Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE

Bc. Martin Hlaváč

Útoky pomocí postranních kanálů

Katedra Algebry

Vedoucí diplomové práce: **Ing. Tomáš Rosa, Ph.D.**

Studijní program: **Matematika**

Studijní obor: **Matematické metody informační bezpečnosti**

Prohlašuji, že jsem diplomovou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce.

V Praze dne 15. dubna 2006 Martin Hlaváč

Contents

Název práce: Útoky pomocí postranních kanálů

Autor: Bc. Martin Hlaváč

Katedra (ústav): Katedra Algebry

Vedoucí diplomové práce: Ing. Tomáš Rosa, Ph.D.

e-mai vedoucího: `trosa@ebanka.cz`

Abstrakt: Práce pojednává o rozšíření původního problému skrytého čísla (PSČ) zavedeného v roce 1996 Bonehem a Venkatesanem. PSČ řeší situaci, kdy jsou známy aproximace nějakých násobků skrytého čísla modulo N, přičemž neuvažuje případnou znalost o něm samotném. V práci je navržena metoda, která PSČ podstatně zobecňuje - zohledňuje situace, kdy je známa informace jak o násobcích, tak i o skrytém čísle samotném. Obě informace mohou být rozděleny do několika dílčích částí, což původní PSČ neuvažoval. Uvedeno je praktické využití pro odhalení privátního klíče DSA v případě znalosti postranní informace o 5 podpisových operacích. Ta může být k dispozici například v případech, kdy je podpis generován v nezabezpečeném prostředí - na platformě Pentium 4 s technologií Hyper-Threading.

Klíčová slova: rozšířený problém skrytého čísla, mřížka, kryptoanalýza, DSA, sliding window, hyper-threading

Title: Side channel attacks

Author: Bc. Martin Hlaváč

Department: Department of Algebra

Supervisor: Ing. Tomáš Rosa, Ph.D.

Supervisor's e-mail address: `trosa@ebanka.cz`

Abstract: The work extends the Hidden Number Problem (HNP) introduced by Boneh and Venkatesan in 1996. HNP is to find an unknown integer if several approximations of its multiples modulo N are known. New method for solving an extension of HNP (EHNP) is elaborated, taking into account the fragmentation of the information on the multiples and on the hidden number itself, as well. A real scenario application of the approach is presented - the private DSA key is extracted with the knowledge of side information on 5 signing operations. Such an information can be obtained if the signatures are generated in the unsecured environment of a Pentium 4 processor with Hyper-Threading technology.

Keywords: extended hidden number problem, lattice, cryptanalysis, DSA, sliding window, hyper-threading

Acknowledgment

The author would like to thank Tomáš Rosa for introducing him to the fascinating area of side channel cryptanalysis and to the basics of the geometry of numbers and the lattice theory. Numerous helpful discussions are gratefully appreciated. Thanks also goes to Colin Percival for providing the source code for his experiments, to Jozef Juríček for clarifying several topics in the probability theory, to Oldřich Ulrych from Mathematical Institute of Charles University for providing several weeks of computing power needed for the experiments conducted and to eBanka, a.s. for lending an experimental hardware for verifying real side channel existence on the Pentium 4 HTT platform.

1 Preface

Motivated by the recently discovered side channel hidden in the design of modern processors with Hyper-Threading technology, we propose a general method that solves the abstract problem in the geometry of numbers arising from Digital Signature Algorithm, the Extended Hidden Number Problem, with an acceptable probability of success.

We extend the original Hidden Number Problem (HNP) introduced by Boneh and Venkatesan in 1996 (see [5]). Let N be a prime and let $\alpha_i, \rho_i, \beta_i \in \mathbb{Z}_N$, $1 \le i \le d$ and $\mu \in \mathbb{Q}$ be known values satisfying $\alpha_i x + \rho_i k_i \equiv \beta_i (\,\mathbf{mod}\, N)$ and $k_i \le 2^\mu$ for unknown values of "hidden number" x and nonces k_i. HNP is to find x.

The extension we propose replaces the unknown values x and k_i with the decompositions $\bar{x} + \sum_{j=1}^{d} 2^{\pi_j} x_j$ and $\bar{k}_i + \sum_{j=1}^{l_i} 2^{\lambda_{i,j}} k_{i,j}$, respectively, where the only unknowns x_j, $1 \le j \le m$ and $k_{i,j}$, $1 \le j \le l_i$, $1 \le i \le d$ are bounded from above with known constants (equivalent to $k_i \le 2^\mu$ in HNP). In other words, such an extension allows us to make use of eventual information on individual bits in the original unknown values x and k_i. We call the extension Extended Hidden Number Problem (EHNP), propose a probabilistic polynomial time algorithm to solve its instances and examine the correctness and probability of success of the algorithm.

The work is organized as follows. Besides several technical lemmas, we define a full-rank lattice in $\mathbb{Q}^n$ in Section 2 and demonstrate several of its properties. We introduce the main algorithmic problems in the lattice theory and some of their solutions, i.e. LLL lattice basis reduction algorithm and Babai's Closest Plane algorithm in the formulations that are widely accepted in cryptographic community (e.g. [10], [5], [14], etc.).

The Extended Hidden Number Problem is defined in Section 3. We mention two existing approaches solving EHNP or its special cases, point out their influence on the newly proposed algorithm, and finally investigate its correctness.

An application to Digital Signature Algorithm is analyzed in Sections 4 and 5 assuming the signatures are generated in the unsecured environment of Pentium 4 processor with Hyper-Threading technology using Sliding Window exponentiation. We propose two algorithms to convert the side channel information to the form suitable for EHNP. We conclude with an extensive series of experiments supporting the usability of the new approach in the real-life situations.

2 Preliminaries

2.1 Symbol $|\cdot|_N$

Definition 2.1. For $a \in \mathbb{Z}$, $N \in \mathbb{N}$ we define

$$|a|_N = \min_{z\in\mathbb{Z}} |a - zN|.$$

Lemma 2.2. *Let $a, b \in \mathbb{Z}$, $N \in \mathbb{N}$. Then*

(i) $|a+b|_N \leq |a|_N + |b|_N$

(ii) $|ab|_N \leq |a|_N |b|_N$

Proof. Let $m_a, m_b \in \mathbb{Z}$ be such that $|a - m_aN| = |a|_N$ and $|b - m_bN| = |b|_N$. Then

(i) $|a|_N + |b|_N = |a - m_aN| + |b - m_bN| \geq |a + b - (m_a + m_b)N| \geq |a + b|_N$

(ii) $|a|_N |b|_N = |a - m_aN||b - m_bN| = |ab - (am_b + bm_a - m_am_b)N| \geq |ab|_N$

□

Corollary 2.3. *Let $d \in \mathbb{N}$, $a_1, \dots, a_d \in \mathbb{Z}$, $N \in \mathbb{N}$. Then*

(i) $\left|\sum_{j=1}^d a_j\right|_N \leq \sum_{j=1}^d |a_j|_N$

(ii) $\left|\prod_{j=1}^d a_j\right|_N \leq \prod_{j=1}^d |a_j|_N$.

Lemma 2.4. *Let $a, b \in \mathbb{Z}$, $N \in \mathbb{N}$. Then*

$$|ab|_N = \big|a|b|_N\big|_N$$

Proof. Let $m_b \in \mathbb{Z}$ be such that $|b - m_bN| = |b|_N$. Then

$$\big|a|b|_N\big|_N = \big|a|b - m_bN|\big|_N = |ab - (am_b)N|_N = |ab|_N.$$

□

Corollary 2.5. *Let $a, b \in \mathbb{Z}$, $N \in \mathbb{N}$. Then*

$$|ab|_N = \big||a|_N|b|_N\big|_N$$

Proof. It suffices to apply Lemma 2.4 twice, i.e.

$$|ab|_N = \big|a|b|_N\big|_N = \big||a|_N\big||b|_N\big|_N\big|_N = \big||a|_N|b|_N\big|_N.$$

□

Corollary 2.6. *Let $d \in \mathbb{N}$, $a_1, \ldots, a_d \in \mathbb{Z}$, $N \in \mathbb{N}$. Then*

$$\left| \prod_{j=1}^{d} a_j \right|_N = \left| \prod_{j=1}^{d} |a_j|_N \right|_N .$$

Lemma 2.7. *Let $N \in \mathbb{N}$ and $a, b \in \mathbb{Z}$. Then statements*

(i) $|a + b|_N = 0$

(ii) $a \equiv -b \; (\mathbf{mod}\, N)$

are equivalent.

Proof. The existence of $k \in \mathbb{Z}$ such that $a+b+kN = 0$ is equivalent to both, (i) and (ii). □

2.2 Lattices

Lattices have been successfully employed to attack various cryptographic algorithms, e.g. Knapsack and DSA-like signature schemes (see [11], [14], [8]). General definition of a lattice can be found in [11]. For the purpose of this work however and for the most of the other approaches as well, the general definition (and the properties related) is unnecessarily wide. Here, we define a *full-rank* lattice and for simplicity call it a lattice without further notice.

2.2.1 Basic definition and properties

Definition 2.8. A *lattice* $\mathcal{L}$ in $\mathbb{Q}^d$ is a set of lattice vectors

$$\left\{ \sum_{i=1}^{d} \alpha_i \mathbf{b}_i \,|\, \alpha_i \in \mathbb{Z} \right\},$$

where $\mathbf{b}_1, \ldots, \mathbf{b}_d \in \mathbb{Q}^d$ are linearly independent and are called basis vectors of lattice $\mathcal{L}$. The matrix, which rows are the basis vectors, is called basis matrix of lattice $\mathcal{L}$. We say the lattice $\mathcal{L}$ is generated by the rows of its basis matrix.

In what follows, we will use symbol $\mathbf{B}$ for the set of basis vectors and for the corresponding basis matrix, as well.

Lemma 2.9. *For $d \geq 2$, any lattice $\mathcal{L}$ in $\mathbb{Q}^d$ has infinitely many basis.*

Proof. Let $\{\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_d}\}$ be a basis of $\mathcal{L}$. Then $\{\mathbf{u_1} + \mathbf{u_2}, \mathbf{u_2}, \ldots, \mathbf{u_d}\}$ is basis of $\mathcal{L}$, as well. □

Lemma 2.10. *Let* $\mathbf{B}$ *and* $\mathbf{B}'$ *be two basis matrices of lattice* $\mathcal{L}$. *Then*

$$|\det \mathbf{B}| = |\det \mathbf{B}'|.$$

Proof. Let $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_d\}$ and $\mathbf{B}' = \{\mathbf{b}'_1, \dots, \mathbf{b}'_d\}$. Since $\mathbf{B}'$ is a basis matrix, we can write $\mathbf{b}_i = \sum_{j=1}^{d} u_{i,j}\mathbf{b}'_j$ for $1 \leq i \leq d$, where $u_{i,j} \in \mathbb{Z}$. Notation $\mathbf{U} = (u_{i,j})$ gives $\mathbf{B} = \mathbf{U}\mathbf{B}'$.

Analogically, there exists an integral matrix $\mathbf{U}'$ such that $\mathbf{B}' = \mathbf{U}'\mathbf{B}$. Since $\mathbf{B} = \mathbf{U}\mathbf{B}' = \mathbf{U}\mathbf{U}'\mathbf{B}$, it holds $\mathbf{U}\mathbf{U}' = \mathbf{I}_d$. Finally, matrices $\mathbf{U}$ and $\mathbf{U}'$ are both integral, thus we can write $\det \mathbf{U} = \det \mathbf{U}' = \pm 1$. $\square$

Definition 2.11. The volume (or determinant) of lattice $\mathcal{L}$ is defined as

$$\mathrm{Vol}(\mathcal{L}) = |\det \mathbf{B}|,$$

where $\mathbf{B}$ is a basis matrix of $\mathcal{L}$.

Remark 2.12. From the geometric point of view, the volume of a lattice corresponds to d-dimensional volume of the parallelepiped spanned by the vectors of the basis matrix.

Definition 2.13. For a lattice $\mathcal{L}$ we define i-th Minkowski's minimum $\lambda_i(\mathcal{L})$ as the minimum of $\max_{1 \leq j \leq i} \|\mathbf{v}_j\|$ over all linearly independent vectors $\mathbf{v}_1$, $\dots$, $\mathbf{v}_i \in \mathcal{L}$.

First Minkowski's minimum $\lambda_1(\mathcal{L})$ is often referenced to as the norm of lattice $\mathcal{L}$, noted $\|\mathcal{L}\|$. Analogically, symbol $\|\mathcal{L}\|_\infty$ is defined if infinity norm is applied in previous expression.

Remark 2.14. The i-th Minkowski's minimum can be seen as the radius of the smallest sphere with the center in origin that contains i linearly independent lattice vectors.

2.2.2 Algorithmic problems

For a given lattice $\mathcal{L}$ in $\mathbb{Q}^d$ one encounters following problems

SVP *Shortest vector problem*
Find $\mathbf{u} \in \mathcal{L}$ such that $\|\mathbf{u}\| = \|\mathcal{L}\|$.

ASVP *Approximate shortest vector problem*
For a fixed $f(d)$ find $\mathbf{u} \in \mathcal{L}$ such that $\|\mathbf{u}\| \leq f(d)\,\|\mathcal{L}\|$.

CVP *Closest vector problem*

For $\mathbf{v} \in \mathbb{Q}^d$ find $\mathbf{u} \in \mathcal{L}$ such that $\|\mathbf{u} - \mathbf{v}\| = \min_{\mathbf{w} \in \mathcal{L}} \|\mathbf{w} - \mathbf{v}\|$.

ACVP *Approximate closest vector problem*
For $\mathbf{v} \in \mathbb{Q}^d$ and a fixed $f(d)$ find $\mathbf{u} \in \mathcal{L}$ such that
$\|\mathbf{u} - \mathbf{v}\| \leq f(d) \min_{\mathbf{w} \in \mathcal{L}} \|\mathbf{w} - \mathbf{v}\|$.

SBP *Smallest basis problem*
Find a basis $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_d\}$ such that $\prod_{i=1}^{d} \|\mathbf{b}_i\|$ is minimal.

The parameter $f(d)$ in the definition of ASVP and ACVP is called the approximation factor and depends only on dimension d.

2.2.3 LLL algorithm

Theorem 2.15 (LLL algorithm)**.** *There exists a polynomial time algorithm, which given a basis of lattice $\mathcal{L}$ in $\mathbb{Q}^d$ as input returns basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_d\}$ of lattice $\mathcal{L}$ that satisfies*

$$\begin{aligned} \|\mathbf{b}_1\| &\leq 2^{\frac{d-1}{4}}\, Vol(\mathcal{L})^{\frac{1}{d}} \\ \|\mathbf{b}_i\| &\leq 2^{\frac{d-1}{2}} \lambda_i(\mathcal{L}), \quad 1 \leq i \leq d \\ \prod_{i=1}^{d} \|\mathbf{b}_i\| &\leq 2^{\binom{d}{2}/2}\, Vol(\mathcal{L}) \end{aligned}$$

Proof. To be found in [9]. □

Definition 2.16. A basis of lattice $\mathcal{L}$ with the properties described in Theorem 2.15 is called *LLL-reduced.*

Remark 2.17. Being given a basis of lattice $\mathcal{L}$, one can solve ASVP by finding its LLL-reduced basis $\{\mathbf{b}_1, \ldots, \mathbf{b}_d\}$. The first vector $\mathbf{b}_1$ solves the problem with approximation factor $f(d) = 2^{\frac{d-1}{2}}$. It is known LLL algorithm in most cases coming from practical cryptanalysis performs much better than what is theoretically guaranteed (see [11]).

2.2.4 Babai's algorithm

Even though the original Babai's Closest Plane algorithm described in [3] solves ACVP with the approximation factor $2^{\frac{d}{2}}$, we present it here with slightly better factor $2^{\frac{d}{4}}$, as is usual in contemporary cryptanalytic practice (see [5], [14]). As stated in [5], the improvement can be achieved by the adjustment of the constants in the definition of LLL reduced basis.

Theorem 2.18 (Babai's algorithm)**.** *Let $\mathcal{L}$ be a lattice in $\mathbb{Q}^d$. There exists a polynomial time algorithm, which given an LLL reduced basis of lattice $\mathcal{L}$ and $\mathbf{v} \in \mathbb{Q}^d$ as input returns lattice vector $\mathbf{u} \in \mathcal{L}$ such that*

$$\|\mathbf{u} - \mathbf{v}\| \leq 2^{\frac{d}{4}} \min_{\mathbf{w} \in \mathcal{L}} \|\mathbf{w} - \mathbf{v}\| .$$

Proof. To be found in [3]. $\square$

3 Extended Hidden Number Problem

The original Hidden Number Problem (HNP) proposed by Boneh and Venkatesan in [5] can be equivalently formulated as follows: Let N be a prime and let x, $x \in \mathbb{Z}_N$ be a particular unknown integer that satisfies d congruences

$$\alpha_i x + \rho_i k_i \equiv \beta_i \ (\mathbf{mod}\, N), \quad 1 \le i \le d,$$

where α_i, $\alpha_i \not\equiv 0 \ (\mathbf{mod}\, N)$, ρ_i and β_i, $1 \le i \le d$ are known values. The unknown integers k_i satisfy $0 \le k_i < 2^{\mu}$, $1 \le i \le d$, where μ is a known rational constant. The Hidden Number Problem (HNP) is to find x (the hidden number).

Let us now formulate an extension to HNP and propose a probabilistic polynomial time algorithm solving its instances.

3.1 Problem definition

Definition 3.1 (Extended hidden number problem)**.** Let N be a prime and let x, $x \in \mathbb{Z}_N$, be a particular unknown integer such that

$$x = \bar{x} + \sum_{j=1}^{m} 2^{\pi_j} x_j, \tag{1}$$

where the integers $\bar{x}$ and π_j, $1 \le j \le m$ are known. The unknown integers x_j satisfy $0 \le x_j < 2^{\nu_j}$, where ν_j are known rational constants $1 \le j \le m$. Furthermore, let us be given d congruences

$$\alpha_i \sum_{j=1}^{m} 2^{\pi_j} x_j + \sum_{j=1}^{l_i} \rho_{i,j} k_{i,j} \equiv \beta_i - \alpha_i \bar{x} \ (\mathbf{mod}\, N), \quad 1 \le i \le d, \tag{2}$$

where α_i, $\alpha_i \not\equiv 0 \ (\mathbf{mod}\, N)$, $1 \le i \le d$, π_j, $1 \le j \le m$, $\rho_{i,j}$, $1 \le i \le d$, $1 \le j \le l_i$ and β_i, $1 \le i \le d$ are known values. The unknown integers $k_{i,j}$ satisfy $0 \le k_{i,j} < 2^{\mu_{i,j}}$, where $\mu_{i,j}$ are known, $1 \le i \le d, 1 \le j \le l_i$. We define $\tau = \sum_{j=1}^{m} \nu_j$, $\xi_i = \sum_{j=1}^{l_i} \mu_{i,j}$, $1 \le i \le d$ and $\xi = \sum_{i=1}^{d} \xi_i$.

The Extended Hidden Number Problem (EHNP) is to find (the hidden number) x and its instance is represented by

$$\left(\bar{x}, N, \{\pi_j, \nu_j\}_{j=1}^{m}, \left\{\alpha_i, \{\rho_{i,j}, \mu_{i,j}\}_{j=1}^{l_i}, \beta_i\right\}_{i=1}^{d}\right). \tag{3}$$

3.2 Existing solutions

Before we introduce the algorithm for solving EHNP, we present two existing approaches solving the problem or its special cases. It will be shown later on, the instances of EHNP may arise from Digital Signature Algorithm (DSA) key disclosure problem when side information about the signing process is known to the attacker for several signatures. This method of deriving EHNP instances is used by both approaches we overview here. The difference is the amount and the fragmentation of the side information they handle.

3.2.1 Nguyen and Shparlinski's approach

In [10], the authors propose a lattice-based probabilistic polynomial time algorithm that solves a special case of EHNP for $l_i = 1$, $\mu_{i,1} = \mu$, $1 \leq i \leq d$ and $\tau = \lceil \log_2 N \rceil$ with the probability of success $P > 1 - \frac{2^{d\mu}}{(N-1)^{d-1}} \left(1 + 2^{\frac{d+1}{4}} (1+d)^{\frac{1}{2}}\right)^d$. Since the dimension of the lattice employed is $d + 1$, i.e. relatively low, it is reasonably fast and may use sophisticated lattice reduction techniques.

To support the formal proof of the method the authors conduct several experiments and are able to solve the instances of the problem with $\mu = 3$ and $d = 100$. Finally, the algorithm is extended to solve EHNP with $l_i = 2$ making use of continued fractions. The extension relies on the conversion to the original algorithm resulting, however, in worse probability of success.

3.2.2 Howgrave-Graham and Smart's approach

The approach presented in [8] is on the first view very different from the one proposed by Nguyen and Shparlinski - the method of Howgrave-Graham and Smart accepts all of the instances of EHNP on the input, however the correctness of its potential success is not supported by a formal proof. The heuristics presented heavily relies on the assumptions about the length of the shortest vector in a random lattice. To support the method experimentally, the authors successfully disclose the hidden number for an EHNP instance with $l_i = 8$, $\xi_i = 80$ for $1 \leq i \leq 2$.

On a closer look, however, we can see that the basic algorithmic ideas of this method are quite similar to the Nguyen-Shparlinski approach.

3.2.3 Motivation for the new method

To summarize, the main drawback of the first approach is its requirement on the sequence of the indexes of known bits to be "continuous" (i.e. increasing with factor 1). Real-life scenarios satisfying this requisite are rare, compared to more frequent side channels providing the information in several blocks of known bits. One such side channel will be shown at the end of the work. On the other hand, the disadvantage of the second approach is the lack of the formal proof.

Both methods solve EHNP using a set of diophantine approximations, i.e. the set of inequalities

$$-2^{\mu_i} < \underbrace{\sum_{j=1}^{l_i} \alpha_{i,j} y_{i,j}}_{\text{“auxiliaries”}} + \underbrace{\sum_{j=1}^{l_0} \alpha_{0,j} y_{0,j}}_{\text{hidden number}} - \underbrace{\beta_i}_{\text{approximation}} < 2^{\mu_i}, \quad 1 \le i \le d$$

where $\alpha_{i,j}$ and μ_i are known values and the solution vector $\mathbf{y}_0$ together with the approximation-supporting vectors $\mathbf{y}_i$ (“auxiliaries”) are restricted to be in $\mathbb{Z}^{l_i}$. Suitable lattices are constructed based on these inequalities, containing hidden vector that reveals the solution vector $\mathbf{y}_0$ and henceforth the hidden number itself. Hidden vector is hoped to be found by Babai’s algorithm with its known approximation vector $\boldsymbol{\beta}$ on input.

In the design of the new method for solving EHNP, we undergo similar route as the two approaches, with an important modification however - we construct the set of diophantine equations

$$\left(\sum_{j=1}^{l_i} \alpha_{i,j} y_{i,j} + \sum_{j=1}^{l_0} \alpha_{0,j} y_{0,j} \right) - \beta_i = 0, \quad 1 \le i \le d$$

which allow us to formally prove the method with minimal heuristic assumptions, as in the first method, while preserving the generality of the second approach. The lattice employed is inspired by the one used by Howgrave-Graham and Smart together with the approach used in [4] while the outline of the proof is strongly motivated by the approach of Nguyen and Shparlinski.

3.3 Algorithm solving EHNP

Definition 3.2. For $\delta > 0$ and a given instance $\mathcal{I} = \mathcal{I}_{EHNP}$ of the EHNP we define $\mathcal{L}(\mathcal{I}, \delta)$ as the lattice spanned by the rows of the matrix

$$\mathbf{B} = \mathbf{B}(\mathcal{I}, \delta) = \begin{pmatrix} N \cdot \mathbf{I}_d & \emptyset & \emptyset \\ \\ \mathbf{A} & \mathbf{X} & \emptyset \\ \begin{matrix} \boldsymbol{\rho}_1^T & & \\ & \ddots & \\ & & \boldsymbol{\rho}_d^T \end{matrix} & \emptyset & \mathbf{K} \end{pmatrix} \in \mathbb{Q}^{D \times D},$$

where we define integers $L = \sum_{i=1}^{d} l_i$ and $D = d + m + L$, vectors

$$\boldsymbol{\rho}_i = (\rho_{i,1}, \dots, \rho_{i,l_i}) \in \mathbb{Z}^{l_i}, \quad 1 \leq i \leq d$$

and matrices

$$\begin{aligned} \mathbf{A} &= (a_{j,i})_{1 \leq i \leq d, 1 \leq j \leq m} \in \mathbb{Z}^{m \times d}, \text{ where } a_{j,i} = \alpha_i 2^{\pi_j} \\ \mathbf{X} &= \mathbf{diag}\left(\frac{\delta}{2^{\nu_1}}, \dots, \frac{\delta}{2^{\nu_m}}\right) \in \mathbb{Q}^{m \times m} \\ \mathbf{K} &= \mathbf{diag}\left(\frac{\delta}{2^{\mu_{1,1}}}, \dots, \frac{\delta}{2^{\mu_{1,l_1}}}, \frac{\delta}{2^{\mu_{2,1}}}, \dots, \frac{\delta}{2^{\mu_{2,l_2}}}, \dots, \frac{\delta}{2^{\mu_{d,1}}} \dots, \frac{\delta}{2^{\mu_{d,l_d}}}\right) \in \mathbb{Q}^{L \times L}. \end{aligned}$$

3.4 Short vectors in $\mathcal{L}$

Lemma 3.3 (Short solutions)**.** *Given prime N and $s \in \mathbb{Z}_N$, let $\rho_1, \dots, \rho_l$ be uniformly and independently distributed on $\mathbb{Z}_N$. Then the probability that the congruence*

$$\sum_{j=1}^{l} \rho_j x_j \equiv s \ (\mathbf{mod}\, N) \tag{4}$$

has a "short" non-trivial solution $(t_1, \dots, t_l) \in (\mathbb{Z}_N)^l$ satisfying $|t_j|_N < T_j$, $1 \leq j \leq l$ is lower than

$$\frac{2^l \prod_{j=1}^{l} T_j}{N}. \tag{5}$$

Algorithm 1 Finding a solution candidate for EHNP

```
Input:   Instance $\mathcal{I}$ of EHNP
Output:  Solution candidate $z \in \mathbb{Z}_N$
1: $\kappa_D \leftarrow 2^{\frac{D}{4}}(m+L)^{\frac{1}{2}} + 1$
2: Choose $\delta \in \mathbb{Q}$ such that $0 < \kappa_D\delta < 1$
3: $\mathbf{v} \leftarrow ((\beta_1 - \alpha_1\bar{x}) \,\mathbf{mod}\, N, \ldots, (\beta_d - \alpha_d\bar{x}) \,\mathbf{mod}\, N, 0, \ldots, 0)$
4: find $\mathbf{W} \in \mathcal{L} = \mathcal{L}(\mathcal{I}, \delta)$, $\mathbf{W} = (W_1, \ldots, W_D)$ such that $\|\mathbf{W} - \mathbf{v}\| \leq 2^{\frac{D}{4}} \min_{\mathbf{B}\in\mathcal{L}} \|\mathbf{v} - \mathbf{B}\|$    //in polynomial time (Lemma 2.15, 2.18)
5: for $j = 1$ to $m$ do
6:     $x'_j \leftarrow \frac{W_{d+j}2^{\nu_j}}{\delta}$    //$x'_j \in \mathbb{Z}$
7: end for
8: $z \leftarrow \bar{x} + \sum_{j=1}^{m} 2^{\pi_j} x'_j \,\mathbf{mod}\, N$
9: return $z$
```

Proof. Let $\mathbf{t} = (t_1, \ldots, t_l)$ be a non-zero l-tuple in $(\mathbb{Z}_N)^l$ with $t_k \neq 0$. There exist exactly N^{l-1} l-tuples

$$(\rho_1, \ldots, \rho_l) = \left(\rho_1, \ldots, \rho_{k-1}, (t_k)^{-1}\left(s - \sum_{j=1, j\neq k}^{l} \rho_j t_j\right) \mathbf{mod}\, N, \rho_{k+1}, \ldots, \rho_l\right)$$

such that $\mathbf{t}$ is a solution of (4). Consequently, there exist no more than

$$N^{l-1}\left(\left(\prod_{j=1}^{l}(2T_j - 1)\right) - 1\right) < N^{l-1}2^l \prod_{j=1}^{l} T_j$$

l-tuples $(\rho_1, \ldots, \rho_l)$ such that "short" non-trivial solution of (4) exists. Since N^l is total number of all l-tuples on $\mathbb{Z}_N$, the lemma follows. □

Lemma 3.4. *Let $\mathcal{I}$ be an instance of the EHNP, where α_i, $1 \leq i \leq d$ and $\rho_{i,j}$, $1 \leq i \leq d$, $1 \leq j \leq l_i$ are uniformly and independently distributed on $\langle 1, N-1\rangle$. Let $\delta, \kappa_D \in \mathbb{Q}$ be such that $0 < \delta$, $0 < \kappa_D$ and $\kappa_D\delta < 1$. Then with the probability*

$$P > 1 - \frac{(2\kappa_D)^{L+m} 2^{\tau+\xi}}{N^d} \tag{6}$$

for each vector $\mathbf{\Delta} \in \mathcal{L} = \mathcal{L}(\mathcal{I}, \delta)$ with coordinates $\mathbf{c} = (e_1, \ldots, e_d, y_1, \ldots, y_m, t_{1,1}, \ldots, t_{1,l_1}, \ldots, \ldots, t_{d,1}, \ldots, t_{d,l_d})$ w.r.t. basis $\mathbf{B} = \mathbf{B}(\mathcal{I}, \delta)$ (i. e. $\mathbf{\Delta} = \mathbf{cB}$), satisfying $\|\mathbf{\Delta}\|_\infty < \kappa_D\delta$

(i) there exists (a witness index) w, $1 \leq w \leq d$ such that

$$t_{w,j} \equiv 0 \;(\mathbf{mod}\, N), \quad 1 \leq j \leq l_w, \tag{7}$$

(ii) $\sum_{j=1}^{m} 2^{\pi_j} y_j \equiv 0 \ (\mathbf{mod}\, N)$ *holds,*

(iii) $\sum_{j=1}^{l_i} \rho_{i,j} t_{i,j} \equiv 0 \ (\mathbf{mod}\, N), 1 \le i \le d$ *holds.*

Proof. Let $\mathbf{\Delta} \in \mathcal{L}$ be such that $\|\mathbf{\Delta}\|_\infty < \kappa_D \delta < 1$. Then

$$\begin{aligned}
\left| e_i N + \alpha_i \sum_{j=1}^{d} 2^{\pi_j} y_j + \sum_{j=1}^{l_i} \rho_{i,j} t_{i,j} \right| &= |\Delta_i| < 1, \quad 1 \le i \le d && (8)\\
\left| \frac{\delta}{2^{\nu_j}} y_j \right| &= |\Delta_{d+j}| < \kappa_D \delta, \ 1 \le j \le m && (9)\\
\left| \frac{\delta}{2^{\mu_{i,j}}} t_{i,j} \right| &= \left|\Delta_{\gamma(i,j)}\right| < \kappa_D \delta, \quad \begin{matrix} 1 \le i \le d \\ 1 \le j \le l_i \end{matrix} && (10)
\end{aligned}$$

where $\gamma(i,j) = d + m + j + \sum_{u=1}^{i-1} l_u$, respectively implying

$$\begin{aligned}
\left| \alpha_i \sum_{j=1}^{d} 2^{\pi_j} y_j + \sum_{j=1}^{l_i} \rho_{i,j} t_{i,j} \right|_N &= 0, \quad 1 \le i \le d && (11)\\
|y_j| &< \kappa_D 2^{\nu_j}, \quad 1 \le j \le m && (12)\\
|t_{i,j}| &< \kappa_D 2^{\mu_{i,j}}, \quad 1 \le i \le d, 1 \le j \le l_i, && (13)
\end{aligned}$$

since the expression on the left-hand side of (8) is an integer. Furthermore, by Lemma 2.7 (11) is equivalent to the congruence

$$\sum_{j=1}^{l_i} \rho_{i,j} t_{i,j} \equiv -\alpha_i \sum_{j=1}^{d} 2^{\pi_j} y_j \ (\mathbf{mod}\, N), \quad 1 \le i \le d. \tag{14}$$

To prove (i), we have to show the probability P_F that for all i, $1 \le i \le d$ there exists j, $1 \le j \le l_i$ such that $t_{i,j} \not\equiv 0 (\mathbf{mod}\, N)$ is bounded above as $P_F < \frac{(2\kappa_D)^{L+m} 2^{\tau+\xi}}{N^d}$. The following probability elaboration is focused on the event that an "unwanted" vector does exist in the lattice at all, rather than investigating the properties of a particular vector chosen. The algorithmic interpretation is then that, obviously, the particular vector computed cannot have the properties that no such vector in the lattice $\mathcal{L}(\mathcal{I}, \delta)$ has.

Fix an m-tuple $(y_1, \ldots, y_m) \in \mathbb{Z}^m$ and define $R_i = -\alpha_i \sum_{j=1}^{d} 2^{\pi_j} y_j \,\mathbf{mod}\, N$. The substitution to congruence (14) gives

$$\sum_{j=1}^{l_i} \rho_{i,j} t_{i,j} \equiv R_i \;(\,\mathbf{mod}\, N), \quad 1 \le i \le d. \tag{15}$$

Lemma 3.3 states non-trivial solution of (15) satisfying (13) exists with the probability

$$p_i(y_1, \ldots, y_m) < \frac{2^{l_i} \prod_{j=1}^{l_i} \kappa_D 2^{\mu_{i,j}}}{N} = \frac{(2\kappa_D)^{l_i}\, 2^{\xi_i}}{N}. \tag{16}$$

For a fixed m-tuple $(y_1, \ldots, y_m)$, the probability that (15) and (13) can be non-trivially satisfied for all i, $1 \le i \le d$ is

$$p(y_1, \ldots, y_m) \le \prod_{i=1}^{d} p_i(y_1, \ldots, y_m) < \prod_{i=1}^{d} \frac{(2\kappa_D)^{l_i}\, 2^{\xi_i}}{N} = \frac{(2\kappa_D)^{L}\, 2^{\xi}}{N^d}. \tag{17}$$

There is no more than $\prod_{j=1}^{m} 2\kappa_D 2^{\nu_j} = (2\kappa_D)^m 2^{\tau}$ m-tuples $(y_1, \ldots, y_m)$ that satisfy (12), therefore

$$P_F \le \sum_{\substack{\mathbf{y} = (y_1, \ldots, y_m) \\ \mathbf{y} \text{ satisfies (12)}}} p(y_1, \ldots, y_m) < \frac{(2\kappa_D)^{L+m}\, 2^{\tau+\xi}}{N^d}. \tag{18}$$

To prove the congruence (ii) holds, it suffices to substitute $t_{w,j} \equiv 0 \;(\,\mathbf{mod}\, N)$, $1 \le j \le l_w$ from (i) to (14), i.e.

$$\sum_{j=1}^{m} 2^{\pi_j} y_j \equiv -(\alpha_w)^{-1} \sum_{j=1}^{l_w} \rho_{w,j} t_{w,j} \equiv 0 \;(\,\mathbf{mod}\, N), \tag{19}$$

since $(\alpha_w)^{-1} \,\mathbf{mod}\, N$ exists, because $\alpha_w \not\equiv 0 \,\mathbf{mod}\, N$ and N is a prime.

Finally, substituting (ii) to the congruence (14), i.e.

$$\sum_{j=1}^{l_i} \rho_{i,j} t_{i,j} \equiv -\alpha_i \sum_{j=1}^{m} 2^{\pi_j} y_j \equiv 0 \;(\,\mathbf{mod}\, N), \tag{20}$$

finishes the proof of (iii). $\square$

3.5 Correctness of the solution

Theorem 3.5. *Let x be the solution of EHNP specified by the instance $\mathcal{I} = \mathcal{I}_{EHNP}$ where N is prime, α_i, $1 \le i \le d$ and $\rho_{i,j}$, $1 \le i \le d$, $1 \le j \le l_i$ are uniformly and independently distributed on $\langle 1, N-1 \rangle$. Then with the probability*

$$P > 1 - \frac{(2\kappa_D)^{L+m}\, 2^{\tau+\xi}}{N^d}, \tag{21}$$

where $\kappa_D = 2^{\frac{D}{4}}(m+L)^{\frac{1}{2}} + 1$, Algorithm 1 returns the correct particular solution of instance $\mathcal{I}$.

Proof. Let $\delta \in \mathbb{Q}$ be such that $0 < \kappa_D \delta < 1$. By Definition 3.1 there exists vector $\mathbf{h} = (c_1, \ldots, c_d,\, x_1, \ldots, x_m,\, k_{1,1}, \ldots, k_{1,l_1},\, \ldots, \ldots,\, k_{d,1}, \ldots, k_{d,l_d}) \in \mathbb{Z}^D$ such that

$$c_i N + \alpha_i \sum_{j=1}^{m} 2^{\pi_j} x_j + \sum_{j=1}^{l_i} \rho_{i,j} k_{i,j} = \beta_i - \alpha_i \bar{x}, \qquad 1 \le i \le d, \tag{22}$$

$$0 \le x_j < 2^{\nu_j}, \qquad 1 \le j \le m, \tag{23}$$

$$0 \le k_{i,j} < 2^{\mu_{i,j}}, \qquad 1 \le i \le d,\ 1 \le j \le l_i \tag{24}$$

hold. Let $\mathbf{B} = \mathbf{B}(\mathcal{I}, \delta)$ be the matrix defined in Definition 3.2 and let

$$\begin{aligned} \mathbf{H} &= \mathbf{hB} \in \mathcal{L},\ \mathcal{L} = \mathcal{L}(\mathcal{I}, \delta), \\ \mathbf{v} &= ((\beta_1 - \alpha_1 \bar{x}) \,\mathbf{mod}\, N, \ldots, (\beta_d - \alpha_d \bar{x}) \,\mathbf{mod}\, N, 0, \ldots, 0) \in \mathbb{Z}^D. \end{aligned}$$

Since the vector $\mathbf{H} - \mathbf{v}$ is equal to

$$\left(0, \ldots, 0, \delta\frac{x_1}{2^{\nu_1}}, \ldots, \delta\frac{x_m}{2^{\nu_m}}, \delta\frac{k_{1,1}}{2^{\mu_{1,1}}}, \ldots, \delta\frac{k_{1,l_1}}{2^{\mu_{1,l_1}}}, \ldots, \ldots, \delta\frac{k_{d,1}}{2^{\mu_{d,1}}}, \ldots, \delta\frac{k_{d,l_d}}{2^{\mu_{d,l_d}}}\right), \tag{25}$$

and the bounds (23), (24) hold, we can write

$$\|\mathbf{v} - \mathbf{H}\|_\infty < \delta.$$

A lattice vector $\mathbf{W}$ found in step 4 of the algorithm satisfies

$$\|\mathbf{v} - \mathbf{W}\| \le 2^{\frac{D}{4}} \min_{\mathbf{A} \in \mathcal{L}} \|\mathbf{v} - \mathbf{A}\| \le 2^{\frac{D}{4}} \|\mathbf{v} - \mathbf{H}\| < 2^{\frac{D}{4}} \delta (m+L)^{\frac{1}{2}}. \tag{26}$$

Let $\mathbf{\Delta} = \mathbf{H} - \mathbf{W} \in \mathcal{L}$. Since $\|\mathbf{a}\|_\infty \le \|\mathbf{a}\|$ for all $\mathbf{a} \in \mathbb{Z}^D$, by triangle inequality we have

$$\|\mathbf{\Delta}\|_\infty \le \|\mathbf{H} - \mathbf{v}\|_\infty + \|\mathbf{v} - \mathbf{W}\| < \delta + 2^{\frac{D}{4}} \delta (m+L)^{\frac{1}{2}} = \kappa_D \delta. \tag{27}$$

Let $\mathbf{w}$, $\boldsymbol{\gamma}$ be the coordinate vectors of $\mathbf{W}$, $\boldsymbol{\Delta}$, respectively, w.r.t. basis $\mathbf{B}$,

$$\begin{aligned} \mathbf{w} &= (c'_1,\dots,c'_d,x'_1,\dots,x'_m,k'_{1,1},\dots,k'_{1,l_1},\dots,\dots,k'_{d,1},\dots,k'_{d,l_d}) \\ \boldsymbol{\gamma} &= (e_1,\dots,e_d,y_1,\dots,y_m,t_{1,1},\dots,t_{1,l_1},\dots,\dots,t_{d,1},\dots,t_{d,l_d}) \end{aligned}$$

and $z = \bar{x} + \sum_{j=1}^{m} 2^{\pi_j} x'_j$ be the candidate returned by Algorithm 1. Since $\mathbf{B}$ is nonsingular, $\boldsymbol{\gamma} = \mathbf{h} - \mathbf{w}$. Then with the probability greater than $1 - \frac{(2\kappa_D)^{L+m} 2^{\tau+\xi}}{N^d}$, guaranteed by Lemma 3.4, it holds

$$\begin{aligned} x - z &\equiv \left(\bar{x} + \sum_{j=1}^{m} 2^{\pi_j} x_j\right) - \left(\bar{x} + \sum_{j=1}^{m} 2^{\pi_j} x'_j\right) \equiv \\ &\equiv \sum_{j=1}^{m} 2^{\pi_j}(x_j - x'_j) \equiv \sum_{j=1}^{m} 2^{\pi_j} y_j \equiv 0 \ (\mathbf{mod}\, N). \end{aligned}$$

Finally, since $x, z \in \mathbb{Z}_N$, we have $x = z$. □

4 Digital Signature Algorithm

Digital signature algorithm (DSA) is U.S. standard for digital signatures. It is defined FIPS 186-2 (see [1]) published by National Institute of Standards and Technology.

4.1 Public parameters

Definition 4.1. DSA *public parameters* are represented by the triplet (p, q, g) with the following properties

1. $p \in \mathbb{P}, \quad 2^{1023} < p < 2^{1024}$
2. $q \in \mathbb{P}, \quad 2^{159} < q < 2^{160}$
3. $q \,| p - 1$
4. $g \in \mathbb{Z}_p^*, \quad g \neq 1, \quad g^q = 1 \,\mathbf{mod}\, p$

The *private key* x is an element of $\mathbb{Z}_q^*$ chosen randomly with uniform distribution, denoted $x \in_r \mathbb{Z}_q^*$. The corresponding *public key* is $y \in \mathbb{Z}_p^*$ is defined as $y = g^x \,\mathbf{mod}\, p$.

Remark 4.2. Originally, the standard FIPS 186 required the prime number p to be at least 512 bits long. However, in 2001 the second revision of the standard specified p to be in the range $2^{1023} < p < 2^{1024}$. This adjustment aims to harden discrete logarithm problem in $\mathbb{Z}_p^*$ which resolution discloses the private key directly ($y = g^x \,\mathbf{mod}\, p$).

In [7], the author encourages the implementers of security concerned applications to use prime numbers p and q larger then the ones prescribed by [1]. Recently, this idea was supported by the third revision of FIPS 186 where the sizes of primes p and q were raised up to 3072 and 256 bits, respectively.

4.2 Signing operation

Let m, $m \in \{0, 1\}^*$ be a message to be signed. Let h be hash function `SHA-1` defined in [2]. The signature of the message m is the pair (r, s) generated as follows

1. $k \in_R \mathbb{Z}_q^*$
2. $r = \left(g^k \,\mathbf{mod}\, p\right) \,\mathbf{mod}\, q$
   $s = k^{-1} \left(h(m) + xr\right) \,\mathbf{mod}\, q$

Remark 4.3. Random integer k is often referenced to as a nonce (number used once). It is essential to keep this number secret entirely during and after the signing process. If an attacker gains access to the nonce k, she can extract the private key x by computing

$$x = r^{-1}\,(sk - h(m)) \mathbin{\mathbf{mod}} q.$$

Later on, we shall show the private key can be exposed even if a little fraction of the nonce k is leaked for multiple signatures.

Remark 4.4. Value of the nonce k should be erased after the signature is generated. Especially, it should never be reused to sign another message.

Suppose an attacker has knowledge of the messages m_1 and m_2 with signatures (r_1, s_1) and (r_2, s_2) signed using the same nonce k. Then $r_1 = r_2$ and

$$\begin{aligned} s_1 &= k^{-1}\,(h(m_1) + xr_1) \mathbin{\mathbf{mod}} q \\ s_2 &= k^{-1}\,(h(m_2) + xr_1) \mathbin{\mathbf{mod}} q. \end{aligned}$$

Multiplying both congruences by k and their subtraction give

$$k(s_2 - s_1) = h(m_2) - h(m_1) \mathbin{\mathbf{mod}} q$$

diclosing the nonce

$$k = (s_2 - s_1)^{-1}\,(h(m_2) - h(m_1)) \mathbin{\mathbf{mod}} q$$

and the private key x, as described in Remark 4.3.

4.3 Verifying operation

To verify the validity of the signature pair (r, s) of message m, verifier first checks if $0 < r < q$ and $0 < s < q$ hold, he rejects the signature otherwise. Secondly, he computes

1. $w = s^{-1} \mathbin{\mathbf{mod}} q$
2. $u_1 = h(m)w \mathbin{\mathbf{mod}} q$
   $u_2 = rw \mathbin{\mathbf{mod}} q$
3. $v = (g^{u_1} y^{u_2} \mathbin{\mathbf{mod}} p) \mathbin{\mathbf{mod}} q$.

The signature is accepted if and only if $v = r$.

Lemma 4.5. *Let (r, s) be a given signature of message m to be verified. If (r, s) was generated using DSA signing operation described in Section 4.2, then the value v computed during signature verification is equal to r.*

Proof. Definition of s yields $l \in \mathbb{Z}$ such that $k = w(h(m) + xr) - lq$. Let $v' = g^{u_1} y^{u_2} \,\mathbf{mod}\, p$. By definition of u_1 and u_2 there exist $l_1, l_2 \in \mathbb{Z}$ such that

$$\begin{aligned} v' &= g^{h(m)w+l_1q} y^{rw+l_2q} \,\mathbf{mod}\, p = g^{w(h(m)+l_1q} g^{xrw+xl_2q} \,\mathbf{mod}\, p = \\ &= g^{w(h(m)+xr)+(l_1+xl_2)q} \,\mathbf{mod}\, p = g^{k+lq} g^{(l_1+xl+2)q} \,\mathbf{mod}\, p = \\ &= g^k \left(g^q\right)^{l+l_1+xl_2} \,\mathbf{mod}\, p = g^k \,\mathbf{mod}\, p, \end{aligned}$$

since $g^q = 1 \,\mathbf{mod}\, p$. Finally,

$$v = v' \,\mathbf{mod}\, q = \left(g^k \,\mathbf{mod}\, p\right) \,\mathbf{mod}\, q = r.$$

$\square$

4.4 DSA Key Disclosure problem

Definition 4.6 (DSA-KDP problem)**.** Let (p, q, g) be public DSA parameters and (x, y) DSA key pair. Let (r_i, s_i) where

$$r_i = \left(g^{k_i} \,\mathbf{mod}\, p\right) \,\mathbf{mod}\, q, \quad 1 \le i \le d \tag{28}$$

$$s_i = k_i^{-1} \left(h(m_i) + xr_i\right) \,\mathbf{mod}\, q, \quad 1 \le i \le d \tag{29}$$

be known signature pairs of known hashed messages $h(m_i)$. Suppose an additional information about the private key x and the nonces k_i is known, as well, i.e.

$$x = \bar{x} + \sum_{j=1}^{m} 2^{\pi_j} x_j, \qquad 0 \le x_j < 2^{\nu_j}, \quad 1 \le j \le m \tag{30}$$

$$k_i = \bar{k}_i + \sum_{j=1}^{l_i} 2^{\lambda_{i,j}} k_{i,j}, \qquad 0 \le k_{i,j} < 2^{\mu_{i,j}}, \quad 1 \le i \le d, 1 \le j \le l_i, \tag{31}$$

where $\bar{x}$, $\{\pi_j, \nu_j\}_{j=1}^{m}$, $\left\{\bar{k}_i, \{\lambda_{i,j}, \mu_{i,j}\}_{j=1}^{l_i}\right\}_{i=1}^{d}$ are known integers satisfying

$$\begin{array}{llll} & & \bar{x} \in \mathbb{Z}_q, \quad \bar{k}_i \in \mathbb{Z}_q, & 1 \le i \le d \\ 2^{\pi_j} \in \mathbb{Z}_q, & 1 \le j \le m, & 2^{\lambda_{i,j}} \in \mathbb{Z}_q, & 1 \le i \le d,\ 1 \le j \le l_i \\ 2^{\nu_j} \in \mathbb{Z}_q, & 1 \le j \le m, & 2^{\mu_{i,j}} \in \mathbb{Z}_q, & 1 \le i \le d,\ 1 \le j \le l_i \end{array}$$

DSA key disclosure problem (DSA-KDP) is to find the private key x and its instance $\mathcal{I}_{DSA}$ is represented by

$$\left(\bar{x}, q, \{r_i, s_i, h(m_i)\}_{i=1}^{d}, \{\pi_j, \nu_j\}_{j=1}^{m}, \left\{\bar{k}_i, \{\lambda_{i,j}, \mu_{i,j}\}_{j=1}^{l_i}\right\}_{i=1}^{d}\right).$$

Lemma 4.7 (Transition from DSA-KDP to EHNP)**.** *Let x be the particular solution of the DSA-KDP problem specified by the instance $\mathcal{I}_{DSA}$. Let*

$$\begin{aligned} N &= q, \\ \alpha_i &= r_i, \quad 1 \le i \le d \\ \rho_{i,j} &= \left(-s_i 2^{\lambda_{i,j}}\right) \mathbf{mod}\, N, \quad 1 \le i \le d, 1 \le j \le l_i, \\ \beta_i &= \left(s_i \bar{k}_i - h(m_i)\right) \mathbf{mod}\, N, \quad 1 \le i \le d. \end{aligned}$$

Then x can be found as the solution of EHNP specified by the instance

$$\left(\bar{x}, N, \{\pi_j, \nu_j\}_{j=1}^{m}, \left\{\alpha_i, \{\rho_{i,j}, \mu_{i,j}\}_{j=1}^{l_i}, \beta_i\right\}_{i=1}^{d}\right). \tag{32}$$

Proof. With regard to Heuristic assumption 4.8 the lemma follows directly when (30) and (31) are substituted into (29) in Definition 4.6. $\square$

It is a common practice in theoretical cryptology to conjecture probabilistic properties of certain variables basing on an assumption that the scheme they come from is computationally unbreakable. Following this approach, we propose

Heuristic assumption 4.8. *Given that DSA is computationally unbreakable, we may assume that the values of α_i and $\rho_{i,j}$ from Lemma 4.7 are computationally indistinguishable from a sequence representing independent observations and coming from the uniform distribution of integers on $\langle 1, N-1\rangle$.*

Justification A possibility to predict the values of α_i implies a possibility to forge signatures by a randomized strategy described e.g. in [6]. Furthermore, let us observe that having given the value of s_i, we can trivially compute the values of $\lambda_{i,j} \in \langle 0, \lceil \log_2 N \rceil)$ from $\rho_{i,j}$, henceforth deducing a great part of the side information. Therefore, a possibility to predict $\rho_{i,j}$ implies a possibility to deduce the side information even without the existence of the side channel itself, implying an ability to break the DSA regardless its implementation details. However, we assume that this is impossible.

5 Real Scenario Application

5.1 Hyper-Threading Technology

So far, we have seen an additional information about nonces k_i and private key x in DSA scheme can lead to the key disclosure. Now, we will show one of the possible ways on how to obtain such information.

In [12], the author explores a side channel hidden in certain processors design employing the Hyper-Threading Technology (HTT). The processors affected are Intel Pentium 4, Mobile Pentium 4 and Xeon.

The size of L1 cache memory in hyper-threaded Pentium 4 is 8 KB (or 16 KB)[1]. It is divided into 32 cache address classes consisting of 4 (or 8) cache lines of 64 bytes and is called 4 (or 8) way associative. When a memory line is requested by a process, the processor first checks whether the line is already loaded in L1 cache in the corresponding cache class. In case it is, we say a cache hit occurs, contrary to a cache miss when the line has to be loaded to L1 cache. Therefore, a cache miss takes a much longer time than a cache hit and that can be recognized by a spying process.

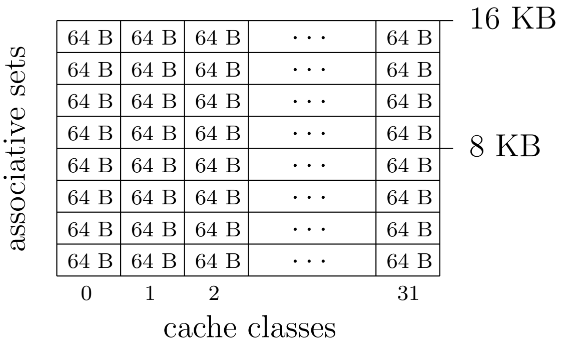

Figure 1: L1 cache memory disposition in a 4 (or 8) way associative Pentium 4 processor.

A hyper-threaded Pentium 4 consists of two separate cores that share access to the same L1 cache memory. To the operating system, the cores appear as two logical CPUs, thus it can schedule two processes to be run at the same time. One process cannot read the data of the other process stored in L1 cache, however, it can determine whether the process running on the second core used a cache line from a certain cache class or not by measuring the amount of time it takes to repeatedly read several data block from the same cache address class.

[1]depending on actual type

In this way, we get a side information which can be used to discriminate between two different operations performed by the process being spied. When the two operations are different enough with respect to the memory access they induce, we can easily identify them basing on a different "footprint" left in the access time measurements (cf. Figure 2).

In [12], this side information was used to break an implementation of RSA scheme. Here, we show that by using the EHNP approach described before, we can break certain implementation of DSA algorithm, as well. In practice, this attack threatens, for instance, applications like SSH [13], when the server runs on an unsecured HTT platform and uses DSA for the server authentication. When the attacker logs on the server, she can run the spy process on one processor core, while she opens another SSH session with the server, hoping that it will run on the second core. In this way, she gains the side information about DSA signature computation when the server authenticates itself for the newly opened connection. Collecting several such measurements, she can get enough information to be able to solve the associated EHNP. From here, she gets the server's private key allowing her to impersonate the server.

5.2 Sliding Window Exponentiation

An OpenSSL-based SSH server uses the sliding window (SW) exponentiation algorithm (cf. Algorithm 2)[2] in the process of DSA authentication when computing $r' = g^k \,\mathbf{mod}\, p$ (followed by the computation $r = r' \,\mathbf{mod}\, q$). Two operations to be discriminated on the HTT platform by the aforesaid technique are squaring (S) and multiplication (M). Being able to identify the SW algorithm execution, the attacker obtains a sequence $\mathcal{S} \in \{S, M\}^*$.

In Figure 2, a part of the execution of SW algorithm from the spy process viewpoint is displayed. The sequence (S, M, S, S, S, S, M, S) is revealed by the latency of cache classes 25 through 29. Typically, the length of the sequence $\mathcal{S}$ is about 200 for the exponent length 160 with sliding window length set to 4 (see Table 1).

To convert $\mathcal{S}$ to an information suitable for EHNP, one sets $\bar{k} = 0$ as the first approximation of the nonce k. Then, she takes the next operation from $\mathcal{S}$ and multiplies $\bar{k}$ by 2 if the operation is S or adds 1 and adds a new "hole" for M. Finally, the holes of zero length are filtered out from the output sequence. The conversion procedure, described by Algorithm 3, outputs the decomposition $k = \bar{k} + \sum_{j=1}^{l} 2^{\lambda_j} k_j$, $0 \le k_j < 2^{\mu_j}$.

[2]Valid for the versions up to 0.9.7g. As from version 0.9.7h, OpenSSL uses fixed window modular exponentiation by default for RSA, DSA, and DH private key operations to prevent cache timing attacks.

Algorithm 2 Sliding window (SW) exponentiation; s is the SW length

```
Input:   $g$, $e = (e_t e_{t-1} \dots e_0)_2$, $e_t = 1$, $s \geq 1$
Output:  $g^e$
1:  $g_1 \leftarrow g$, $g_2 \leftarrow g^2$
2:  for $i = 1$ to $2^{s-1} - 1$ do
3:      $g_{2i+1} \leftarrow g_{2i-1} g_2$
4:  end for
5:  $A \leftarrow 1$, $i \leftarrow t$
6:  while $i \geq 0$ do
7:      if $e_i = 0$ then
8:          $A \leftarrow A^2$                                          //squaring
9:          $i \leftarrow i - 1$
10:     else
11:         find longest string $(e_i e_{i-1} \dots e_l)$ such that $i - l + 1 \leq s$ and $e_l = 1$
12:         $A \leftarrow A^{2^{i-l+1}} g_{(e_i e_{i-1} \dots e_l)_2}$    //$2^{i-l+1}$ squarings, 1 multiplication
13:         $i \leftarrow l - 1$
14:     end if
15: end while
16: return $A$
```

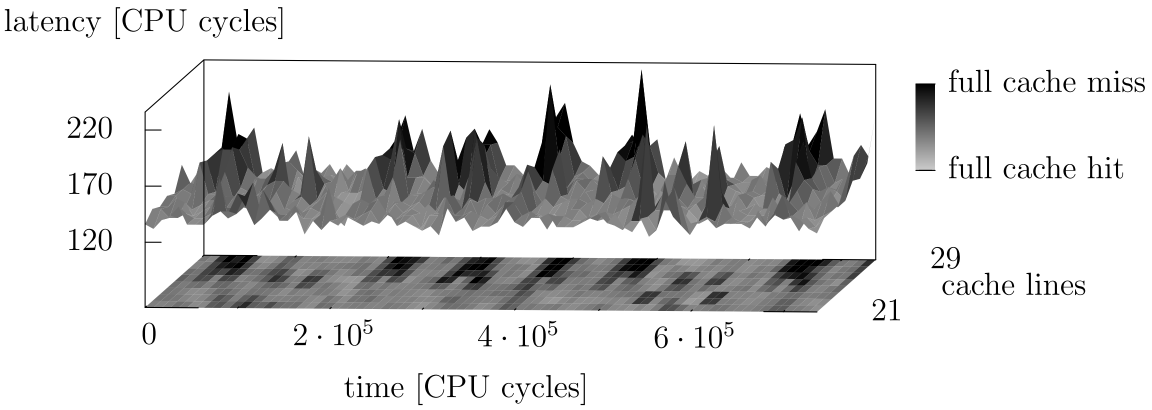

Figure 2: Side channel timing information on selected L1 cache classes latency on a 4-way associative Pentium 4 2.8GHz with HTT enabled during SW exponentiation performed by OpenSSL 0.9.7e running under GNU/Linux Debian 3.1 Sarge.

As stated by Theorem 3.5, the probability of success of Algorithm 1 is significantly affected by the total number of holes in nonces k_i and in the hidden number x, which is $L + m$. The dimension D of the basis matrix $\mathbf{B}$ of

exponent binary size	SW length
672 or greater	6
$\langle 240, 671\rangle$	5
$\langle 80, 239\rangle$	4
$\langle 24, 79\rangle$	3
$\langle 1, 23\rangle$	1

Table 1: Sliding window length used for different exponent (binary) lengths in Algorithm 2 as implemented in OpenSSL 0.9.7e.

Algorithm 3 Conversion of sequence from $\{S, M\}^*$ to the decomposition of k; s is the SW length; (N is the upper bound on k)

```
Input:   $\mathcal{S} \in \{S, M\}^*$, $s \geq 1$, ($N > 1$)
Output:  $\bar{k}, l, \{\lambda_j, \mu_j\}_{j=1}^{l}$
1: $\bar{k} \leftarrow 0$, $L \leftarrow 0$, $\Lambda_0 \leftarrow 1$, $w_0 \leftarrow s - 1$        //L, Λ, and w are internal only
2: $A \leftarrow \mathrm{first}(\mathcal{S})$
3: repeat
4:    if $A = S$ then
5:       $\bar{k} \leftarrow 2\bar{k}$
6:       for $j = 0$ to $L$ do
7:          $\Lambda_j \leftarrow \Lambda_j + 1$                                  //shift existing holes
8:       end for
9:    else
10:      $\bar{k} \leftarrow \bar{k} + 1$
11:      $L \leftarrow L + 1$, $\Lambda_L \leftarrow 1$
12:      $w_L \leftarrow \min(s - 1, \Lambda_{L-1} + w_{L-1} - s - 1)$     //new, possibly empty, hole
13:   end if
14: until $A \leftarrow \mathrm{next}(\mathcal{S})$
15: if $N$ is defined and $\Lambda_1 + w_1 > \lceil \log_2 N \rceil$ then
16:    $w_1 = \lceil \log_2 N \rceil - \Lambda_1$                             //adjust the first hole
17: end if
18: $l \leftarrow 0$
19: for $j = 1$ to $L$ do
20:    if $w_j > 0$ then
21:       $l \leftarrow l + 1$, $\lambda_l \leftarrow \Lambda_j$, $\mu_l \leftarrow w_j$          //keep the nonempty holes
22:    end if
23: end for
```

associated lattice $\mathcal{L}$ defined in Definition 3.2 and consequently the running time of the Algorithm 1 are affected, as well. Thus, it makes sense to consider eliminating short blocks of known bits (especially single bits), as each of them adds one hole to the nonce decomposition, leading to an expensive increase of the basis matrix dimension. This procedure is described in Algorithm 4 with the minimal block size m_b to be considered for the decomposition on input.

Algorithm 4 Exclude blocks of known bits shorter than m_b

```
Input:   $\bar{k}, l, \{\lambda_j, \mu_j\}_{j=1}^{l}$, $m_b \geq 1$
Output:  $\bar{k}', l', \{\lambda'_j, \mu'_j\}_{j=1}^{l'}$
1: $l' \leftarrow 1,\ \bar{k}' \leftarrow \bar{k}$
2: $\lambda'_1 \leftarrow \lambda_1,\ \mu'_1 \leftarrow \mu_1$
3: for j=2 to l do
4:    if $\lambda'_{l'} - (\lambda_j + \mu_j) < m_b$ then
5:       $\bar{k}' \leftarrow \bar{k}' - 2^{\lambda_j+\mu_j}\left(\left(\bar{k}' \,\mathbf{div}\, 2^{\lambda_j+\mu_j}\right) \,\mathbf{mod}\, 2^{\lambda'_{l'}-(\lambda_j+\mu_j)}\right)$
6:       $\mu'_{l'} \leftarrow \mu'_{l'} + (\lambda'_{l'} - \lambda_j),\ \lambda'_{l'} \leftarrow \lambda_j$
7:    else
8:       $l' \leftarrow l' + 1,\ \lambda'_{l'} \leftarrow \lambda_j,\ \mu'_{l'} \leftarrow \mu_j$
9:    end if
10: end for
```

We ran several experiments implementing Algorithms 2, 3 and 4 for different exponent sizes suitable for DSA-like schemes for the purpose of this work and for the sizes suitable for RSA, as well, for informational purpose only. We use the notation $\sigma = \lceil \log_2 N \rceil - \sum_{j=1}^{l} \mu_j$ for the number of bits known in the decomposition. All statistics are based upon a sample of 10^4 random exponent values from $\mathbb{Z}_N$. The sliding window length s is set to match its definition in OpenSSL 0.9.7e (see Table 1). The amount (σ) and the fragmentation of the side channel information on the nonces (or exponents), depending on the minimal (known) block size m_b, gained via L1 cache timing are displayed in Figures 3 and 4. The details can be found in Appendix A.

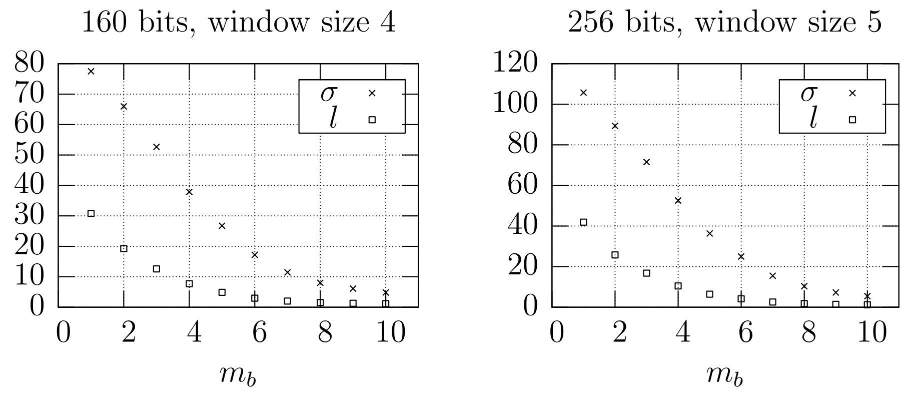

Figure 3: Side channel information on nonces suitable for DSA-like schemes.

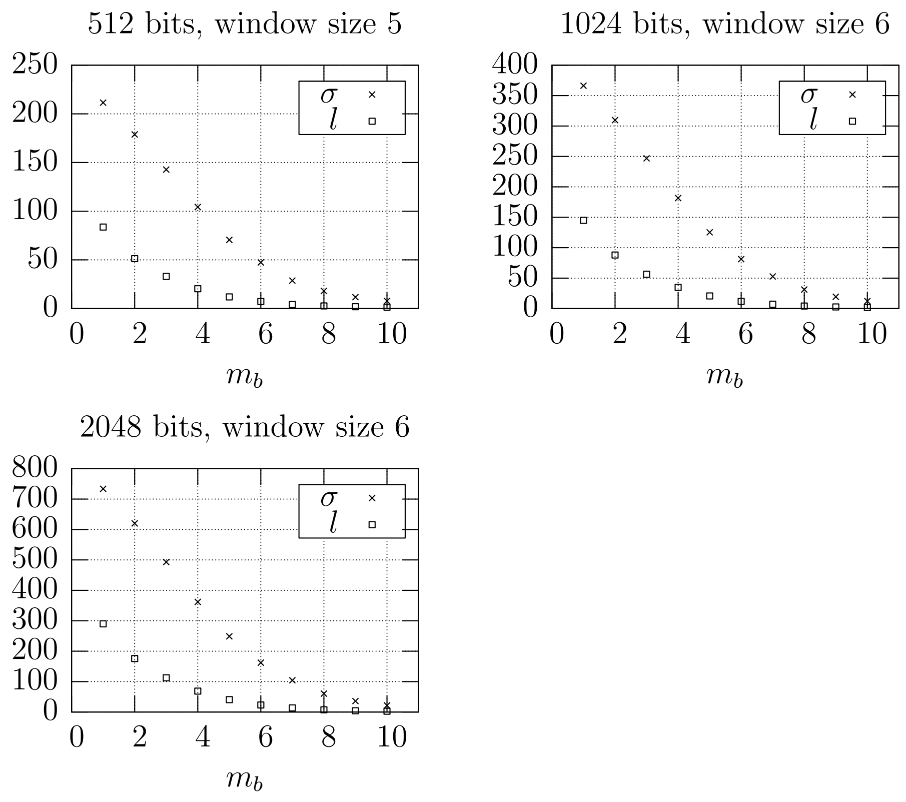

Figure 4: Side channel information on exponents suitable for RSA.

5.3 Practical Experiments with EHNP

Algorithm 1 was implemented in C++ employing Shoup's NTL library [15]. The experiments were run for different number of signatures d and minimal known block size m_b to be considered. Each experiment consists of 10 instances of DSA-KDP with random public parameters, random key pair and the signature pairs for d random messages converted to the associated EHNP instance (Lemma 4.7). The results of the experiments with the size of DSA prime q set to 160 bits are displayed in Figures 5 through 9. The details can be found in Appendix B. We used the side channel emulation in these computations. Its real existence and usability was successfully verified by technical experiments with an SSH server powered by OpenSSL 0.9.7e running under GNU/Linux Debian 3.1 Sarge on an unprotected Pentium 4 HTT platform, as well.

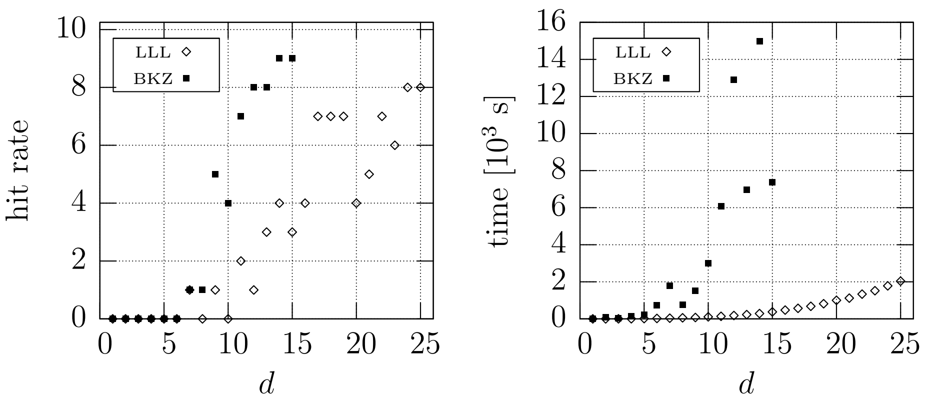

Figure 5: The hit rate and the average duration of EHNP algorithm solving 10 random instances of DSA-KDP for each d, with minimal known block size $m_b = 1$.

If the side channel information is available for sufficient number of signatures, it seems reasonable to ignore shorter blocks of known bits, as the running time of Algorithm 1 drops significantly while the hit rate stays on an acceptable level. The experiments with 14 signatures and $m_b = 1$ in Figure 5 and with 18 signature and $m_b = 5$ in Figure 9 support this hypothesis.

The bases matrices of associated lattices were reduced with LLL reduction (`LLL_XD()`) with parameter `Delta = 0.99` (marked "LLL") in NTL (see [15]). If such reduction did not lead to the key disclosure, stronger Block Korkin Zolotarev reduction with Givens rotations (`G_BKZ_XD()`) was employed in several experiments, with parameters `BlockSize = 40` and `Prune = 15` (marked “BKZ”). The experiments were run for different values of minimal (known) block size m_b to see how much information can be sacrificed while preserving the hit rate on an acceptable level. The computing platform employed was running GNU/Linux Debian on AMD Opteron 844.

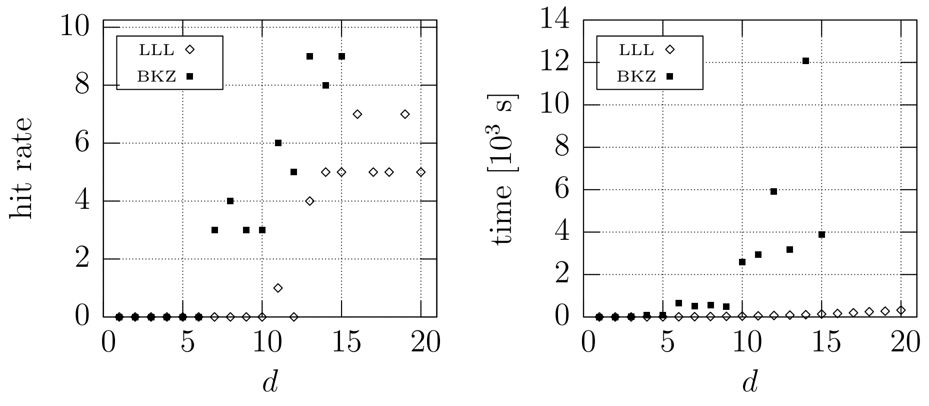

Figure 6: The hit rate and the average duration of EHNP algorithm solving 10 random instances of DSA-KDP for each d, with minimal known block size $m_b = 2$.

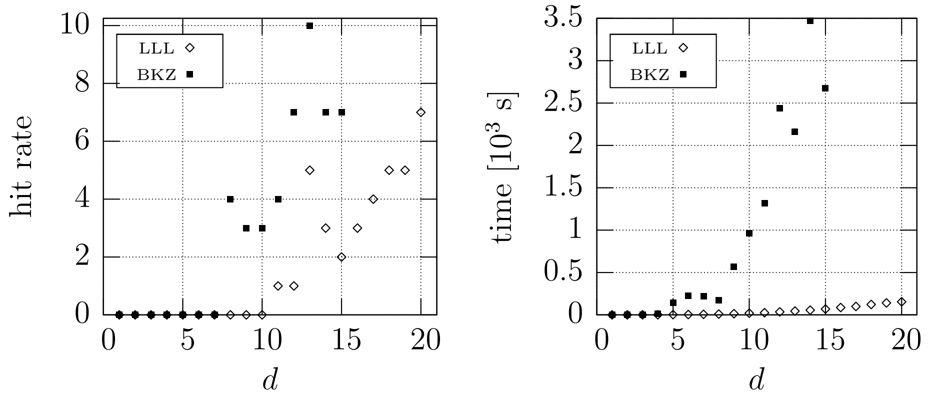

Figure 7: The hit rate and the average duration of EHNP algorithm solving 10 random instances of DSA-KDP for each d, with minimal known block size $m_b = 3$.

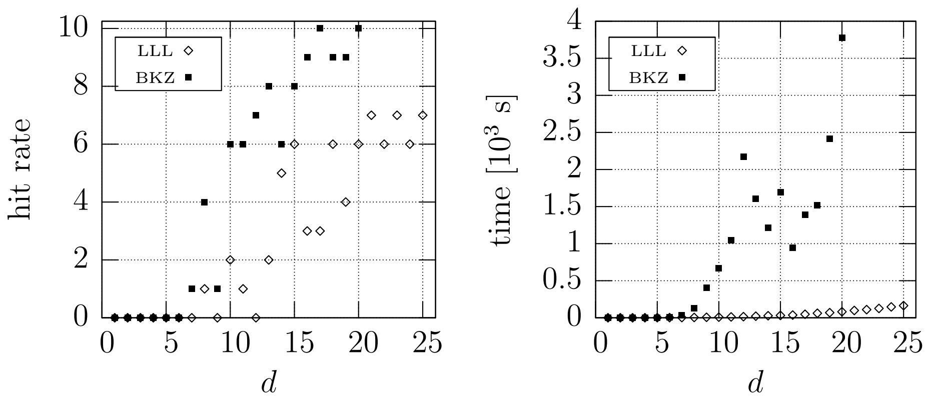

Figure 8: The hit rate and the average duration of EHNP algorithm solving 10 random instances of DSA-KDP for each d, with minimal known block size $m_b = 4$.

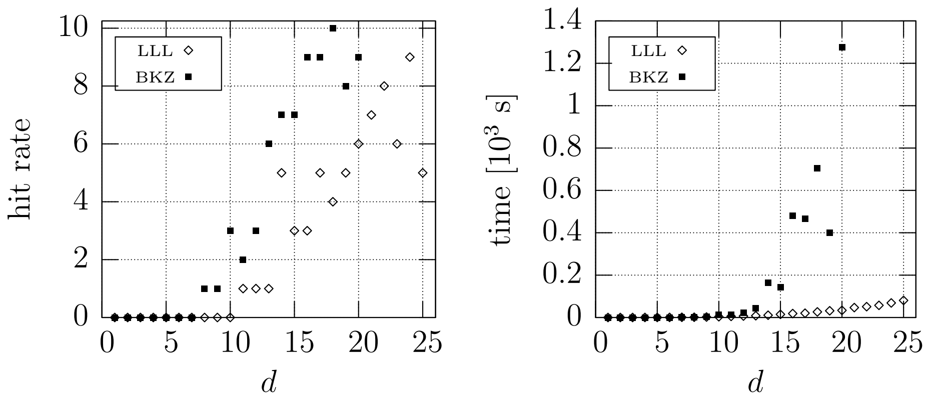

Figure 9: The hit rate and the average duration of EHNP algorithm solving 10 random instances of DSA-KDP for each d, with minimal known block size $m_b = 5$.

A limited series of experiments was run with a sample of 100 random DSA-KDP instances with $m_b = 1$ and $\lceil \log_2 q \rceil = 160$ bits, to see how many signatures are needed to disclose the private key. Here, for each instance we define σ as the total number of known bits in the nonces, i.e. $\sigma = \sum_{i=1}^{d} \left(\lceil \log_2 N \rceil - \sum_{j=1}^{l_i} \mu_{i,j} \right)$. It turns out that in one case out of one hundred instances the key can be extracted from as few as 5 signatures.

d	m_b	D	$\bar{\sigma}$	$\bar{t}_{LLL}$ $[s]$	$\bar{t}_{Babai}$ $[s]$	$\bar{t}_{BKZ}$ $[s]$	hit rate
3	1	96.88	222.02	2.32	0.2	49.5	0
4	1	128.47	298.51	6.34	0.37	113.04	0
5	1	161.19	369.88	13.47	0.65	202.64	1
6	1	192.46	446.61	23.27	0.97	752.69	7

Table 2: Experiments employing BKZ reduction with $m_b = 1$ on 100 random DSA-KDP instances.

Finally, we ran several experiments with random DSA-KDP instances with the size of DSA prime q set to 256 bits (which is the highest size allowed by the draft of Third revision of FIPS 186). The dimension of lattices associated is higher resulting in longer running time, however, as shown in Figure 10, the private key can be extracted for these instances, as well. The details about these experiments can be found in Appendix C.

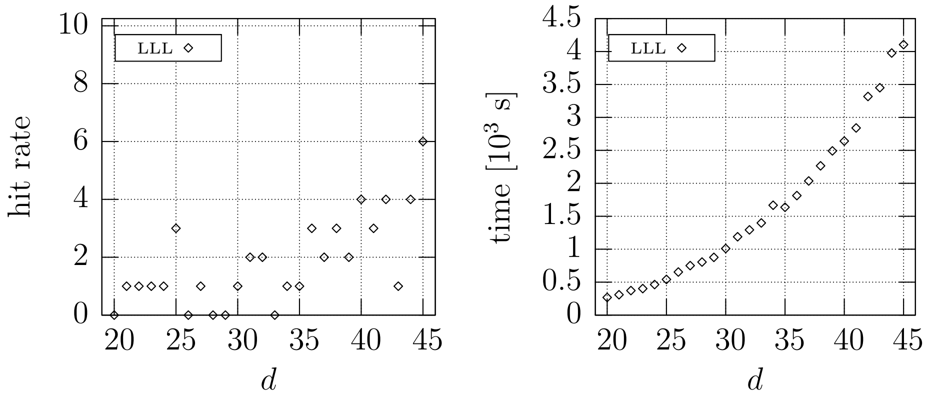

Figure 10: Comparison of the hit rate and the average duration of EHNP algorithm solving 10 random instances of DSA-KDP for each d, with minimal known block size $m_b = 5$ and $\lceil \log_2 q \rceil = 256$ bits.

6 Conclusion

We presented the algorithm for solving an extension of Hidden Number Problem and examined its correctness. We demonstrated a real life cryptanalytic application on Digital Signature Algorithm running in an unsecured environment. We were able to disclose the private key knowing certain side information for as few as 5 signing operations. The method we elaborated can be employed to solve other problems that may be converted to instances of EHNP, as well, e.g. cryptanalysis of Diffie-Hellman key agreement protocol, as shown in [5].

References

[1] *Digital signature standard.* National Institute of Standards and Technology, Washington, 2000. URL: `http://csrc.nist.gov/publications/fips/`. Note: Federal Information Processing Standard 186-2.

[2] *Secure hash standard.* National Institute of Standards and Technology, Washington, 2002. URL: `http://csrc.nist.gov/publications/fips/`. Note: Federal Information Processing Standard 180-2.

[3] L. Babai. On Lovász' lattice reduction and the nearest lattice point problem. In *Proc. on STACS 85 2nd Annual Symposium on Theoretical Aspects of Computer Science*, pages 13–20, New York, NY, USA, 1985. Springer-Verlag New York, Inc.

[4] M. Bellare, S. Goldwasser, and D. Micciancio. “Pseudo-random” generators within cryptographic applications: the DSS case. In *Advances in Cryptology—CRYPTO '97*, pages 277–291, Santa Barbara, California, 17–21 August 1997. Springer-Verlag.

[5] D. Boneh and R. Venkatesan. Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes. In *Proc. of CRYPTO '96*, pages 129–142, London, UK, 1996. Springer-Verlag.

[6] D. R. Brown. Generic groups, collision resistance, and ecdsa. *Designs, Codes and Cryptography*, 35(1):119–152, 2005.

[7] D. Eastlake 3rd. DSA KEYs and SIGs in the Domain Name System (DNS). RFC 2536 (Proposed Standard), March 1999.

[8] N. A. Howgrave-Graham and N. P. Smart. Lattice attacks on digital signature schemes. *Designs, Codes and Cryptography*, 23(3):283–290, 2001.

[9] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Ann.*, 261:513–534, 1982.

[10] P. Q. Nguyen and I. Shparlinski. The insecurity of the digital signature algorithm with partially known nonces. *J. Cryptology*, 15(3):151–176, 2002.

[11] P. Q. Nguyen and J. Stern. The two faces of lattices in cryptology. In *CaLC '01: Revised Papers from the International Conference on Cryptography and Lattices*, pages 146–180, London, UK, 2001. Springer-Verlag.

[12] C. Percival. Cache missing for fun and profit. 2005. URL: `http://www.daemonology.net/papers/htt.pdf`.

[13] OpenBSD project members. OpenSSH Suite. URL: `http://www.openssh.com/`.

[14] T. Rosa. One-Time HNP or Attacks on a Flawed El Gamal Revisited. Cryptology ePrint Archive, Report 2005/460, 2005. `http://eprint.iacr.org/`.

[15] V. Shoup. NTL: A Library for doing Number Theory. URL: `http://www.shoup.net/ntl/`.

Appendix A

Tables 3 and 4 provide further details on experiments implementing Algorithm 3 for the conversion of the sequence $\mathcal{S} \in \{S, M\}^*$ obtained by L1 cache timing to the decomposition $k = \bar{k} + \sum_{j=1}^{d} 2^{\lambda_j} k_j$, $0 \leq k_j < 2^{\mu_j}$ of the unknown nonce (or exponent) k. The statistics are based upon 10^4 random values from $\mathbb{Z}_N$ processed by SW algorithm to obtain the sequence $\mathcal{S}$.

$\log_2 N$	s	m_b	$\overline{\sigma}$	$\sigma_{\min}$	$\sigma_{\max}$	$\bar{l}$	$l_{\min}$	$l_{\max}$	$\overline{\sigma}/\log_2 N$
160	4	1	77.51	60	102	30.79	23	36	0.485
160	4	2	65.97	41	93	19.24	13	26	0.412
160	4	3	52.67	20	86	12.58	5	21	0.329
160	4	4	37.88	1	81	7.70	1	14	0.237
160	4	5	26.76	1	66	4.90	1	11	0.167
160	4	6	17.19	1	66	2.97	1	8	0.107
160	4	7	11.43	1	51	2.02	1	6	0.072
160	4	8	8.01	1	52	1.53	1	6	0.050
160	4	9	6.10	1	50	1.30	1	5	0.038
160	4	10	4.84	1	34	1.15	1	4	0.030
256	5	1	105.67	82	136	41.95	35	47	0.413
256	5	2	89.53	59	118	25.76	18	34	0.350
256	5	3	71.59	25	116	16.79	6	26	0.280
256	5	4	52.92	11	110	10.55	3	20	0.207
256	5	5	36.12	1	92	6.37	1	15	0.141
256	5	6	24.79	1	77	4.10	1	11	0.097
256	5	7	15.55	1	67	2.55	1	8	0.061
256	5	8	10.32	1	55	1.81	1	6	0.040
256	5	9	7.2	1	59	1.42	1	7	0.028
256	5	10	5.44	1	44	1.22	1	4	0.021

Table 3: Experiments for the modulus N suitable for DSA-like schemes.

$\log_2 N$	s	m_b	$\overline{\sigma}$	$\sigma_{\min}$	$\sigma_{\max}$	$\bar{l}$	$l_{\min}$	$l_{\max}$	$\overline{\sigma}/\log_2 N$
512	5	1	211.49	177	249	83.70	75	91	0.413
512	5	2	178.86	134	225	51.27	39	63	0.349
512	5	3	142.74	85	206	33.08	20	45	0.279
512	5	4	104.26	45	170	20.34	9	32	0.204
512	5	5	70.58	15	139	11.95	3	23	0.138
512	5	6	47.37	1	123	7.30	1	17	0.093
512	5	7	28.75	1	95	4.19	1	12	0.056
512	5	8	18.10	1	72	2.67	1	8	0.035
512	5	9	11.68	1	61	1.86	1	7	0.023
512	5	10	7.78	1	74	1.43	1	7	0.015
1024	6	1	366.33	321	414	145.02	135	154	0.358
1024	6	2	309.73	238	375	88.01	71	104	0.303
1024	6	3	246.83	164	334	56.53	38	72	0.241
1024	6	4	181.60	99	281	34.80	20	52	0.177
1024	6	5	125.28	40	226	20.74	8	35	0.122
1024	6	6	81.27	14	169	11.97	3	25	0.079
1024	6	7	52.72	1	152	7.20	1	17	0.052
1024	6	8	31.25	1	96	4.12	1	11	0.031
1024	6	9	19.37	1	85	2.63	1	9	0.019
1024	6	10	12.18	1	67	1.84	1	7	0.012
2048	6	1	733.67	670	821	289.75	276	303	0.358
2048	6	2	620.17	529	716	175.87	151	197	0.303
2048	6	3	493.07	372	617	112.53	87	137	0.241
2048	6	4	361.9	236	500	68.85	47	95	0.177
2048	6	5	248.8	119	381	40.66	20	60	0.122
2048	6	6	162.02	64	299	23.21	10	41	0.079
2048	6	7	104.24	18	223	13.59	3	29	0.051
2048	6	8	60.46	1	147	7.34	1	17	0.030
2048	6	9	35.95	1	129	4.27	1	12	0.017
2048	6	10	21.62	1	95	2.69	1	9	0.011

Table 4: Experiments for the modulus N suitable for RSA.

Appendix B

We present detailed statistics on the experiments with random instances of DSA-KDP converted to the instances of EHNP and solved by Algorithm 1. The size of DSA prime q is set to 160 bits to match its definition in FIPS 186 (Second revision). Each line in Tables 5 through 14 represents an experiment of 10 random DSA-KDP instances.

d	m_b	D	$\bar{\sigma}$	$\bar{t}_{LLL}$ $[s]$	$\bar{t}_{Babai}$ $[s]$	hit rate
5	1	160.9	369.8	13.16	0.65	0
6	1	193.9	442.6	23.30	1.03	0
7	1	226.5	512.7	38.23	1.60	1
8	1	255.4	596.2	56.44	2.17	0
9	1	287.9	672.2	73.52	3.06	1
10	1	323.7	734.0	109.24	4.31	0
11	1	353.2	812.6	138.70	5.58	2
12	1	388.1	883.2	181.11	7.09	1
13	1	417.0	966.5	229.56	8.73	3
14	1	449.4	1032.5	287.71	11.04	4
15	1	483.0	1110.7	376.98	13.78	3
16	1	513.9	1182.2	474.33	16.11	4
17	1	547.2	1253.6	570.28	18.84	7
18	1	576.9	1334.3	680.13	22.24	7
19	1	607.5	1418.8	815.89	26.55	7
20	1	641.7	1473.7	1001.00	30.97	4
21	1	672.8	1562.0	1118.30	34.51	5
22	1	706.6	1627.8	1335.44	40.35	7
23	1	742.7	1693.1	1519.61	46.72	6
24	1	769.3	1785.9	1781.56	51.43	8
25	1	798.8	1849.7	2033.85	57.84	8

Table 5: Experiments employing LLL reduction with $m_b = 1$.

d	m_b	D	$\bar{\sigma}$	$\bar{t}_{LLL}$ $[s]$	$\bar{t}_{Babai}$ $[s]$	hit rate
5	2	95.7	305.5	4.18	0.23	0
6	2	114.4	356.5	7.47	0.35	0
7	2	136.4	425.2	13.27	0.52	0
8	2	151.8	483.1	19.66	0.71	0
9	2	172.7	540.9	28.65	0.99	0
10	2	192.5	613.8	38.81	1.28	0
11	2	210.8	667.5	49.30	1.68	1
12	2	232.3	734.1	67.96	2.11	0
13	2	251.7	796.1	85.43	2.53	4
14	2	271.1	871.5	106.48	3.19	5
15	2	293.6	937.8	136.93	3.92	5
16	2	310.8	991.3	164.35	4.68	7
17	2	322.6	1032.0	194.43	5.53	5
18	2	343.7	1088.1	246.62	6.54	5
19	2	363.3	1169.3	271.32	7.43	7
20	2	382.2	1221.9	320.61	8.71	5

Table 6: Experiments employing LLL reduction with $m_b = 2$.

d	m_b	D	$\bar{\sigma}$	$\bar{t}_{LLL}$ $[s]$	$\bar{t}_{Babai}$ $[s]$	hit rate
5	3	64.4	236.3	2.32	0.09	0
6	3	77.4	284.5	3.68	0.16	0
7	3	91.0	339.2	6.06	0.26	0
8	3	106.4	398.2	9.27	0.39	0
9	3	116.7	426.7	13.85	0.50	0
10	3	131.8	486.0	19.39	0.69	0
11	3	143.3	534.1	25.39	0.84	1
12	3	159.9	594.7	35.39	1.13	1
13	3	169.9	630.4	44.79	1.35	5
14	3	180.6	672.8	55.97	1.59	3
15	3	199.8	747.9	68.49	2.03	2
16	3	212.3	790.3	87.09	2.50	3
17	3	224.2	835.5	99.50	2.86	4
18	3	239.3	895.0	122.87	3.42	5
19	3	249.9	946.4	141.24	3.88	5
20	3	254.7	950.4	154.30	4.44	7

Table 7: Experiments employing LLL reduction with $m_b = 3$.

d	m_b	D	$\bar{\sigma}$	$\bar{t}_{LLL}$ $[s]$	$\bar{t}_{Babai}$ $[s]$	hit rate
5	4	45.6	189.5	1.12	0.04	0
6	4	52.5	212.6	1.85	0.06	0
7	4	61.8	249.4	3.18	0.09	0
8	4	69.6	282.4	4.55	0.15	1
9	4	78.5	321.1	6.72	0.23	0
10	4	85.4	348.1	8.39	0.33	2
11	4	95.1	388.5	11.76	0.46	1
12	4	100.6	411.4	15.46	0.56	0
13	4	111.0	455.3	21.32	0.73	2
14	4	121.2	502.6	27.29	0.92	5
15	4	126.1	524.2	31.37	1.07	6
16	4	133.4	545.6	36.34	1.24	3
17	4	141.6	571.5	47.62	1.52	3
18	4	155.2	643.5	61.41	1.88	6
19	4	162.0	659.8	69.31	2.13	4
20	4	171.0	707.1	81.09	2.36	6
21	4	183.1	760.8	98.92	2.85	7
22	4	188.0	778.1	109.73	3.33	6
23	4	196.1	803.8	126.21	3.68	7
24	4	206.3	852.8	146.41	4.34	6
25	4	212.9	885.8	163.96	4.63	7

Table 8: Experiments employing LLL reduction with $m_b = 4$.

d	m_b	D	$\bar{\sigma}$	$\bar{t}_{LLL}$ $[s]$	$\bar{t}_{Babai}$ $[s]$	hit rate
5	5	27.0	105.1	0.23	0.01	0
6	5	32.4	130.8	0.45	0.01	0
7	5	39.8	164.1	0.97	0.03	0
8	5	41.7	167.7	1.21	0.04	0
9	5	50.0	208.4	2.25	0.05	0
10	5	55.6	227.9	3.49	0.08	0
11	5	63.2	267.0	5.21	0.12	1
12	5	66.5	276.3	6.68	0.15	1
13	5	70.7	289.8	8.22	0.21	1
14	5	77.4	327.2	10.71	0.35	5
15	5	85.3	361.0	14.96	0.60	3
16	5	92.1	392.6	18.66	0.74	3
17	5	93.3	394.2	20.33	0.86	5
18	5	99.7	424.1	26.54	1.09	4
19	5	105.2	439.3	31.80	1.25	5
20	5	109.5	468.5	33.75	1.37	6
21	5	117.3	499.9	46.84	1.72	7
22	5	122.1	509.3	51.16	1.92	8
23	5	124.0	518.6	57.40	2.11	6
24	5	132.5	559.7	68.65	2.55	9
25	5	136.8	577.8	80.55	2.76	5

Table 9: Experiments employing LLL reduction with $m_b = 5$.

d	m_b	D	$\bar{\sigma}$	$\bar{t}_{LLL}$ $[s]$	$\bar{t}_{Babai}$ $[s]$	$\bar{t}_{BKZ}$ $[s]$	hit rate
5	1	160.9	374.9	13.30	0.66	228.59	0 (0+0)
6	1	196.4	438.4	23.92	1.02	739.48	0 (0+0)
7	1	224.8	517.2	37.60	1.51	1790.52	1 (0+1)
8	1	256.8	591.3	54.22	2.08	758.57	1 (0+1)
9	1	290.1	662.9	76.88	3.00	1521.70	5 (0+5)
10	1	322.8	732.6	105.73	4.20	3014.15	4 (2+2)
11	1	352.6	816.5	139.85	5.38	6075.52	7 (2+5)
12	1	387.2	883.2	184.82	6.89	12909.33	8 (1+7)
13	1	415.7	967.6	230.94	8.77	6971.49	8 (3+5)
14	1	448.1	1032.4	293.67	10.99	14970.03	9 (4+5)
15	1	482.5	1111.2	376.10	12.74	7377.91	9 (5+4)

Table 10: Experiments employing BKZ reduction with $m_b = 1$.

d	m_b	D	$\bar{\sigma}$	$\bar{t}_{LLL}$ $[s]$	$\bar{t}_{Babai}$ $[s]$	$\bar{t}_{BKZ}$ $[s]$	hit rate
5	2	96.9	306.1	4.61	0.23	84.62	0 (0+0)
6	2	116.6	370.1	7.65	0.36	653.17	0 (0+0)
7	2	138.1	431.6	13.21	0.55	499.35	3 (0+3)
8	2	155.8	490.2	19.29	0.78	548.27	4 (1+3)
9	2	174.7	564.9	25.88	0.98	483.71	3 (0+3)
10	2	191.8	608.0	36.79	1.27	2596.34	3 (0+3)
11	2	213.7	676.8	47.83	1.64	2946.76	6 (3+3)
12	2	233.4	730.0	64.55	2.19	5908.30	5 (0+5)
13	2	248.3	792.7	81.82	2.48	3179.12	9 (4+5)
14	2	271.8	853.7	113.02	3.25	12069.42	8 (2+6)
15	2	290.7	916.9	136.79	4.03	3885.29	9 (3+6)

Table 11: Experiments employing BKZ reduction with $m_b = 2$.

d	m_b	D	$\bar{\sigma}$	$\bar{t}_{LLL}$ $[s]$	$\bar{t}_{Babai}$ $[s]$	$\bar{t}_{BKZ}$ $[s]$	hit rate
5	3	67.6	246.3	2.53	0.11	142.54	0 (0+0)
6	3	76.9	285.1	3.64	0.16	224.96	0 (0+0)
7	3	90.5	336.7	5.78	0.25	221.19	0 (0+0)
8	3	102.8	383.3	8.62	0.36	174.97	4 (0+4)
9	3	121.3	449.7	13.88	0.55	569.09	3 (1+2)
10	3	128.6	476.8	17.66	0.67	964.61	3 (0+3)
11	3	138.8	516.4	23.02	0.78	1315.60	4 (1+3)
12	3	155.1	590.7	32.67	1.02	2438.11	7 (3+4)
13	3	171.2	638.1	43.35	1.33	2162.14	10 (4+6)
14	3	183.8	693.9	57.37	1.73	3470.40	7 (3+4)
15	3	199.1	739.2	70.07	1.98	2672.28	7 (2+5)

Table 12: Experiments employing BKZ reduction with $m_b = 3$.

d	m_b	D	$\bar{\sigma}$	$\bar{t}_{LLL}$ $[s]$	$\bar{t}_{Babai}$ $[s]$	$\bar{t}_{BKZ}$ $[s]$	hit rate
5	4	44.1	181.7	1.04	0.03	1.31	0 (0+0)
6	4	53.4	212.6	1.99	0.06	7.48	0 (0+0)
7	4	61.9	249.8	3.24	0.09	34.71	1 (0+1)
8	4	71.8	299.8	4.77	0.15	125.85	4 (0+4)
9	4	78.5	317.9	6.47	0.23	404.78	1 (0+1)
10	4	87.3	360.1	9.00	0.36	666.69	6 (1+5)
11	4	95.2	396.0	11.32	0.44	1042.91	6 (1+5)
12	4	106.6	441.5	17.32	0.61	2169.34	7 (1+6)
13	4	111.8	465.0	20.13	0.72	1607.15	8 (5+3)
14	4	114.5	466.5	23.05	0.87	1214.87	6 (3+3)
15	4	126.1	518.4	31.27	1.05	1696.46	8 (5+3)
16	4	134.8	554.4	39.14	1.29	942.24	9 (6+3)
17	4	144.7	595.8	47.64	1.58	1386.65	10 (5+5)
18	4	155.6	651.9	60.35	1.94	1520.36	9 (4+5)
19	4	165.8	680.6	73.52	2.18	2417.96	9 (3+6)
20	4	170.8	697.0	80.26	2.59	3777.19	10 (4+6)

Table 13: Experiments employing BKZ reduction with $m_b = 4$.

d	m_b	D	$\bar{\sigma}$	$\bar{t}_{LLL}$ $[s]$	$\bar{t}_{Babai}$ $[s]$	$\bar{t}_{BKZ}$ $[s]$	hit rate
5	5	28.1	118.5	0.24	0.01	0.03	0 (0+0)
6	5	33.1	135.3	0.47	0.02	0.09	0 (0+0)
7	5	41.5	175.0	1.07	0.03	0.74	0 (0+0)
8	5	47.5	201.0	1.76	0.05	1.70	1 (0+1)
9	5	52.4	225.6	2.56	0.06	3.12	1 (0+1)
10	5	58.4	250.0	3.90	0.09	12.18	3 (2+1)
11	5	61.9	254.9	4.96	0.11	13.72	2 (1+1)
12	5	67.9	283.1	7.01	0.16	23.44	3 (2+1)
13	5	71.6	299.0	8.62	0.23	43.63	6 (3+3)
14	5	78.0	325.0	10.72	0.39	162.86	7 (3+4)
15	5	83.6	349.1	14.32	0.58	143.79	7 (4+3)
16	5	90.5	379.5	18.00	0.75	480.48	9 (2+7)
17	5	95.6	405.2	22.22	0.89	465.93	9 (4+5)
18	5	97.3	405.4	25.19	1.06	703.87	10 (5+5)
19	5	101.6	420.3	29.70	1.21	399.98	8 (5+3)
20	5	108.8	455.8	37.03	1.41	1275.56	9 (6+3)

Table 14: Experiments employing BKZ reduction with $m_b = 5$.

Appendix C

We present detailed statistics on the experiments with random instances of DSA-KDP converted to the instances of EHNP and solved by Algorithm 1. The size of DSA prime q is set to 256 bits to match its definition in the draft of FIPS 186 (Third revision). Each line in Table 15 represents an experiment of 10 random DSA-KDP instances.

d	m_b	D	$\bar{\sigma}$	$\bar{t}_{LLL}$ $[s]$	$\bar{t}_{Babai}$ $[s]$	hit rate
20	5	148.4	700.7	267.59	5.85	0
21	5	151.3	706.9	309.31	6.69	1
22	5	163.9	764.9	372.54	7.70	1
23	5	162.6	752.1	402.33	8.53	1
24	5	174.4	813.1	463.96	9.41	1
25	5	181.1	854.6	541.15	11.42	3
26	5	191.9	899.0	654.09	12.62	0
27	5	199.8	933.4	751.25	14.90	1
28	5	204.5	958.4	805.70	16.12	0
29	5	205.2	950.3	876.88	17.85	0
30	5	219.4	1038.3	1010.06	19.89	1
31	5	228.0	1071.7	1189.26	22.82	2
32	5	235.8	1111.1	1293.63	24.35	2
33	5	235.8	1110.9	1398.55	27.08	0
34	5	251.0	1173.8	1666.97	31.49	1
35	5	252.2	1174.8	1636.32	30.41	1
36	5	260.9	1229.4	1813.62	34.12	3
37	5	266.4	1231.0	2035.25	36.14	2
38	5	280.8	1320.7	2265.42	38.02	3
39	5	281.0	1307.4	2491.76	42.95	2
40	5	286.4	1337.7	2640.81	44.92	4
41	5	297.2	1397.1	2839.71	46.08	3
42	5	307.5	1430.1	3317.86	55.31	4
43	5	307.8	1424.5	3450.49	56.51	1
44	5	330.0	1553.6	3977.16	60.88	4
45	5	329.7	1545.1	4104.91	65.11	6

Table 15: Experiments employing LLL reduction with $m_b = 1$ $\lceil \log_2 q \rceil = 256$ bits.