Implementational aspects of code-based cryptography

Ph.D. Defence of

Bhaskar Biswas

Supervised by

Nicolas Sendrier and Luc Bouganim

INRIA-Rocquencourt and École Polytechnique, France

École Polytechnique, France

October 1, 2010

Introduction

Cryptology

My research domain is cryptology.
Cryptology ($\kappa\rho\upsilon\pi\tau\acute{o}\sigma$ (*adj.* secret); $\lambda o\gamma\iota\alpha$ (*n.* sayings or body of knowledge)): is the science of making communication incomprehensible to every one but those who have a right to access it. It has two wings,

$\rightarrow$ Cryptography: concerns with the system design.

$\rightarrow$ Cryptanalysis: concerns with system analysis (breaking?).

Two types of systems,

- Symmetric key; same key is used for encryption and decryption.
- Asymmetric key; different keys for encryption and decryption.

Asymmetric systems

The concept of public key (asymmetric) cryptography came to existence to address the following two difficult problems associated with the secret key (symmetric) cryptosystems.

- Key distribution and
- Non-repudiation.

To address both problems, Diffie and Hellman achieved one of the greatest breakthrough in cryptography in 1976.The proposed scheme used a pair of different keys in contrast to one shared key and it was based on mathematical functions instead of substitution and permutation. The use of two keys has profound consequences in facilitating key distribution and providing digital signature.

Public Key Encryption schemes

All the systems have three distinct steps.

1. **Key generation**, generates a set of *public* and *private* keys.
2. **Encryption**, encryption of the *plain text* with the help of *public* key.
3. **Decryption**, decryption of the ciphertext with the help of *private* key, which is hard to deduce from the *public* key.

The security of these systems depend upon some underlying hard problem.

Short history : (my) view

Almost solely used public key encryption scheme is RSA, it is based on the problem of factorization.

Next possible alternative is Elliptic curve systems, because of the better performance for high security levels.

Alternative systems are always of interest. They are attracting more attention as *quantum alternatives*.

Thus, subjects like lattice based crypto, code based crypto and multivariate crypto are getting more interest.

My research area is code based cryptography.

McEliece encryption scheme

An asymmetric key system, proposed by Robert McEliece in 1978.

Based on the well known and well studied problem of "hardness of decoding a general linear code".

Private key is the description of t-error-correcting code with efficient decoding algorithm (binary Goppa code).

Public key is the disguised form of the code selected, as a general linear code (pseudo-randomness of Goppa code).

McEliece encryption scheme cont...

- ▲ One of the fastest public key system (160 Mbits/s).
- ▲ Secure; resisted cryptanalysis till date.
- ▲ Remain secure; will resist the threat posed by quantum computers.
- ▲ Good scalability; security grows exponentially if we raise parameters.
- ▼ Rather large key size (100 KB; 88 bit security).
- ▼ Lesser information rate.
- ▼ Rarely used in practice, implementation has not been studied before with full state of the art.

Overview of McEliece scheme

Let $\mathcal{F}$ be a family of binary t-error-correcting (n, k) codes. We describe McEliece cryptosystem as,

- public key: a $k \times n$ binary *generator matrix* G of a code $\in \mathcal{F}$.
- secret key: a decoder for the said code.
- encryption: $x \mapsto xG + e$ where, hamming weight of e is $w_H(e) \leq t$.
- decryption: Essentially the decoding process.

It was introduced by R. McEliece in 1978 with irreducible binary Goppa codes.

Brief time line

- Proposed in 1978 by Robert J. McEliece.
- Niederreiter proposed a variant! Knapsack type. 1986.
- Attacks:

Decoding attacks (Information set decoding)	Lee and Brickell - '88.
	Leon - '88.
	J. van Tilburg - '88.
	J. Stern - '89.
	A. Canteaut and F. Chabaud - '95.
Structural attacks (Goppa codes)	Support splitting algorithm
	P. Loidreau and N. Sendrier - '01.

- D. Bernstein, T. Lange and C. Peters - Successful attack, '08.
- N. Courtois, M. Finiasz and N. Sendrier - Signature Scheme, '01.

Niederreiter cryptosystem

Proposed by Harald Niederreiter in 1986 a variant of McEliece scheme.

Uses parity check matrix instead of generator matrix.

Embeds information in the error pattern.

The security reduction is identical as the McEliece scheme.

Theoretically it is the **dual** system of McEliece.

Our objectives

To study the scope of McEliece system in real life; its practicality, efficiency and security aspects.

- Full state of the art implementation of McEliece public key cryptosystem.
- Reference platform.
- Algorithmic improvement.
- Optimized implementation.

The Hybrid McEliece Encryption Scheme (HyMES)

Our proposal

1. Generator matrix in row echelon form, reduces key size.
2. Encode information in the error, increases the *information rate*.

In spite of the changes we preserve security.

We improve,

- Key size.
- Encryption speed.
- Information rate.

As we mix the ideas of McEliece and Niederreiter schemes, we call our system **Hybrid**.

HyMES Description

We define an injective mapping $\varphi : \{0,1\}^{\ell} \to W_{n,t}$, a set of n length words weight t. Both φ and φ^{-1} should be easy to compute.

- Key generation:
  - We construct the necessary tools to design and operate on $\mathbf{GF}$ along with polynomial representation.
  - generate a support $L = (\alpha_1, \ldots, \alpha_n)$ of n distinct elements of $\mathbf{F}_{2^m}$,
  - generate a monic irreducible generator polynomial $g(z) \in \mathbf{F}_{2^m}[z]$ of degree t.
  - The secret key is the pair (L, g) (i.e. the Goppa code $\Gamma(L, g)$ and its decoder)
  - The public key is a binary $k \times (n-k)$ matrix R where $G = (Id \mid R)$ is a generator matrix of $\Gamma(L, g)$ in row echelon form.

Description cont...

- Encryption: the plaintext is in $\{0,1\}^k \times \{0,1\}^\ell$ and the ciphertext in $\{0,1\}^n$

$$\begin{array}{ccc} \{0,1\}^k \times \{0,1\}^\ell & \longrightarrow & \{0,1\}^n \\ (x, x') & \longmapsto & (x, xR) + \varphi(x') \end{array}$$

- Decryption: the ciphertext has the form $y = xG + e$, with $e = \varphi(x')$ of Hamming weight $\leq t$. Applying the decoder of $\Gamma(L, g)$ on y will provide x and $x' = \varphi^{-1}(e)$.

Remark : We try to chose ℓ, as close as $\log \binom{n}{t}$ for better information rate and security.

Security

- One Way Encryption(OWE) scheme: A public key encryption scheme is a *One Way Encryption* scheme if the probability of success of any of its adversary running in polynomial time is negligible.

In practice, one needs more than just an OWE scheme.

- McE, though it is OWE, is vulnerable to many attacks.
- Given perfect hash functions exists, there are generic conversions which, starting from an OWE scheme, provide a scheme resistant against adaptative chosen ciphertext attack.

We can prove, given the two following assumptions, the hybrid McEliece encryption scheme is OWE.

Security assumptions

- Hardness of decoding in the average case: The success probability

$$\Pr\left(\mathcal{A}(xG + e, G) = (x, e)\right)$$

  of any adversary $\mathcal{A}$ running in polynomial time is negligible.

- Pseudo-randomness of binary Goppa Codes: there exists no efficient distinguisher for Goppa codes. In other words, the generator matrix of a Goppa code looks random.

Theorem 1 *Under the two assumptions stated above, the Hybrid McEliece Encryption Scheme is secure.*

Bird's eye view comparison

- Parameters: m, t, $n \leq 2^m$, $W_{n,t}$ set of n length words of weight t.
- Secret key: The pair (L, g).
  - generator: $g(z) \in \mathbf{F}_{2^m}[z]$, monic irreducible of degree t.
  - support: L a sequence of n distinct elements in $\mathbf{F}_{2^m}[z]$.
  - code: $\Gamma(L, g)$, t error correcting Goppa code.
  - decoder: $\Psi : \{0,1\}^n \to \Gamma(L, g)$, a decoding procedure for Γ.
- Public key:

A random generator matrix	A systematic generator matrix
G of $\Gamma(L, g)$.	$G = (Id\|R)$ of $\Gamma(L, g)$.

Cont...

- Encryption:

$$\begin{array}{ccc} \{0,1\}^k & \longrightarrow & \{0,1\}^n \\ x & \longmapsto & xG+e \end{array} \quad \Big| \quad \begin{array}{ccc} \{0,1\}^k \times \{0,1\}^\ell & \longrightarrow & \{0,1\}^n \\ (x,x') & \longmapsto & (x,xR)+\varphi(x') \end{array}$$

  Where e is random error of length n and weight t.

- Decryption:

$$\begin{array}{ccc} \{0,1\}^n & \longrightarrow & \{0,1\}^k \\ y & \longmapsto & \Psi(y)G^* \end{array} \quad \Big| \quad \begin{array}{ccc} \{0,1\}^n \times \{0,1\}^\ell & \longrightarrow & \{0,1\}^n \\ \{0,1\}^n \times \{0,1\}^\ell & \longmapsto & (x,x') \end{array}$$

  Where $GG^* = Id$ is the identity matrix.

  x is the first k bits of $\Psi(y)$ and $x' = \varphi^{-1}(y - \Psi(y))$.

Implementation

Running time overview

Key generation is one time process, it is fast enough.

Encryption is very fast.

For **decryption**, which is the *bottleneck* of the scheme, there are 3 steps and their time consumption are,

- Syndrome computation $\mathcal{O}(nt)$; 11.3%.
- Key equation solving $\mathcal{O}(t^2)$; 12.0%.
- Error locator polynomial root finding
  - BTA $\mathcal{O}(mt^2)$; 72.1%
  (• Chien search $\mathcal{O}(nt)$; 85.6%)
- Other tasks 4.6%

Binary Goppa Code

Let $m > 0$, $n \leq 2^m$ and $a = (a_1, ..., a_n) \in \mathbb{F}_2^n$.

The n-length binary Goppa code $\Gamma(L, g)$ is defined by:

- Support $L = (\alpha_1, ..., \alpha_n)$ n-tuple of distinct elements of $\mathbb{F}_{2^m}$;
- Goppa polynomial $g(z) \in \mathbb{F}_{2^m}[z]$, square-free, monic of degree $t > 0$ with no root in L.

$\Gamma(L, g)$ is a subfield subcode over $\mathbb{F}_2$ of a particular Goppa code over the binary field $\mathbb{F}_{2^m}$.
We have $a \in \Gamma(L, g)$ if and only if:

$$R_a(z) := \sum_{i=1}^{n} \frac{a_i}{z - \alpha_i} = 0 \text{ over } \mathbb{F}_{2^m}[z]/(g(z)).$$

Decode Binary Goppa Codes

Let e, x, y be n-length binary vectors. We have to find x, the sent codeword knowing $y = x + e$ where y is the received word and e the error word. We can correct up to t errors.

Algebraic decoding is carried out in three steps:

1. Syndrome computation

$$R_y(z) = R_e(z) = \sum_{i=1}^{n} \frac{e_i}{z - \alpha_i} \text{ over } \mathbb{F}_{2^m}[z]/(g(z)).$$

2. Solving the Key Equation to obtain the error locator polynomial

$$R_e(z) \cdot \sigma_e(z) = \sigma'_e(z) \text{ over } \mathbb{F}_{2^m}[z]/(g(z)).$$

3. Error Locator Polynomial Root Finding

$$\sigma_e(z) := \prod_{i=1}^{n} (z - \alpha_i)^{e_i};\ \sigma_e(\alpha_i) = 0 \Leftrightarrow e_i \neq 0.$$

Step 2 : Patterson algorithm

The Patterson algorithm solves the Goppa code key equation: given $R(z)$ and $g(z)$ in $\mathbf{F}_{2^m}[z]$, with $g(z)$ of degree t respectively, find $\sigma(z)$ of degree t such that

$$R(z)\sigma(z) = \frac{d}{dz}\sigma(z) \bmod g(z)$$

We write $\sigma(z) = \sigma_0(z)^2 + z\sigma_1(z)^2$. Since $\frac{d}{dz}\sigma(z) = \sigma_1(z)^2$, we have

$$(1 + zR(z))\sigma_1(z)^2 = R(z)\sigma_0(z)^2 \bmod g(z).$$

Because $g(z)$ is irreducible, $R(z)$ can be inverted modulo $g(z)$. We put $h(z) = z + R(z)^{-1} \bmod g(z)$ and we have

$$h(z)\sigma_1(z)^2 = \sigma_o(z)^2 \bmod g(z).$$

Patterson algorithm cont...

The mapping $f(z) \mapsto f(z)^2 \bmod g(z)$ is bijective and linear over $\mathbf{F}_2^{tm}$, there is a unique polynomial $S(z)$ such that $S(z)^2 = h(z) \bmod g(z)$. We have

$$S(z)\sigma_1(z) = \sigma_0(z) \bmod g(z).$$

The polynomial $\sigma_0(z), \sigma_1(z)$ are the unique solution of the equation

$$\begin{cases} S(z)\sigma_1(z) = \sigma_0(z) \bmod g(z) \\ \deg \sigma_0 \leq t/2 \\ \deg \sigma_1 \leq (t-1)/2 \end{cases} \tag{1}$$

Patterson algorithm cont...

The three steps of the algorithm are the following

1. Compute $h(z) = z + R(z)^{-1} \bmod g(z)$ using the extended Euclidian algorithm.
2. Compute $S(z) = \sqrt{h(z)} \bmod g(z)$

   If $s(z)$ such that $s(z)^2 = z \bmod g(z)$ has been precomputed and $h(z) = h_0 + h_1 z + \ldots + h_{t-1} z^{t-1}$, we have

$$S(z) = \sum_{i=0}^{(t-1)/2} h_{2i}^{2^{m-1}} z^i + \sum_{i=0}^{t/2-1} h_{2i+1}^{2^{m-1}} z^i s(z)$$

3. Compute $(\sigma_0(z), \sigma_1(z))$ using the extended Euclidian algorithm.

The polynomial $\sigma(z) = \sigma_0(z)^2 + z\sigma_1(z)^2$ is returned.

Step 3 : Find the roots efficiently

Several approaches are possible, their efficiency depends on the size of the parameters m and t.

- Chien search computes roots by evaluating artfully the polynomial in all points of L. This method is recommended for hardware implementations and coding theory applications in which m is small.
- BTA is a **recursive** algorithm using trace function properties. It is a faster method for secure parameters in McEliece-type cryptosystems.

Berlekamp Trace Algorithm (BTA)

Trace function $\mathsf{Tr}(\cdot) : \mathbb{F}_{2^m} \to \mathbb{F}_2$

$$\mathsf{Tr}(z) := z + z^2 + z^{2^2} + \ldots + z^{2^{m-1}}.$$

The function $\mathsf{Tr}(\cdot)$ is $\mathbb{F}_2$-linear and onto.

We know that:

$$\forall i \in \mathbb{F}_2 \ , \ \mathsf{Tr}(z) - i = \prod_{\gamma \ s.t. \ \mathsf{Tr}(\gamma)=i} (z - \gamma).$$

Moreover, we have: $z^{2^m} - z = \mathsf{Tr}(z) \cdot (\mathsf{Tr}(z) - 1)$.

Berlekamp Trace Algorithm (BTA) cont...

Let $B = (\beta_1, \ldots, \beta_m)$ a basis of $\mathbb{F}_{2^m}$ over $\mathbb{F}_2$.
BTA splits any $f \in \mathbb{F}_{2^m}[z]$ s.t. $f(z)|(z^{2^m} - z)$ into factors of degree 1,

by computing iteratively on $\beta \in B$ and recursively on f:

$$g(z) := \gcd(f(z), \mathrm{Tr}(\beta \cdot z)) \text{ and } h(z) := \frac{f(z)}{g(z)}.$$

Theorem 2 *(Berlekamp) Berlekamp Trace Algorithm always successfully returns the linear factors of any polynomial.*

Reduce time complexity

Though efficient, BTA has a large number of recursive calls when the system parameters grow.

We reduce it by mixing BTA and Zinoviev's algorithms, ad-hoc methods for finding roots of polynomials of degree ≤ 10 over $\mathbb{F}_{2^m}$.
We call this process BTZ in the following.

BTZ depends on a parameter d_{max} which is the maximum degree up to which we use Zinoviev's methods.

V.A. Zinoviev, On the solution of equations of degree ≤ 10 over finite fields GF(2^m), Research Report INRIA n° 2829, 1996

Pseudocode of a simplified version of BTZ

Algorithm 1 - $BTZ(f, d, i)$

First call: $f \leftarrow \sigma_e$; $d \leftarrow d_{max} \in \{2, \ldots, 10\}$; $i \leftarrow 1$.

if degree$(f) \leq d$ **then**

  return $ZINOVIEV(f, d)$;

else

  $g \leftarrow \gcd(f, \mathrm{Tr}(\beta_i \cdot z))$;

  $h \leftarrow f/g$;

  return $BTZ(g, d, i+1) \cup BTZ(h, d, i+1)$;

end if

Zinoviev's algorithms

Zinoviev's methods find an affine multiple of any polynomial of deg ≤ 10 over $\mathbb{F}_{2^m}$. The methods differ according to this degree.

Affine Polynomial

$A(z) = L(z) + c$ where L is a linearized polynomial, $c \in \mathbb{F}_{2^m}$.

Linearized Polynomial

$$L(z) = \sum_{i=0}^{n} l_i \cdot z^{2^i}$$

with $l_i \in \mathbb{F}_{2^m}$ and $l_n = 1$.

After that, finding roots of affine polynomial is easier than in the general case.

Get an affine multiple of a polynomial of degree 2 or 3

Let us have an equation: $z^2 + \alpha z + \beta = 0, \qquad \alpha,\ \beta \in \mathbb{F}_{2^m}$.
Notice $z^2 + \alpha z$ is already a linearized polynomial. Nothing to do here.

Now consider the equation: $z^3 + az^2 + bz + c = 0, \qquad a,\ b,\ c \in \mathbb{F}_{2^m}$
We have to decimate the non-linear terms.

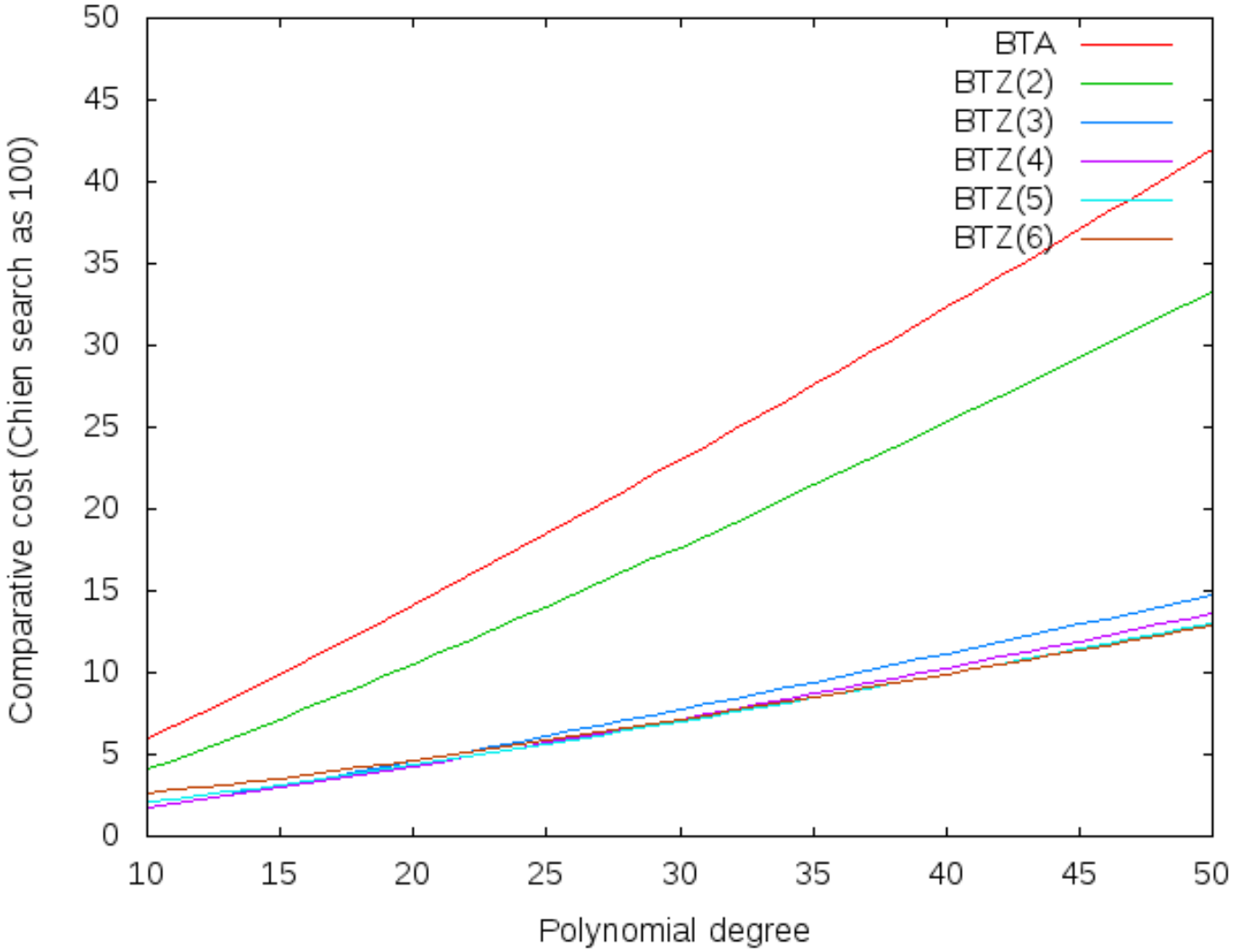
For this, we add one particular root by multiplying the left side by $(z + a)$.
We obtain $z^4 + dz^2 + ez + f = 0$ with $d = a^2 + b,\ e = ab + c,\ f = ac$.

Got what we wanted, an affine multiple of a polynomial of degree 3.

Results we obtained

We specify a recurrence complexity formula for BTZ.
We then use dynamic programming to estimate its theoretical complexity in the worst case.
We thus determine the best d_{max} to use to have the optimal efficiency on the following range of parameters:
$m = 8, 11, 12, 13, 14, 15, 16, 20, 30, 40$; $t = 10..300$; $d_{max} = 2..10$.
Let K be the cost function of any operation over $\mathbb{F}_{2^m}$.
We take $K(+) = 1$; $K(\times) = 1$ or $K(\times) = m$.

Comparison of BTA and BTZ with Chien



Experimental results

We provide experimental data for finding the roots of a polynomial of degree $t = 32$ over $\mathbb{F}_{2^m} = \mathbb{F}_{2048}$. BTZ_d means that we use BTZ with $d_{max} = d$, for all suitable d.

$n = 2048,\quad t = 32,\ m = 11$		$Chien$	BTA	BTZ_2	BTZ_3	BTZ_4
# CPU cycles per byte for	root finding	3200	1300	900	800	800
	decrypting	3700	1800	1400	1300	1300
percentage of time spent for	syn. computation	5	10	13	14	14
	solve key equation	7	14	18	19	19
	root finding	87	72	65	61	60

HyMES simulation results

(m,t)	cycles/byte		key size	security
	encrypt	decrypt		
$(10,50)$	243	7938	32 kB	60
$(11,32)$	178	1848	73 kB	88
$(12,21)$	126	573	118 kB	88
$(13,29)$	149	535	360 kB	129
$(14,15)$	132	229	415 kB	91
$(15,13)$	132	186	775 kB	90
$(16,12)$	132	166	1532 kB	91

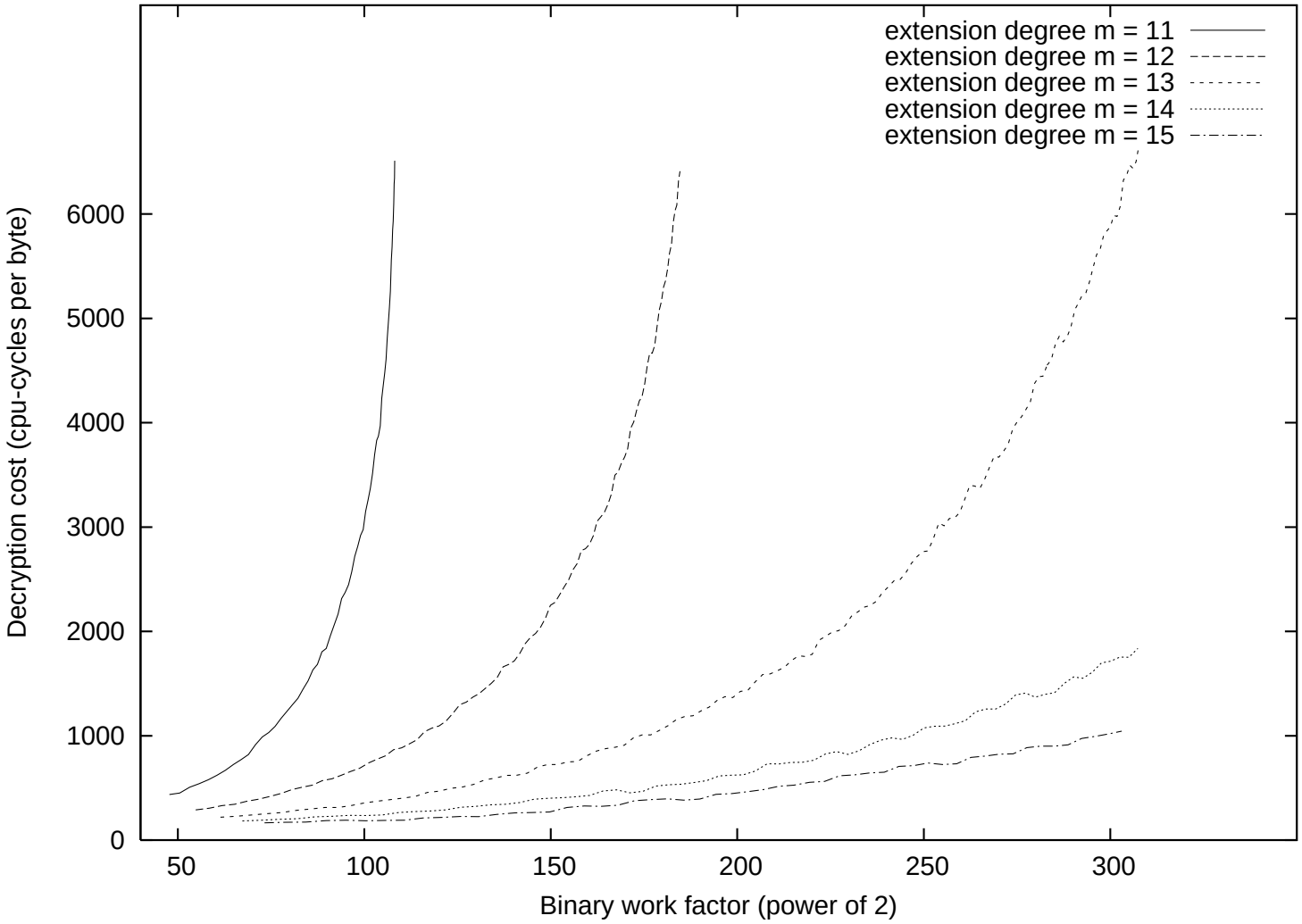
HyMES: selected parameters at a glance (Pentium Core 2)

	cycles/byte	
	encrypt	decrypt
RSA 1024 (1)	800	23100
RSA 2048 (1)	834	55922
NTRU (2)	4753	8445

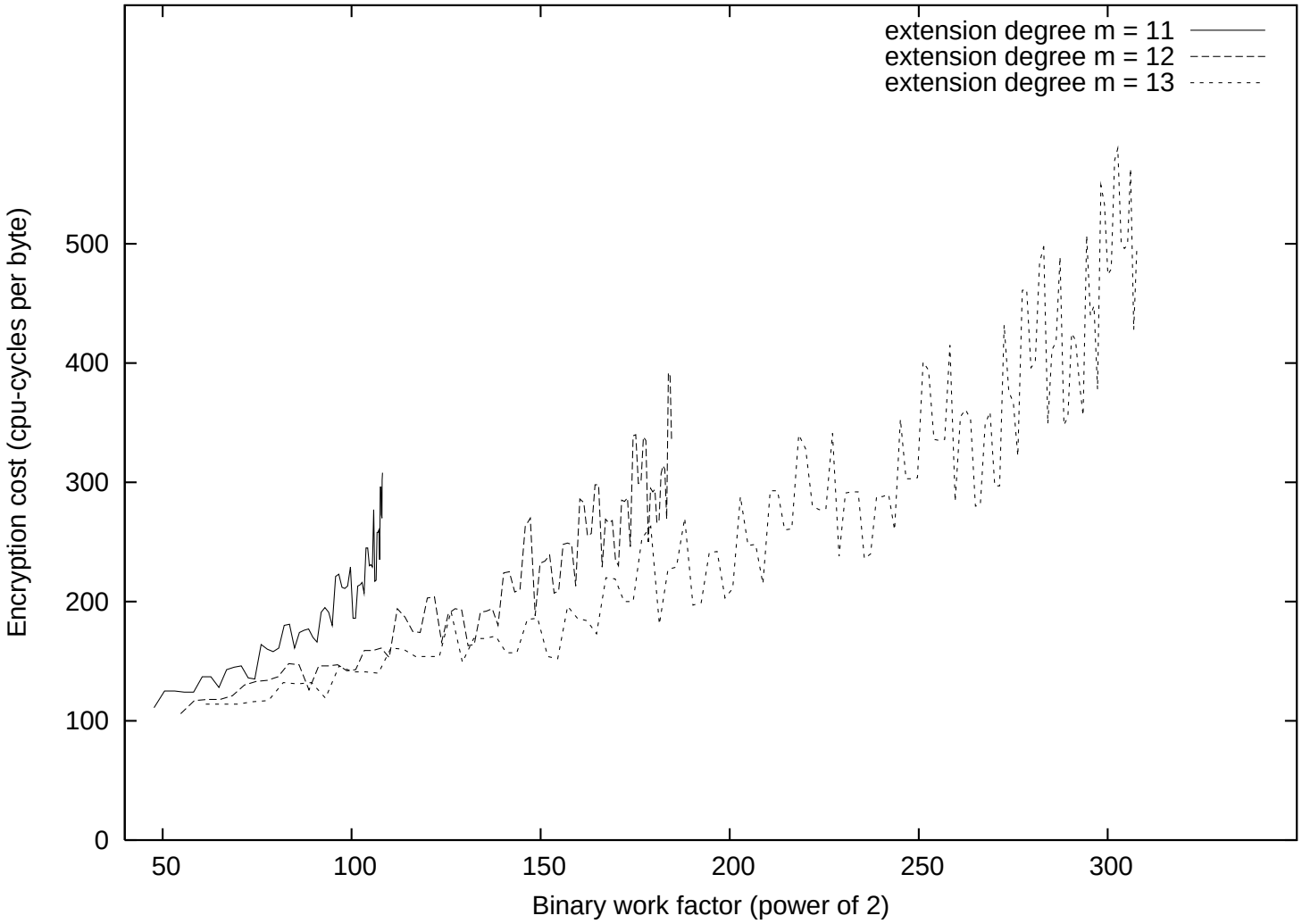
(1) RSA encryption (with malleability defense) using OpenSSL.

(2) `ntru-enc 1 ees787ep1` NTRU encryption with $N = 787$ and $q = 587$. Software written by Mark Etzel (NTRU Cryptosystem).

Decryption results



Encryption results



Conclusion

Summary

- We made 2 major changes to the original system.
- The security is preserved.
- We implemented the Hybrid system.
- The system is faster than other number theory based PKE's.
- We have made algorithmic improvement to root finding of polynomials and thus to decode faster.
- The full project is freely available at
  `http://www-roc.inria.fr/secret/CBCrypto/index.php?pg=hymes`.
- Goppa code distinguisher ?
  (Faugère, Otmani, Perret and Tillich)