An efficient structural attack on NIST submission DAGS

Élise Barelli[1] and Alain Couvreur [2,3]

[1]Université de Versailles Saint Quentin
[2]INRIA
[3]LIX, École polytechnique

Asiacrypt 2018

Context

- **DAGS** is a proposal to NIST call for post quantum cryptography.
- McEliece-like public key encryption scheme (+ conversion to a KEM).
- Based on quasi–dyadic alternant codes.
- Original parameters :

Security	n	$\dim \mathscr{C}_{\text{pub}}$	Ground field	$\mathcal{G}$	Key size
128	832	416	$\mathbb{F}_{32}$	$(\mathbb{Z}/2\mathbb{Z})^4$	6.8 kB
192	1216	512	$\mathbb{F}_{64}$	$(\mathbb{Z}/2\mathbb{Z})^5$	8.5 kB
256	2112	704	$\mathbb{F}_{64}$	$(\mathbb{Z}/2\mathbb{Z})^6$	11.6 kB

Note. Parameters have been updated (see further).

(Generalised) Reed–Solomon codes

Definition 1 (Reed–Solomon codes)

Let n, k be positive integers $k \leqslant n$. Let $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{F}_q^n$ be a vector with distinct entries

$$\mathbf{RS}_k(\boldsymbol{x}) \stackrel{\text{def}}{=} \{(f(x_1), \ldots, f(x_n)) \mid \deg(f) < k\}.$$

(Generalised) Reed–Solomon codes

Definition 1 (Reed–Solomon codes)

Let n, k be positive integers $k \leqslant n$. Let $\boldsymbol{x} = (x_1, \dots, x_n) \in \mathbb{F}_q^n$ be a vector with distinct entries

$$\mathbf{RS}_k(\boldsymbol{x}) \stackrel{\text{def}}{=} \{(f(x_1), \dots, f(x_n)) \mid \deg(f) < k\}.$$

Definition 2 (Generalised Reed–Solomon codes)

Let n, k be positive integers $k \leqslant n$. Let $\boldsymbol{x} = (x_1, \dots, x_n) \in \mathbb{F}_q^n$ be a vector with distinct entries and $\boldsymbol{y} = (y_1, \dots, y_n) \in (\mathbb{F}_q^\times)^n$.

$$\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \stackrel{\text{def}}{=} \{(y_1 f(x_1), \dots, y_n f(x_n)) \mid \deg(f) < k\}.$$

(Generalised) Reed–Solomon codes

Definition 1 (Reed–Solomon codes)

Let n, k be positive integers $k \leqslant n$. Let $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{F}_q^n$ be a vector with distinct entries

$$\mathbf{RS}_k(\boldsymbol{x}) \stackrel{\text{def}}{=} \{(f(x_1), \ldots, f(x_n)) \mid \deg(f) < k\}.$$

Definition 2 (Generalised Reed–Solomon codes)

Let n, k be positive integers $k \leqslant n$. Let $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{F}_q^n$ be a vector with distinct entries and $\boldsymbol{y} = (y_1, \ldots, y_n) \in (\mathbb{F}_q^\times)^n$.

$$\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \stackrel{\text{def}}{=} \{(y_1 f(x_1), \ldots, y_n f(x_n)) \mid \deg(f) < k\}.$$

Claim. For such codes one can correct up to $\frac{n-k}{2}$ errors in polynomial time.

Alternant codes

Definition 3 (Alternant codes)

Let n, k be positive integers $k \leqslant n$. Let $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{F}_{q^m}^n$ be a vector with distinct entries and $\boldsymbol{y} = (y_1, \ldots, y_n) \in (\mathbb{F}_{q^m}^{\times})^n$. An alternant code is a code of the form

$$\mathbf{GRS}_r(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n.$$

Alternant codes

Definition 3 (Alternant codes)

Let n, k be positive integers $k \leqslant n$. Let $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{F}_{q^m}^n$ be a vector with distinct entries and $\boldsymbol{y} = (y_1, \ldots, y_n) \in (\mathbb{F}_{q^m}^\times)^n$. An alternant code is a code of the form

$$\mathbf{GRS}_r(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n.$$

Alternant codes

Definition 3 (Alternant codes)

Let n, k be positive integers $k \leqslant n$. Let $\boldsymbol{x} = (x_1, \dots, x_n) \in \mathbb{F}_{q^m}^n$ be a vector with distinct entries and $\boldsymbol{y} = (y_1, \dots, y_n) \in (\mathbb{F}_{q^m}^\times)^n$. An alternant code is a code of the form

$$\mathbf{GRS}_r(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n.$$

Fact 1. Alternant codes inherit from generalised Reed–Solomon decoding algorithms.

Alternant codes

Definition 3 (Alternant codes)

Let n, k be positive integers $k \leqslant n$. Let $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{F}_{q^m}^n$ be a vector with distinct entries and $\boldsymbol{y} = (y_1, \ldots, y_n) \in (\mathbb{F}_{q^m}^\times)^n$. An alternant code is a code of the form

$$\mathbf{GRS}_r(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n.$$

Fact 1. Alternant codes inherit from generalised Reed–Solomon decoding algorithms.

Fact 2. Their parameters are not as good as GRS codes, but they are much less structured which is interesting for cryptography.

History – McEliece (1978)

- 1978 : McEliece's original proposal based on binary Goppa codes (special case of alternant codes). Public key : 32kB for ≈ 80 bits of security[1].
- 2018 : NIST proposal : Classic McEliece. Public key $> 1MB$ for > 256 bits of security.

[1] With respect to Prange algorithm

History – McEliece (1978)

- 1978 : McEliece's original proposal based on binary Goppa codes (special case of alternant codes). Public key : 32kB for $\approx$ 80 bits of security[1].
- 2018 : NIST proposal : Classic McEliece. Public key $> 1MB$ for $>$ 256 bits of security.

During these 40 years many attempts to get shorter keys.

[1] With respect to Prange algorithm

History – McEliece (1978)

- 1978 : McEliece's original proposal based on binary Goppa codes (special case of alternant codes). Public key : 32kB for $\approx$ 80 bits of security[1].
- 2018 : NIST proposal : Classic McEliece. Public key $> 1MB$ for > 256 bits of security.

During these 40 years many attempts to get shorter keys. **How?**

[1] With respect to Prange algorithm

Idea 1 : Reducing the extension degree

$$
\begin{tikzcd}
\mathbb{F}_{q^m} \arrow[d, dash, "m"] & \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \arrow[d, dash] \\
\mathbb{F}_q & \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n
\end{tikzcd}
$$

Fact. The larger the m the worse the parameters. But:

Idea 1 : Reducing the extension degree

$$
\begin{array}{ccc}
\mathbb{F}_{q^m} & \qquad & \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \\
\Big| \Big) \, m & & \Big| \\
\mathbb{F}_q & & \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n
\end{array}
$$

Fact. The larger the m the worse the parameters. But:

- Case $m = 1$ is broken (Sidelnikov, Shestakov 1992);
- Some specific cases of $m = 2$ and 3 called *wild Goppa codes* are broken too:
  - C., Otmani, Tillich, 2014;
  - Faugère, Perret, de Portzamparc, 2014

Idea 2 : Using codes with a non trivial automorphism group

- **Advantage.** Permits to reduce the public key size with almost no incidence on the security

Idea 2 : Using codes with a non trivial automorphism group

- **Advantage.** Permits to reduce the public key size with almost no incidence on the security **w.r.t. message security attacks.**

Idea 2 : Using codes with a non trivial automorphism group

- **Advantage.** Permits to reduce the public key size with almost no incidence on the security **w.r.t. message security attacks.**
- **But, may affect the security w.r.t. key recovery attacks.**

Idea 2 : Using codes with a non trivial automorphism group

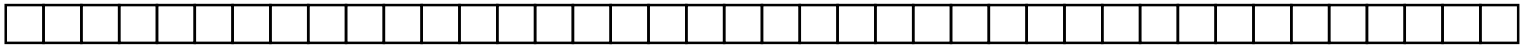
- **Advantage.** Permits to reduce the public key size with almost no incidence on the security **w.r.t. message security attacks.**
- **But, may affect the security w.r.t. key recovery attacks.**

Some tempting choices of using large groups lead to key recovery attacks:

- Otmani, Tillich, Dallot (2008);
- Faugère, Otmani, Perret, Tillich (2010);
- Faugère, Otmani, Perret, Tillich, de Portzamparc (2016).

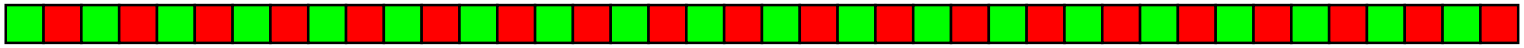
DAGS

DAGS scheme's public keys are Quasi–dyadic alternant codes. i.e. $\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n$ with an automorphism group acting as:



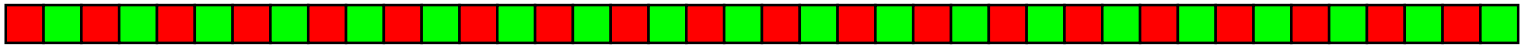
DAGS

DAGS scheme's public keys are Quasi–dyadic alternant codes. i.e. $\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n$ with an automorphism group acting as:



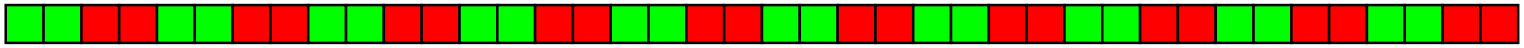
DAGS

DAGS scheme's public keys are Quasi–dyadic alternant codes. i.e. $\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n$ with an automorphism group acting as:



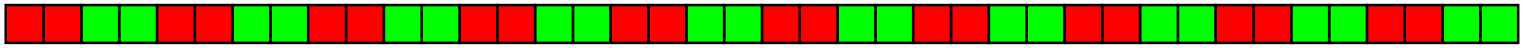
DAGS

DAGS scheme's public keys are Quasi–dyadic alternant codes. i.e. $\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n$ with an automorphism group acting as:



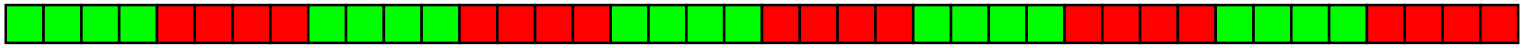
DAGS

DAGS scheme's public keys are Quasi–dyadic alternant codes. i.e. $\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n$ with an automorphism group acting as:



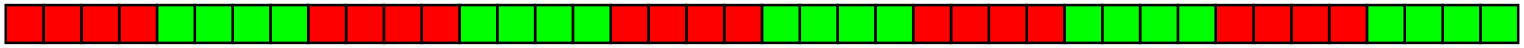
DAGS

DAGS scheme's public keys are Quasi–dyadic alternant codes. i.e. $\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n$ with an automorphism group acting as:



DAGS

DAGS scheme's public keys are Quasi–dyadic alternant codes. i.e. $\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n$ with an automorphism group acting as:



DAGS

In short :automorphism group $\mathcal{G}$ is $\cong (\mathbb{Z}/2\mathbb{Z})^\gamma$ for some $\gamma > 0$.

- **Public key.** An $\mathbb{F}_q[\mathcal{G}]$–basis of $\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n$;
- **Secret Key.** The pair $(\boldsymbol{x}, \boldsymbol{y})$.

Important. The extension degree m is 2.

$$
\begin{tikzcd}
\mathbb{F}_{q^2} \arrow[d, dash, "m=2"] & & \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \subseteq \mathbb{F}_{q^2}^n \arrow[d, dash] \\
\mathbb{F}_q & & \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n
\end{tikzcd}
$$

Section 2

Description of the attack

Tool 1 : the conductor

In $\mathbb{F}_q^n$ we denote by $\star$ the component wise product:

$$\boldsymbol{u} \star \boldsymbol{v} \stackrel{\text{def}}{=} (u_1 v_1, \dots, u_n v_n).$$

Then, the star product of two codes $\mathscr{A}, \mathscr{B} \subseteq \mathbb{F}_q^n$:

$$\mathscr{A} \star \mathscr{B} \stackrel{\text{def}}{=} \mathbf{Span}\{\boldsymbol{a} \star \boldsymbol{b} \mid \boldsymbol{a} \in \mathscr{A},\ \boldsymbol{b} \in \mathscr{B}\}$$

Definition 4

Let $\mathscr{U}, \mathscr{V} \subseteq \mathbb{F}_q^n$ be two codes:

$$\mathbf{Cond}(\mathscr{U}, \mathscr{V}) = \{\boldsymbol{x} \in \mathbb{F}_q^n \mid \boldsymbol{x} \star \mathscr{U} \subseteq \mathscr{V}\}$$

Remark

Equivalently, the conductor is the largest code $\mathscr{X}$ satisfying $\mathscr{X} \star \mathscr{U} \subseteq \mathscr{V}$.

Why are conductors good for?

Illustrative example.

- Suppose the public key is $\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y})$
- Suppose we obtained $\mathbf{GRS}_{k-1}(\boldsymbol{x}, \boldsymbol{y})$ (for instance by brute force search)

Lemma 5

$$\mathbf{Cond}(\mathbf{GRS}_{k-1}(\boldsymbol{x}, \boldsymbol{y}), \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y})) = \mathbf{RS}_2(\boldsymbol{x}) = \mathbf{Span}\{\mathbf{1}, \boldsymbol{x}\}.$$

Idea of the proof.

The largest space of polynomials S such that

$$S \cdot \mathbb{F}_q[X]_{<k-1} \subseteq \mathbb{F}_q[X]_{<k}$$

is $\mathbb{F}_q[X]_{<2} = \mathrm{Span}\{1, X\}$. □

Why are conductors good for?

Illustrative example.

- Suppose the public key is $\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y})$
- Suppose we obtained $\mathbf{GRS}_{k-1}(\boldsymbol{x}, \boldsymbol{y})$ (for instance by brute force search)

Lemma 6

$$\mathbf{Cond}(\mathbf{GRS}_{k-1}(\boldsymbol{x}, \boldsymbol{y}), \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y})) = \mathbf{RS}_2(\boldsymbol{x}) = \mathbf{Span}\{\mathbf{1}, \boldsymbol{x}\}.$$

Fundamental fact : the result does not depend on $\boldsymbol{y}$!

With alternant codes, things become harder...

Lemma 7

$$\mathbf{Cond}(\mathbf{GRS}_{k-1}(\boldsymbol{x},\boldsymbol{y}) \cap \mathbb{F}_q^n, \mathbf{GRS}_k(\boldsymbol{x},\boldsymbol{y}) \cap \mathbb{F}_q^n) \supseteq \mathbf{RS}_2(\boldsymbol{x}) \cap \mathbb{F}_q^n.$$

With alternant codes, things become harder...

Lemma 7

$$\mathbf{Cond}(\mathbf{GRS}_{k-1}(\boldsymbol{x},\boldsymbol{y}) \cap \mathbb{F}_q^n, \mathbf{GRS}_k(\boldsymbol{x},\boldsymbol{y}) \cap \mathbb{F}_q^n) \supseteq \mathbf{RS}_2(\boldsymbol{x}) \cap \mathbb{F}_q^n.$$

- Good news : typically equality holds.

With alternant codes, things become harder...

Lemma 7

$$\mathbf{Cond}(\mathbf{GRS}_{k-1}(\boldsymbol{x},\boldsymbol{y}) \cap \mathbb{F}_q^n, \mathbf{GRS}_k(\boldsymbol{x},\boldsymbol{y}) \cap \mathbb{F}_q^n) \supseteq \mathbf{RS}_2(\boldsymbol{x}) \cap \mathbb{F}_q^n.$$

- Good news : typically equality holds.
- Bad news : typically $\mathbf{RS}_2(\boldsymbol{x}) \cap \mathbb{F}_q^n = \mathbf{Span}\{(1,\ldots,1)\}$.

With alternant codes, things become harder...

One has to increase the gap between the degrees.

Lemma 8

For any $0 \leqslant a < k$*,*

$$\mathbf{Cond}(\mathbf{GRS}_{k-a}(\boldsymbol{x}, \boldsymbol{y}) \qquad , \mathbf{GRS}_{k}(\boldsymbol{x}, \boldsymbol{y}) \qquad) = \mathbf{RS}_{a+1}(\boldsymbol{x}) \qquad .$$

With alternant codes, things become harder...

One has to increase the gap between the degrees.

Lemma 8

For any $0 \leqslant a < k$,

$$\mathbf{Cond}(\mathbf{GRS}_{k-a}(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n, \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n) \supseteq \mathbf{RS}_{a+1}(\boldsymbol{x}) \cap \mathbb{F}_q^n.$$

With alternant codes, things become harder...

One has to increase the gap between the degrees.

Lemma 8

For any $0 \leqslant a < k$,

$$\mathbf{Cond}(\mathbf{GRS}_{k-a}(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n, \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n) \supseteq \mathbf{RS}_{a+1}(\boldsymbol{x}) \cap \mathbb{F}_q^n.$$

Idea. Choose a so that $\mathbf{RS}_{a+1}(\boldsymbol{x}) \cap \mathbb{F}_q^n \neq \mathbf{Span}\{(1, \ldots, 1)\}$.

With alternant codes, things become harder...

One has to increase the gap between the degrees.

Lemma 8

For any $0 \leqslant a < k$,

$$\mathbf{Cond}(\mathbf{GRS}_{k-a}(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n, \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n) \supseteq \mathbf{RS}_{a+1}(\boldsymbol{x}) \cap \mathbb{F}_q^n.$$

Idea. Choose a so that $\mathbf{RS}_{a+1}(\boldsymbol{x}) \cap \mathbb{F}_q^n \neq \mathbf{Span}\{(1, \ldots, 1)\}$.

For instance $\mathbf{RS}_{q+1}(\boldsymbol{x}) \cap \mathbb{F}_q^n$ contains $\boldsymbol{x}^q + \boldsymbol{x}$ (image of $\boldsymbol{x}$ by $\mathrm{Tr}_{\mathbb{F}_{q^2}/\mathbb{F}_q}$).

(Very Naive attack)

Recall that $\mathscr{C}_{\text{pub}} = \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n$ and $m = 2$.

We look for $\mathbf{GRS}_{k-q}(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n$

- For any $\mathscr{D} \subseteq \mathscr{C}_{\text{pub}} \cap \mathbb{F}_q^n$ of codimension $2q$, compute $\mathbf{Cond}(\mathscr{D}, \mathscr{C}_{\text{pub}})$.
- If the conductor $\neq \mathbf{Span}\{(1, \ldots, 1)\}$, you probably found $\mathbf{RS}_{q+1}(\boldsymbol{x}) \cap \mathbb{F}_q^n$. Deducing $\boldsymbol{x}$ from this code is rather easy.

(Very Naive attack)

Recall that $\mathscr{C}_{\text{pub}} = \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n$ and $m = 2$.

We look for $\mathbf{GRS}_{k-q}(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n$

- For any $\mathscr{D} \subseteq \mathscr{C}_{\text{pub}} \cap \mathbb{F}_q^n$ of codimension $2q$, compute $\mathbf{Cond}(\mathscr{D}, \mathscr{C}_{\text{pub}})$.
- If the conductor $\neq \mathbf{Span}\{(1, \dots, 1)\}$, you probably found $\mathbf{RS}_{q+1}(\boldsymbol{x}) \cap \mathbb{F}_q^n$. Deducing $\boldsymbol{x}$ from this code is rather easy.

$\rightarrow$ Cost $\widetilde{O}(q^{2q\cdot(\dim \mathscr{C}_{\text{pub}} - 2q)})$. e.g. For `DAGS_1` : $> 2^{112640}$ operations.

(Very Naive attack)

Recall that $\mathscr{C}_{\text{pub}} = \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n$ and $m = 2$.

We look for $\mathbf{GRS}_{k-q}(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n$

- For any $\mathscr{D} \subseteq \mathscr{C}_{\text{pub}} \cap \mathbb{F}_q^n$ of codimension $2q$, compute $\mathbf{Cond}(\mathscr{D}, \mathscr{C}_{\text{pub}})$.
- If the conductor $\neq \mathbf{Span}\{(1, \ldots, 1)\}$, you probably found $\mathbf{RS}_{q+1}(\boldsymbol{x}) \cap \mathbb{F}_q^n$. Deducing $\boldsymbol{x}$ from this code is rather easy.

$\rightarrow$ Cost $\widetilde{O}(q^{2q \cdot (\dim \mathscr{C}_{\text{pub}} - 2q)})$. e.g. For `DAGS_1` : $> 2^{112640}$ operations.

$\rightarrow$ Up to now we never used the automorphism group.

Tool 2 : The invariant code

Consider the code

$$\mathscr{C}_{\text{pub}}{}^{\mathcal{G}} \stackrel{\text{def}}{=} \{\boldsymbol{c} \in \mathscr{C}_{\text{pub}} \mid \forall \sigma \in \mathcal{G},\ \sigma(\boldsymbol{c}) = \boldsymbol{c}\}.$$

Theorem 9 (Proved under some heuristic)

$$\mathbf{Cond}\left(\left(\mathbf{GRS}_{k-q}(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n\right)^{\mathcal{G}}, \mathscr{C}_{pub}\right) = \mathbf{RS}_{q+2}(\boldsymbol{x}) \cap \mathbb{F}_q^n.$$

Tool 2 : The invariant code

Consider the code

$$\mathscr{C}_{\mathrm{pub}}{}^{\mathcal{G}} \stackrel{\mathrm{def}}{=} \{\boldsymbol{c} \in \mathscr{C}_{\mathrm{pub}} \mid \forall \sigma \in \mathcal{G},\ \sigma(\boldsymbol{c}) = \boldsymbol{c}\}.$$

Theorem 9 (Proved under some heuristic)

$$\mathbf{Cond}\left(\left(\mathbf{GRS}_{k-q}(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n\right)^{\mathcal{G}}, \mathscr{C}_{pub}\right) = \mathbf{RS}_{q+2}(\boldsymbol{x}) \cap \mathbb{F}_q^n.$$

Tool 2 : The invariant code

Consider the code

$$\mathscr{C}_{\mathrm{pub}}{}^{\mathcal{G}} \stackrel{\text{def}}{=} \{\boldsymbol{c} \in \mathscr{C}_{\mathrm{pub}} \mid \forall \sigma \in \mathcal{G},\ \sigma(\boldsymbol{c}) = \boldsymbol{c}\}.$$

Theorem 9 (Proved under some heuristic)

$$\mathbf{Cond}\big(\big(\mathbf{GRS}_{k-q}(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n\big)^{\mathcal{G}}, \mathscr{C}_{pub}\big) = \mathbf{RS}_{q+2}(\boldsymbol{x}) \cap \mathbb{F}_q^n.$$

→ Enumerate $\mathscr{D} \subseteq \mathscr{C}_{\mathrm{pub}}{}^{\mathcal{G}}$ of codimension $\frac{2q}{|\mathcal{G}|}$.

→ Cost $\widetilde{O}\left(q^{\frac{2q}{|\mathcal{G}|} \cdot \frac{\dim \mathscr{C}_{\mathrm{pub}} - 2q}{|\mathcal{G}|}}\right)$.

Tool 2 : The invariant code

Consider the code

$$\mathscr{C}_{\mathrm{pub}}{}^{\mathcal{G}} \stackrel{\mathrm{def}}{=} \{\boldsymbol{c} \in \mathscr{C}_{\mathrm{pub}} \mid \forall \sigma \in \mathcal{G},\ \sigma(\boldsymbol{c}) = \boldsymbol{c}\}.$$

Theorem 9 (Proved under some heuristic)

$$\mathbf{Cond}\left(\left(\mathbf{GRS}_{k-q}(\boldsymbol{x}, \boldsymbol{y}) \cap \mathbb{F}_q^n\right)^{\mathcal{G}}, \mathscr{C}_{pub}\right) = \mathbf{RS}_{q+2}(\boldsymbol{x}) \cap \mathbb{F}_q^n.$$

- $\rightarrow$ Enumerate $\mathscr{D} \subseteq \mathscr{C}_{\mathrm{pub}}{}^{\mathcal{G}}$ of codimension $\frac{2q}{|\mathcal{G}|}$.
- $\rightarrow$ Cost $\widetilde{O}(q^{\frac{2q}{|\mathcal{G}|} \cdot \frac{\dim \mathscr{C}_{\mathrm{pub}} - 2q}{|\mathcal{G}|}})$.
- $\rightarrow$ Next, using some classical coding theoretic operations (shortening) we can reduce the cost to $\widetilde{O}(q^{\frac{4q}{|\mathcal{G}|}})$.

Section 3

Complexity and implementation

In practice

The average work factor will be:

	Claimed security	q	$\|\mathcal{G}\|$	Work factor
`DAGS_1`	128 bits	2^5	2^4	$\approx 2^{70}$
`DAGS_3`	192 bits	2^6	2^5	$\approx 2^{80}$
`DAGS_5`	256 bits	2^6	2^6	$\approx 2^{58}$

Second approach using polynomial system solving

Brute force search can be replaced by the resolution of a **system of polynomial equations** of degree 2.

Note. Magma implementation on personal computer.

Second approach using polynomial system solving

Brute force search can be replaced by the resolution of a **system of polynomial equations** of degree 2.

	Claimed security	q	$\lvert\mathcal{G}\rvert$	1st approach Work factor	2nd approach Running times
DAGS_1	128 bits	2^5	2^4	2^{70}	$\approx$ 20mn
DAGS_3	192 bits	2^6	2^5	2^{80}	-
DAGS_5	256 bits	2^6	2^6	2^{58}	$<$ 1mn

Note. Magma implementation on personal computer.

Second approach using polynomial system solving

Brute force search can be replaced by the resolution of a **system of polynomial equations** of degree 2.

	Claimed security	q	$\lvert\mathcal{G}\rvert$	1st approach Work factor	2nd approach Running times
DAGS_1	128 bits	2^5	2^4	2^{70}	$\approx$ 20mn
DAGS_3	192 bits	2^6	2^5	2^{80}	-
DAGS_5	256 bits	2^6	2^6	2^{58}	$<$ 1mn

Note. Magma implementation on personal computer.

Note 1. **DAGS** authors changed their proposal to be out of reach of the first version of the attack (see **DAGS**' website).

Second approach using polynomial system solving

Brute force search can be replaced by the resolution of a **system of polynomial equations** of degree 2.

	Claimed security	q	$\|\mathcal{G}\|$	1st approach Work factor	2nd approach Running times
`DAGS_1`	128 bits	2^5	2^4	2^{70}	$\approx$ 20mn
`DAGS_3`	192 bits	2^6	2^5	2^{80}	-
`DAGS_5`	256 bits	2^6	2^6	2^{58}	$<$ 1mn

Note. Magma implementation on personal computer.

Note 1. DAGS authors changed their proposal to be out of reach of the first version of the attack (see **DAGS**' website).

Note 2. Bardet, Bertin and Otmani, are currently working on improving the 2nd version. They are able to break original `DAGS_3` in $<$ 20mn.

Questions?