Attacking and defending the McEliece cryptosystem

(Joint work with Daniel J. Bernstein and Tanja Lange)

Christiane Peters

EIPSI Seminar

October 1, 2008

Outlook

On the attack side

- present improvements to Stern's attack on the McEliece cryptosystem.
- show that the McEliece cryptosystem with original parameters $n = 1024$, $k = 524$, $t = 50$ can be broken in just 1400 days by a single 2.4GHz Core 2 Quad CPU, or 7 days by a cluster of 200 CPUs.

On the defense side

- can make use of list decoding for Goppa codes.
- use codes whose length is not a power of 2.
- achieve considerably smaller public-key sizes than previous parameter choices for the same security level.

1. Review of the McEliece cryptosystem

2. Stern's attack

3. Attack optimization and comparison

4. A successfull attack on the original McEliece parameters

5. Defending the McEliece cryptosystem

Linear codes

A binary $[n, k]$ code is a binary linear code of length n and dimension k, i.e., a k-dimensional subspace of $\mathbf{F}_2^n$.

A generator matrix of an $[n, k]$ code C is a $k \times n$ matrix G such that $C = \{\mathbf{x}\,G : \mathbf{x} \in \mathbf{F}_2^k\}$.

The matrix G corresponds to a map $\mathbf{F}_2^k \to \mathbf{F}_2^n$ sending a message of length k to an n-bit string.

A parity-check matrix of an $[n, k]$ code C is an $(n-k) \times n$ matrix H such that $C = \{\mathbf{c} \in \mathbf{F}_2^n : H\,\mathbf{c}^T = 0\}$.

A systematic generator matrix is a generator matrix of the form $(I_k|Q)$ where I_k is the $k \times k$ identity matrix and Q is a $k \times (n-k)$ matrix (redundant part).

The matrix $H = (Q^T|I_{n-k})$ is then a parity-check matrix for C.

Decoding problem

The Hamming weight of an element $\mathbf{c} \in \mathbf{F}_2^n$ is the number of nonzero entries of $\mathbf{c}$.

The minimum distance of an $[n, k]$ code C with $k > 0$ is the smallest Hamming weight of any nonzero element of C.

Classical decoding problem: find the closest codeword $\mathbf{x} \in C$ to a given $\mathbf{y} \in \mathbf{F}_2^n$, assuming that there is a unique closest codeword.

Close means that the difference has small Hamming weight. Uniqueness is guaranteed if there exists a codeword $\mathbf{x}$ whose distance from $\mathbf{y}$ is less than half the minimum distance of C.

Decoding a generic binary code of length n without knowing anything about its structure is a difficult problem: about $2^{(0.5+o(1))n/\log_2(n)}$ binary operations required.

The McEliece cryptosystem

Given a length-n binary Goppa code Γ of dimension k with minimum distance $2t+1$ where $t \approx (n-k)/\log_2(n)$.
(original parameters: $n = 1024$, $k = 524$, $t = 50$)

The McEliece secret key consists of a generator matrix G for Γ, an efficient t-error correcting decoding algorithm for Γ; an $n \times n$ permutation matrix P and a nonsingular $k \times k$ matrix S.

n, k, t are public; but Γ, P, S are randomly generated secrets.

The McEliece public key is the $k \times n$ matrix SGP.

Encryption of a message $\mathbf{m}$ of length k: Compute $\mathbf{m}SGP$ and add a random error vector $\mathbf{e}$ of weight t and length n.
Send $\mathbf{y} = \mathbf{m}SGP + \mathbf{e}$.

McEliece decryption: Compute $\mathbf{y}P^{-1} = \mathbf{m}SG + \mathbf{e}P^{-1}$.
Use decoding algorithm to find $\mathbf{m}S$ and thereby $\mathbf{m}$.

Note on CCA2-secure variants of the PKC

McEliece's system in the original setting doesn't resist chosen-ciphertext attacks ("IND-CCA2 security").

Reduce computational costs of an attack by making use of

- partially known plaintexts
- relations between messages

Engelbert et al. in 2006 and Overbeck in 2008:

- scrambling the message inputs;
  destroy any relations of two dependent messages which an adversary might be able to exploit.

CCA2-secure McEliece PKC requires less storage for public key: suffices to store the redundant part of systematic generator matrix; reduce public-key size from kn bits to $k(n-k)$ bits.

1. Review of the McEliece cryptosystem
2. Stern's attack
3. Attack optimization and comparison
4. A successfull attack on the original McEliece parameters
5. Defending the McEliece cryptosystem

Look for minimum weight words in a (slightly larger) code

McEliece ciphertext $\mathbf{y} \in \mathbf{F}_2^n$ has distance t from a unique closest codeword $\mathbf{c} = \mathbf{m}G$ in a code C which has minimum distance at least $2t + 1$.

Find $\mathbf{e}$ of weight t such that $\mathbf{c} = \mathbf{y} - \mathbf{e}$:

- append $\mathbf{y}$ to the list of generators
- and form a generator matrix for $C + \{0, \mathbf{y}\}$.

Then

$$\mathbf{e} = (\mathbf{m}, 1) \left(\frac{G}{\mathbf{m}G + \mathbf{e}} \right)$$

is a codeword in $C + \{0, \mathbf{y}\}$; and it is the only weight-t word.

bottleneck in all of these attacks is finding the weight-t codeword in $C + \{0, \mathbf{y}\}$ which has slightly larger dimension, namely $k + 1$.

Stern's attack

Given $w \geq 0$ and an $(n-k) \times n$ parity check matrix H for a binary $[n,k]$ code C. Find $\mathbf{c} \in C$ of weight w.
($H\mathbf{c}^T = 0 \rightarrow$ find w cols adding up to 0)

Fix integers ℓ and p.

Step 1 Select randomly $n-k$ linearly independent columns of H. (form identity matrix using Gauss elimination)

Select randomly a size-ℓ subset Z of those $n-k$ columns.

Partition the remaining k columns into two sets X and Y (independently, uniformly distributed)

Step 2 Search for a codeword $\mathbf{c}$ having exactly $p, p, 0$ ones in the column sets X, Y, Z and exactly $w-2p$ nonzero bits in the remaining columns.

Step 3 If Step 2 was successfull return $\mathbf{c}$.

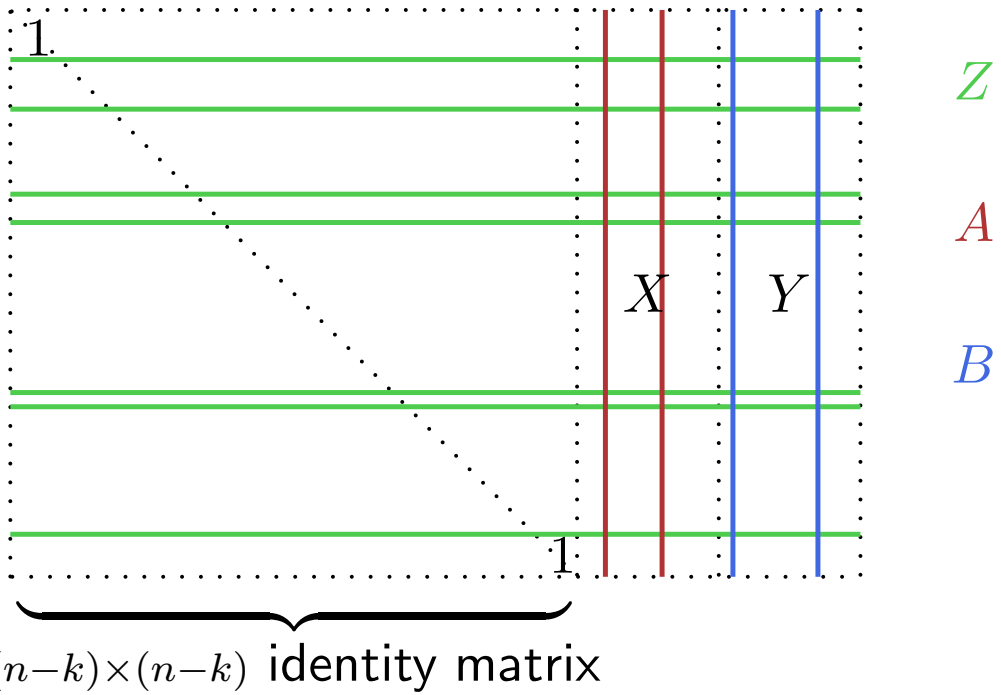
Else go back to Step 1.

Step 2 of Stern's algorithm

Consider all p-element subsets A of X.
Compute the ℓ-bit vector $\pi(A)$ by adding up the columns of the matrix

$$H' = (H_{i,j})_{i \in Z, j \in A}.$$

Similarly, compute $\pi(B)$.



- For each collision $\pi(A) = \pi(B)$ compute the sum of the $2p$ columns in $A \cup B$. This sum is an $(n-k)$-bit vector.
- If the sum has weight $w - 2p$, we obtain 0 by adding the corresponding $w - 2p$ columns in the $(n-k) \times (n-k)$ submatrix.

Get a weight-w word c with $Hc^T = 0$ by setting $c_m = 1$ for $A \cup B$ and $c_m = 1$ for m indexing the $w - 2p$ columns in the submatrix.

1. Review of the McEliece cryptosystem
2. Stern's attack
3. Attack optimization and comparison
4. A successfull attack on the original McEliece parameters
5. Defending the McEliece cryptosystem

The new attack (1)

Step 1 Reusing existing pivots in each iteration of Stern's algorithm

Forcing more existing pivots: *reuse exactly* $n - k - c$ *column selections (Canteaut et al.:* $c = 1$*)*

Faster pivoting: *suppose that we defer additions of* r *rows; after precomputing all* $2^r - 1$ *sums of nonempty subsets of these rows, we can handle each remaining row with, on average,* $1 - 1/2^r$ *vector additions, rather than* $r/2$ *vector additions.*

Multiple choices of Z: *allow* m *disjoint sets* $Z_1, \ldots, Z_m$ *s.t. the word we're looking for has weight* $p, p, 0 \ldots, 0$ *on the sets* $X, Y, Z_1, \ldots, Z_m$

The new attack (2)

Step 2 Reusing additions of the ℓ-bit vectors when considering p-element subsets A of X: *Caching additions leaves a little more that ℓ additions and not $p\ell$ as Stern and Canteaut et al. claim*

Faster additions after collisions: *After computing $2(w - 2p + 1)$ rows one has, on average, $w - 2p + 1$ errors; abandon pair (A, B) as soon as number of errors exceeds $w - 2p$*

Success chance of one iteration of the attack

The probability of a weight-w word having exactly $w-2p$ errors in a uniform random set of $n-k$ columns is $\binom{w}{2p}\binom{n-w}{k-2p}/\binom{n}{k}$.

We select randomly $\lfloor k/2 \rfloor$ columns for X and $\lceil k/2 \rceil$ columns for Y. So the conditional probability of $2p$ errors splitting as p, p between X, Y is $\binom{\lfloor k/2 \rfloor}{p}\binom{\lceil k/2 \rceil}{p}/\binom{k}{2p}$,

The conditional probability of the remaining $w-2p$ errors avoiding at least one of the disjoint ℓ-size sets $Z_1, Z_2, \ldots, Z_m$ is

$$m\frac{\binom{n-k-(w-2p)}{\ell}}{\binom{n-k}{\ell}} - \binom{m}{2}\frac{\binom{n-k-(w-2p)}{2\ell}}{\binom{n-k}{2\ell}} + \binom{m}{3}\frac{\binom{n-k-(w-2p)}{3\ell}}{\binom{n-k}{3\ell}} - \cdots$$

by the inclusion-exclusion principle.

The product of these probabilities is the chance that the *first* iteration succeeds.

Iterations

Stern: iterations are independent (in each step $n-k$ linearly independent columns are randomly chosen); the average number of iterations be simply the reciprocal of the product of the probabilities.

Our attack: iterations are not independent!
Number of errors in the selected $n-k$ columns is correlated with the number of errors in the columns selected in the next iteration.

Extreme case $c=1$ considered by Canteaut et al.:
swapping one selected column for one deselected column is quite likely to preserve the number of errors in the selected columns.

The effect decreases in magnitude as c increases, but iterations also become slower as c increases.

Analyze the impact of selecting c new columns:

Compute a Markov chain for the number of errors; generalizing the analysis of Canteaut et al. from $c = 1$ to arbitrary c.
States of the chain:

0: There are 0 errors in the deselected k columns.
1: There is 1 error in the deselected k columns.
...
w: There are w errors in the deselected k columns.
Done: The attack has succeeded.

Starting from state u, the attack replaces c selected columns, moving to states $u-c, \ldots, u-2, u-1, u, u+1, u+2, \ldots, u+c$ with various probabilities (three types of choosing new cols). The attack then checks for success, moving from state $2p$ to state Done with probability

$$\beta = \frac{\binom{\lfloor k/2 \rfloor}{p}\binom{\lceil k/2 \rceil}{p}}{\binom{k}{2p}}\left(m\frac{\binom{n-k-(w-2p)}{\ell}}{\binom{n-k}{\ell}} - \binom{m}{2}\frac{\binom{n-k-(w-2p)}{2\ell}}{\binom{n-k}{2\ell}} + \cdots\right)$$

and otherwise staying in the same state.

Complexity

Canteaut, Chabaud, and Sendrier: an attacker can decode 50 errors in a $[1024, 524]$ code over $\mathbf{F}_2$ in $2^{64.1}$ bit operations.

Choosing parameters $p = 2$, $m = 2$, $\ell = 20$, $c = 7$, and $r = 7$ in our new attack shows that the same computation can be done in only $2^{60.55}$ bit operations, almost a $12\times$ improvement over Canteaut et al.

The number of iterations drops from $9.85 \cdot 10^{11}$ to $4.21 \cdot 10^{11}$, and the number of bit operations per iteration drops from $20 \cdot 10^6$ to $4 \cdot 10^6$.

1. Review of the McEliece cryptosystem
2. Stern's attack
3. Attack optimization and comparison
4. A successfull attack on the original McEliece parameters
5. Defending the McEliece cryptosystem

Implementation

Our attack software extracts a plaintext from a ciphertext by decoding 50 errors in a $[1024, 524]$ binary code.

Attack on a single computer with a 2.4GHz Intel Core 2 Quad Q6600 CPU would need, on average, approximately 1400 days (2^{58} CPU cycles) to complete the attack.

Running the software on 200 such computers would reduce the average time to one week.

Canteaut, Chabaud, and Sendrier: implementation on a 433MHz DEC Alpha CPU; one such computer would need approximately 7400000 days (2^{68} CPU cycles).

Hardware improvements only reduce 7400000 days to 220000 days.

The remaining speedup factor of 150 comes from our improvements of the attack itself.

We gratefully acknowledge contributions of CPU time from

- 38 cores of the Coding and Cryptography Computer Cluster (C4) at TU/e;
- 32 cores in the Department of Electrical Engineering at National Taiwan University;
- 22 cores in the CACAO cluster at Laboratoire Lorrain de Recherche en Informatique et ses Applications (LORIA);
- 16 cores of the System Architecture and Networking Distributed and Parallel Integrated Terminal (sandpit) at TU/e;
- 8 cores of the Argo cluster at the Academic Computing and Communications Center at the University of Illinois at Chicago (UIC);
- 6 cores at the Center for Research and Instruction in Technologies for Electronic Security (RITES) at UIC; and
- 4 cores owned by D. J. Bernstein and Tanja Lange.

We are carrying out about $3.26 \cdot 10^9$ attack iterations each day. At the moment our chances of success are 2% per day.

1. Review of the McEliece cryptosystem
2. Stern's attack
3. Attack optimization and comparison
4. A successfull attack on the original McEliece parameters
5. Defending the McEliece cryptosystem

Improvements

Increasing n The most obvious way to defend McEliece's cryptosystem is to increase n, the length of the code used in the cryptosystem.

Allowing values of n between powers of 2 allows considerably better optimization of (e.g.) the McEliece public-key size.

Using list decoding to increase w

2008: Bernstein introduced a list-decoding algorithm for classical irreducible binary Goppa codes.

The receiver can efficiently decode approximately $n - \sqrt{n(n-2t-2)} \geq t+1$ errors instead of t errors. The sender can introduce correspondingly more errors.

If list decoding returns more that one codeword within a specified distance, CCA2-secure variants of the McEliece system ensure that the valid codeword is identified.

Proposed parameters $[n, k]$ for various security levels

Using CCA2-secure variants of the McEliece PKC the public key needs only $k(n-k)$ bits.

For 80-bit security against our attack we propose $[1632, 1269]$ Goppa codes (degree $t = 33$), with 34 errors added by the sender. Public-key size: 460647 bits.

Without list decoding, and restriction $n = 2^d$: $[2048, 1751]$ Goppa codes ($t = 27$). Public key size: 520047 bits.

For 128-bit security: we propose $[2960, 2288]$ Goppa codes ($t = 56$), with 57 errors added by the sender. Public-key size: 1537536 bits.

For 256-bit security: $[6624, 5129]$ Goppa codes ($t = 115$), with 117 errors added by the sender. Public-key size: 7667855 bits.

Thank you for your attention!