Mixed-radix Naccache–Stern encryption

Rémi Géraud and David Naccache

Information security group, Département d'informatique de l'ÉNS,
École normale supérieure, CNRS, PSL Research University, Paris, France.
first_name.family_name@ens.fr

Abstract In this work we explore a combinatorial optimization problem stemming from the Naccache–Stern cryptosystem. We show that solving this problem results in bandwidth improvements, and suggest a polynomial-time approximation algorithm to find an optimal solution. Our work suggests that using optimal radix encoding results in an asymptotic 50% increase in bandwidth.

1 Introduction

The Naccache–Stern (NS) modular knapsack cryptosystem encodes messages $m = \{m_i\} \in \{0,1\}^n$ as $p_1^{m_1} \cdots p_n^{m_n} \bmod p$. For decryption to be possible, one must choose a large enough modulus p, namely $p > p_1 \cdots p_n$.

In this work we consider the possibility to encode m as a mixed-radix representation — which is just another way to dispatch m's bits. This gives encodings of the form $p_1^{\mu_1} \cdots p_\ell^{\mu_\ell}$, where for all i, $0 \leq \mu_i \leq w_i$ are integers. The original NS corresponds to $w_i = 1$.

One interest of such a representation is that some more weight could be put on the smallest primes p_i, which are much smaller than the largest primes. As a result, more bits are available for encoding, using the same p_i (and p) as the original NS. Note that decoding is not much harder than in the original NS case, as it suffices to iterate over the smoothness base at most $\max_i w_i$ times.

Mathematically, we consider the following problem:

$$\begin{cases} \text{Maximize} & \sum_{i=1}^{\ell} \log_2(1+w_i) \\ \text{Subject to} & w_i \in \mathbb{N}, \sum_{i=1}^{\ell} w_i \log_2(p_i) < \log_2(p) \end{cases} \tag{1}$$

Example 1. NS with $\ell = 15$ and 64-bit modulus p can encode 15-bit messages. Using

$$\mathbf{w} = \{7,4,2,2,1,1,1,1,1,1,1,1,1,0,0\}$$

we can encode 17-bit messages. Notice that the last two primes p_{14}, p_{15} are not used. This is a 13% increase in bandwidth, but it is arguably a small gain.

We now face several interesting questions: Does an optimal $\mathbf{w}$ always exist? How to find it? Is the gain marginal, or does it provide an (asymptotic or practical) advantage?

2 The original Naccache-–Stern cryptosystem

The NS cryptosystem [13] uses the following sub-algorithms:

- Setup: Pick a large prime p and a positive integer n. Let $\mathfrak{P} = \{p_0 = 2, \ldots, p_{n-1}\}$ be the set of the n first primes, so that $\prod_{i=0}^{n-1} p_i < p$. We leave aside a one-bit leakage dealt with in [13]).
- KeyGen: Pick a secret integer $s < p-1$, such that $\gcd(p-1, s) = 1$. Set $v_i = \sqrt[s]{p_i} \bmod p$. The private key is s. The public key is $(p, n, v_0, \ldots, v_{n-1})$.

– Encrypt: To encrypt an n-bit message m, compute the ciphertext c:

$$c \leftarrow \prod_{i=0}^{n-1} v_i^{m_i} \bmod p$$

where m_i is the i-th bit of m.

– Decrypt: To decrypt c, compute

$$m \leftarrow \sum_{i=0}^{n-1} 2^i \mu_i(c, s, p)$$

where $\mu_i(c, s, p) \in \{0, 1\}$ is the function defined by:

$$\mu_i(c, s, p) = \frac{\gcd(p_i, c^s \bmod p) - 1}{p_i - 1}.$$

The security of this scheme relies on the conjectured hardness of a multiplicative variant of the knapsack problem[1]:

Definition 1 (Multiplicative Knapsack Problem). *Given p, c, and a set $\{v_i\}$, find a binary vector x such that*

$$c = \prod_{i=0}^{n-1} v_i^{x_i} \bmod p.$$

Just as in additive knapsacks, this problem is NP-hard in general but can be solved efficiently in some situations; the secret key enabling precisely to transform the ciphertext into an easily-solvable instance.

Unlike additive knapsacks, this multiplicative knapsack doesn't lend itself to lattice reduction attacks, which completely break many additive knapsack-based cryptosystems [1,4,11,6,12,10].

Over the past years, several NS variants were published, these notably seek to either increase efficiency [7], or extend NS to polynomial rings [10], or to achieve semantic security [5]; to the best of our knowledge, no efficient attacks against the original NS are known.

3 A daunting optimization problem

Equation 1 can be reformulated, and simplified somewhat. First we may rewrite the objective as a polynomial in w_i; then we may further impose $w_{i+1} \leq w_i$ for all $1 \leq i \leq \ell$. The resulting problem is not strictly equivalent as eq. 1, but a solution for this modified problem is also an optimal solution for the original problem.

$$\begin{cases} \text{Maximize} & f(\mathbf{w}) = \prod_{i=1}^{\ell}(1 + w_i) \\ \text{Subject to} & w_i \in \mathbb{N}, \\ & w_{i+1} \leq w_i, \\ & \sum_{i=1}^{\ell} w_i \log_2(p_i) < \log_2(p). \end{cases} \tag{2}$$

The linear constraints delimit a simplex, however the objective function is not itself linear, which prevents us from directly leveraging efficient linear programming techniques. Equation 2 belongs to a class of *polynomial* programming problems. Unfortunately,

Theorem 1 ([9,8]). *The problem of minimizing a degree-4 polynomial over the lattice points of a convex polygon is* NP*-hard.*

[1] This can also be described as a modular variant of the 'subset product' problem.

3.1 Brute-force exploration

Despite the problem's complexity, we may hope to exhaust all solutions for small enough instances. This can be achieved by backtracking, where we explore the combinatorial graph "from the end", i.e. with values of w_ℓ as roots, $w_{\ell-1}$ as subroots, etc.

One advantage of this approach is its simplicity; however it quickly runs out of steam as larger instances are considered. The solution of Example 1 was first computed this way.

It is interesting to compute the number of configurations that can be explored, i.e. the number of points in $P \cap \mathbb{Z}^n$, where P is the polytope defined by the constraints of Eq. 2. In dimension 2 one can use Pick's theorem, however there are no simple generalisations to higher dimensions of such a result. Rather, we may estimate the number of integer points as the volume of P:

$$N = |P \cap \mathbb{Z}^n| \approx \mathrm{vol}(P) = \frac{1}{n!} \prod_{i=1}^{n} \log_{p_i} p.$$

Assuming that all the p_i are of similar size, this is approximately $(\log_{p_n} p)^n / n!$; by the prime number theorem we have $\log_{p_n} p = \ln p / \ln(p_n) \approx \ln p / \ln(n \ln n)$, so that

$$N \approx \frac{(\ln p)^n}{n!(n \ln n)^n}$$

Using $p \geq \#75 \approx 2^{512}$, we find $N \approx 2^{82}$, which is nearly beyond reach — and in any case impractical.

3.2 Fully polynomial-time approximation scheme

In the face of Theorem 1, and of the comments above, we will alter the problem: First we replace the logarithms by their rational approximation, the precision of which will be discussed later; Second we look for an approximation to the optimum, rather than the optimum itself.

Approximation problem. Having an approximate value for the maximum value $f^* = f(\widehat{\mathbf{w}})$ taken by f on the optimal solution $\widehat{\mathbf{w}}$ provides us with a strategy:

Lemma 1. *Let P be an n-dimensional rational polytope. Assume that there exists a polynomial-time algorithm that computes f^* over $P \cap \mathbb{Z}^n$. Then there is a polynomial-time algorithm that finds a feasible solution $\overline{\mathbf{w}}$ such that $f^* - f(\overline{\mathbf{w}}) \leq O(\epsilon)$.*

Proof. We proceed by iterative bisection of P.

Remark 1. The expression "polynomial-time" refers here to an algorithm whose complexity is bounded above by a polynomial function of the encoding of f and P.

Therefore we first focus on finding efficiently an approximation for f^*, from which an immediate algorithm follows (by Lemma 1), that provides an approximation to $\widehat{\mathbf{w}}$.

Recall the classical fact that, for $\{x_1, \ldots, x_n\}$ a collection of non-negative real numbers,

$$\max_i x_i = \|\mathbf{x}\|_\infty = \lim_{k \to \infty} \|\mathbf{x}\|_k$$

where $\|\cdot\|_k$ is the ℓ_k-norm, from which we get

$$n^{-\frac{1}{k}} \|\mathbf{x}\|_k \leq \|\mathbf{x}\|_\infty \leq \|\mathbf{x}\|_k.$$

Now, denote $x_i = f(\mathbf{w})$ for all w in the rational polytope P, and $\mathbf{x} = (x_1, \ldots, x_N)$, where $N = |P \cap \mathbb{Z}^n|$. Then for all $k > 0$,

$$N^{-\frac{1}{k}} \|\mathbf{x}\|_k \leq \|\mathbf{x}\|_\infty \leq \|\mathbf{x}\|_k.$$

Phrased equivalently, the optimal solution, $\widehat{\mathbf{w}} = \|\mathbf{x}\|_\infty$, can be approximated by *summing* a polynomial in the points of $P \cap \mathbb{Z}^n$. Namely:

Lemma 2. *Let $\epsilon > 0$, then for $k = \lceil (1 + 1/\epsilon) \log \ell \rceil$,*

$$\|\mathbf{x}\|_k \left(1 - \ell^{-\frac{1}{k}}\right) \leq \epsilon f\left(\widehat{\mathbf{w}}\right)$$

Remark 2. It is important to note at this point that we can expand the polynomial function f^k as a list of monomials in polynomial time.

The only remaining question is whether we can compute $\|x\|_k$ in polynomial time.

Polynomial-time computation over $P \cap \mathbb{Z}^n$. One difficulty is that P contains a priori an exponential number of integer points. Here is how to circumvent this apparent problem: Consider the *generating formal (Laurent) series*:

$$g(P; \mathbf{z}) = \sum_{\mathbf{w} \in P \cap \mathbb{Z}^n} \mathbf{z}^{\mathbf{w}}. \tag{3}$$

Since P is bounded, the sum is in fact finite, and $g(P; z)$ is a formal (Laurent) polynomial for which we may expect a short rational representation.

Example 2. Consider P the interval $[0, k]$, so that $P \cap \mathbb{Z} = \{0, \ldots, k\}$. We have

$$\begin{aligned} g(P; z) &= \sum_{w \in P \cap \mathbb{Z}} z^w = \sum_{j=0}^{k} z^j = z^0 + z^1 + \cdots + z^k \\ &= \frac{1 - z^{k+1}}{1 - z}. \end{aligned}$$

The second line of the above equation gives a compact representation of $g(P; z)$, which is *linear* in n. In particular, one may count the points in P — i.e. compute the ℓ_1-norm over the integer points of P — by carefully evaluating $g(P; z)$ at $z = 1$. *Carefully* refers to the $1 - z$ denominator, which requires us to use either residue methods, the Bernoulli-l'Hôpital rule, or a numerical approximation of the limit $\|\mathbf{w}\|_1 = \lim_{z \to 1} g(P; z)$.

Theorem 2 **([2,3]).** *There exists a polynomial-time algorithm for computing the rational generating function of a polyhedron $P \subseteq \mathbb{R}^n$ given by rational inequalities.*

Theorem 2 is constructive and provides an explicit algorithm, that we reproduce below. But first, observe that the knowledge of $g(P; \mathbf{z})$ is enough to compute the function

$$g(P, h; \mathbf{z}) = \sum_{\mathbf{w} \in P \cap \mathbb{Z}^n} h(\mathbf{w}) \mathbf{z}^{\mathbf{w}}.$$

Example 3. Consider the same setting as in Example 2, and let D be the differential operator $D = \left(z \frac{\mathrm{d}}{\mathrm{d}z}\right)^2$. Then

$$\begin{aligned} Dg(P; z) &= D \frac{1 - z^{k+1}}{1 - z} \\ &= \left(z \frac{\mathrm{d}}{\mathrm{d}z}\right) \left(z \frac{\mathrm{d}}{\mathrm{d}z} \frac{1 - z^{k+1}}{1 - z}\right) \\ &= \left(z \frac{\mathrm{d}}{\mathrm{d}z}\right) \left(\frac{z(1 - (1 + k) z^k + k z^{1+k})}{(1 - z)^2}\right) \\ &= \frac{z((k(z - 1)(k(z - 1) - 2) + z + 1) z^k - z - 1)}{(z - 1)^3} \\ &= z^1 + 4z^2 + 9z^3 + \cdots + k^2 z^k \\ &= g(P, h; z) \end{aligned}$$

where $h(z) = z^2$.

Lemma 3. *Let $h \in \mathbb{Z}[w_1, \ldots, w_n]$ be a polynomial, then there is a differential operator D_h such that $D_h g(P; \mathbf{z}) = g(P, h; \mathbf{z})$.*

Proof. This is a straightforward extension of Example 3, using

$$D_h = h\left(w_1 \frac{\partial}{\partial w_1}, \ldots, w_n \frac{\partial}{\partial w_n}\right).$$

Combining this with Lemma 2, we see that the knowledge of $g(P, f^k; \mathbf{z})$ gives the desired approximate solution. Hence everything hinges on Theorem 2 being polynomial-time.

Definition 2. *An algorithm $\mathcal{A}$ is an* ϵ-approximation algorithm *for a constrained optimization problem with optimal cost f^* if, for each instance of the problem of encoding length N, $\mathcal{A}$ runs in $\mathsf{poly}(n)$ and returns a feasible solution with cost $f_{\mathcal{A}}$, such that $f_{\mathcal{A}} \geq (1 - \epsilon) f^*$.*

Definition 3. *A family $\{\mathcal{A}_\epsilon\}_\epsilon$ of ϵ-approximation algorithms is a* fully polynomial-time approximation scheme (FPTAS) *if the running time of $\mathcal{A}_\epsilon$ is $\mathsf{poly}(n, 1/\epsilon)$.*

Theorem 3. *Let f be a polynomial function over the integer points of a rational polytope P, and a rational number $\epsilon > 0$, where f is given as a list of monomials with rational coefficients and integer exponents, then there exists a FPTAS for the maximisation problem for all polynomial functions f that are non-negative on the feasible region, i.e. an polynomial-time algorithm that computes a feasible solution $\mathbf{x}_\epsilon$ such that*

$$|f(\mathbf{x}_\epsilon) - f^*| \leq \epsilon f^*$$

Proof. The algorithm is as follows: On input P, f, $\epsilon > 0$,

1. Compute $k = (1 + 1/\epsilon) \log(|P \cap \mathbb{Z}^n|)$ (as in Lemma 2)
2. Compute f^k as a list of monomials
3. Use Lemma 3 to compute D_{f^k}
4. Use Barvinok's algorithm (Algorithm 1) to get the function $g(P; \mathbf{z})$
5. Apply D_{f^k} to $g(P; \mathbf{z})$ to get $g(P, f^k; \mathbf{z})$
6. Using residue techniques, compute

$$LB_k = \left\lceil \left(\frac{g(P, f^k; \mathbf{1})}{g(P; \mathbf{1})}\right)^{\frac{1}{k}} \right\rceil$$
$$UB_k = \left\lfloor \left(g\left(P, f^k; \mathbf{1}\right)\right)^{\frac{1}{k}} \right\rfloor$$

   These bounds satisfy

$$UB_k - LB_k \leq f^* \left((|P \cap \mathbb{Z}^n|)^{\frac{1}{k}} - 1\right)$$

7. Iteratively bisecting P (as in Lemma 1) we get a feasible solution that is $(1 - \epsilon)$-optimal.

Every step is easily checked to run in polynomial time.

```
Algorithm 1: Barvinok's Algorithm [2]
Input: Polyhedron $P$.
Output: Short generating function $g(P;\mathbf{z})$.

1. Compute all vertices $\mathbf{v}_i$, and corresponding supporting cones $C_i$, of $P$
2. Triangulate $C_i$ into simplicial cones $C_{i,j}$, keeping track of all the intersecting proper faces
3. Apply signed decomposition to the cones $\mathbf{v}_i + C_{i,j}$, to obtain unimodular cones $\mathbf{v}_i + C_{i,j,l}$ (again keeping
   track of all the intersecting proper faces)
4. Compute the unique integer point $\mathbf{a}_i$ in the fundamental parallelepiped of every resulting cone $\mathbf{v}_i + C_{i,j,l}$.
5. For each unimodular cone, the generating function is
       $z^{\mathbf{a}} / \prod_{j=1}^{n} (1 - \mathbf{z}^{\mathbf{b}_j}),$
   where $\mathbf{a}$ is the integer point and $\mathbf{b}_j$ are the cone's spanning vectors.
6. Compute the signed summation of all the cones, resulting in $g(P;\mathbf{z})$.
```

4 Mixed-radix linear-bandwidth Naccache–Stern encryption

Using [7], messages can be encoded by "packing" primes together. The resulting construction results in asymptotically linear bandwidth encryption.

Table 1. Comparison of the original and pack-adjusting mixed-radix linear-bandwidth Chevallier-Mames–Naccache–Stern encryption, for $\gamma = 4, \ell = 1$. Both outperform the original Naccache–Stern bandwidth at a given security level.

n	Minimal p	CMNS	MR-CMNS	Radix
1	11	2 bits	2 bits	4
2	137	4 bits	4 bits	3, 6
3	4931	6 bits	7 bits	2, 7, 11
4	260849	8 bits	10 bits	2, 4, 9, 17
5	18517753	10 bits	13 bits	2, 2, 8, 15, 23
6	1648077367	12 bits	16 bits	2, 2, 4, 12, 19, 30
7	176344276177	14 bits	19 bits	2, 2, 3, 7, 14, 23, 37
8	23101100172959	16 bits	23 bits	2, 2, 2, 5, 10, 19, 28, 44
9	3488266126107761	18 bits	26 bits	2, 2, 2, 3, 9, 12, 22, 32, 51

– $\mathsf{Setup}(1^\kappa) \to \mathsf{pp}$. Let $\ell \geq 1$ and γ be two integers. We construct n "packs" containing γ small primes each, and pick a prime p such that

$$\prod_{i=1}^{n} p_{\gamma i}^{\ell} < p,$$

where p_i is the i-th prime number ($p_1 = 2$). Denoting

$$b = \binom{\gamma + \ell}{\ell},$$

we introduce an invertible function unrank that maps an integer $0 \leq m < b$ to a γ-tuple $(d_1, \ldots, d_\gamma)$ such that $0 \leq d_k$ and $d_1 + \cdots + d_k \leq \ell$. Set $\mathsf{pp} \leftarrow (p, n, \gamma, \ell)$.

– $\mathsf{KeyGen}(\mathsf{pp}) \to (\mathsf{sk}, \mathsf{pk})$. Choose a random integer $s \nmid (p-1)$, and let $v_i \leftarrow p_i^{1/s} \bmod p$ for $i = 1$ to γn. Set $\mathsf{sk} \leftarrow q$, $\mathsf{pk} \leftarrow \{v_i\}$.

– $\mathsf{Encrypt}(\mathsf{pp}, \mathsf{pk}, m) \to c$. Write m in base b as $m_0, \dots, m_{n-1}$, then $\{d_{i,j}\} \leftarrow \mathrm{unrank}(m_i)$. Finally,

$$c \leftarrow \prod_{i=0}^{n-1} \prod_{j=1}^{\gamma} v_{i\gamma+j}^{d_{i,j}} \bmod p.$$

– $\mathsf{Decrypt}(\mathsf{pp}, \mathsf{sk}, c) \to m$. Compute $c^s \bmod p$ in $\mathbb{N}$, factor over the smoothness base $\{p_i\}$ and recover each m_i as $m_i \leftarrow \mathrm{rank}(\{d_{i,j}\})$.

The main appeal of this approach is that p can be made much smaller compared to the original cryptosystem. We can use the techniques described above to optimize each "pack". Let's illustrate this on an example.

Example 4. Let $n = 3$, $\gamma = 4$, $\ell = 1$, hence we will use the primes p_1 to p_{12}, for which an admissible value of p is 4931.[2] Let $s = 3079$. The public key consists in the $\{v_i\}$:

$$\text{pack 1}\begin{cases} v_1 = \sqrt[s]{2} \bmod p = 1370 \\ v_2 = \sqrt[s]{3} \bmod p = 1204 \\ v_3 = \sqrt[s]{5} \bmod p = 1455 \\ v_4 = \sqrt[s]{7} \bmod p = 3234 \end{cases}$$
$$\text{pack 2}\begin{cases} v_5 = \sqrt[s]{11} \bmod p = 2544 \\ v_6 = \sqrt[s]{13} \bmod p = 3366 \\ v_7 = \sqrt[s]{17} \bmod p = 1994 \\ v_8 = \sqrt[s]{19} \bmod p = 3327 \end{cases}$$
$$\text{pack 3}\begin{cases} v_9 = \sqrt[s]{23} \bmod p = 4376 \\ v_{10} = \sqrt[s]{29} \bmod p = 1921 \\ v_{11} = \sqrt[s]{31} \bmod p = 3537 \\ v_{12} = \sqrt[s]{37} \bmod p = 3747 \end{cases}$$

To encrypt a message, it is written in base $\gamma = 4$ and the appropriate v_i's are multiplied; thus if $m = 35 = 203_4$,

$$c = v_4 \cdot v_5 \cdot v_{11} \bmod p = 4484.$$

This cryptosystems allows the encoding of 6-bit messages.

There are two ways to introduce the mixed-radix approach: by choosing different pack sizes, for instance making the first pack larger, or by using varying powers of primes. The two approaches are related, as it might be more efficient to use e.g. a higher power of a small prime, rather than introducing a new large prime.

To illustrate the effect of pack-adjusting, assume that the packs have dimension $\gamma_1, \gamma_2, \gamma_3$; this allows the encoding of messages of $\log_2 \gamma_1\gamma_2\gamma_3$ bits, under the condition that $p_{\gamma_1} p_{\gamma_2+\gamma_1} p_{\gamma_1+\gamma_2+\gamma_3} < p$. The choice $\gamma_1 = 2, \gamma_2 = 4, \gamma_3 = 24$ satisfies the constraints and allows for encoding 192 different messages, i.e. a little more than 7-bit messages. Alternatively, using $\gamma_1 = \gamma_2 = 2, \gamma_3 = 16$, we still encode 6-bit messages, but we can safely bring p down to 1493.

The optimization problem corresponding to pack-adjusting can readily be addressed using the same tools as in Section 3, and we give the results for small values of n in Table 1. As is visible already in this table, the gain of pack-adjusting increases, achieving a 50% bandwidth improvement for larger values of n.

[2] In the original NS setting, p would be at least 7420738134871.

Considering parameters of cryptographic size, $n = 58$ gives a minimal CMNS modulus of 513 bits:

$$\begin{aligned} p_{\min} = {} & \texttt{0x172027ebd81c90acf8b8f7276279f5e5} \\ & \texttt{c5dfa4bdb9bd2391937db041ec981aa7} \\ & \texttt{cb82dc468791ad498fb808f111f74d9b} \\ & \texttt{8e53e96e6b4f37b06ae81bff4b04b0487}. \end{aligned}$$

Using NS the bandwidth is 75 bits.[3] Using CMNS the bandwidth reaches 116 bits. Using our mixed-radix encoding with CMNS (MR-CMNS) we achieve a bandwidth of 174 bits, or about 49% improvement over plain CMNS.

5 Conclusion

In this paper we set out to use a more flexible encoding of messages for the Naccache–Stern public-key encryption scheme, namely a mixed-radix representation. This gave rise to an optimization problem for which a polynomial-time approximation scheme exists, which relies on geometric computations. This in turn provides efficient weights for the encoding — which only need to be computed once for a given modulus. The resulting scheme extends on, and outperforms, earlier work on linear-bandwidth Naccache–Stern encryption, with an overall bandwidth improvement of 50%.

References

1. Adleman, L.M.: On breaking the iterated Merkle-Hellman public-key cryptosystem. In: D. Chaum, R.L. Rivest, A.T. Sherman (eds.) Advances in Cryptology – CRYPTO'82, pp. 303–308. Plenum Press, New York, USA, Santa Barbara, CA, USA (1982)
2. Barvinok, A.I.: A polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed. Mathematics of Operations Research **19**(4), 769–779 (1994)
3. Barvinok, A.I., Pommersheim, J.E.: An algorithmic theory of lattice points. New perspectives in algebraic combinatorics **38**, 91 (1999)
4. Brickell, E.F.: Breaking iterated Knapsacks. In: G.R. Blakley, D. Chaum (eds.) Advances in Cryptology – CRYPTO'84, *Lecture Notes in Computer Science*, vol. 196, pp. 342–358. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (1984)
5. Brier, É., Géraud, R., Naccache, D.: Exploring naccache-stern knapsack encryption. In: P. Farshim, E. Simion (eds.) Innovative Security Solutions for Information Technology and Communications - 10th International Conference, SecITC 2017, Bucharest, Romania, June 8-9, 2017, Revised Selected Papers, *Lecture Notes in Computer Science*, vol. 10543, pp. 67–82. Springer (2017). DOI 10.1007/978-3-319-69284-5_6. URL https://doi.org/10.1007/978-3-319-69284-5_6
6. Chee, Y.M., Joux, A., Stern, J.: The cryptoanalysis of a new public-key cryptosystem based on modular Knapsacks. In: J. Feigenbaum (ed.) Advances in Cryptology – CRYPTO'91, *Lecture Notes in Computer Science*, vol. 576, pp. 204–212. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (1992)
7. Chevallier-Mames, B., Naccache, D., Stern, J.: Linear bandwidth Naccache-Stern encryption. In: R. Ostrovsky, R.D. Prisco, I. Visconti (eds.) SCN 08: 6th International Conference on Security in Communication Networks, *Lecture Notes in Computer Science*, vol. 5229, pp. 327–339. Springer, Heidelberg, Germany, Amalfi, Italy (2008)
8. De Loera, J.A., Hemmecke, R., Köppe, M., Weismantel, R.: Integer polynomial optimization in fixed dimension. Mathematics of Operations Research **31**(1), 147–153 (2006)
9. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman (1979)
10. Herold, G., Meurer, A.: New attacks for knapsack based cryptosystems. In: I. Visconti, R.D. Prisco (eds.) SCN 12: 8th International Conference on Security in Communication Networks, *Lecture Notes in Computer Science*, vol. 7485, pp. 326–342. Springer, Heidelberg, Germany, Amalfi, Italy (2012)
11. Joux, A., Stern, J.: Cryptanalysis of another Knapsack cryptosystem. In: H. Imai, R.L. Rivest, T. Matsumoto (eds.) Advances in Cryptology – ASIACRYPT'91, *Lecture Notes in Computer Science*, vol. 739, pp. 470–476. Springer, Heidelberg, Germany, Fujiyoshida, Japan (1993)

[3] Indeed, $p_{\min}$ is closest to the 75-th primorial.

12. Lenstra Jr., H.W.: On the Chor-Rivest knapsack cryptosystem. Journal of Cryptology **3**(3), 149–155 (1991)
13. Naccache, D., Stern, J.: A new public-key cryptosystem. In: W. Fumy (ed.) Advances in Cryptology – EUROCRYPT'97, *Lecture Notes in Computer Science*, vol. 1233, pp. 27–36. Springer, Heidelberg, Germany, Konstanz, Germany (1997)