_k \beta_k x^{q^{w_k}} + \gamma.$$

Note

- Greater $d \implies$ greater security
- Complexity of computing p^{-1} depends quadratically on d.

$$\implies d \leq 512.$$

Generate public key

Represent p publicly such that original structure and inversion are hidden:

Generate public key

Represent p publicly such that original structure and inversion are hidden:

- Represent F_{q^n} as F_q vector space.
- Choose 2 linear transformations S and T.

Represent p publicly such that original structure and inversion are hidden:

- Represent F_{q^n} as F_q vector space.
- Choose 2 linear transformations S and T.

$$\Longrightarrow \textbf{public key } T \circ p \circ S.$$

Represent p publicly such that original structure and inversion are hidden:

- Represent F_{q^n} as F_q vector space.
- Choose 2 linear transformations S and T.

$$\Longrightarrow \textbf{public key } T \circ p \circ S.$$

Assume $q = 2$

Frobenius map on F_{2^n} is a linear transformation over F_2 on F_{2^n}:

$$\begin{array}{ccc} \alpha_{i,j} x^{2^{u_{i,j}} + 2^{v_{i,j}}} & \longrightarrow & \text{quadratic term} \\ \sum_k \beta_k x^{2^{w_k}} & \longrightarrow & \text{linear term} \\ \gamma & \longrightarrow & \text{constant term} \end{array}$$

Represent p publicly such that original structure and inversion are hidden:

- Represent F_{q^n} as F_q vector space.
- Choose 2 linear transformations S and T.

$$\Longrightarrow \textbf{public key } T \circ p \circ S.$$

Assume $q = 2$

Frobenius map on F_{2^n} is a linear transformation over F_2 on F_{2^n}:

$$\begin{array}{ccc} \alpha_{i,j} x^{2^{u_{i,j}} + 2^{v_{i,j}}} & \longrightarrow & \text{quadratic term} \\ \sum_k \beta_k x^{2^{w_k}} & \longrightarrow & \text{linear term} \\ \gamma & \longrightarrow & \text{constant term} \end{array}$$

system of n quadratic equations in n variables over F_2

Public key: n multivariate polynomials $(p_1, \ldots, p_n)$ over F_q.

HFE Encryption

Public key: n multivariate polynomials $(p_1, \ldots, p_n)$ over F_q.

$\implies$ Transform message $M \in F_{q^n}$ to F_q^n, i.e. $M = (x_1, \ldots, x_n)$.

HFE Encryption

Public key: n multivariate polynomials $(p_1, \ldots, p_n)$ over F_q.

$\Longrightarrow$ Transform message $M \in F_{q^n}$ to F_q^n, i.e. $M = (x_1, \ldots, x_n)$.

Encryption: Evaluate each p_i at M.

$\Longrightarrow$ Ciphertext $C = (p_1(x_1, \ldots, x_n), \ldots, p_n(x_1, \ldots, x_n)) \in F_q^n$.

HFE Encryption

Public key: n multivariate polynomials $(p_1, \ldots, p_n)$ over F_q.

$\Longrightarrow$ Transform message $M \in F_{q^n}$ to F_q^n, i.e. $M = (x_1, \ldots, x_n)$.

Encryption: Evaluate each p_i at M.

$\Longrightarrow$ Ciphertext $C = (p_1(x_1, \ldots, x_n), \ldots, p_n(x_1, \ldots, x_n)) \in F_q^n$.

Or in terms of p, S and T (those are not available to the public):

Public key: n multivariate polynomials $(p_1, \ldots, p_n)$ over F_q.

$\Longrightarrow$ Transform message $M \in F_{q^n}$ to F_q^n, i.e. $M = (x_1, \ldots, x_n)$.

Encryption: Evaluate each p_i at M.

$\Longrightarrow$ Ciphertext $C = (p_1(x_1, \ldots, x_n), \ldots, p_n(x_1, \ldots, x_n)) \in F_q^n$.

Or in terms of p, S and T (those are not available to the public):

- Apply S to M: $S(x_1, \ldots, x_n) \Longrightarrow x'$ $(\in F_{q^n})$.

HFE Encryption

Public key: n multivariate polynomials $(p_1, \ldots, p_n)$ over F_q.

$\Longrightarrow$ Transform message $M \in F_{q^n}$ to F_q^n, i.e. $M = (x_1, \ldots, x_n)$.

Encryption: Evaluate each p_i at M.

$\Longrightarrow$ Ciphertext $C = (p_1(x_1, \ldots, x_n), \ldots, p_n(x_1, \ldots, x_n)) \in F_q^n$.

Or in terms of p, S and T (those are not available to the public):

- Apply S to M: $S(x_1, \ldots, x_n) \Longrightarrow x'$ $(\in F_{q^n})$.
- Evaluate $p(x') = y' \Longrightarrow (y'_1, \ldots, y'_n) \in F_q^n$.

Public key: n multivariate polynomials $(p_1, \ldots, p_n)$ over F_q.

$\Longrightarrow$ Transform message $M \in F_{q^n}$ to F_q^n, i.e. $M = (x_1, \ldots, x_n)$.

Encryption: Evaluate each p_i at M.

$\Longrightarrow$ Ciphertext $C = (p_1(x_1, \ldots, x_n), \ldots, p_n(x_1, \ldots, x_n)) \in F_q^n$.

Or in terms of p, S and T (those are not available to the public):

- Apply S to M: $S(x_1, \ldots, x_n) \Longrightarrow x'$ $(\in F_{q^n})$.
- Evaluate $p(x') = y' \Longrightarrow (y'_1, \ldots, y'_n) \in F_q^n$.
- Apply $T \Longrightarrow C = Ty' \in F_q^n$.

HFE Decryption

Simply put: Take C and apply T^{-1}, p^{-1} and S^{-1}.

Simply put: Take C and apply T^{-1}, p^{-1} and S^{-1}.

- Computing S^{-1} and T^{-1} is easy.

Simply put: Take C and apply T^{-1}, p^{-1} and S^{-1}.

- Computing S^{-1} and T^{-1} is easy.
- Finding solutions for $p(x') = y'$ is crucial:
  - $\deg(p) = d \Longrightarrow$ at most d **different** solutions for one y' (p not nec. one-to-one).
  - Redundancy r is added to message M to get a unique solution.

Simply put: Take C and apply T^{-1}, p^{-1} and S^{-1}.

- Computing S^{-1} and T^{-1} is easy.
- Finding solutions for $p(x') = y'$ is crucial:
  - $\deg(p) = d \Longrightarrow$ at most d **different** solutions for one y' (p not nec. one-to-one).
  - Redundancy r is added to message M to get a unique solution.

How to break the system ?

Simply put: Take C and apply T^{-1}, p^{-1} and S^{-1}.

- Computing S^{-1} and T^{-1} is easy.
- Finding solutions for $p(x') = y'$ is crucial:
  - $\deg(p) = d \Longrightarrow$ at most d **different** solutions for one y' (p not nec. one-to-one).
  - Redundancy r is added to message M to get a unique solution.

How to break the system ?

Solve a system of multivariate quadratic polynomials over F_q:

$$\begin{array}{rcl} p_1(x_1, \ldots, x_n) & = & y_1 \\ \vdots & \vdots & \vdots \\ p_n(x_1, \ldots, x_n) & = & y_n \end{array}$$

Patarin defined the so-called **HFE Challenge 1** by

- $d = 96$,
- $q = 2$,
- $n = 80$.

Patarin defined the so-called **HFE Challenge 1** by

- $d = 96$,
- $q = 2$,
- $n = 80$.

Faugère broke this system computing a Gröbner basis of the corresponding system of quadratic multivariate polynomials over F_2 in 2002 using a specialized $F5$ Algorithm:

Patarin defined the so-called **HFE Challenge 1** by

- $d = 96$,
- $q = 2$,
- $n = 80$.

Faugère broke this system computing a Gröbner basis of the corresponding system of quadratic multivariate polynomials over F_2 in 2002 using a specialized $F5$ Algorithm:

96 hours of CPU time on an HP workstation with an alpha EV68 processor at 1 GHz and 4 GB RAM
(Whole computation approx. 7.65 GB.)

Bibliography

[AP10] M. Albrecht und J. Perry. F4/5

[AP11] A. Arri und J. Perry. The F5 Criterion revised

[EGP11] C. Eder, J. Gash and J. Perry. Modifying Faugère's F5 Algorithm to ensure termination

[EP10] C. Eder and J. Perry. F5C: A variant of Faugère's F5 Algorithm with reduced Gröbner bases

[EP11] C. Eder and J. Perry. Signature-based algorithms to compute Gröbner bases

[ER13] C. Eder and B. H. Roune. Signature Rewriting in Gröbner Basis Computation

[F99] J.-C. Faugère. A new efficient algorithm for computing Gröbner bases (F4)

[F02] J.-C. Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero F_5

[FJ03] J.-C. Faugère and A. Joux. Algebraic Cryptanalysis of Hidden Field Equation (HFE) Cryptosystems Using Gröbner Bases

[FL10] J.-C. Faugère and S. Lachartre. Parallel Gaussian Elimination for Gröbner bases computations in finite fields

[GGV10] S. Gao, Y. Guan and F. Volny IV. A New Incremental Algorithm for Computing Gröbner Bases

[GVW11] S. Gao, F. Volny IV and M. Wang. A New Algorithm For Computing Grobner Bases

[P96] J. Patarin. Hidden Field Equations (HFE) and Isomorphic Polynomial (IP): two new families of asymmetric algorithm

[RS12] B. H. Roune and M. Stillman. Practical Gröbner Basis Computation