Generalized Reed-Solomon codes in the McEliece cryptosystem

- Generalized Reed-Solomon (GRS) codes would allow reducing the public key size (since they are MDS codes)
- They are widespread and already widely implemented in software and hardware
- But they have more structure than Goppa codes, and this facilitates attacks

We have recently proposed a new GRS code-based variant of the McEliece cryptosystem which provides a higher protection to the structure of the secret GRS code [1].

An attack procedure already devised in [2] has been improved in [3], resulting in a polynomial-time attack against the system parameters proposed in [1].

The attack can be avoided by changing some parameters, without modifying the system

But this way the advantage of using GRS codes is lost

We propose a modification of the system which maintains security with practical parameters at a small cost in terms of public key size.

Original System

- Niederreiter version of the system
- Bob's secret key is a GRS code:
  - with length $n$ and dimension $k$
  - defined over $\mathbb{F}_q$
  - able to correct $t$ errors
  - described through its $r \times n$ parity-check matrix $H$
- Bob's public key:

$$H = S^{-1} \cdot H \cdot Q^T$$

where:
- $S$ is a non-singular $r \times r$ scrambling matrix
- $Q$ is a transformation matrix replacing the classical permutation matrix
- $Q = R + T$
- $R$ is a dense $n \times n$ matrix with rank $z < n$
- $T$ is a sparse $n \times n$ matrix with average row and column weight $m < n$
- $R = a^T \cdot b$
- $a$ and $b$ are two $z \times n$ matrices with rank $z$

Encryption:

$$x = H \cdot e^T$$

where:
- $e$ is the error vector corresponding to the message
- $e$ has weight $t_{pub} = \lfloor \frac{1}{2} t \rfloor$

Decryption:

$$x' = S \cdot x = H \cdot Q^T \cdot e^T = H \cdot (e \cdot Q)^T$$

then:

- Bob guesses the value of $\gamma$
- Bob computes $x' = x - H \cdot b^T \cdot \gamma = H \cdot T^T \cdot e^T$
- $x'$ is a correctable syndrome since $T^T \cdot e^T$ has weight $\leq m \cdot t_{pub} \leq t$
- Bob recovers $e_T = T^T \cdot e^T$, with weight $\leq t$, through syndrome decoding
- Bob multiplies the result by $(T^T)^{-1}$ to recover $e$

Main issue

In the original system, both $m$ and $z$ must be small, since:
- for a given $t_{pub}$, increasing $m$ requires to increase $t$ and, hence, the code size and the public key size
- Bob needs $q^2/2$ attempts on average to guess the value of $\gamma$, hence increasing $z$ increases the decryption complexity

Keeping both $z$ and $m$ small exposes the system to polynomial-time attacks [3].

New system

Modifications:
- Make a public
- Choose $b$ such that it has rank $z' < z$
- Make a basis of the kernel of $b^T$ public
- This basis is represented through a $z' \times z$ matrix $B$, having rank $z'$
- Alice computes $\gamma' = \gamma + v$, where $v$ is a $z \times 1$ vector in the kernel of $b^T$ (i.e., $b^T \cdot v = 0$)
- Alice sends $\gamma'$ along with the ciphertext
- Bob computes $b^T \cdot \gamma' = b^T \cdot \gamma$, thus he no longer needs to guess $\gamma$
- This allows to choose high values of $z'$

Using high values of $z$ prevents attacks exploiting the subcode defined by the parity-check matrix $H_S = H - [1]$

$m = 1$ can be used, with code rate $> 2/3$ to avoid the attack in [4] against a shortened version of the public code.

Assessment and comparisons

- Classical binary Goppa code-based Niederreiter cryptosystem
- New system with:
  - $m = 1$
  - $z = k$
  - $z' = \lfloor k/2 \rfloor$
  - codes with rate $> 2/3$

Work factor (WF) of the most dangerous attacks (Information Set Decoding) estimated according to [5] and public key size (KS) in Kib:

<table>
<thead>
<tr>
<th>$m$</th>
<th>$z$</th>
<th>$z'$</th>
<th>KS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>304</td>
<td>97</td>
<td>468.6</td>
</tr>
<tr>
<td>2</td>
<td>98</td>
<td>94</td>
<td>434.3</td>
</tr>
<tr>
<td>3</td>
<td>116</td>
<td>90</td>
<td>342</td>
</tr>
</tbody>
</table>

We propose a modification of the system which maintains security with practical parameters at a small cost in terms of public key size.

References