Code–Based Public–Key Cryptosystems: Constructions and Attacks

Nicholas E. Kolokotronis

Dept. of Informatics and Telecommunications
University of Peloponnese
End of Karaiskaki Str., 22100 Tripolis, Greece

E–mail: nkolok@uop.gr
Web: http://www.uop.gr/~nkolok/

AtheCrypt 2014 — Athens Cryptography Day

January 7, 2014 • National Technical University of Athens, Athens, Greece

Talk outline

Background on codes

Definition (code)

A binary (n, k) linear code $\mathscr{C}$ is a k-th dimensional subspace of $\mathbb{F}_2^n$ and $R = \frac{k}{n}$ is called the *information rate* of $\mathscr{C}$.

The code $\mathscr{C}$ can defined by means of (Lin & Costello '04)

- The row space of a *generator matrix* $\boldsymbol{G} \in \mathbb{F}_2^{k \times n}$

$$\mathscr{C} = \left\{ \boldsymbol{v} = \boldsymbol{u}\boldsymbol{G} : \boldsymbol{u} \in \mathbb{F}_2^k \right\}$$

- The null space of a *parity–check matrix* $\boldsymbol{H} \in \mathbb{F}_2^{n-k \times n}$

$$\mathscr{C} = \left\{ \boldsymbol{v} \in \mathbb{F}_2^n : \boldsymbol{v}\boldsymbol{H}^T = \boldsymbol{0} \right\}$$

satisfying $\boldsymbol{G}\boldsymbol{H}^T = \boldsymbol{0}$

Background on codes (cont.)

Systematic forms

$\boldsymbol{G}, \boldsymbol{H}$ have full row–rank and can be given in their *systematic form*

$$\boldsymbol{G} = \begin{pmatrix} \boldsymbol{A} & \boldsymbol{I}_k \end{pmatrix} \quad \text{and} \quad \boldsymbol{H} = \begin{pmatrix} \boldsymbol{I}_{n-k} & \boldsymbol{B} \end{pmatrix}$$

where $\boldsymbol{B} = \boldsymbol{A}^T$, that allows for efficient implementations

Definition (parameters)

The *minimum distance* d of the (n,k) code $\mathscr{C}$ is defined as

$$d = \min_{\substack{\boldsymbol{x},\boldsymbol{y} \in \mathscr{C} \\ \boldsymbol{x} \neq \boldsymbol{y}}} \mathrm{d}(\boldsymbol{x},\boldsymbol{y}) = \min_{\boldsymbol{x} \in \mathscr{C} \setminus \{\boldsymbol{0}\}} \mathrm{wt}(\boldsymbol{x})$$

and $\left\lfloor \frac{d-1}{2} \right\rfloor$ is its *error–correcting capability*

Background on codes (cont.)

Encoding and decoding

- The function $\mathsf{Encode} : \mathbb{F}_2^k \to \mathscr{C}$ is injective $(\boldsymbol{u} \mapsto \boldsymbol{v} = \boldsymbol{uG})$
- The function $\mathsf{Decode} : \mathbb{F}_2^n \to \mathscr{C}$ should be such that

$$\mathrm{d}\big(\boldsymbol{x}, \mathsf{Decode}(\boldsymbol{x})\big) = \mathrm{d}\big(\boldsymbol{x}, \mathscr{C}\big), \qquad \forall \boldsymbol{x} \in \mathbb{F}_2^n$$

i.e. compute the closest codeword to a given vector (MD decoding)

Decoding strategies

Given a noisy version $\boldsymbol{x} = \boldsymbol{v} + \boldsymbol{e}$ of the codeword $\boldsymbol{v}$ find a *minimal weight*

- representative of the coset $\boldsymbol{x} + \mathscr{C} = \boldsymbol{e} + \mathscr{C}$ called *coset leader*
- solution to the equation $\boldsymbol{s} = \boldsymbol{xH}^T = \boldsymbol{eH}^T$, where $\boldsymbol{s}$ is the *syndrome*

Background on codes (cont.)

Syndrome decoding

The standard array of a systematic $(7,4)$ Hamming code $\mathscr{C}$ with $d=3$

	$\boldsymbol{v}_0$	$\boldsymbol{v}_1$	$\cdots$	$\boldsymbol{v}_{14}$	$\boldsymbol{v}_{15}$	$\boldsymbol{s}$
$\boldsymbol{e}_0$	(0000000)	(1101000)	$\cdots$	(0010111)	(1111111)	(000)
$\boldsymbol{e}_1$	(1000000)	(0101000)	$\cdots$	(1010111)	(0111111)	(100)
$\boldsymbol{e}_2$	(0100000)	(1001000)	$\cdots$	(0110111)	(1011111)	(010)
$\boldsymbol{e}_3$	(0010000)	(1111000)	$\cdots$	(0000111)	(1101111)	(001)
$\boldsymbol{e}_4$	(0001000)	(1100000)	$\cdots$	(0011111)	(1110111)	(110)
$\boldsymbol{e}_5$	(0000100)	(1101100)	$\cdots$	(0010011)	(1111011)	(011)
$\boldsymbol{e}_6$	(0000010)	(1101010)	$\cdots$	(0010101)	(1111101)	(111)
$\boldsymbol{e}_7$	(0000001)	(1101001)	$\cdots$	(0010110)	(1111110)	(101)

- standard array has size $2^{n-k} \times (2^k + 1)$
- 1^{st} column indicates the coset leader

Hardness results

Definition (CSD_ω)

Given a parity–check matrix $\boldsymbol{H} \in \mathbb{F}_2^{n-k\times n}$, the syndrome $\boldsymbol{s} \in \mathbb{F}_2^{n-k}$, and a weight $\omega \in \mathbb{N}$, find a vector $\boldsymbol{e} \in \mathbb{F}_2^n$ with $\mathrm{wt}(\boldsymbol{e}) \leq \omega$ such that $\boldsymbol{s} = \boldsymbol{e}\boldsymbol{H}^T$

- Its decisional form is proved to be NP–complete for *random linear codes* (Berlekamp *et al.* '78)
  - reduced to the 3–DIMENSIONAL MATCHING problem in Karp's list
- The same was proved for the special case $\boldsymbol{s} = \boldsymbol{0}$ but with $\mathrm{wt}(\boldsymbol{e}) = \omega$
- CSD_ω is equivalent to its decisional variant DSD_ω
  - Given access to an oracle $\mathcal{O}_\omega(\boldsymbol{H}, \boldsymbol{s})$ for DSD_ω, recursively apply
    (1) Write $\boldsymbol{s} = \boldsymbol{e}'\boldsymbol{H}'^T + e_n\boldsymbol{h}_n$
    (2) Query $\mathcal{O}_\omega(\boldsymbol{H}', \boldsymbol{s})$ and if Y is received then $e_n = 0$, else $e_n = 1$
    (3) Set $\omega \leftarrow \omega - e_n$, $\boldsymbol{s} \leftarrow \boldsymbol{s} + e_n\boldsymbol{h}_n$ and $\boldsymbol{H} \leftarrow \boldsymbol{H}'$, $\boldsymbol{e} \leftarrow \boldsymbol{e}'$

Code–based one–way functions

Constructions

Let $\mathscr{C}$ be an (n,k) <u>*random*</u> linear code. There are two ways to build an OWF f based on the hardness of the CSD problem.

- Define $f : \mathbb{F}_2^k \times \mathrm{W}^n(\omega) \to \mathbb{F}_2^n$ via the generator (McEliece '78)

$$f(\boldsymbol{m}, \boldsymbol{e}) = \boldsymbol{m}\boldsymbol{G} + \boldsymbol{e}$$

- Define $f : \mathrm{W}^n(\omega) \to \mathbb{F}_2^{n-k}$ via the parity–check (Niederreiter '86)

$$f(\boldsymbol{m}) = \boldsymbol{m}\boldsymbol{H}^T$$

where $\mathrm{W}^n(\omega) \subset \mathbb{F}_2^n$ is the Hamming sphere of radius ω

Both are efficiently computable and (computationally) hard to invert

Code–based one–way functions (cont.)

Embedding a trapdoor

PKCs require efficient inversion of f given some auxiliary information

- Take a family of codes equipped with an efficient alg. Decode
  - Choose a member code $\mathscr{C}_{\mathsf{sec}}$ secretly

  The family and the decoding alg. are public
- Hide the structure of $\mathscr{C}_{\mathsf{sec}}$ by generating an equivalent code $\mathscr{C}_{\mathsf{pub}}$ via a <u>*random*</u> transformation

Implications

- One–wayness of the above T–OWF functions $\neq$ to the average–case hardness of CSD for random linear codes
- Choosing an *exponentially large* family of codes is necessary

Code–based cryptosystems

McEliece PKC

1. System setup: take the family of (n, k) irreducible binary Goppa codes with error correcting capability ω
   - fix a generator matrix $\boldsymbol{G}_{\mathsf{sec}} \in \mathbb{F}_2^{k \times n}$
   - generate *random* invertible $\boldsymbol{S} \in \mathbb{F}_2^{k \times k}$ and permutation $\boldsymbol{P} \in \mathbb{F}_2^{n \times n}$
   - set $\boldsymbol{G}_{\mathsf{pub}} = \boldsymbol{S}\boldsymbol{G}_{\mathsf{sec}}\boldsymbol{P}$

   Then $\mathcal{K}_{\mathsf{pub}} = \left(\boldsymbol{G}_{\mathsf{pub}}, \omega\right)$ and $\mathcal{K}_{\mathsf{sec}} = \left(\boldsymbol{S}, \boldsymbol{G}_{\mathsf{sec}}, \boldsymbol{P}\right)$
2. Encryption: given $\mathcal{K}_{\mathsf{pub}}$ and $\boldsymbol{m}$, output the ciphertext

$$\boldsymbol{c} = \boldsymbol{m}\boldsymbol{G}_{\mathsf{pub}} + \boldsymbol{e}, \qquad \boldsymbol{e} \in_{\mathsf{R}} \mathsf{W}^n(\omega)$$

3. Decryption: given $\mathcal{K}_{\mathsf{sec}}$ and $\boldsymbol{c}$, compute $\boldsymbol{m}' = \mathrm{Decode}\left(\boldsymbol{c}\boldsymbol{P}^T\right)$, where $\boldsymbol{c}\boldsymbol{P}^T = \left(\boldsymbol{m}\boldsymbol{S}\right)\boldsymbol{G}_{\mathsf{sec}} + \boldsymbol{e}\boldsymbol{P}^T$, and output the plaintext $\boldsymbol{m} = \boldsymbol{m}'\boldsymbol{S}^{-1}$

Code–based cryptosystems (cont.)

McEliece PKC (cont.)

Some basic facts about the construction:

- The family of irreducible binary Goppa codes is *exponentially large*

$$J = \frac{1}{\omega} \sum_{d|\omega} \mu\left(\frac{\omega}{d}\right) n^d \geq \frac{n^\omega}{\omega}\left(1 - 2n^{-\omega/2}\right)$$

  where usually $n = 2^m$ for some $m \in \mathbb{N}$ and μ is the Möbius function
- Admits fast decoding in $O(\omega n)$ time via the alg. of Patterson
- Parameters originally suggested in (McEliece '78) are

$$(n, k, \omega) = (1024, 524, 50)$$

  for an estimated security level of 80.7 bits, whereas $J \simeq 2^{494.4}$

Code–based cryptosystems (cont.)

Niederreiter PKC

1. System setup: take the family of (n, k) irreducible binary Goppa codes with error correcting capability ω
   - fix a parity–check matrix $\boldsymbol{H}_{\text{sec}} \in \mathbb{F}_2^{n-k \times n}$
   - generate *random* invertible $\boldsymbol{S} \in \mathbb{F}_2^{n-k \times n-k}$ and permutation $\boldsymbol{P} \in \mathbb{F}_2^{n \times n}$
   - set $\boldsymbol{H}_{\text{pub}} = \boldsymbol{S}^T \boldsymbol{H}_{\text{sec}} \boldsymbol{P}$

   Then $\mathcal{K}_{\text{pub}} = (\boldsymbol{H}_{\text{pub}}, \omega)$ and $\mathcal{K}_{\text{sec}} = (\boldsymbol{S}, \boldsymbol{H}_{\text{sec}}, \boldsymbol{P})$
2. Encryption: given $\mathcal{K}_{\text{pub}}$ and $\boldsymbol{m}$, output the ciphertext

$$\boldsymbol{c} = \boldsymbol{m}\boldsymbol{H}_{\text{pub}}^T, \qquad \boldsymbol{m} \in \mathrm{W}^n(\omega)$$

3. Decryption: given $\mathcal{K}_{\text{sec}}$ and $\boldsymbol{c}$, compute $\boldsymbol{m}' = \mathrm{Decode}(\boldsymbol{c}\boldsymbol{S}^{-1})$, where $\boldsymbol{c}\boldsymbol{S}^{-1} = (\boldsymbol{m}\boldsymbol{P}^T)\boldsymbol{H}_{\text{sec}}^T$, and output the plaintext $\boldsymbol{m} = \boldsymbol{m}'\boldsymbol{P}$

Code–based cryptosystems (cont.)

Niederreiter PKC (cont.)

Some basic facts about the construction:

- It was originally proposed with generalized Reed–Solomon (GRS) codes (they include Goppa codes)
- Attacked successfully in (Sidelnikov & Shestakov '92)
  - Exploited the factorization of $\boldsymbol{H}_{\text{sec}}$
  - Alg. to find trapdoor(s) $\mathcal{K}'_{\text{sec}} = \left(\boldsymbol{S}', \boldsymbol{H}'_{\text{sec}}, \boldsymbol{P}'\right)$ in polynomial time
  - McEliece PKC is not affected
- Niederreiter and McEliece PKCs are equivalent in terms of security, assuming the same setup (Li *et al.* '94)
  - M $\Leftarrow$ N: from $\boldsymbol{s} = \boldsymbol{c}\boldsymbol{H}^T_{\text{pub}} = \left(\boldsymbol{m}\boldsymbol{G}_{\text{pub}} + \boldsymbol{e}\right)\boldsymbol{H}^T_{\text{pub}}$ solve $\boldsymbol{s} = \boldsymbol{e}\boldsymbol{H}^T_{\text{pub}}$
  - N $\Leftarrow$ M: from $\boldsymbol{c} = \boldsymbol{m}\boldsymbol{H}^T_{\text{pub}}$ solve $\boldsymbol{y} = \boldsymbol{x}\boldsymbol{G}_{\text{pub}} + \boldsymbol{m}$ for appropriate $\boldsymbol{x}, \boldsymbol{y}$

Structural attacks

Definition

Given $\mathcal{K}_{\mathsf{pub}}$, structural attacks aim to recover the underlying structure of the code, i.e. $\mathcal{K}'_{\mathsf{sec}} \sim \mathcal{K}_{\mathsf{sec}}$

How many trapdoors?

- Initially thought to be unique (Adams & Meijer '87)
- It was later shown that *at least* $2\binom{n}{2}\log n$ exist (Gibson '91)
  - consider that $g \in \mathbb{F}_{2^m}[x]$ is irreducible with $\deg(g) = \omega$
  - equivalent Goppa polynomials g yield equivalent codes
- Weak constructions are obtained when $g \in \mathbb{F}_2[x]$ (Sendrier '00)
  - the *support–splitting algorithm* (SSA) tells if $\mathscr{C}_{\mathsf{pub}} \sim \mathscr{C}'$
  - its complexity is $O(\mathrm{poly}(n))$
  - uses properties (e.g. *hull*) invariant by a permutation

Structural attacks (cont.)

Overview of attacks

- GRS codes, and certain subcodes (Berger & Loidreau '05)
  - polynomial time algs. in both (Sidelnikov & Shestakov '92) and (Wieschebrink '10), with $O(n^3)$ in the worst case
- q–ary algebraic geometry codes (Janwa & Moreno '96)
  - polynomial time $O(n^4)$ alg. in (Faure & Minder '08)
  - only works for AG codes over low–genus hyperelliptic curves, e.g. $g = 1, 2$
- Alternant/QC–Goppa codes (Berger *et al.* '09) and QD–Goppa codes (Misoczki & Barreto '09)
  - broken in (Faugère *et al.* '10) using algebraic cryptanalysis
  - the added structure allows a drastic reduction of the number of unknowns

Structural attacks (cont.)

Overview of attacks (cont.)

- QC–LDPC (Baldi & Chiaraluce '07) as well as MDPC codes (Misoczki *et al.* '12)
  - polynomial time $O(n^3)$ alg. in (Otmani *et al.* '10) for QC codes
  - trapdoor for a punctured version of the secret code is obtained
- Reed–Muller codes (Sidelnikov '94)
  - subexponential time $O(\mathrm{poly}(n))\, e^{O(\mathrm{poly}(\log n))}$ alg. for any fixed order r in (Minder & Shokrollahi '07)
- Convolutional/Goppa codes (Löndahl & Johansson '12)
  - exponential time alg. in (Landais & Tillich '13) — claimed feasible
  - same idea of puncturing is applied, combined with ISD

Ciphertext indistinguishability

Property (McEliece)

In its current form the McEliece PKC is *not* IND–CPA

1. The adversary $\mathcal{A}$ submits $\boldsymbol{m}_0$ and $\boldsymbol{m}_1$
2. The system $\mathcal{E}$, flips a fair coin $a \in_{\mathsf{R}} \mathbb{F}_2$, and sends $\boldsymbol{c} = \boldsymbol{m}_a \boldsymbol{G}_{\mathsf{pub}} + \boldsymbol{e}$
3. If $\mathrm{wt}(\boldsymbol{c} + \boldsymbol{m}_0 \boldsymbol{G}_{\mathsf{pub}}) = \omega$ then
    - $\mathcal{A}$ knows $\boldsymbol{m}_0$ was encrypted, else
    - $\mathcal{A}$ knows $\boldsymbol{m}_1$ was encrypted

The adversary knows with certainty!

A simpler statement also holds for Niederreiter's PKC

- $\boldsymbol{c} \stackrel{?}{=} \boldsymbol{m}_0 \boldsymbol{H}_{\mathsf{pub}}^T$

Ciphertext indistinguishability (cont.)

Related plaintext attacks (Berson '97)

$\mathcal{A}$ chooses a desired target difference Δ

1. System: $\mathcal{E}$ encrypts $\boldsymbol{c}_i = \boldsymbol{m}_i \boldsymbol{G}_{\mathsf{pub}} + \boldsymbol{e}_i$ such that $\boldsymbol{m}_0 + \boldsymbol{m}_1 = \Delta$
2. Detection phase: $\mathcal{A}$ computes $\Delta' = \boldsymbol{c}_0 + \boldsymbol{c}_1 + \Delta \boldsymbol{G}_{\mathsf{pub}}$ $(= \boldsymbol{e}_0 + \boldsymbol{e}_1)$

$$\mathrm{wt}(\Delta') = 2\ell \leq 2\omega \;\Leftrightarrow\; \boldsymbol{m}_0 + \boldsymbol{m}_1 = \Delta$$

3. Attack's idea: Δ' reveals the positions of 2ℓ ones from $\boldsymbol{e}_0 + \boldsymbol{e}_1$

$$\begin{aligned} \boldsymbol{e}_0 &= \begin{pmatrix} 1_\ell & \mathbf{1}_{\omega-\ell} & 0_\ell & \mathbf{0}_{n-\omega-\ell} \end{pmatrix} \\ \boldsymbol{e}_1 &= \begin{pmatrix} 0_\ell & \mathbf{1}_{\omega-\ell} & 1_\ell & \mathbf{0}_{n-\omega-\ell} \end{pmatrix} \\ \Delta' &= \begin{pmatrix} 1_\ell & \mathbf{0}_{\omega-\ell} & 1_\ell & \mathbf{0}_{n-\omega-\ell} \end{pmatrix} \end{aligned}$$

Avoid $\omega - \ell$ positions from $n - 2\ell$: gives $\mathrm{Pr}_{\mathsf{isd}} = \binom{n-\omega-\ell}{k} / \binom{n-2\ell}{k}$
and $\mathrm{Pr}_{\mathsf{isd}}^{-1} \simeq 12.08$ if $\ell = 47$ with McEliece parameters

Ciphertext indistinguishability (cont.)

How to achieve IND–CPA

- In (Sun '98) a number of constructions are proposed

$$\begin{aligned} \boldsymbol{c} &= \big(\boldsymbol{m} + f(\boldsymbol{e})\big)\boldsymbol{G}_{\mathsf{pub}} + \boldsymbol{e}\,, & f &: \mathsf{W}^n(\omega) \to \mathbb{F}_2^k \\ \boldsymbol{c} &= \quad g\big(\boldsymbol{m}, \boldsymbol{e}\big)\boldsymbol{G}_{\mathsf{pub}} + \boldsymbol{e}\,, & g &: \mathbb{F}_2^k \times \mathsf{W}^n(\omega) \to \mathbb{F}_2^k \end{aligned}$$

  where f is OWF and g is T–OWF

- In (Berson '97), (Nojima '08) the construction proposed is

$$\boldsymbol{c} = \big(\boldsymbol{m} \;\; \boldsymbol{r}\big)\boldsymbol{G}_{\mathsf{pub}} + \boldsymbol{e}\,, \qquad \boldsymbol{m} \in \mathbb{F}_2^{\rho k} \text{ and } \boldsymbol{r} \in_{\mathsf{R}} \mathbb{F}_2^{(1-\rho)k}$$

  for some $\rho \in (0,1)$ with $\rho \ll \frac{1}{2}$ suggested

  - semantic security in the standard model is proved in (Nojima '08)
  - drawback: information rate decreased to ρR

ISD attacks

Overview

- Information set decoding (ISD) algorithms are generic decoding algorithms for solving the CSD_ω problem for *random linear codes*
  - seek for *information sets*, i.e. error–free positions in the ciphertext, from which the plaintext can be obtained
  - hence, they constitute per–message attacks
- They have exponential running time for any constant asymptotic rate R and error fraction W (Coffey & Goodman '90)

$$\alpha(R, W) = \lim_{n \to \infty} \frac{1}{n} \log_2 \mathcal{N}(n, Rn, Wn)$$

  is the *complexity coefficient*, where $\frac{k}{n} \to R$ and $\frac{\omega}{n} \to W$ as $n \to \infty$
- Currently, the best known algs. for attacking McEliece PKC

Plain ISD (Prange '62)

Main idea

- Compute the systematic form of a randomly permuted version of $\boldsymbol{H}_{\mathrm{pub}}$
- The syndrome's weight reveals when no errors occur in the first k coordinates of vector $\boldsymbol{e}$

Evaluation

In general $\alpha(R, W)$ implies $2^{\alpha(R,W)n+o(n)}$ complexity

```
input:  H, c, weight ω
1: compute s = cH^T  (= eH^T)
2: repeat
3:     select permutation P randomly
4:     apply Gauss elimination on HP to get
           H̃ = (B I_{n−k})
       where H̃ = QHP
5:     set s̃ = sQ^T
6:     set ẽ = (0 s̃)
7: until wt(ẽ) = ω
8: set e = ẽP^T                          » ẽ = eP
output:  error e
```

Transform performed:

$$\boldsymbol{s}^T = \boldsymbol{H}\boldsymbol{e}^T \Leftrightarrow \boldsymbol{Q}\boldsymbol{s}^T = \boldsymbol{Q}\boldsymbol{H}\boldsymbol{P}(\boldsymbol{P}^T\boldsymbol{e}^T)$$
$$\Leftrightarrow \tilde{\boldsymbol{s}}^T = (\boldsymbol{B}\ \boldsymbol{I}_{n-k})\tilde{\boldsymbol{e}}^T = \boldsymbol{B}\tilde{\boldsymbol{e}}_L^T + \tilde{\boldsymbol{e}}_R^T$$

Plain ISD (Prange '62) (cont.)

Complexity aspects

- Find a *good* permutation

$$\mathrm{Pr}_{\mathsf{p\text{-}isd}} = \frac{\binom{n-\omega}{k}}{\binom{n}{k}}$$

- Gauss elimination requires $T_{\mathsf{GE}}(n-k)$ operations:

$$T_{\mathsf{GE}}(x) \simeq \frac{1}{2} x^2 n$$

- Total number of operations

$$\mathcal{N}_{\mathsf{p\text{-}isd}}(n,k,\omega) = \frac{T_{\mathsf{GE}}(n-k)}{\mathrm{Pr}_{\mathsf{p\text{-}isd}}}$$

and $\alpha(R, W) = 0.1208$

input: $\boldsymbol{H}, \boldsymbol{c}$, weight ω

1: compute $\boldsymbol{s} = \boldsymbol{c}\boldsymbol{H}^T \;\; (= \boldsymbol{e}\boldsymbol{H}^T)$
2: **repeat**
3: select permutation $\boldsymbol{P}$ randomly
4: apply Gauss elimination on $\boldsymbol{HP}$ to get

$$\tilde{\boldsymbol{H}} = \begin{pmatrix} \boldsymbol{B} & \boldsymbol{I}_{n-k} \end{pmatrix}$$

where $\tilde{\boldsymbol{H}} = \boldsymbol{QHP}$
5: set $\tilde{\boldsymbol{s}} = \boldsymbol{s}\boldsymbol{Q}^T$
6: set $\tilde{\boldsymbol{e}} = \begin{pmatrix} \boldsymbol{0} & \tilde{\boldsymbol{s}} \end{pmatrix}$
7: **until** $\mathrm{wt}(\tilde{\boldsymbol{e}}) = \omega$
8: set $\boldsymbol{e} = \tilde{\boldsymbol{e}}\boldsymbol{P}^T$ » $\tilde{\boldsymbol{e}} = \boldsymbol{eP}$

output: error $\boldsymbol{e}$

Lee–Brickel ISD (Lee & Brickel '88)

Main idea

- Works on a randomly permuted version of $\boldsymbol{G}_{\text{pub}}$
- Rewrites $\boldsymbol{c} = \boldsymbol{m}\boldsymbol{G}_{\text{pub}} + \boldsymbol{e}$ as the system

$$\tilde{\boldsymbol{c}}_L = \boldsymbol{m}\boldsymbol{Q}^{-1} + \tilde{\boldsymbol{e}}_L$$
$$\tilde{\boldsymbol{c}}_R = \boldsymbol{m}\boldsymbol{Q}^{-1}\boldsymbol{A} + \tilde{\boldsymbol{e}}_R$$

- Allow $\leq p$ errors to occur in the first k coordinates of $\boldsymbol{e}$
  - Suggests taking p small

input: $\boldsymbol{G}$, $\boldsymbol{c}$, weight ω, parameter p

repeat
  select permutation $\boldsymbol{P}$ randomly
  apply Gauss elimination on $\boldsymbol{GP}$ to get

$$\tilde{\boldsymbol{G}} = \begin{pmatrix} \boldsymbol{I}_k & \boldsymbol{A} \end{pmatrix}$$

  where $\tilde{\boldsymbol{G}} = \boldsymbol{QGP}$
  set $\tilde{\boldsymbol{c}} = \boldsymbol{cP}$ » $\tilde{\boldsymbol{c}} = \begin{pmatrix} \tilde{\boldsymbol{c}}_L & \tilde{\boldsymbol{c}}_R \end{pmatrix}$
  for all $\tilde{\boldsymbol{e}}_L \in \mathrm{B}^k(p)$ **do**
    set $\tilde{\boldsymbol{e}}_R = \tilde{\boldsymbol{c}}_R + \left(\tilde{\boldsymbol{c}}_L + \tilde{\boldsymbol{e}}_L\right)\boldsymbol{A}$
    set $\tilde{\boldsymbol{e}} = \begin{pmatrix} \tilde{\boldsymbol{e}}_L & \tilde{\boldsymbol{e}}_R \end{pmatrix}$
    break if error is found
  end
until $\mathrm{wt}(\tilde{\boldsymbol{e}}) = \omega$
set $\boldsymbol{e} = \tilde{\boldsymbol{e}}\boldsymbol{P}^T$ » $\tilde{\boldsymbol{e}} = \boldsymbol{eP}$

output: error $\boldsymbol{e}$, message $\boldsymbol{m} = \left(\tilde{\boldsymbol{c}}_L + \tilde{\boldsymbol{e}}_L\right)\boldsymbol{Q}$

Note: $\mathrm{B}^k(p)$ is the Hamming ball of radius p

Lee–Brickel ISD (Lee & Brickel '88) (cont.)

Complexity aspects

- Find a *good* permutation

$$\Pr_{\text{lb-isd}} = \sum_{i=0}^{p} \frac{\binom{\omega}{i}\binom{n-\omega}{k-i}}{\binom{n}{k}}$$

- Gauss elimination requires $T_{\text{GE}}(k)$ operations
- Total number of operations $\mathcal{N}_{\text{lb-isd}}(n,k,\omega) =$

$$\frac{\left(T_{\text{GE}}(k) + |\mathrm{B}^k(p)|\,M\right)}{\Pr_{\text{lb-isd}}}$$

- Small improvement over P–ISD

input: $\boldsymbol{G}, \boldsymbol{c}$, weight ω, parameter p

1: **repeat**
2: select permutation $\boldsymbol{P}$ randomly
3: apply Gauss elimination on $\boldsymbol{GP}$ to get

$$\tilde{\boldsymbol{G}} = \left(\boldsymbol{I}_k \;\; \boldsymbol{A}\right)$$

where $\tilde{\boldsymbol{G}} = \boldsymbol{QGP}$

4: set $\tilde{\boldsymbol{c}} = \boldsymbol{cP}$ » $\tilde{\boldsymbol{c}} = \left(\tilde{\boldsymbol{c}}_L \;\; \tilde{\boldsymbol{c}}_R\right)$
5: **for all** $\tilde{\boldsymbol{e}}_L \in \mathrm{B}^k(p)$ **do**
6: set $\tilde{\boldsymbol{e}}_R = \tilde{\boldsymbol{c}}_R + \left(\tilde{\boldsymbol{c}}_L + \tilde{\boldsymbol{e}}_L\right)\boldsymbol{A}$
7: set $\tilde{\boldsymbol{e}} = \left(\tilde{\boldsymbol{e}}_L \;\; \tilde{\boldsymbol{e}}_R\right)$
8: break if error is found
9: **end**
10: **until** $\mathrm{wt}(\tilde{\boldsymbol{e}}) = \omega$
11: set $\boldsymbol{e} = \tilde{\boldsymbol{e}}\boldsymbol{P}^T$ » $\tilde{\boldsymbol{e}} = \boldsymbol{eP}$

output: error $\boldsymbol{e}$, message $\boldsymbol{m} = \left(\tilde{\boldsymbol{c}}_L + \tilde{\boldsymbol{e}}_L\right)\boldsymbol{Q}$

Leon ISD (Leon '88)

Main idea

- Works on the matrix $\boldsymbol{G}_{\text{pub}}$
- Rewrites $\boldsymbol{c} = \boldsymbol{m}\boldsymbol{G}_{\text{pub}} + \boldsymbol{e}$ as

$$\tilde{\boldsymbol{c}}_L = \boldsymbol{m}\boldsymbol{Q}^{-1} + \tilde{\boldsymbol{e}}_L$$
$$\tilde{\boldsymbol{c}}_M = \boldsymbol{m}\boldsymbol{Q}^{-1}\boldsymbol{B} + \tilde{\boldsymbol{e}}_M$$
$$\tilde{\boldsymbol{c}}_R = \boldsymbol{m}\boldsymbol{Q}^{-1}\boldsymbol{A} + \tilde{\boldsymbol{e}}_R$$

- Allow $\leq p$ errors to occur in the first k coordinates of $\boldsymbol{e}$
  - Suggests taking $p = 2$
- Assumes <u>no errors</u> in the next ℓ coordinates

input: $\boldsymbol{G}$, $\boldsymbol{c}$, weight ω, parameters p, ℓ

repeat
    select permutation $\boldsymbol{P}$ randomly
    apply Gauss elimination on $\boldsymbol{G}\boldsymbol{P}$ to get

$$\tilde{\boldsymbol{G}} = \begin{pmatrix} \boldsymbol{I}_k & \boldsymbol{B} & \boldsymbol{A} \end{pmatrix}$$

    where $\tilde{\boldsymbol{G}} = \boldsymbol{Q}\boldsymbol{G}\boldsymbol{P}$
    set $\tilde{\boldsymbol{c}} = \boldsymbol{c}\boldsymbol{P}$ » $\tilde{\boldsymbol{c}} = \begin{pmatrix} \tilde{\boldsymbol{c}}_L & \tilde{\boldsymbol{c}}_M & \tilde{\boldsymbol{c}}_R \end{pmatrix}$
    for all $\tilde{\boldsymbol{e}}_L \in \mathrm{B}^k(p)$ **do**
        if $\tilde{\boldsymbol{c}}_M = (\tilde{\boldsymbol{c}}_L + \tilde{\boldsymbol{e}}_L)\boldsymbol{B}$ **then**
            set $\tilde{\boldsymbol{e}}_R = \tilde{\boldsymbol{c}}_R + (\tilde{\boldsymbol{c}}_L + \tilde{\boldsymbol{e}}_L)\boldsymbol{A}$
            set $\tilde{\boldsymbol{e}} = \begin{pmatrix} \tilde{\boldsymbol{e}}_L & \boldsymbol{0} & \tilde{\boldsymbol{e}}_R \end{pmatrix}$
            break if error is found
        end
until $\mathrm{wt}(\tilde{\boldsymbol{e}}) = \omega$
set $\boldsymbol{e} = \tilde{\boldsymbol{e}}\boldsymbol{P}^T$ » $\tilde{\boldsymbol{e}} = \boldsymbol{e}\boldsymbol{P}$

output: error $\boldsymbol{e}$, message $\boldsymbol{m} = (\tilde{\boldsymbol{c}}_L + \tilde{\boldsymbol{e}}_L)\boldsymbol{Q}$

Leon ISD (Leon '88) (cont.)

Complexity aspects

- Find a *good* permutation

$$\Pr_{\text{l–isd}} = \sum_{i=0}^{p} \frac{\binom{\omega}{i}\binom{n-\omega}{k+\ell-i}}{\binom{n}{k+\ell}}$$

- Gauss elimination requires $T_{\text{GE}}(k)$ operations
- Total number of operations $\mathcal{N}_{\text{l–isd}}(n,k,\omega) =$

$$\frac{\left(T_{\text{GE}}(k) + |\mathrm{B}^k(p)|\widetilde{M}\right)}{\Pr_{\text{l–isd}}}$$

- Small improvement over LB–ISD

input: $\boldsymbol{G}, \boldsymbol{c}$, weight ω, parameters p, ℓ

repeat
    select permutation $\boldsymbol{P}$ randomly
    apply Gauss elimination on $\boldsymbol{GP}$ to get

$$\tilde{\boldsymbol{G}} = \begin{pmatrix} \boldsymbol{I}_k & \boldsymbol{B} & \boldsymbol{A} \end{pmatrix}$$

    where $\tilde{\boldsymbol{G}} = \boldsymbol{QGP}$
    set $\tilde{\boldsymbol{c}} = \boldsymbol{cP}$ » $\tilde{\boldsymbol{c}} = \begin{pmatrix} \tilde{\boldsymbol{c}}_L & \tilde{\boldsymbol{c}}_M & \tilde{\boldsymbol{c}}_R \end{pmatrix}$
    for all $\tilde{\boldsymbol{e}}_L \in \mathrm{B}^k(p)$ **do**
        if $\tilde{\boldsymbol{c}}_M = (\tilde{\boldsymbol{c}}_L + \tilde{\boldsymbol{e}}_L)\boldsymbol{B}$ **then**
            set $\tilde{\boldsymbol{e}}_R = \tilde{\boldsymbol{c}}_R + (\tilde{\boldsymbol{c}}_L + \tilde{\boldsymbol{e}}_L)\boldsymbol{A}$
            set $\tilde{\boldsymbol{e}} = \begin{pmatrix} \tilde{\boldsymbol{e}}_L & \boldsymbol{0} & \tilde{\boldsymbol{e}}_R \end{pmatrix}$
            break if error is found
        end
until $\mathrm{wt}(\tilde{\boldsymbol{e}}) = \omega$
set $\boldsymbol{e} = \tilde{\boldsymbol{e}}\boldsymbol{P}^T$ » $\tilde{\boldsymbol{e}} = \boldsymbol{eP}$

output: error $\boldsymbol{e}$, message $\boldsymbol{m} = (\tilde{\boldsymbol{c}}_L + \tilde{\boldsymbol{e}}_L)\boldsymbol{Q}$

Stern ISD (Stern '89), (Chabaud '94)

Main idea

- Allow $2p$ $(= p + p)$ errors in the first k bits of $\boldsymbol{e}$
- Assumes no errors in the next ℓ coordinates
  - ℓ should be small
- Rewrites $\boldsymbol{s} = \boldsymbol{e}\boldsymbol{H}^T$ as shown

$$\tilde{\boldsymbol{s}}_\ell = \tilde{\boldsymbol{e}}_{L_1}\boldsymbol{A}^T + \tilde{\boldsymbol{e}}_{L_2}\boldsymbol{B}^T + \tilde{\boldsymbol{e}}_M$$

$$\tilde{\boldsymbol{s}}_R = \tilde{\boldsymbol{e}}_{L_1}\boldsymbol{C}^T + \tilde{\boldsymbol{e}}_{L_2}\boldsymbol{D}^T + \tilde{\boldsymbol{e}}_R$$

- Introduces collisions to improve performance

input: $\boldsymbol{H}$, $\boldsymbol{c}$, weight ω, parameters p, ℓ

compute $\boldsymbol{s} = \boldsymbol{c}\boldsymbol{H}^T \ (= \boldsymbol{e}\boldsymbol{H}^T)$

repeat

  select permutation $\boldsymbol{P}$ randomly

  apply Gauss elimination on $\boldsymbol{H}\boldsymbol{P}$ to get

$$\tilde{\boldsymbol{H}} = \left(\begin{array}{cc|cc} \boldsymbol{A} & \boldsymbol{B} & \boldsymbol{I}_\ell & \\ \boldsymbol{C} & \boldsymbol{D} & & \boldsymbol{I}_{n-k-\ell} \end{array}\right)$$

  where $\tilde{\boldsymbol{H}} = \boldsymbol{Q}\boldsymbol{H}\boldsymbol{P}$

  set $\tilde{\boldsymbol{c}} = \boldsymbol{c}\boldsymbol{P}$ » $\tilde{\boldsymbol{c}} = \begin{pmatrix} \tilde{\boldsymbol{c}}_{L_1} & \tilde{\boldsymbol{c}}_{L_2} & \tilde{\boldsymbol{c}}_M & \tilde{\boldsymbol{c}}_R \end{pmatrix}$

  set $\tilde{\boldsymbol{s}} = \boldsymbol{s}\boldsymbol{Q}^T$ » $\tilde{\boldsymbol{s}} = \begin{pmatrix} \tilde{\boldsymbol{s}}_\ell & \tilde{\boldsymbol{s}}_R \end{pmatrix}$

  for all $\tilde{\boldsymbol{e}}_{L_1}, \tilde{\boldsymbol{e}}_{L_2} \in \mathrm{B}^{k/2}(p)$ **do**

    if $\tilde{\boldsymbol{s}}_\ell = \tilde{\boldsymbol{e}}_{L_1}\boldsymbol{A}^T + \tilde{\boldsymbol{e}}_{L_2}\boldsymbol{B}^T$ **then**

      set $\tilde{\boldsymbol{e}}_R = \tilde{\boldsymbol{s}}_R + \tilde{\boldsymbol{e}}_{L_1}\boldsymbol{C}^T + \tilde{\boldsymbol{e}}_{L_2}\boldsymbol{D}^T$

      set $\tilde{\boldsymbol{e}} = \begin{pmatrix} \tilde{\boldsymbol{e}}_{L_1} & \tilde{\boldsymbol{e}}_{L_2} & \boldsymbol{0} & \tilde{\boldsymbol{e}}_R \end{pmatrix}$

      break if error is found

    end

until $\mathrm{wt}(\tilde{\boldsymbol{e}}) = \omega$

set $\boldsymbol{e} = \tilde{\boldsymbol{e}}\boldsymbol{P}^T$ » $\tilde{\boldsymbol{e}} = \boldsymbol{e}\boldsymbol{P}$

output: error $\boldsymbol{e}$

Stern ISD (Stern '89), (Chabaud '94) (cont.)

Complexity aspects

- Find a *good* permutation

$$\Pr_{\text{s–isd}} = \frac{\binom{\omega}{p}\binom{n-\omega}{\frac{k}{2}-p}}{\binom{n}{\frac{k}{2}}} \times \frac{\binom{\omega-p}{p}\binom{n-\omega-(\frac{k}{2}-p)}{\frac{k}{2}-p}}{\binom{n-\frac{k}{2}}{\frac{k}{2}}} \times \frac{\binom{n-k-(\omega-2p)}{\ell}}{\binom{n-k}{\ell}}$$

- Gauss elimination requires $T_{\text{GE}}(n-k)$ operations

```
input: H, c, weight ω, parameters p, ℓ
compute s = cH^T  (= eH^T)
repeat
    select permutation P randomly
    apply Gauss elimination on HP to get
```

$$\tilde{\boldsymbol{H}} = \left(\begin{array}{cc|cc} \boldsymbol{A} & \boldsymbol{B} & \boldsymbol{I}_\ell & \\ \boldsymbol{C} & \boldsymbol{D} & & \boldsymbol{I}_{n-k-\ell} \end{array}\right)$$

```
    where H̃ = QHP
    set c̃ = cP                              » c̃ = (c̃_{L1} c̃_{L2} c̃_M c̃_R)
    set s̃ = sQ^T                            » s̃ = (s̃_ℓ s̃_R)
    for all ẽ_{L1}, ẽ_{L2} ∈ B^{k/2}(p) do
        if s̃_ℓ = ẽ_{L1}A^T + ẽ_{L2}B^T then
            set ẽ_R = s̃_R + ẽ_{L1}C^T + ẽ_{L2}D^T
            set ẽ = (ẽ_{L1} ẽ_{L2} 0 ẽ_R)
            break if error is found
    end
until wt(ẽ) = ω
set e = ẽP^T                                » ẽ = eP
output: error e
```

Stern ISD (Stern '89), (Chabaud '94) (cont.)

Complexity aspects

- Total number of operations

  $\mathcal{N}_{\text{s-isd}}(n,k,\omega) =$

  $$\frac{\left(T_{\text{GE}}(n-k) + 2\,|\mathrm{B}^{k/2}(p)|\,\widehat{M}\right)}{\Pr_{\text{s-isd}}}$$

  and $\alpha(R,W) = 0.1166$

- Considerable improvement over previous algs.

```
input:  H, c, weight ω, parameters p, ℓ
compute s = cH^T  (= eH^T)
repeat
    select permutation P randomly
    apply Gauss elimination on HP to get
```

$$\tilde{\boldsymbol{H}} = \left(\begin{array}{cc|cc} \boldsymbol{A} & \boldsymbol{B} & \boldsymbol{I}_\ell & \\ \boldsymbol{C} & \boldsymbol{D} & & \boldsymbol{I}_{n-k-\ell} \end{array}\right)$$

```
    where H̃ = QHP
    set c̃ = cP                          » c̃ = ( c̃_{L1} c̃_{L2} c̃_M c̃_R )
    set s̃ = sQ^T                        » s̃ = ( s̃_ℓ s̃_R )
    for all ẽ_{L1}, ẽ_{L2} ∈ B^{k/2}(p) do
        if s̃_ℓ = ẽ_{L1} A^T + ẽ_{L2} B^T then
            set ẽ_R = s̃_R + ẽ_{L1} C^T + ẽ_{L2} D^T
            set ẽ = ( ẽ_{L1} ẽ_{L2} 0 ẽ_R )
            break if error is found
        end
until wt(ẽ) = ω
set e = ẽP^T                            » ẽ = eP
output:  error e
```

Ball–collision decoding (Bernstein *et al.* '11)

Main idea

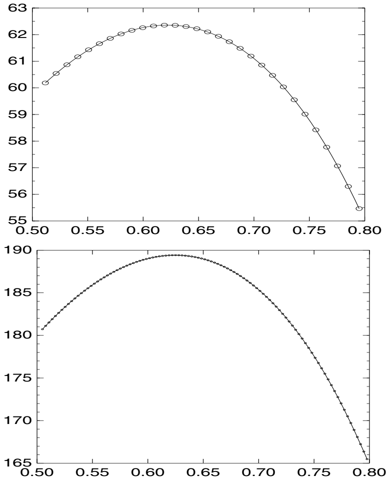
- Is one of the many generalizations of Stern's algorithm
- Takes the equivalent parity–check matrix $\tilde{\boldsymbol{H}} = \boldsymbol{QHP}$, where

$$\tilde{\boldsymbol{H}} = \left(\begin{array}{cc|cc} \boldsymbol{A} & \boldsymbol{B} & \boldsymbol{I}_\ell & \\ \boldsymbol{C} & \boldsymbol{D} & & \boldsymbol{I}_{n-k-\ell} \end{array}\right)$$

  but splits $k = k_1 + k_2$, and the parameters $\ell = \ell_1 + \ell_2$ and $2p = p_1 + p_2$
- Moreover, introduces q_1, q_2 to be the number of errors in the middle part
  - k_i, ℓ_i, p_i, q_i not necessarily equal
- Complexity is reduced w.r.t. S–ISD, with $\alpha(R, W) = 0.1163$

Ball–collision decoding (Bernstein *et al.* '11) (cont.)

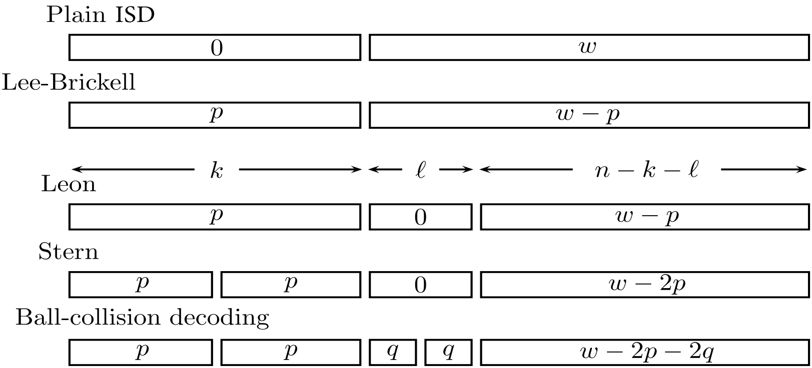
B–ISD costs for $n = i \cdot 1024,\ i = 1, 2, 3, 4$ (code rate in h–axis, and $\log_2(\text{cost})$ in v–axis)



Summary of ISD algorithms

Errors' structure

Assumptions w.r.t. the error positions by the above ISD algorithms



The LPN problem

Problem formulation

- Let $\boldsymbol{s} \in \mathbb{F}_2^k$ be some secret value and $p \in \left(0, \frac{1}{2}\right)$
- The LPN oracle $\mathcal{O}_p(\boldsymbol{s})$, at each query, returns independent random noisy samples

$$\left(\boldsymbol{a}, z\right) = \left(\boldsymbol{a}, \boldsymbol{s}\boldsymbol{a}^T + e\right)$$

  where $\boldsymbol{a} \in_{\mathsf{R}} \mathbb{F}_2^k$ and $\Pr[e = 1] = p$
- Adversaries may perform n queries to obtain $\left(\boldsymbol{A}, \boldsymbol{z}\right) = \left(\boldsymbol{A}, \boldsymbol{s}\boldsymbol{A} + \boldsymbol{e}\right)$, i.e. the system of equations

$$\begin{pmatrix} z_1 & \cdots & z_n \end{pmatrix} = \boldsymbol{s}\begin{pmatrix} \boldsymbol{a}_1^T & \cdots & \boldsymbol{a}_n^T \end{pmatrix} + \begin{pmatrix} e_1 & \cdots & e_n \end{pmatrix}$$

  where $\boldsymbol{e} \sim \mathcal{B}(n, p)$ with expected weight pn

The LPN problem (cont.)

Definition (LPN$_{k,p}$)

The algorithm $\mathcal{A}$ is said to (n, t, m, ε)–solve the LPN$_{k,p}$ problem if

$$\Pr\left[\mathcal{A}^{\mathcal{O}_p(\boldsymbol{s})}(1^k) = \boldsymbol{s}\right] \geq 1 - \varepsilon, \qquad \forall \boldsymbol{s} \in \mathbb{F}_2^k$$

making at most n queries to $\mathcal{O}_p(\boldsymbol{s})$, running in time at most t, and using memory at most m.

Attacking the problem

Best known algorithm is BKW (Blum *et al.* '00)

- Splits $\boldsymbol{s} = (\boldsymbol{s}_1 \ \cdots \ \boldsymbol{s}_b)$ into b blocks of length l
- Finds $\boldsymbol{s}_i$ independently by writing basis vectors $(1\ 0\ \cdots\ 0), \ \ldots$ as the sum of small number $(= \frac{1}{2}\sqrt{k})$ of samples
- Much smaller than that of Gaussian elimination $(\simeq k)$

The BKW algorithm

input: $k \times n$ matrix $\boldsymbol{G}$, vector $\boldsymbol{c}$, noise rate $p = \frac{1}{2} - \frac{1}{2}\eta$ » $\boldsymbol{c} = \boldsymbol{m}\boldsymbol{G} + \boldsymbol{e}$

initialization: choose $l, b \in \mathbb{Z} : lb \geq k$, set $r = \mathsf{poly}\left(\eta^{-2^b}, l\right)$

for $i = 1, \ldots, b$ **do**
  while $s \leq r$ **do**
    choose $\boldsymbol{A} \in_{\mathsf{R}} \boldsymbol{G}$ not previously used » has $O\left(b\,2^l\right)$ columns
    for $j \neq i$ **do**
      order $\boldsymbol{A} = (\boldsymbol{A}_0 \;\cdots\; \boldsymbol{A}_{2^l-1})$ w.r.t. the j-th block » l–bits long
      for all $\boldsymbol{A}_v \neq \emptyset$ **do**
        choose column $\boldsymbol{g}^T \in_{\mathsf{R}} \boldsymbol{A}_v$
        add $\boldsymbol{g}^T$ to the rest of the columns in $\boldsymbol{A}_v$
        remove $\boldsymbol{g}^T$ from $\boldsymbol{A}_v$
      end
    end
    find $\boldsymbol{I}_l$ in $\boldsymbol{A}$ w.r.t. the i-th block » has $O\left(2^l\right)$ columns
    get the s-th sample $\tilde{\boldsymbol{c}}_s = \boldsymbol{m}_i + \tilde{\boldsymbol{e}}_s$ » sum of 2^{b-1} values
  end
  $m_{ij} = \mathsf{Decode}_{\mathsf{MLG}}\left((\tilde{c}_{1j} \;\cdots\; \tilde{c}_{rj}), q\right)$, $j = 1, \ldots, l$ » $q = \frac{1}{2} - \frac{1}{2}\eta^{2^{b-1}}$
end

output: estimate $\boldsymbol{m} = (\boldsymbol{m}_1 \;\cdots\; \boldsymbol{m}_b)$

The BKW algorithm (cont.)

Iteration for a single estimate

BKW determines e.g. $\boldsymbol{m}_1$ in $\boldsymbol{m} = (\boldsymbol{m}_1 \ \cdots \ \boldsymbol{m}_b)$ as follows

$$\begin{pmatrix} \boldsymbol{A}_{1,1}^{(0)} & \boldsymbol{A}_{1,2}^{(0)} & \cdots & \boldsymbol{A}_{1,2^l}^{(0)} \\ \vdots & \vdots & & \vdots \\ \boldsymbol{A}_{b-1,1}^{(0)} & \boldsymbol{A}_{b-1,2}^{(0)} & \cdots & \boldsymbol{A}_{b-1,2^l}^{(0)} \\ \boldsymbol{0}^{(0)} & \boldsymbol{1}_1^{(0)} & \cdots & \boldsymbol{1}_{2^l-1}^{(0)} \end{pmatrix} \longrightarrow \begin{pmatrix} \boldsymbol{A}_{1,1}^{(1)} & \boldsymbol{A}_{1,2}^{(1)} & \cdots & \boldsymbol{A}_{1,2^l}^{(1)} \\ \vdots & \vdots & & \vdots \\ \boldsymbol{0}^{(1)} & \boldsymbol{1}_1^{(1)} & \cdots & \boldsymbol{1}_{2^l-1}^{(1)} \\ \hline & & & \end{pmatrix} \longrightarrow \cdots$$

$$\vdots$$

$$\begin{array}{c} \text{pick columns labeled} \\ \text{with } \boldsymbol{1}_{2^i}^{(b-1)},\ 0 \le i < l \end{array} \longleftarrow \begin{pmatrix} \boldsymbol{0}^{(b-1)} & \boldsymbol{1}_1^{(b-1)} & \cdots & \boldsymbol{1}_{2^l-1}^{(b-1)} \\ \hline & & & \\ & & & \end{pmatrix} \longleftarrow \cdots$$

The BKW algorithm (cont.)

Theorem (Blum *et al.* '00)

For $k = lb$, the $\mathrm{LPN}_{k,p}$ problem is solved with $\mathrm{poly}\left(\eta^{-2^b}, 2^l\right)$ sample size and computation time.

Suggested values

For $k = lb$, and every *fixed* noise rate (i.e. η), take

$$b = \frac{1}{2}\log k \quad \text{and} \quad l = 2k/\log k$$

to get $2^{O(k/\log k)}$ sample size and computation time

Theorem (Levieil & Fouque '06)

For $k = lb$, the BKW algorithm $\left(n, O(kbn), kn, \frac{1}{2}\right)$–solves the $\mathrm{LPN}_{k,p}$ problem, where $n = 20\ln(4k)\,2^l\,\eta^{-2^b}$.

Conclusions

What key sizes?

- As stated, ISD algorithms have superior performance

Parameters	n	1632	2960	6624	30332
	k	1269	2288	5129	22968
	ω	34	57	117	494
Bit security	S–ISD	82.23	129.84	258.61	1007.4
	B–ISD	81.33	127.89	254.15	996.22
	BKW	123.09	205.02	416.16	1585.4

- However, BKW is more robust to high noise rate

Conclusions (cont.)

How secure?

McEliece PKC over *rational Goppa codes* and Niederreiter PKC over *classical Goppa codes* resist quantum attacks (Dinh *et al.* '11)

- The *strong Fourier sampling* on which all known exponential speedups by quantum algorithms are based, is not applicable

Open problems?

- Strong need for other (capacity–approaching) codes
  - Attempts to use LDPC/QC–LDPC have failed (MDPC?)
  - Recent attempts to use polar codes
- Great need for space–efficient ISD algorithms
  - ISD algorithms generate large lists to find collisions

References

E. Berlekamp, R. McEliece, and H. Van Tilborg
On the inherent intractability of certain coding problems
IEEE Trans. Inform. Theory 24 (3) pp. 384–386, 1978

R. McEliece
A public–key cryptosystem based on algebraic coding theory
DSN progress report 42–44 pp. 114–116, 1978

H. Niederreiter
Knapsack–type cryptosystems and algebraic coding theory
Probl. Control Inform. Theory 15 (2) pp. 157–166, 1986

P. J. Lee and E. F. Brickell
An observation on the security of McEliece's public–key cryptosystem
In proc. EUROCRYPT '88 pp. 275–280, 1988

J. Stern
A method for finding codewords of small weight
In proc. Coding Theory Appl. pp. 106–113, 1989

References (cont.)

T. A. Berson
Failure of McEliece public–key cryptosystem under message–resend and related–message attack
In proc. CRYPTO '97 pp. 213–220, 1997

A. Blum, A. Kalai, and H. Wasserman
Noise–tolerant learning, the parity problem, and the statistical query model
In proc. ACM STOC '00 pp. 435–440, 2000

D. J. Bernstein, T. Lange, and C. Peters
Smaller decoding exponents: ball–collision decoding
In proc. CRYPTO '11 pp. 743–760, 2011

H. Dinh, C. Moore, and A. Russell
McEliece and Niederreiter cryptosystems that resist quantum Fourier sampling attacks
In proc. CRYPTO '11 pp. 761–779, 2011

A. Becker, A. Joux, A. May, A. Meurer
Decoding random binary linear codes in $2^{n/20}$: how $1 + 1 = 0$ improves ISD
In proc. EUROCRYPT '12 pp. 520–536, 2012

Questions & Answers

Thank you for your attention!