Information-set decoding for linear codes over $\mathbf{F}_q$

Christiane Peters

Technische Universiteit Eindhoven

PQCrypto 2010

May 26, 2010

1. Introduction

2. Information-set decoding over $\mathbf{F}_q$

3. Improvements and cost analysis

4. Applications

1. Introduction

2. Information-set decoding over $\mathbf{F}_q$

3. Improvements and cost analysis

4. Applications

Motivation

Bernstein-Lange-P., PQCrypto 2008:

- For 128-bit security of the McEliece cryptosystem take a length-2960, dimension-2288 classical binary Goppa code ($t = 56$), with 57 errors added by the sender.
- The public-key size here is 1537536 bits.

Goal: reduce the key size!

Go $\mathbf{F}_q$

(1) Janwa-Moreno 1996: McEliece Public Key Cryptosystems Using Algebraic-Geometric Codes

(2) Berger-Loidreau 2005: How to Mask the Structure of Codes for a Cryptographic Use

(3) Berger-Cayrel-Gaborit-Otmani 2009: Reducing Key Length of the McEliece Cryptosystem

(4) Misoczki-Barreto 2009: Compact McEliece Keys from Goppa Codes

Note (2), (3), (4) have fallen prey to structural attacks.

This talk

- Provides a cost analysis of a state-of-the-art information-set-decoding algorithm to decode a linear code over $\mathbf{F}_q$ without any known structure.

- Proposes alternative setups of the McEliece PKC based on classical Goppa codes over $\mathbf{F}_q$ halving the key size.

- Sneak Preview: Wild McEliece (joint work with Daniel J. Bernstein and Tanja Lange)

1. Introduction

2. Information-set decoding over $\mathbf{F}_q$

3. Improvements and cost analysis

4. Applications

Information-set-decoding algorithms

Over $\mathbf{F}_2$ we have information-set-decoding algorithms due to

Prange (1962), Leon (1988), Lee and Brickell (1988), Krouk (1989), Stern (1989), Dumer (1989), van Tilburg (1990) (independently from Omura in the 1960's), Dumer (1991), Canteaut and Chabanne (1994), Canteaut and Chabaud (1998), and Canteaut and Sendrier (1998), Bernstein-Lange-P. (2008), Finiasz-Sendrier (2009), Bernstein-Lange-P. (2010; see recent results session coming Friday).

ISD is the best non-structural attack to decrypt a message without knowing the McEliece secret key.

Information-set decoding

Given a $k \times n$ generator matrix G of a linear code over $\mathbf{F}_q$. An information set $I \subset \{1, \ldots, n\}$ is a set of size k which indexes k linearly independent columns of G.

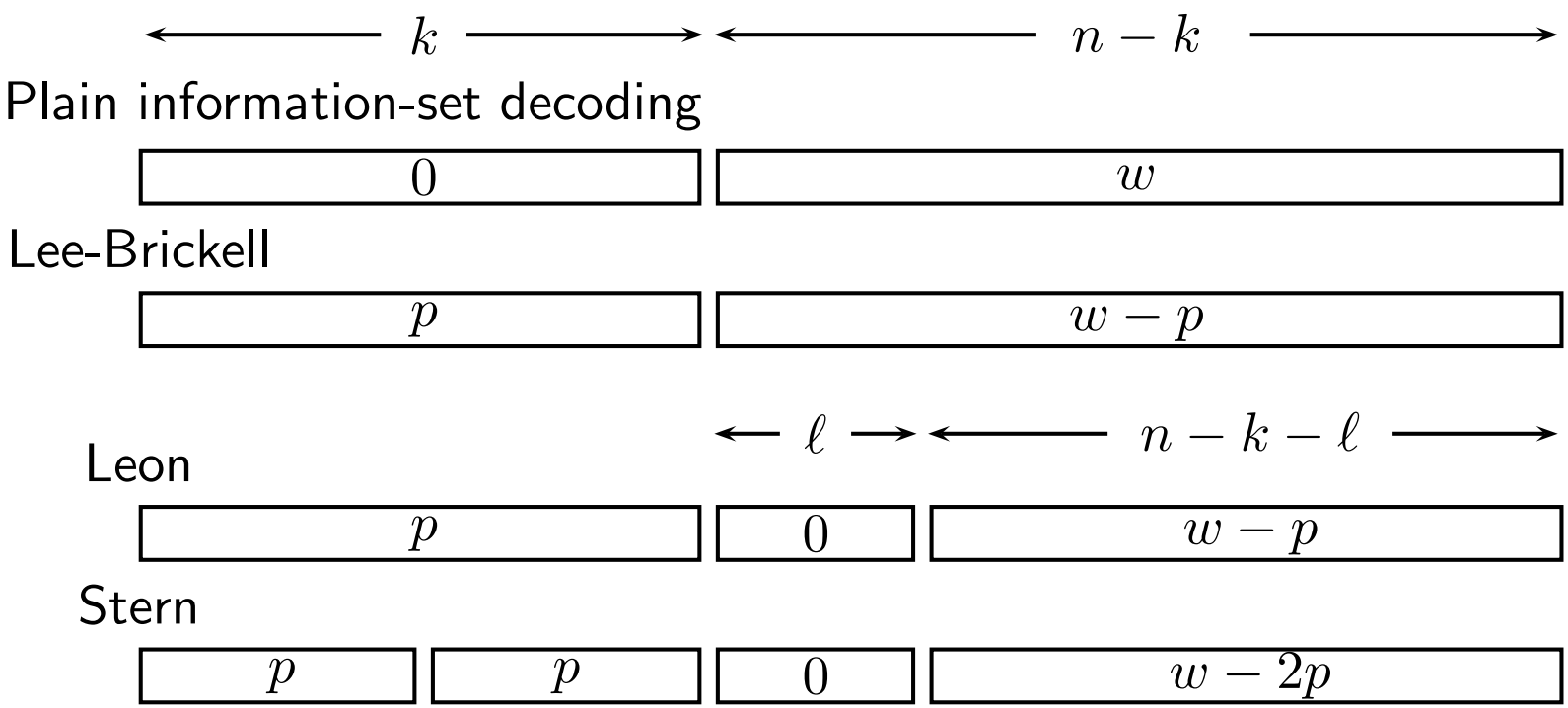
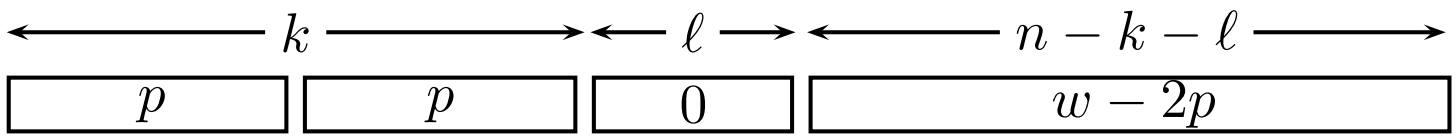


Figure from Overbeck and Sendrier: *Code-based Cryptography*, in *Post-Quantum Cryptography* (eds.: Bernstein, Buchmann, and Dahmen)

ISD$\mathbf{F}_q$: Stern's algorithm for $\mathbf{F}_q$



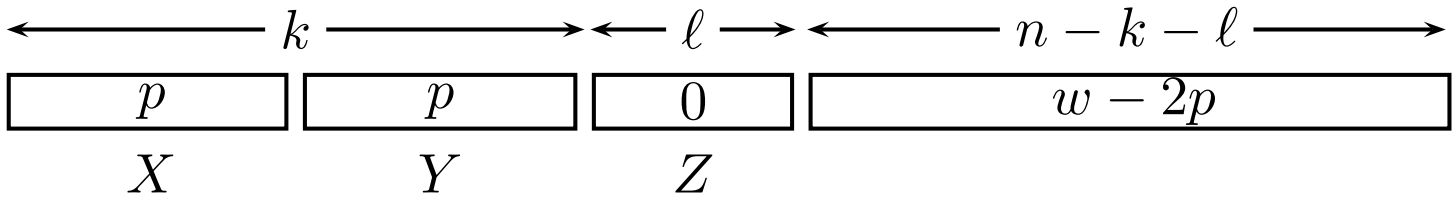
Given a generator matrix G and a word $\mathbf{y}$ in $\mathbf{F}_q^n$ which is at distance w from the code $\mathbf{F}_q^k G$ (for simplicity assume $w < \frac{1}{2}$min distance of $\mathbf{F}_q^k G$).

Step 1 Choose an information set I.

Step 2 Bring G in systematic form,
i.e., replace G by $G_I^{-1}G = (I_k|Q)$.

Step 3 Use G to eliminate the I-indexed columns of $\mathbf{y}$,
i.e., replace $\mathbf{y}$ by $\mathbf{y} - \mathbf{y}_I G_I^{-1} G$.

ISD$\mathbf{F}_q$: Stern's algorithm for $\mathbf{F}_q$

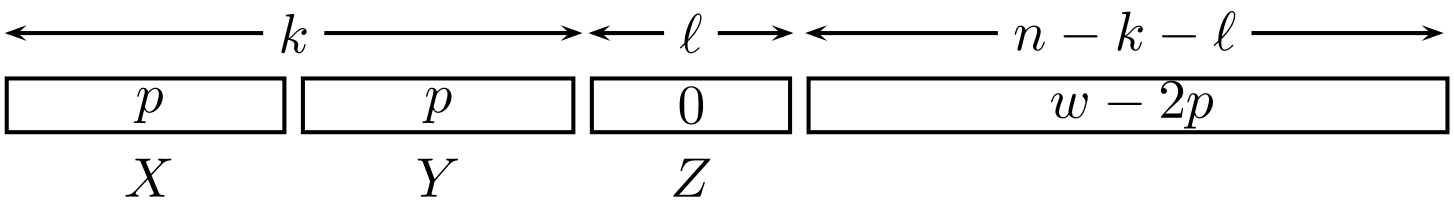


Step 4 Consider subsets X and Y of I.

Step 5 Compute all possible combinations of p weighted rows $\mathbf{g}_{a_i}$ with indices in X on ℓ positions outside I: $(\mathbf{y} - \sum_{i=1}^{p} m_i \mathbf{g}_{a_i})_\ell$.

Step 6 Compute all possible combinations of p weighted rows $\mathbf{g}_{b_j}$ with indices in Y on ℓ positions outside I: $(\sum_{j=1}^{p} m'_j \mathbf{g}_{b_j})_\ell$.

ISD$\mathbf{F}_q$: Stern's algorithm for $\mathbf{F}_q$



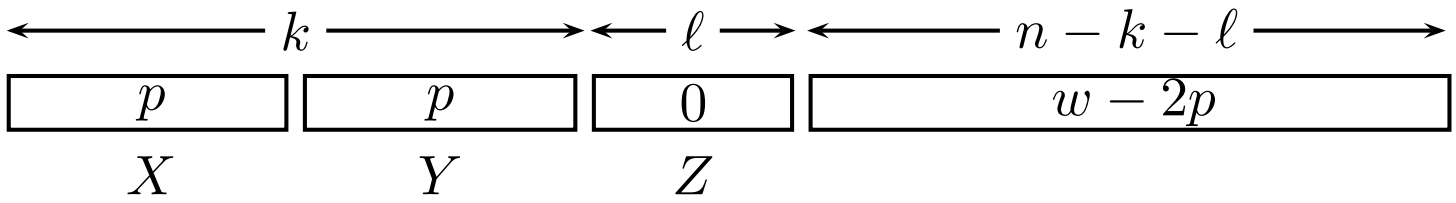
Step 7 Look for collisions on those ℓ positions.

Step 8 For each collision compute the whole vector coming from $2p$ weighted rows of G

$$\mathbf{y} - \sum_{i=1}^{p} m_i \mathbf{g}_{a_i} - \sum_{j=1}^{p} m'_j \mathbf{g}_{b_j}$$

and check if its weight equals w.

ISD$\mathbf{F}_q$: Stern's algorithm for $\mathbf{F}_q$



- If the algorithm finds an information set I together with sets X, Y, and an size-ℓ subset Z in $\{1,\ldots,n\}\setminus I$ such that $\mathbf{e}$ has weights p, p, 0 on the positions indexed by X, Y, and Z, respectively, then it finds the error vector.
- Stern: $I = X \cup Y$. Finiasz-Sendrier: $X, Y \subset I$.
- Steps 1–8 form one iteration of the generalized Stern algorithm.
- If the set I chosen in Step 1 does not lead to a weight-w word in Step 8 another iteration has to be performed.

1. Introduction

2. Information-set decoding over $\mathbf{F}_q$

3. Improvements and cost analysis

4. Applications

Success probability

- Assume that $I = X \cup Y$ (even split).
- The chance that Stern's algorithm finds $\mathbf{e}$ after the first round is

$$P_{\text{success}} = \frac{\binom{k/2}{p}^2 \binom{n-k-\ell}{w-2p}}{\binom{n}{w}}.$$

- If one chooses in each iteration an information set uniformly at random the expected number of iterations is $1/P_{\text{success}}$.
- Ignore extremely unusual codes for which the average number of iterations is significantly different from the reciprocal of the success probability of a single iteration.
- If parts of the information set of the previous round are reused the iterations are not longer independent. Compute the number of iterations with a Markov chain.

Cost: Updating the matrix G

Bring G in systematic form, i.e., replace G by $G_I^{-1}G = (I_k|Q)$.

- Assume that iterations are independent and in each round $G_I^{-1}G$ has to be computed for a new choice of I.
- Reasonable estimate for the cost for Gaussian elimination over $\mathbf{F}_q$ $(q > 2)$: $(n-k)^2(n+k)$ field operations.

Disclaimer for the whole cost analysis:
Assume that multiplications in $\mathbf{F}_q$ are implemented as table-lookups. Cost about the same as additions in $\mathbf{F}_q$.

Cost: Hashing step

Compute all possible combinations of p weighted rows $\mathbf{g}_{a_i}$ with indices in X of G on ℓ positions outside I: $(\mathbf{y} - \sum_{i=1}^{p} m_i \mathbf{g}_{a_i})_\ell$.

- Naively computing $(\mathbf{y} - \sum_{i=1}^{p} m_i \mathbf{g}_{a_i})_\ell$ takes $p\ell$ multiplications (m_i's) and $\ell + (p-1)\ell$ additions ($-$, $\sum$).
- However, all elements can be reached by repeated addition (be careful if q is not a prime!). Cost for $(\sum_{i=1}^{p} m_i \mathbf{g}_{a_i})_\ell$ boils down to ℓ additions $=$ 1 vector addition in $\mathbf{F}_q^\ell$.
- Adding $\mathbf{y}$ can also done using precomputations: $(k/2 - p + 1)\ell$ additions in $\mathbf{F}_q$.

Total for Steps 5 and 6:

$$\left(\left(\frac{k}{2} - p + 1 \right) + 2 \binom{k/2}{p} (q-1)^p \right) \ell$$

Cost: Collision handling

For each collision compute the whole vector coming from $2p$ weighted rows: $\mathbf{y} - \sum_{i=1}^{p} m_i \mathbf{g}_{a_i} - \sum_{j=1}^{p} m'_j \mathbf{g}_{b_j}$ and check if its weight equals w.

- Expected number of collisions on ℓ positions

$$\frac{\binom{k/2}{p}^2 (q-1)^{2p}}{q^\ell}.$$

- Naive: compute the sum on $n-k-\ell$ positions.
- Use early abort: compute the vector in a column-by-column fashion. If the weight exceeds $w-2p$ abandon vector. On average can abort after $\sim (q/(q-1))(w-2p+1)$ columns are handled.

Total:

$$\frac{q}{q-1}(w-2p+1)2p\left(1+\frac{q-2}{q-1}\right)\frac{\binom{k/2}{p}^2 (q-1)^{2p}}{q^\ell}.$$

Notes

- Choose ℓ to balance the hashing step with the collision-handling step. A reasonable choice is

$$\ell \approx \log_q \binom{k/2}{p} + p \log_q(q-1).$$

- For small q the parameter p is at least 2 for codes achieving at least 80-bit security.
- For larger fields, e.g., $q = 256$, one can choose $p = 1$ since the sets handled in Steps 5 and 6 are quite large.
- Note that the analysis also holds for $\mathbf{F}_q$ where q is a prime power.

Analysis for depending iterations

Depending iterations (reuse parts of the information set of the previous iteration):

- Saves time on row reduction at the cost of more iterations (for details have a look at the paper).
- On `www.win.tue.nl/~cpeters/isdfq.html` a C-implementation of the Markov chain process to estimate the number of iterations for an algorithm with depending iterations can be found.
  (uses GMP, MPFR, MPFI libraries for multiprecision)
- Investigate in particular the parameters proposed by Berger et al at (AFRICACRYPT 2009) and parameters proposed by Misoczki and Barreto (SAC 2009).

Crude analysis

Fast simple program (here with PARI) yields the following security exponents for ISD$\mathbf{F}_q$ with independent iterations:

```
q=4, n=2560, k=1536, w=128
without Gauss    181.545876020555
with Gauss       182.30540037296

q=16, n=1408, k=896, w=128
without Gauss    209.409946211931
with Gauss       211.292457298855

q=256, n=640, k=512, w=64
without Gauss    180.520078719529
with Gauss       184.202777659868

q=256, n=768, k=512, w=128
without Gauss    251.833665994276
with Gauss       255.432627408043
```

1. Introduction

2. Information-set decoding over $\mathbf{F}_q$

3. Improvements and cost analysis

4. Applications

Recommendation #1

- Note that a Goppa code $\Gamma_q(a_1, \ldots, a_n, g)$ for a degree-t polynomial $g(x)$ can only correct up to $t/2$ errors if $q > 2$.
- For moderate-size fields $\mathbf{F}_q$ such as $\mathbf{F}_{31}$ one can halve the key size in comparison to using a binary Goppa code:
- $\Gamma_{31}(a_1, \ldots, a_n, g)$ with length $n = 961$, dimension $k = 771$, and a degree-95 polynomial achieves 128-bit security against the generalized ISD attack.
- The public key needs ≈ 725741 bits to store.
- Recall the **1537536** bits for McEliece with a binary Goppa code for 128-bit security.
- However, note that Goppa codes over $\mathbf{F}_3$ and $\mathbf{F}_4$ look worse than binary Goppa codes since only $t/2$ errors can be efficiently corrected.

Recommendation #2 (Sneak Preview)

Bernstein-Lange-P. (preprint): Wild McEliece

Based on:

Theorem (Sugiyama-Kasahara-Hirasawa-Namekawa, 1976)

$$\Gamma_q(a_1, \ldots, a_n, g^{q-1}) = \Gamma_q(a_1, \ldots, a_n, g^q)$$

for a monic squarefree polynomial $g(x)$ *in* $\mathbf{F}_{q^m}[x]$ *of degree* t*.*

- Well known for $q = 2$.
- Consequence: minimum distance of $\Gamma_q(a_1, \ldots, a_n, g^{q-1})$ is at least qt (and not just $(q-1)t$).

More advertising:

Bernstein-Lange-P. (preprint): Wild McEliece

- Propose to use the wild Goppa code $\Gamma_q(a_1, \ldots, a_n, g^{q-1})$ over small finite fields $\mathbf{F}_q$.
- Give efficient decoding algorithm for $\Gamma_q(a_1, \ldots, a_n, g^{q-1})$ which can correct $\lfloor qt/2 \rfloor$ errors.
- Give efficient list decoding algorithm for $\Gamma_q(a_1, \ldots, a_n, g^{q-1})$ which can correct $n - \sqrt{n(n-qt)}$ errors.
- Review structural attacks and their applicability to the Wild McEliece cryptosystem.

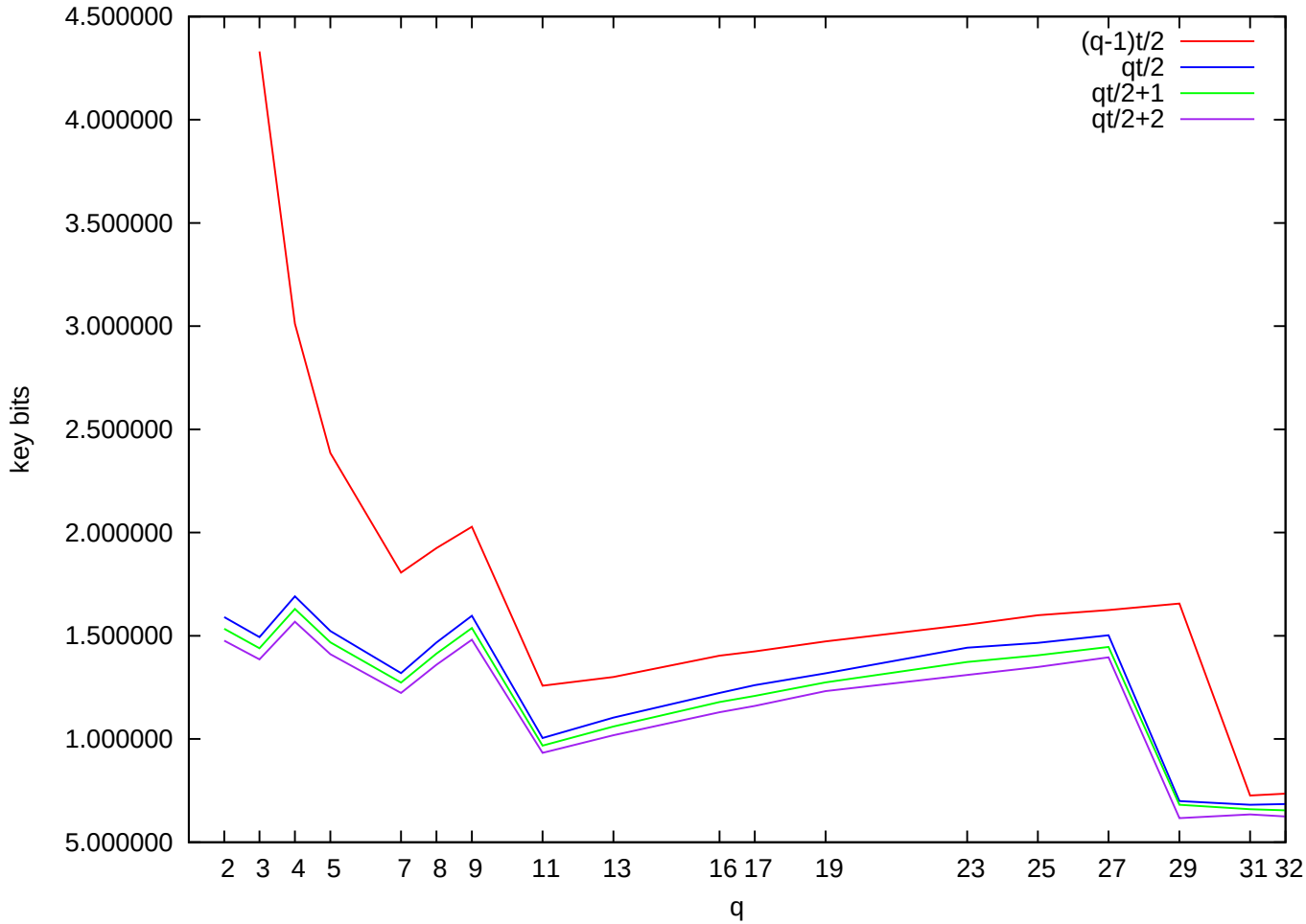
Even more advertising:

Bernstein-Lange-P. (preprint): Wild McEliece

- Present parameters for different base fields achieving 128-bit security against the attack in today's talk.
- Show that base fields $\mathbf{F}_q$ with $q \leq 32$ are interesting alternatives to $\mathbf{F}_2$.
- For $\mathbf{F}_{32}$ the increase factor $q/(q-1)$ is close to 1 (results are close to the $\mathbf{F}_{31}$-example presented before).
- For $q = 3, 4$, or 5 the change of the error-correction capacity from $\lfloor (q-1)t/2 \rfloor$ to $\lfloor qt/2 \rfloor$ significantly influences the key sizes.

Key sizes for various q

McEliece with $\Gamma_q(a_1, \ldots, a_n, g^{q-1})$ and $\lfloor (q-1)t/2 \rfloor$, $\lfloor qt/2 \rfloor$, $\lfloor qt/2 \rfloor + 1$, or $\lfloor qt/2 \rfloor + 2$ added errors.



Thank you for your attention!