Attacking (EC)DSA Given Only an Implicit Hint

Jean-Charles Faugère[1⋆], Christopher Goyet[1,2⋆] and Guénaël Renault[1⋆]

[1] UPMC, Université Paris 6, LIP6
INRIA, Centre Paris-Rocquencourt, PolSys Project-team
CNRS, UMR 7606, LIP6
4, place Jussieu
75252 Paris, Cedex 5, France
[2] Thales Communications & Security
160 Boulevard de Valmy
92700 Colombes, France
jean-charles.faugere@inria.fr, christopher.goyet@thalesgroup.com, guenael.renault@lip6.fr

Abstract. We describe a lattice attack on DSA-like signature schemes under the assumption that implicit information on the ephemeral keys is known. Inspired by the implicit oracle of May and Ritzenhofen presented in the context of RSA (PKC2009), we assume that the ephemeral keys share a certain amount of bits without knowing the value of the shared bits. This work also extends results of Leadbitter, Page and Smart (CHES2004) which use a very similar type of partial information leakage. By eliminating the shared blocks of bits between the ephemeral keys, we provide lattices of small dimension (e.g. equal to the number of signatures) and thus obtain an efficient attack. More precisely, by using the LLL algorithm, the complexity of the attack is polynomial. We show that this method can work when ephemeral keys share certain amount of MSBs and/or LSBs, as well as contiguous blocks of shared bits in the middle. Under the Gaussian heuristic assumption, theoretical bounds on the number of shared bits in function of the number of signed messages are proven. Experimental results show that we are often able to go a few bits beyond the theoretical bound. For instance, if only 2 shared LSBs on each ephemeral keys of 200 signed messages (with no knowledge about the secret key) then the attack reveals the secret key. The success rate of this attack is about 90% when only 1 LSB is shared on each ephemeral keys associated with about 400 signed messages.

Keywords: DLP, ECDSA, Lattice attack, Oracle, Implicit information

1 Introduction

The security of the main public-key cryptosystems is based on the difficulty of solving certain mathematical problems. In this context, the most commonly used problems come from Number Theory, most notably the integer factorization problem and the discrete logarithm on finite cyclic groups. For instance, an efficient factorization leads immediately to an attack on the RSA cryptosystem. The security of RSA is then partly based on the presumed difficulty of factoring large integers. Indeed, the most efficient published factoring algorithms have sub exponential asymptotic running times ([Pom84,Len87,LL93]) and it is not known whether efficient factorization can be done in polynomial time on a classical Turing machine. Another classical example is the discrete logarithm problem on a finite cyclic group, upon which is based the security of the ElGamal encryption system, the Diffie-Hellman key exchange and the DSA-like signature schemes. Despite the proven fact that a generic algorithm for computing discrete logarithms in any group is necessarily an exponential algorithm ([Tes01,Sho97,Sha71,Pol78,Pol00]), once again, subexponential algorithms are known to solve some instances of the discrete logarithm problem, i.e. on well-defined classes of groups. For instance, discrete logarithms can be computed in subexponential time on the multiplicative group of finite fields ([AD93]) or on some hyperelliptic curves ([ADH94,ADH99]). Conversely, there is no known subexponential algorithm to solve the discrete logarithm problem on the group of rational points of a well-chosen elliptic curve.

⋆ This work was partly supported by the Commission of the European Communities through the ICT program under contract ICT-2007-216676 (ECRYPT-II) and by the French ANR under the Computer Algebra and Cryptography (CAC) project (ANR-09-JCJCJ-0064-01).

Instead of trying to solve directly a hard mathematical problem, we can rather look at which information should be added in order to solve this problem in polynomial time. With this objective in mind, Rivest and Shamir introduced in [RS86] the notion of oracle to formalize this approach and showed that a RSA modulus $N = pq$ of bit size n (with p and q balanced prime factors of n) can be factored in polynomial time as soon as the $n/3$ most significant bits (MSB) from one of the factors are known. This result was next improved with the so-called Coppersmith's method based on lattice basis reduction as introduced in [Cop96], and which reduced the number of needed bits known to only one-half of the MSB of one of the factors. Beyond the theoretical interest, this additional information can be provided for instance with the help of side-channel analysis and the discovery of some leaks of secret information in some cryptographic systems, and brought to the attacker under the form of an oracle.

In this article, we focus on the Digital Signature Algorithm (DSA) [FIP94] of which the security is based on the difficulty of computing discrete logarithms. The Elliptic Curve Digital Signature Algorithm (ECDSA) [JMV01] is the elliptic curve variant of DSA. The ElGamal [ElG85] and the Schnorr [Sch90] digital signature schemes are also variants of DSA but are rarely used in practice. Anyway, we note that all results presented in this article could be applied to any of these variants. Without redefining the DSA-like schemes (see section 2 for details), we only recall that each user is associated with a pair of private key/public key, such that the private key is the discrete logarithm in a given group of the public key. The user private key and a randomly generated number, called the ephemeral key, are required to compute the signature of a message. The ephemeral key must remain secret and is to be renewed for any new message to be signed.

The first proposal of using an oracle on DSA comes from Howgrave-Graham and Smart in [HGS01] using the LLL lattice reduction algorithm ([LLL82]) to take benefit from the knowledge of a small number of bits in many ephemeral keys. For instance, they show experimentally that if only 8 bits out of 160 bits are known from each ephemeral keys for 30 signed messages, then the secret key is known in less than 10 seconds. However, these results were only heuristics, even though confirmed by experimentation. Nguyen and Shparlinski then presented in [NS02] the first polynomial time algorithm that provably recovers the secret DSA key if about $\log^{1/2}(q)$ LSB (or MSB) of each ephemeral key are known (q denoting the order of the chosen group, see section 2) for a polynomially bounded number of corresponding signed messages. They also show that the case of arbitrary consecutive bits requires much more known bits (about twice as much). Finally, in addition of proving the heuristic attack of [HGS01], Nguyen and Shparlinski ([NS02]) also improved the experimental results of [HGS01] by showing that only 3 known bits of each ephemeral key for 100 signed messages are enough to make the attack feasible. The previous attack is adapted to the case of ECDSA [NS03] and other DSA-like signature schemes like the Nyberg-Rueppel variants of DSA [MNS01]. It is also necessary to mention another analysis that shows the threats associated with the use of private keys generated from an imperfect source of randomness. The attack of Bellare, Goldwasser and Micciancio [BGM97] shows that DSA is totally insecure if private keys are produced by weak pseudo-random number generator such as the Knuth's linear congruential generator. When private keys are smaller than a certain bound, Poulakis proposed in [Pou09] an attack on (EC)DSA using the LLL algorithm and an algorithm to compute the integral points of a class of conics.

Note that unlike the case of RSA, where the oracle gives a way to directly compute the factors of the modulus, all these methods against the DSA-Like cryptosystems bypass the problem of computing discrete logarithm, but rather take advantage of the particular form of the modular equality defining the signature (see (1) in section 2).

At PKC 2009 [MR09], May and Ritzenhofen, in the context of factorization, highly restricted the power of the oracle. They did not assume that the oracle explicitly outputs bits but rather provides only implicit information. This unusual oracle applied against the cryptosystem RSA was formalized as follows: given an RSA modulus $N_1 = p_1q_1$ as input, the oracle outputs a different RSA modulus $N_2 = p_2q_2$ such that the factors of N_2 shared a certain amount of bits with the factors of the modulus N_1. The implicit nature of the information given by the oracle is due to the fact that the value of the shared bits remains unknown as long as the modulus N_1 or N_2 are not factored. Surprisingly, May and Ritzenhofen give an efficient lattice-based algorithm that provably factors N_1 and N_2 in quadratic time provided that factors of the two moduli shared enough of their least significant bits (LSB). They also showed that this algorithm extends to an algorithm with more than one oracle query, which improves upon the required number of shared LSB. This cryptanalysis with the help of an implicit oracle was next extended to the case of shared MSB (and both LSB/MSB) in [FMR10,SM09] and the bound on the required number of shared bits was also improved.

In the case of DSA, an attack using implicit information of a totally different kind was already proposed by Leadbitter, Page and Smart in [LPS04] and made effective in [Tak06a,Tak06b]. From a theoretical point of view, this

attack can also be formalized by queries to an oracle which returns a message signed with an ephemeral key of the form $k = y + 2^w y + 2^{2w} x$, i.e. such that the w first bits of k are equal to the w next bits. Experiments show that the repetition of a 4-bits window in the ephemeral keys of 20 signatures is enough to recover the secret key. This attack is motivated by side-channel analysis. According to the authors, recovering some relation amongst the bits of the secret keys (called "second order leakage") is much more probable than determining the values of such bits because of implementation protections against side-channel analysis. They also described many realistic scenarii where this type of leakage could occur.

Inner product	Canonical		Weighted Euclidean	
Lattice spanned by	the basis M (6) of section 4.1	linearly dependent rows vectors of the matrix (8) obtained by removing the second column of M	the basis M (6) of section 4.1	linearly dependent rows vectors of the matrix (8) obtained by removing the second column of M
constraints on the secret key $\mathbf{a}$	$\mathbf{a} \leq 2^{N-\delta}$ or exhaustive search on the δ msb (section 4.2.1)	No constraint ($\mathbf{a}$ can be up to N bits)	$\mathbf{a} \leq 2^{N-\delta}$ or exhaustive search on the δ msb (section 4.2.1)	No constraint ($\mathbf{a}$ can be up to N bits)
Bounds on the number of shared bits δ function of the number of messages n	$\delta \geq \frac{2N+(n-1)}{n+1} + \frac{c-\log_2(\frac{n+1}{n})}{2}$ (Theorem 1)	$\delta \geq \frac{2N+(n-2)}{n} + \frac{c-\log_2(\frac{n}{n-1})}{2}$ (Theorem 2)	$\delta \geq \frac{N+(n-1)}{n} + \frac{c(n+1)}{2n}$ (Theorem 3a)	$\delta \geq \frac{N+(n-2)}{n-1} + \frac{cn}{2(n-1)}$ (Theorem 3b)

Table 1. Summary of our results

Our contribution is to define an attack on DSA-like schemes using implicit information, like in [LPS04], but with an oracle very similar to the one introduced by May and Ritzenhofen ([MR09]). More precisely, we assume that on input of a signed message (m_1, s_1) an oracle outputs different signed message (m_2, s_2) signed with the same secret key and such that the ephemeral keys k_1 and k_2 (used to signed the messages m_1 and m_2 respectively) share a certain amount δ of bits. This oracle only gives implicit information about the bits of the ephemeral keys because the value of the shared bits remains unknown as long as the ephemeral keys stay unknown (or equivalently as long as the secret key stays unknown). In other words, we only know that there are equalities between δ bits of the unknown ephemeral keys used to sign the given messages. We show that this implicit information should be extracted by constructing a lattice which contains a very short vector such that its components yield the secret key. The attack succeeds when this vector is found by the LLL lattice reduction algorithm ([LLL82]), that is when it is small enough. This happens when the ephemeral keys share enough bits δ. This method also works for an arbitrary number n of oracle's queries, each new piece of information decreasing the number of required shared bits.

As usual in lattice basis reduction problems, we have to use the Gaussian heuristic to find a condition on the number of shared bits δ in function of the number of messages n for this vector to be the shortest of the lattice. This condition can be improved by the use of a weighted Euclidean inner product instead of the canonical inner product during the reduction algorithm. A variant of this lattice is also proposed so that the complexity of the attack becomes independent of the secret key size and is polynomial time in n (assuming that the bound on δ is verified). In this case, the lattice is spanned by a set of linearly dependent vectors and the condition on δ is slightly deteriorated. A summary of our method and the proposed improvements can be found in table 1.

As an example of our results, the theorem 3 proves that under the Gaussian heuristic assumption, only 4 LSBs shared on each ephemeral keys of 100 signed messages are enough to make a never-failing attack and that with only 3 LSBs shared, the method needs about 200 signed messages. The result of experiments confirms these theoretical values with a computation time less than 5s, and they even show that the number of messages can be most of the time reduced to an amount comparable to the one which is described in [NS02] despite of the weakness of the oracle. However, these experiments also showed that the success rate of this attack is about 90% when only 1 LSB is shared on each ephemeral key of about 400 signed messages. Interestingly enough, this improves the experimental results of [NS02] where the best experiment corresponds to 3 LSB known (even though LSB are not even known in our contribution).

Throughout this paper, we use common results on euclidean lattice summarized in Appendix A. Section 2 recall the (EC)DSA signature algorithm. Section 3 provides a concrete scenario to justify the existence of this implicit information. In section 4, we present our method by beginning with the case of shared MSB and LSB (subsection 4.1). Essential improvements are proposed in subsection 4.2. In subsection 4.3, we present our method when they are many blocks of shared bits. Finally, we present the result of our experiments in section 5.

2 DSA-Style Signature Scheme

The Digital Signature Algorithm was adopted in 1993 by the U.S. government's National Institute of Standards and Technology (NIST) to become the Digital Signature Standard (DSS) [FIP09]. It is much more used than ElGamal [ElG85] and Schnorr [Sch90] digital signature schemes which are variants of DSA. Thus, we focus on DSA although our attack is transferable to others.

The (EC)DSA algorithm ([FIP09,JMV01]) is defined over a finite abelian group G of prime order q. The group G is chosen as a subgroup of $\mathbb{F}^\times$ (resp. $E(\mathbb{F})$) for DSA (resp. ECDSA) where $\mathbb{F}$ is a finite field (resp. $E(\mathbb{F})$ the group of rational points of an elliptic curve defined over $\mathbb{F}$). For security reason, the size of q is chosen to be at least 160 bits. More precisely, the last revision of the standard ([FIP09,JMV01]) specifies that the parameter q must verify $2^{N-1} < q < 2^N$ where $N \in \{160, 224, 256\}$. The private key is an integer $\mathbf{a} \in \{1, \ldots, q-1\}$ and the public key is the group element $A = g^{\mathbf{a}}$ where g is a publicly known generator of G. Let the function $f : G \longrightarrow \mathbb{F}_q$ be defined by

$$f : \begin{cases} x \in G \subset \mathbb{F}_p^\times & \longmapsto x \bmod q \text{ for DSA} \\ (x,y) \in G \subset E(\mathbb{Z}_p) & \longmapsto x \bmod q \text{ for ECDSA} \end{cases}$$

The signer chooses an hash function h mapping messages to G. To sign a message m, he chooses a random number $\mathbf{k} \in \{1, \ldots, q-1\}$ called the *ephemeral key* and computes (here we present the computation in the case of DSA only)

$$r = f(g^{\mathbf{k}}) \quad \text{and} \quad s = \mathbf{k}^{-1}(h(m) + \mathbf{a}r) \bmod q \tag{1}$$

The signature on the message m is then the pair (r,s). The verification of the signature is performed by checking

$$f(g^{s^{-1}h(m) \bmod q} \quad A^{s^{-1}r \bmod q}) = f(g^{s^{-1}(h(m)+\mathbf{a}r) \bmod q}) = f(g^{\mathbf{k}}) = r$$

3 Possible application scenario

As stated in [LPS04] in reference to side-channel analysis, "the assumption that an attacker may be able to determine a specific set of bits from the ephemeral secrets is less probable than when the original attacks were first published. It is far more probable that second order, seemingly innocuous information can still be recovered and used by the attacker, even if a defense against the first order leakage is implemented". Indeed, there are always many situations where implicit information can be found despite the implementation of typically recommended countermeasures. In this paper, as in [LPS04], the attacker is only assumed to be able to determine relations of equality between bits of the ephemeral keys. In addition to the three scenarii given as examples in [LPS04] where implicit information is collected by a power analysis or a timing attack (involving a fixed table implementation of elliptic curve point multiplication, address-bit DPA and cache analysis), we suggest other scenarii which are specifically relevant to our model.

Using invasive attacks, an attacker could lock some bits of the register or memory containing the ephemeral key. This kind of attack largely depends on the implementation and requires a good knowledge of the target, which presupposes at least a partial reverse engineering of the chip. Lasers are then used to cut some wires or to modify the chip (see Skorobogatov thesis [Sko05]). More generally, a lot of fault attacks assume that some bits of the memory are flipped to zero (as in [NNTW05]). But it seems more general to assume that the attacked bits take an indeterminate value.

In addition to weak and wrong implementations (e.g. [GJQ97]), we could also think to destructive applications with a malicious manipulation of random generators for instance. This application could be possible on both embedded systems and software implementation, for instance with physical disturbances, invasive attacks or with malicious softwares. Moreover, the presence of a random number generator testing suite ([Bro11,RSN+10]) does not seem to be an effective countermeasure (see experimental results in section 5.3).

4 Embedding into a Lattice Problem

In this section, we study the security of (EC)DSA given a set of messages signed with the same secret key, and such that the secret ephemeral keys share a certain amount of bits. We recall that the values of these common bits are unknown to us. Thus, the information about the unknown ephemeral keys are implicitly given by the set of signed messages. In other words, we only know that there are some relations amongst the bits of the ephemeral keys used to sign the messages. To ease the exposition of this method, the bit length of the modulus q is noted by N (i.e. we have $2^{N-1} < q < 2^N$).

We will show how the secret key can be revealed by lattice basis reduction provided that there are enough messages or relations between the bits of the ephemeral keys. These constraints are estimated by relating our method to the Gaussian heuristic (see theorem 6 of Appendix A). Since all the complexities given in this paper depend on the complexity of computing a shortest vector in a given lattice, we set the following notation.

Notation 1 *The time complexity of computing a shortest vector of a d-dimensional lattice L of $\mathbb{Z}^n$ will be denoted by $\mathscr{C}(d,B)$, where $B = \log \max_i(\|\mathbf{b_i}\|)$. Notice that computing a shortest vector of any lattice is a NP-hard problem ([Ajt98]) called the Shortest Vector Problem (SVP).*

For certain families of lattices (i.e. under certain conditions on lattices), the shortest vector can be computed with the LLL Algorithm. In this case, we have that $\mathscr{C}(d,B) = \mathscr{O}(d^5(d+B)B)$, i.e. polynomial time in d and B ([NS05], see Appendix A). In this paper, we seek to always stay in this situation.

We first present the case when most significant bits (MSB) and/or less significant bits (LSB) are shared and next the case when blocks of bits are shared.

4.1 Shared MSB and LSB

We first assume that we have n messages m_i $(i = 1,\ldots,n)$ with the associated signatures (r_i,s_i) such that all the corresponding ephemeral keys $\mathbf{k}_i$ share a total of δ bits between the MSB and LSB independently of i (see Figure 1). Thus, they are of the form

$$\mathbf{k}_i = \mathbf{k} + 2^t \tilde{\mathbf{k}}_i + 2^{t'} \mathbf{k}' \quad \text{for all } i = 1,\ldots,n \tag{2}$$

where

$$0 \leq \mathbf{k} < 2^t, \quad 0 \leq \mathbf{k}' < 2^{N-t'}, \quad \delta = N - t' + t \quad \text{and} \quad 0 \leq \tilde{\mathbf{k}}_i < 2^{N-\delta}$$

with $\mathbf{k}$ and $\mathbf{k}'$ common for all the $\mathbf{k}_i$ (i.e. independent of i).

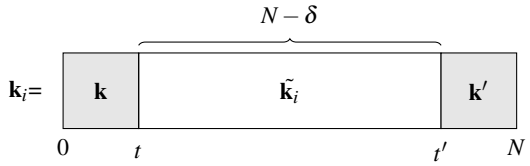

Fig. 1. Ephemeral keys

Note that all the values of $\mathbf{k}_i$, $\mathbf{k}$, $\tilde{\mathbf{k}}_i$ and $\mathbf{k}'$ are unknown. In the n equations (1) defining the signature

$$\begin{cases} m_1 + \mathbf{a} r_1 - s_1 \mathbf{k}_1 \equiv 0 \pmod q \\ m_2 + \mathbf{a} r_2 - s_2 \mathbf{k}_2 \equiv 0 \pmod q \\ \quad \vdots \quad \vdots \quad \vdots \quad \vdots \\ m_n + \mathbf{a} r_n - s_n \mathbf{k}_n \equiv 0 \pmod q \end{cases} \tag{3}$$

we substitute the $\mathbf{k}_i$ by (2) and eliminate the common variables $\mathbf{k}$ and $\mathbf{k}'$. Then we have

$$\begin{cases} (s_1^{-1}m_1 - s_2^{-1}m_2) + \mathbf{a}(s_1^{-1}r_1 - s_2^{-1}r_2) - 2^t(\tilde{\mathbf{k}_1} - \tilde{\mathbf{k}_2}) \equiv 0 \pmod q \\ (s_1^{-1}m_1 - s_3^{-1}m_3) + \mathbf{a}(s_1^{-1}r_1 - s_3^{-1}r_3) - 2^t(\tilde{\mathbf{k}_1} - \tilde{\mathbf{k}_3}) \equiv 0 \pmod q \\ \quad \vdots \qquad\qquad \vdots \qquad\qquad \vdots \qquad \vdots \\ (s_1^{-1}m_1 - s_n^{-1}m_n) + \mathbf{a}(s_1^{-1}r_1 - s_n^{-1}r_n) - 2^t(\tilde{\mathbf{k}_1} - \tilde{\mathbf{k}_n}) \equiv 0 \pmod q \end{cases} \tag{4}$$

Let α_i, β_i, $\kappa_i \in \mathbb{Z}$ be such that

$$\begin{cases} \alpha_i := 2^{-t}(s_1^{-1}m_1 - s_i^{-1}m_i) \mod q \\ \beta_i := 2^{-t}(s_1^{-1}r_1 - s_i^{-1}r_i) \mod q \\ \kappa_i := \tilde{\mathbf{k}_1} - \tilde{\mathbf{k}_i} \end{cases}$$

then (4) becomes

$$\begin{cases} \alpha_2 + \mathbf{a}\beta_2 - \kappa_2 \equiv 0 \pmod q \\ \alpha_3 + \mathbf{a}\beta_3 - \kappa_3 \equiv 0 \pmod q \\ \quad \vdots \quad \vdots \quad \vdots \quad \vdots \\ \alpha_n + \mathbf{a}\beta_n - \kappa_n \equiv 0 \pmod q \end{cases} \tag{5}$$

where $\mathbf{a}$ and κ_i are unknown, β_i and α_i are known. The set of solutions

$$L = \{(x_0, x_1, \ldots, x_n) \in \mathbb{Z}^{n+1} | x_0\alpha_i + x_1\beta_i - x_i \equiv 0 \pmod q \text{ for all } i = 2, \ldots, n\}$$

forms an $(n+1)$-dimensional lattice spanned by the row vectors of the following basis matrix

$$M = \begin{pmatrix} 1 & 0 & \alpha_2 & \ldots & \alpha_n \\ 0 & 1 & \beta_2 & \ldots & \beta_n \\ 0 & 0 & q & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & q \end{pmatrix} \tag{6}$$

Note that $v_0 = (1, \mathbf{a}, \kappa_2, \kappa_3, \ldots, \kappa_n)$ is an element of the lattice L. Indeed by (5), there are $\lambda_2, \ldots, \lambda_n \in \mathbb{Z}$ such that

$$(1, \mathbf{a}, \lambda_2, \ldots, \lambda_n) \cdot M = v_0 \tag{7}$$

If we were able to find this vector v_0 in L, then we could recover the secret key $\mathbf{a}$. Thus, we would like to give some conditions so that the vector v_0 be a short vector in L, and therefore may be obtained by lattice basis reduction.

However, it is easy to see that the norm of v_0 is lower bounded by the secret key $\mathbf{a}$, which can be an integer of roughly N bits. The second component of v_0 is then much bigger than the next ones which are $(N-\delta)$-bits integers. Actually, v_0 has no reason to be a short vector of L while $\mathbf{a}$ is so high. Therefore we will first assume that the secret key $\mathbf{a}$ is smaller than $2^{N-\delta}$, before adapting the lattice to be able to find the secret keys up to N-bit size.

This temporary assumption makes v_0 short in the lattice L, but we are still unable to prove that it is the shortest vector of L. Therefore, the Gaussian heuristic (see Appendix A), which is usually applied in this situation, gives us a way to estimate the required number δ of shared bits in function of the number of available messages so that v_0 is likely to be the shortest vector of L.

Assumption 1 *The Gaussian heuristic (theorem 6 of Appendix A) holds with the lattice L. Thus, if v_0 is shorter than the Gaussian heuristic $\lambda_1(L) \approx \sqrt{\frac{d}{2\pi e}}\,\mathrm{Vol}(L)^{\frac{1}{d}}$ then it is a shortest vector of L.*

Experiments of the section 5 confirm this assumption which seems to be true in practice with the lattices used in our method.

Theorem 1. *Let n messages m_i $(i = 1,\ldots,n)$ with the associated signatures (r_i, s_i) such that the ephemeral keys $\mathbf{k}_i$ share a total of δ bits between the MSB and LSB. Under assumption 1 and, under the assumption that the secret key $\mathbf{a}$ is smaller than $2^{N-\delta}$, then $\mathbf{a}$ can be computed in time $\mathscr{C}(n+1, \frac{1}{2}\log_2(n-1)+N)$ as soon as*

$$\delta \geq \frac{2N+(n-1)}{n+1} + \frac{1+\log_2(\pi e)-\log_2(\frac{n+1}{n})}{2}$$

Proof. First of all, we find an upper-bound for the norm of v_0. Under our assumptions, each coefficient of v_0 is an integer of about $(N-\delta)$ bits (except the first one which is equal to 1). Thus, we have the following inequality:

$$\|v_0\|^2 \leq \sum_{i=1}^{n} 2^{2(N-\delta)} = 2^{2(N-\delta)+\log_2 n}$$

On the other hand, thanks to the upper-triangular shape of the matrix M, the volume of L is easily computed as $\mathrm{Vol}(L) = q^{(n-1)} > 2^{(N-1)(n-1)}$. We now seek the condition on δ and n under which the norm of v_0 is smaller than the Gaussian heuristic:

$$2^{2(N-\delta)+\log_2(n)} \leq \frac{n+1}{2\pi e} 2^{2(N-1)\frac{n-1}{n+1}}$$

which is equivalent to

$$\delta \geq \frac{2N+(n-1)}{n+1} + \frac{1+\log_2(\pi e)-\log_2(\frac{n+1}{n})}{2}$$

□

4.2 Proposal for improvements

Until now, we made the assumption that $\mathbf{a} \leq 2^{N-\delta}$ to simplify the presentation. Actually, this assumption is rarely verified by the secret key, so we have to adapt the previous attack to be able to find the secret key up to N-bit long. We suggest three ways, which may be concurrent to each other, to reach this goal.

4.2.1 Exhaustive search: If δ is reasonably small then the method comes down to the previous one with an exhaustive search on the δ-most significant bits of $\mathbf{a}$. Indeed, we have $\mathbf{a} = \tilde{\mathbf{a}} + 2^{N-\delta}a'$ with $\tilde{\mathbf{a}} < 2^{N-\delta}$ and $a' < 2^{\delta}$. An exhaustive search is then made on a' with the previous lattice L in which we set $\alpha_i = 2^{-t}(s_1^{-1}m_1 - s_i^{-1}m_i) + a'2^{-t}(s_1^{-1}r_1 - s_i^{-1}r_i)$. The bound given by the theorem 1 remains true with this method.

4.2.2 Remove the second column: There is an other way to make the previous attack independent of the size of the secret key, but this trick needs a slight modification of the LLL Reduction Algorithm ([Poh87]). Let the lattice L' spanned by the row vectors of the following matrix

$$M' = \begin{pmatrix} 1 & \alpha_2 & \ldots & \alpha_n \\ 0 & \beta_2 & \ldots & \beta_n \\ 0 & q & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & q \end{pmatrix}, \quad n > 2 \tag{8}$$

The matrix M' is the matrix M without the second column. As previously, we have that $v'_0 = (1, \kappa_2, \kappa_3, \ldots, \kappa_n)$ is an element of the lattice L'. Indeed, there are $\lambda_2, \ldots, \lambda_n \in \mathbb{Z}$ such that

$$(1, \mathbf{a}, \lambda_2, \ldots, \lambda_n) \cdot M' = v'_0 \tag{9}$$

However, the row vectors of this matrix M' are not linearly independent. Thus, they do not form a basis of the lattice L' and the original LLL algorithm can not be applied directly. In this case, we use the MLLL algorithm ([Poh87]) which is a variant of LLL in which the input vectors can be linearly dependent, and has the same complexity as the LLL algorithm.

Remark 1. Note that the secret key must be read in the transformation vector $(1, \mathbf{a}, \lambda_2, \ldots, \lambda_n)$ and not in the reduced basis. We could also have removed the first column, but experiments show that the required number of messages is greater in this case.

The lattice L' used in this case is different from the lattice L of theorem 1. The bound and the volume must be updated.

Lemma 1. *The volume of the lattice L' defined by the matrix M' given as (8) is equal to q^{n-2}.*

Proof. The sublattice S of L' spanned by the row vectors of the following matrix

$$\begin{pmatrix} 1 & \alpha_2 & \ldots & \alpha_n \\ 0 & q & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & q \end{pmatrix}$$

is a n-dimensional lattice. The dimension of the lattice L' defined by the matrix M' given as (8) is equal to the dimension of its sublattice S, therefore S is a full-rank sublattice of L' (see [NV09]). We consider a lattice as a group and we have the classical relation between volume and index: $\mathrm{Vol}(S) = \mathrm{Vol}(L')[L' : S]$. Now, it is easy to see that $\mathrm{Vol}(S) = q^{n-1}$ and $[L' : S] = q$, from which we get $\mathrm{Vol}(L') = q^{n-2}$. □

The following theorem is directly derived from this lemma.

Theorem 2. *Let n messages m_i ($i = 1, \ldots, n$, $n > 2$) with associated signatures (r_i, s_i) such that the ephemeral keys $\mathbf{k}_i$ share a total of δ bits between the MSB and LSB (see Figure 1). Under the assumption 1, the secret key $\mathbf{a}$ can be computed in time $\mathscr{C}(n, \frac{1}{2}\log_2(n-1)+N)$ as soon as*

$$\delta \geq \frac{2N + (n-2)}{n} + \frac{1 + \log_2(\pi e) - \log_2(\frac{n}{n-1})}{2}$$

The required number of shared bits δ is slightly larger with the lattice L' defined by the matrix M' given as (8) than the one given in the theorem 1. Experiments confirm this fact but the success rate is now independent of the secret key size (see section 5).

Remark 2. As a referee pointed out to us, this result may be reworded by CVP instead of SVP. This will be detailed in an extended version of this paper.

4.2.3 Weighted Euclidean inner product: In order to obtain the v_0 vector (7) (or similarly the v'_0 vector (9)) within a LLL-reduced basis, we can also use a weighted Euclidean inner product, to take advantage of the knowledge of the components size of the targeted vector. For example, we can take the following inner product of two vectors

$$\langle (x_0, \ldots, x_n), (y_0, \ldots, y_n) \rangle := \sum_{i=0}^{n} x_i y_i 2^{2(N - \lceil \log_2(v_{0,i}) \rceil)}$$

during the LLL algorithm. In practice, this trick drastically reduces the required number of shared bits δ (see section 5).

Remark 3. Weights can be used without needing to change the norm. Indeed, it is equivalent to multiplying all columns of the lattice by the corresponding weight.

As previously, a bound can be obtained with Gaussian Heuristic.

Theorem 3. *Let n messages m_i ($i = 1, \ldots, n$, $n > 2$) with associated signatures (r_i, s_i) such that the ephemeral keys $\mathbf{k}_i$ share a total of δ bits between the MSB and LSB. Under the assumption 1, the secret key $\mathbf{a}$ can be computed*

a. with the exhaustive search method in time $\delta\mathscr{C}(n+1, \frac{1}{2}\log_2(n)+\delta N)$ *as soon as*

$$\delta \geq \frac{N+(n-1)}{n} + \frac{(n+1)(1+\log_2(\pi e))}{2n} \tag{10}$$

b. with the lattice L' *(8) in time* $\mathscr{C}(n, \frac{1}{2}\log_2(n-1)+\delta N)$ *as soon as*

$$\delta \geq \frac{N+(n-2)}{n-1} + \frac{n(1+\log_2(\pi e))}{2(n-1)} \tag{11}$$

Proof. Let an integer $k \geq N-\delta$. We use a weighted Euclidean inner product such that each component of the targeted vector v_0 have the same size k (i.e. the i-th weight is equal to $k - \lceil \log_2(v_{0,i}) \rceil$).

a. With the exhaustive search method, a close approximation of the vector v_0 (7) is computed. Its norm is given by $\|v_0\|^2 = \sum_{i=1}^{n+1} v_{0,i}^2 2^{2(k-\lceil \log_2(v_{0,i}) \rceil)} \leq \sum_{i=1}^{n+1} 2^{2k} = 2^{2k+\log_2(n+1)}$ and the volume of the lattice (6) is

$$\mathrm{Vol}(L) = 2^k 2^{k-(N-\delta)} (q 2^{k-(N-\delta)})^{(n-1)} \geq 2^{k(n+1)+n(\delta-1)-N+1}.$$

Using the Gaussian heuristic assumption, we have $2^{2k+\log_2(n+1)} \leq \frac{n+1}{2\pi e} (2^{k(n+1)+n(\delta-1)-N+1})^{\frac{2}{n+1}}$ which is equivalent to (10).

b. We apply the same method with the n-dimensional lattice L' (8) and the seek vector v'_0 (9). We obtain $\|v'_0\|^2 = \sum_{i=1}^{n} v'^2_{0,i} 2^{2(k-\lceil \log_2(v'_{0,i}) \rceil)} \leq \sum_{i=1}^{n} 2^{2k} = 2^{2k+\log_2(n)}$ and $\mathrm{Vol}(L') = \frac{2^k (q 2^{k-(N-\delta)})^{(n-1)}}{q} \geq 2^{n(k-1)+\delta(n-1)-N+2}$. The Gaussian heuristic assumption gives $2^{2k+\log_2(n)} \leq \frac{n}{2\pi e} (2^{n(k-1)+\delta(n-1)-N+2})^{\frac{2}{n}}$ which is equivalent to (11).

Note that both results are independent of the variable k. □

4.3 Blocks of shared bits

The previous attack can be generalized to the case of ephemeral keys sharing several blocks of bits. Thus, we now assume that we have n messages m_i ($i = 1, \ldots, n$) with associated signatures (r_i, s_i) such that the ephemeral keys $\mathbf{k}_i$ share a total of δ bits dispatched between l blocks of bits. We denote by δ_i the number of bits of the i-th block $\mathbf{b}_i$ at position p_i (see Figure 2). For convenience we simplify the notation as follows: let $\underline{t} = (t_1, \ldots, t_l)$ be a l-tuple of integers then we set $2^{\underline{t}} = (2^{t_1}, \ldots, 2^{t_l})$ Then the ephemeral key $\mathbf{k}_i$ is of the form

$$\mathbf{k}_i = 2^{\underline{p}} \cdot \underline{\mathbf{b}} + 2^{\underline{t}} \cdot \underline{\mathbf{k}_i} \quad \text{for all } i = 1, \ldots, n \tag{12}$$

where $\underline{\mathbf{b}} = (\mathbf{b}_1, \ldots, \mathbf{b}_l)$ is the vector of shared bits blocks, with the position vector $\underline{p} = (p_1, \ldots, p_l)$ of the l blocks, and $\underline{\mathbf{k}_i} = (\mathbf{k}_{i,0}, \ldots, \mathbf{k}_{i,l})$ the vector of no shared bits blocks at positions $\underline{t} = (t_0, \ldots, t_l)$. After stating that $t_0 := 0$ and $p_{l+1} := N$, it follows that for all $i = 1, \ldots, n$, and for all $j = 1, \ldots, l$, we must have

$$t_j = p_j + \delta_j, \quad \delta = \sum_j \delta_j, \quad 0 \leq \mathbf{b}_j < 2^{\delta_j} \quad \text{and} \quad 0 \leq \mathbf{k}_{i,j} < 2^{(p_{j+1}-t_j)}$$

Note that the values of $\mathbf{k}_i$, $\mathbf{k}_{i,j}$ and $\mathbf{b}_j$ are all unknown. In the n signature equations (1), we substitute the $\mathbf{k}_i$ by (12) and we eliminate the common variable $\underline{\mathbf{b}}$, then we have

$$\begin{cases} (s_1^{-1}m_1 - s_2^{-1}m_2) + \mathbf{a}(s_1^{-1}r_1 - s_2^{-1}r_2) - \sum_{j=0}^{l} 2^{t_j}(\mathbf{k}_{1,j} - \mathbf{k}_{2,j}) \equiv 0 \pmod q \\ (s_1^{-1}m_1 - s_3^{-1}m_3) + \mathbf{a}(s_1^{-1}r_1 - s_3^{-1}r_3) - \sum_{j=0}^{l} 2^{t_j}(\mathbf{k}_{1,j} - \mathbf{k}_{3,j}) \equiv 0 \pmod q \\ \quad \vdots \qquad\qquad \vdots \qquad\qquad\qquad\qquad \vdots \qquad \vdots \\ (s_1^{-1}m_1 - s_n^{-1}m_n) + \mathbf{a}(s_1^{-1}r_1 - s_n^{-1}r_n) - \sum_{j=0}^{l} 2^{t_j}(\mathbf{k}_{1,j} - \mathbf{k}_{n,j}) \equiv 0 \pmod q \end{cases} \tag{13}$$

Let $\alpha_i, \beta_i \in \mathbb{Z}$ and $\underline{\kappa_i}, \underline{t} \in \mathbb{Z}^l$ be such that

$$\begin{cases} \alpha_i := (s_1^{-1}m_1 - s_i^{-1}m_i) \bmod q \\ \beta_i := (s_1^{-1}r_1 - s_i^{-1}r_i) \bmod q \\ \underline{\kappa_i} := (\mathbf{k}_{1,1}, \ldots, \mathbf{k}_{1,l}) - (\mathbf{k}_{i,1}, \ldots, \mathbf{k}_{i,l}) \\ \underline{t} := (t_1, \ldots, t_l) \end{cases}$$

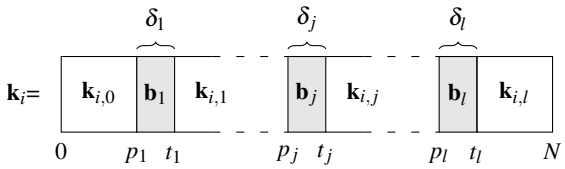

Fig. 2. Ephemeral keys

then (13) becomes

$$\begin{cases} \alpha_2 + \mathbf{a}\beta_2 - 2^t \cdot \underline{\kappa_2} \equiv \mathbf{k}_{1,0} - \mathbf{k}_{2,0} \pmod{q} \\ \alpha_3 + \mathbf{a}\beta_3 - 2^t \cdot \underline{\kappa_3} \equiv \mathbf{k}_{1,0} - \mathbf{k}_{3,0} \pmod{q} \\ \quad \vdots \quad \vdots \quad \vdots \quad \vdots \\ \alpha_n + \mathbf{a}\beta_n - 2^t \cdot \underline{\kappa_n} \equiv \mathbf{k}_{1,0} - \mathbf{k}_{n,0} \pmod{q} \end{cases} \tag{14}$$

where $\mathbf{a}$ and $\underline{\kappa_i}$ are unknown, β_i and α_i are known. Embedding these equations into the lattice L spanned by the row vectors of the following basis matrix

$$M = \left(\begin{array}{c|c} I_{l(n-1)+2} & \begin{array}{c} \alpha_2 \ \ldots \ \alpha_n \\ \beta_2 \ \ldots \ \beta_n \\ \hline 2^{t_1} I_{(n-1)} \\ \hline \vdots \\ \hline 2^{t_l} I_{(n-1)} \end{array} \\ \hline 0 & qI_{(n-1)} \end{array} \right) \tag{15}$$

we obtain that

$$v_0 = (1, \mathbf{a}, \underbrace{\mathbf{k}_{1,1} - \mathbf{k}_{2,1}, \ldots, \mathbf{k}_{1,1} - \mathbf{k}_{n,1}}_{\kappa_{i,1}}, \underbrace{\ldots}_{\kappa_{i,j}}, \underbrace{\mathbf{k}_{1,l} - \mathbf{k}_{2,l}, \ldots, \mathbf{k}_{1,l} - \mathbf{k}_{n,l}}_{\kappa_{i,l}}, \underbrace{\mathbf{k}_{1,0} - \mathbf{k}_{2,0}, \ldots, \mathbf{k}_{1,0} - \mathbf{k}_{n,0}}_{\alpha_i + \mathbf{a}\beta_i - 2^{t'} \cdot \underline{\kappa_i} \bmod q}) \tag{16}$$

is an element of the lattice L. If we were able to find this vector v_0 in L, then we could recover the secret key $\mathbf{a}$.

Example 1. For instance, if we have only one block (*i.e.* $l = 1$) of δ shared bits in the middle, then the n ephemeral keys are of the form

$$\mathbf{k}_i = \mathbf{k}_{i,0} + 2^{p_1}\mathbf{b}_1 + 2^{t_1}\mathbf{k}_{i,1} \quad \text{for all } i = 1, \ldots, n$$

where

$$\delta = \delta_1, \quad t_1 = p_1 + \delta_1, \quad 0 \le \mathbf{k}_{i,0} < 2^{p_1}, \quad 0 \le \mathbf{k}_{i,1} < 2^{N-t_1} \quad \text{and} \quad 0 \le \mathbf{b}_1 < 2^{\delta_1}$$

and α_i, β_i, $\kappa_i \in \mathbb{Z}$ be such that

$$\begin{cases} \alpha_i := (s_1^{-1} m_1 - s_i^{-1} m_i) \bmod q \\ \beta_i := (s_1^{-1} r_1 - s_i^{-1} r_i) \bmod q \\ \kappa_i := \mathbf{k}_{1,1} - \mathbf{k}_{i,1} \end{cases}$$

In this case, (14) becomes

$$\begin{cases} \alpha_2 + \mathbf{a}\beta_2 + 2^{t_1}\kappa_2 \equiv \mathbf{k}_{1,0} - \mathbf{k}_{2,0} \pmod{q} \\ \alpha_3 + \mathbf{a}\beta_3 + 2^{t_1}\kappa_3 \equiv \mathbf{k}_{1,0} - \mathbf{k}_{3,0} \pmod{q} \\ \quad \vdots \quad \vdots \quad \vdots \quad \vdots \\ \alpha_n + \mathbf{a}\beta_n + 2^{t_1}\kappa_n \equiv \mathbf{k}_{1,0} - \mathbf{k}_{n,0} \pmod{q} \end{cases}$$

and the lattice L is spanned by the row vectors of the following basis matrix

$$M = \left(\begin{array}{cc|ccc|ccc} 1 & 0 & 0 & \dots & 0 & \alpha_2 & \dots & \alpha_n \\ 0 & 1 & 0 & \dots & 0 & \beta_2 & \dots & \beta_n \\ \hline 0 & 0 & 1 & \dots & 0 & 2^{t_1} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 & \dots & 2^{t_1} \\ \hline 0 & 0 & 0 & \dots & 0 & q & \dots & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 0 & \dots & q \end{array}\right)$$

The targeted element of L is the vector

$$v_0 = (1, \mathbf{a}, \kappa_2, \dots, \kappa_n, \mathbf{k}_{1,0} - \mathbf{k}_{2,0}, \dots, \mathbf{k}_{1,0} - \mathbf{k}_{n,0}) = (1, \mathbf{a}, \kappa_2, \dots, \kappa_n, \lambda_2, \dots, \lambda_n) \cdot M$$

The proposed improvements of the section 4.2 can also be (directly) applied in the case of shared bits blocks. Especially, a weighted Euclidean inner product gives rise to the following estimations of the required number of shared bits in function of the number of blocks and the number of available messages.

Theorem 4. *Let n messages m_i ($i = 1, \dots, n$, $n > 2$) with associated signatures (r_i, s_i) such that the ephemeral keys $\mathbf{k}_i$ share a total of δ bits dispatched between l blocks of bits. Under the assumption 1, the secret key $\mathbf{a}$ can be computed*

a. with the exhaustive search method in time $\delta\mathscr{C}((l+1)(n-1)+2, \frac{1}{2}\log_2(n-1))$ as soon as

$$\delta \geq \frac{N+(n-1)}{n} + (1+\log_2(\pi e))\frac{(l+1)(n-1)+2}{2n} \tag{17}$$

b. with the lattice L' obtained by removing the second column (8) in time $\mathscr{C}((l+1)(n-1)+1, \frac{1}{2}\log_2(n-1))$ as soon as

$$\delta \geq \frac{N+(n-2)}{n-1} + (1+\log_2(\pi e))\frac{(l+1)(n-1)+1}{2(n-1)} \tag{18}$$

The proof of this theorem is almost similar as the one developed in Section 4.2 and is given in the Appendix B.

5 Experimental results

In order to check the validity and the quality of the bounds on δ, we implemented the methods on the computational algebra system Magma V2.17-1 ([BCP97]). All the tests have essentially validated the Gaussian Heuristic assumption (assumption 1) and the fact that the first gap of the lattices (defined as λ_2/λ_1, see Annex A) is high enough so that the shortest vector can be computed with the LLL algorithm. Hence they show the effectiveness of our method. In the following, the length of q is fixed to $N = 160$.

5.1 Shared MSB and LSB

The figure 3 is the graph of the four bounds on δ given by the theorems 1, 2 and 3 in function of the number of messages. The table 2 gives more details by listing some theoretical minimal integer values of the necessary number of LSB/MSB shared bits δ for a given number of messages.

We conducted experiments of this attack when the ephemeral keys have their δ LSB in common. For the same reason as the one explained in [NS02], the results for the case of MSB or both MSB/LSB are not as good as in the LSB case, about one more bit being required. As the secret key is 160 bits long, we used the independent key size method by removing the corresponding column of the lattice (see section 4.2.2). Additionally, we use a weighted

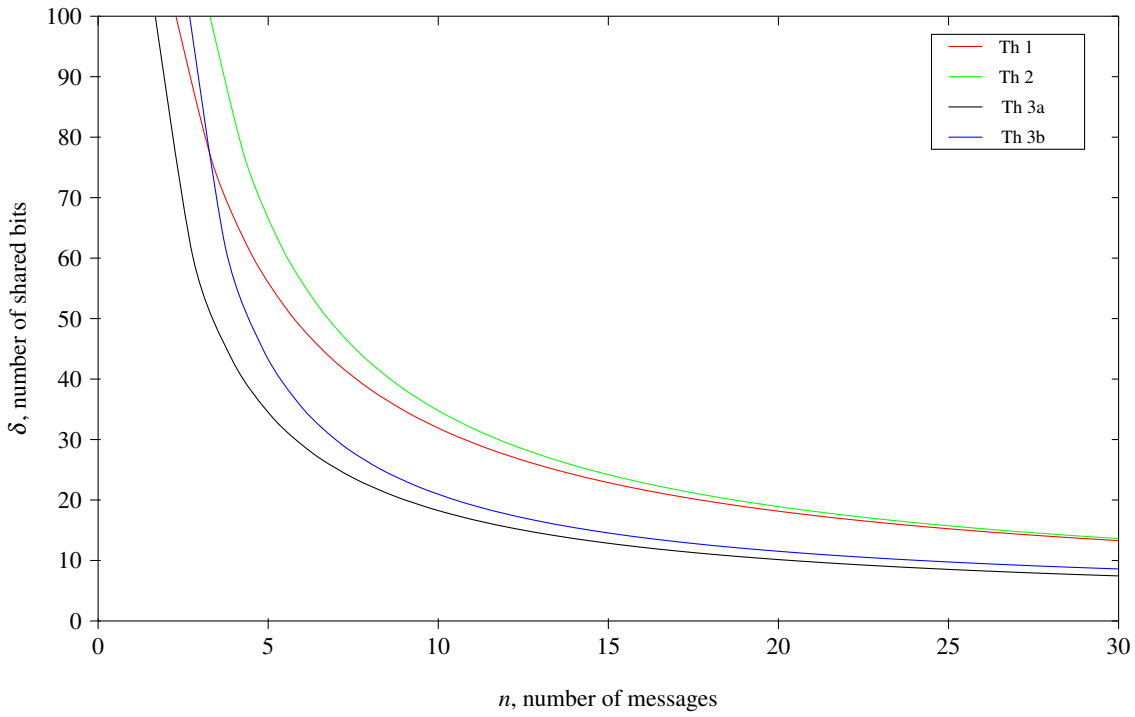

Fig. 3. Theoretical bounds of Theorems 1, 2 and 3 with $N = 160$

Euclidean inner product during the LLL algorithm phase which gives better results than the canonical inner product (more precisely, we use the inner product given as an example in section 4.2.3). The experiments are then conducted under the theorem 3b conditions. Note that we consider an attack to be successful when the secret key is found, that is when the targeted vector $\pm v_0$ (9) is a vector of the reduced basis. For each δ and each n, we generated 100 tests and store the success rate in the table 3. To compare experimental values to the theoretical bound, the success rates corresponding to the theoretical minimal value of δ for a given number of messages are written in red.

Bound on δ of theorem	n, number of messages																			
	3	4	5	6	7	8	9	10	15	20	30	40	50	60	70	80	90	100	200	∞
1	83	67	56	49	43	39	35	32	23	19	14	11	10	9	8	7	7	7	5	≈3.05
2	109	83	67	56	49	43	39	35	25	19	14	11	10	9	8	8	7	7	5	≈3.05
3a	57	44	36	30	27	24	21	20	14	12	9	8	7	6	6	6	5	5	4	≈3.05
3b	84	57	44	36	30	27	24	21	15	12	9	8	7	6	6	6	5	5	4	≈3.05

Table 2. Theoretical minimum for δ with $N = 160$

First of all, the results show that we have a 100% success rate when δ is the bound (11) of theorem 3 and then we could say that they confirm that the assumptions we made are justified. Moreover, we observe that we can often go a few bits beyond the theoretical bound on δ. Then, the success rate gradually decreases to zero percent with δ.

Another interesting result is that the attack still works with only 1 or 2 shared bits ($\delta = 1, 2$) when there are enough messages. This result is particularly surprising considering that the theoretical limit is 3 and that the success rate can be higher than 90% (not to say 99% or 100% when $n > 500$).

Note also that the step of lattice reduction of this attack is very fast (always less than a minute, or even less than a second up to $n \approx 70$).

δ	n, Number of messages																					
	40	50	60	70	80	90	100	110	120	130	140	150	160	170	180	190	200	250	300	400	500	600
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	8	10	35	56	91	99	99
2	0	0	0	0	0	2	6	12	24	33	42	58	63	73	80	85	100	100	100	100	100	100
3	0	2	19	34	60	74	82	94	96	97	99	99	99	99	99	100	100	100	100	100	100	100
4	34	76	90	99	99	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
5	96	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
Time (s)	0.35	0.58	0.78	0.94	1.2	1.4	1.7	1.9	2.1	2.4	2.6	2.9	3.2	3.5	3.8	4.1	4.2	6.3	8.5	15	27	44

Table 3. Success rate of LSB attack of theorem 3b

5.2 Blocks of shared bits

Following the bound (18) (better than the bound (17)) of theorem 4, the table 4 gives some theoretical minimal integer values of the necessary number of total shared bits δ in function of the number of blocks and the number of messages. Under theses conditions, we conducted a large number of experiments (100 tests for each δ and each n) whose results are summarized in table 5. Once more, we wrote in red the success rates corresponding to the theoretical minimal value of δ for a given number of messages.

Contrary to the case of shared LSB/MSB, we may have a success rate lower than 100% when δ was within the bound with the attack of blocks of shared bits (see Section 4.3). The reason is that the assumption of the theorem 4 may fail because of the occurrence of exceptionally short vectors. However, we observe that we can always go a few bits beyond the theoretical bound on δ keeping a good success rate.

We can also note that the computation time is longer than in the case LSB/MSB because of the larger dimension of lattices.

Number of blocks l	n, number of messages													
	3	4	5	6	7	8	9	10	15	20	30	40	50	∞
1	86	59	46	38	32	29	26	23	17	14	11	10	9	6
2	88	61	48	40	34	31	28	26	19	16	13	12	11	8
3	90	63	50	42	37	33	30	28	21	18	15	14	13	10
10	105	78	64	56	51	47	44	42	36	32	30	28	27	24
20	125	98	85	77	71	67	65	62	56	53	50	49	48	44
30	145	119	105	97	92	88	85	83	76	73	71	69	68	65

Table 4. Theoretical minimum for δ with theorem 4b

δ	n, number of messages															
	13	14	15	16	17	18	19	20	30	40	50	60	70	80	90	100
4	0	0	0	0	0	0	0	0	0	0	6	13	19	28	26	49
5	0	0	0	0	0	0	0	0	0	19	32	49	67	78	82	77
6	0	0	0	0	0	0	0	0	13	49	78	89	90	94	100	99
7	0	0	0	0	0	0	0	1	63	89	96	99	97	99	97	96
8	0	0	0	0	2	1	5	20	93	99	99	100	99	98	98	98
9	0	0	0	1	12	21	49	59	99	100	99	99	100	99	100	99
10	0	1	1	21	60	82	95	94	100	100	100	100	99	99	100	100
Time (s)	0.08	0.10	0.13	0.16	0.20	0.23	0.28	0.3	0.8	1.4	2.1	3.0	4.1	5	6	6.8

Table 5. Success rate of theorem 4b with one block

5.3 Random number generator tests

In the scenario with malicious PRNG, we verified that a defect is experimentally undetectable by conventional tests. Indeed, a 8 GByte bit sequence from the AES_OFB random number generator in Dieharder ([Bro11]), manipulated to contain enough implicit information, was used as input for the Dieharder test suite. More precisely, sequences of some randomly selected bits are repeated in a predictable way. For instance, a sequence of 4 bits can be repeated 100 times following a predictable pattern in a random sequence corresponding to 2^{10} ephemeral keys. The pattern that describes the positions of corrupted nonces (i.e. ephemeral keys sharing some bits) can be, for example, a function of the position of the first corrupted nonce. Therefore, in this case we need an additional step containing an exhaustive search to find the first corrupted nonce (of complexity of about 2^{10} with this example). All tests of the two referenced statistical test suites: Dieharder statistical test suite ([Bro11]) and the NIST statistical test suite (STS) ([RSN+10]) have shown a random behavior at a high confidence level, our manipulations being then unnoticed when the number of shared bits matches the number of corrupted nonces to have a 100% success rate (see Table 3). These experiments remind that these tests are not a proof of randomness even though they are common tools for initial validation. Finally, it has been shown that an exploitable bias is currently undetectable by conventional statistical tests.

6 Further developments

Throughout this work, we assumed that all ephemeral keys used in the attack shared a same block of bits. In this scenario, by the pigeonhole principle, our attack needs, in the worst case, $2^{\delta}+1$ samples before obtaining only two signatures that have ephemeral keys with δ bits in common. However, from a practical perspective, we would like to use all the signatures generated by a signer, which is not possible for the moment. Assuming, as in [LPS04], that an attacker can determine, in practice, some relation amongst the bits of the secret ephemeral keys rather than their specific values, we present below how to solve this problem by slightly adapting the lattices of our method.

In this context, our attack can be naturally extended to the more general case where each ephemeral key $\mathbf{k}_i$ ($i = 1 \dots n$) shares δ bits with at least one other key $\mathbf{k}_j$ ($i \neq j$) and not necessarily with all of them. For instance, we look at shared MSB/LSB, with the same notation as in section 4.1. For two fixed positions t and t', we can take the partition P of the set of all ephemeral keys corresponding to the equivalence relation $\mathscr{R}_{t,t'}$, defined such that related ephemeral keys share the first t MSB and the last t' LSB (which represent a total of δ bits). In a given equivalence class $[\mathbf{k}_j]$, if we have $\#[\mathbf{k}_j] \geq 2$ then we can apply the method of Section 4.1. For each class $[\mathbf{k}_j]$, we obtain a system of $\#[\mathbf{k}_j]-1$ modular equations as (5) with

$$\forall \mathbf{k}_i \in [\mathbf{k}_j] \text{ s.t. } i \neq j, \begin{cases} \alpha_i := 2^{-t}(s_j^{-1}m_j - s_i^{-1}m_i) \mod q \\ \beta_i := 2^{-t}(s_j^{-1}r_j - s_i^{-1}r_i) \mod q \\ \kappa_i := \tilde{\mathbf{k}}_j - \tilde{\mathbf{k}}_i \end{cases}$$

The set of all common solutions of the $\#P = \#\left(\{\mathbf{k}_i, i = 1..n\}/R_{t,t'}\right)$ systems described as above forms a lattice similar to (6) of dimension equal to $n - \#P$.

In the same way, other shared bits in other positions (i.e. when t and t' are not fixed) can be exploited by expanding the lattice with the corresponding columns. Also the same improvement can be developped in the case of shared bits blocks in the middle. All these further developments will be detailed in an extended version of this paper. More generally, we could also imagine other forms of implicit information, i.e. an other relationship than just the equality between bits. For instance, an interesting open question is whether we can exploit inequalities or simple relationships between unknown bits.

7 Acknowledgments

This work has greatly benefited from highly relevant comments of anonymous referees. The authors would like to thank them sincerely for their suggestions that improved this paper. They also thank Olivier Orcière for all his constructive comments on a preliminary version of this paper. Finally, they thank Martin Albrecht for presenting this paper at the SAC2012 conference.

References

[AD93] Leonard M. Adleman and Jonathan DeMarrais. A Subexponential Algorithm for Discrete Logarithms over All Finite Fields. In *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, Lecture Notes in Computer Science, pages 147–158, 1993.

[ADH94] Leonard M. Adleman, Jonathan DeMarrais, and Ming-Deh A. Huang. A subexponential algorithm for discrete logarithms over the rational subgroup of the jacobians of large genus hyperelliptic curves over finite fields. In *Algorithmic Number Theory, First International Symposium, ANTS-I, Ithaca, NY, USA, May 6-9, 1994, Proceedings*, pages 28–40. Springer, 1994.

[ADH99] Leonard M. Adleman, Jonathan DeMarrais, and Ming-Deh A. Huang. A Subexponential Algorithm for Discrete Logarithms over Hyperelliptic Curves of Large Genus over GF(q). *Theoretical Computer Science*, 226(1-2):7–18, 1999.

[Ajt98] Miklós Ajtai. The shortest vector problem in l_2 is *np*-hard for randomized reductions (extended abstract). In *Proceedings of the 30th Symposium on the Theory of computing (STOC 1998)*, pages 10–19. ACM Press, 1998.

[Ajt06] Miklós Ajtai. Generating random lattices according to the invariant distribution. draft, March 2006.

[BCP97] Wieb Bosma, John Cannon, and Catherine Playoust. The MAGMA algebra system: the user language. *J. Symb. Comput.*, 24:235–265, October 1997.

[BGM97] Mihir Bellare, Shafi Goldwasser, and Daniele Micciancio. "Pseudo-Random" Number Generation Within Cryptographic Algorithms: The DSS Case. In *CRYPTO*, volume 1294 of *Lecture Notes in Computer Science*, pages 277–291. Springer, 1997.

[Bro11] Robert G. Brown. DieHarder: A Random Number Test Suite. URL `http://www.phy.duke.edu/~rgb/General/dieharder.php`, 2011. C program archive dieharder, version 3.29.4b.

[Cop96] Don Coppersmith. Finding a Small Root of a Bivariate Integer Equation; Factoring with High Bits Known. In *EUROCRYPT*, pages 178–189, 1996.

[ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 10–18. Springer-Verlag New York, Inc., 1985.

[FIP94] FIPS. *Digital Signature Standard (DSS)*. National Institute of Standards and Technology (NIST), 1994.

[FIP09] FIPS. *Digital Signature Standard (DSS)*. pub-NIST, pub-NIST:adr, 2009.

[FMR10] Jean-Charles Faugère, Raphaël Marinier, and Guénaël Renault. Implicit Factoring with Shared Most Significant and Middle Bits. In *Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 70–87. Springer, 2010.

[GJQ97] A. Gillet, Marc Joye, and Jean-Jacques Quisquater. Cautionary note for protocols designers: Security proof is not enough. In H. Orman and C. Meadows, editors, *DIMACS Workshop on Design and Formal Verification of Security Protocols*, 1 1997.

[HGS01] Nick Howgrave-Graham and Nigel P. Smart. Lattice Attacks on Digital Signature Schemes. *Des. Codes Cryptography*, 23(3):283–290, 2001.

[JMV01] Don Johnson, Alfred Menezes, and Scott A. Vanstone. The Elliptic Curve Digital Signature Algorithm (ECDSA). *Intern. J. of Information Security*, 1(1):36–63, 2001.

[Len87] Hendrik W. Lenstra. Factoring Integers with Elliptic Curves. In *The Annals of Mathematics*, volume 126(3), pages 649–673, 1987.

[LL93] Arjen K. Lenstra and Hendrik W. Lenstra. The Development of the Number Field Sieve. volume 1554 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 1993.

[LLL82] Arjen Lenstra, Hendrik Lenstra, and László Lovász. Factoring polynomials with rational coefficients. In *Mathematische Annalen*, volume 261, no 4, pages 515–534, 1982.

[LPS04] Peter J. Leadbitter, Dan Page, and Nigel P. Smart. Attacking DSA Under a Repeated Bits Assumption. In *CHES*, volume 3156 of *Lecture Notes in Computer Science*, pages 428–440. Springer, 2004.

[MNS01] Edwin El Mahassni, Phong Q. Nguyen, and Igor Shparlinski. The Insecurity of Nyberg-Rueppel and Other DSA-Like Signature Schemes with Partially Known Nonces. In *CaLC*, volume 2146 of *Lecture Notes in Computer Science*, pages 97–109. Springer, 2001.

[MR09] Alexander May and Maike Ritzenhofen. Implicit Factoring: On Polynomial Time Factoring Given Only an Implicit Hint. In *Public Key Cryptography*, volume 5443 of *Lecture Notes in Computer Science*, pages 1–14. Springer, 2009.

[NNTW05] David Naccache, Phong Q. Nguyen, Michael Tunstall, and Claire Whelan. Experimenting with Faults, Lattices and the DSA. In *Public Key Cryptography*, volume 3386 of *Lecture Notes in Computer Science*, pages 16–28. Springer, 2005.

[NS02] Phong Q. Nguyen and Igor Shparlinski. The Insecurity of the Digital Signature Algorithm with Partially Known Nonces. *Journal of Cryptology*, 15:151–176, 2002.

[NS03] Phong Q. Nguyen and Igor Shparlinski. The Insecurity of the Elliptic Curve Digital Signature Algorithm with Partially Known Nonces. *Designs, Codes and Cryptography*, 30(2):201–217, 2003.

[NS05] Phong Q. Nguyen and Damien Stehlé. Floating-Point LLL Revisited. In *EUROCRYPT*, pages 215–233, 2005.

[NV09] Phong Q. Nguyen and Brigitte Vallée. *Hermite's Constant and Lattice Algorithms*. Information Security and Cryptography. Springer, 2009.

[Poh87] Michael Pohst. A Modification of the LLL Reduction Algorithm. In *Journal of Symbolic Computation*, volume 4, pages 123–127, 1987.

[Pol78] John M. Pollard. Monte Carlo methods for index computation (mod p). 32:918–924, 1978.

[Pol00] John M. Pollard. Kangaroos, Monopoly and discrete logarithms. 13:437–447, 2000.

[Pom84] Carl Pomerance. The Quadratic Sieve Factoring Algorithm. In *Proceedings of EUROCRYPT 84 workshop on Advances in cryptology: theory and application of cryptographic techniques*, volume 209 of *Lecture Notes in Computer Science*, pages 169–182. Springer-Verlag, 1984.

[Pou09] Dimitrios Poulakis. Some lattices attacks on dsa and ecdsa. Cryptology ePrint Archive, Report 2009/363, 2009. `http://eprint.iacr.org/`.

[RS86] Ronald Linn Rivest and Adi Shamir. Efficient factoring based on partial information. In *Proc. of a workshop on the theory and application of cryptographic techniques on Advances in cryptology—EUROCRYPT '85*, pages 31–34, New York, NY, USA, 1986. Springer-Verlag New York, Inc.

[RSN+10] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Elaine Barker, Stefan Leigh, Mark Levenson, Mark Vangel, David Banks, Alan Heckert, James Dray, and San Vo. A Statistical Test Suite of Random and Pseudorandom Number Generators for Cryptographic Applications. Tech. rep., National Institute of Standards and Technology (NIST), Special Publication 800-22 Revision 1a, 2010. `http://csrc.nist.gov/groups/ST/toolkit/rng/documentation_software.html`.

[Sch90] Claus-Peter Schnorr. Efficient Identification and Signatures for Smart Cards. In *Proceedings of the 9th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '89, pages 239–252. Springer-Verlag, 1990.

[Sha71] Daniel Shanks. Class number, a theory of factorization, and genera. In *Proceedings of Symposia in Pure Mathematics*, volume 20, pages 415–440. American Mathematical Society, Providence, 1971.

[Sho97] Victor Shoup. Lower Bounds for Discrete Logarithms and Related Problems. In *EUROCRYPT*, volume 1233, pages 256–266. Springer, 1997.

[Sko05] Sergei P. Skorobogatov. *Semi-invasive attacks - A new approach to hardware security analysis*. PhD thesis, University of Cambridge, 2005.

[SM09] Santanu Sarkar and Subhamoy Maitra. Further Results on Implicit Factoring in Polynomial Time. *Advances in Mathematics of Communications*, 3(2):205–217, 2009.

[Tak06a] Katsuyuki Takashima. Practical Application of Lattice Basis Reduction Algorithm to Side-Channel Analysis on (EC)DSA. *IEICE Transactions*, 89-A(5):1255–1262, 2006.

[Tak06b] Katsuyuki Takashima. Practical modifications of Leadbitter et al.'s repeated-bits side-channel analysis on (EC)DSA. In *Proceedings of the 6th international conference on Information Security Applications*, WISA'05, pages 259–270. Springer-Verlag, 2006.

[Tes01] Edlyn Teske. Square-Root Algorithms For The Discrete Logarithm Problem (a Survey). In *Public Key Cryptography and Computational Number Theory, Walter de Gruyter*, pages 283–301, 2001.

A Common results on lattice

In this section, we state common results on lattices that are used throughout this paper. Readers interested in getting more details and proofs can refer to [NV09].

An integer lattice L is a discrete additive subgroup of $\mathbb{Z}^n$. It can be generated from a basis of d independent vectors $(b_1,\ldots,b_d)$ of $\mathbb{Z}^n$ by linear combinations with integer coefficients. A lattice may be described by many different bases. All the bases are then related by an unimodular transformation. The integer d is called the dimension of L. If $d = n$ then L is said to be full-rank.

Definition 1. *The Gram determinant of $b_1,\ldots,b_d \in \mathbb{R}^n$ denoted by $\Delta(b_1,\ldots,b_d)$, is the determinant of the $d \times d$ Gram matrix $(\langle b_i, b_j\rangle)_{1\leq i,j\leq d}$.*

Definition 2. *The volume of L is defined by $\mathrm{Vol}(L) = \Delta(b_1,\ldots,b_d)^{1/2}$. In other words, it is the d-dimensional volume of the parallelepiped spanned by the vectors of a basis.*

The dimension and the volume are independent of the choice of this basis.

The volume of a lattice is easy to compute with Definition 2 if at least one explicit basis is known. But, in this article, we also need a way to compute the volume of lattices spanned by a set of linearly dependent vectors. In this case, the following results on sublattices is very useful.

Definition 3. *A sublattice of L is a lattice M included in L. Clearly, the sublattices of L are the subgroups of L.*

Lemma 2 ([NV09]). *A sublattice M of L is full-rank if and only if the group index $[L : M]$ is finite, in which case we have*

$$\mathrm{Vol}(M) = \mathrm{Vol}(L) \times [L : M].$$

We also need the following results about lattice basis reduction. More particularly, we require a way to provide a precise estimation of the expected length of the shortest vector, which is called the Gaussian heuristic, and a way to get an approximation of this small vector in polynomial time, which is done by using the LLL algorithm ([LLL82]).

Definition 4. *For $1 \leq r \leq d$, let $\lambda_r(L)$ be the least real number such that there exists at least r linearly independent vectors of L of euclidean norm smaller or equal to $\lambda_r(L)$. We call $\lambda_1(L),\ldots,\lambda_d(L)$ the d minima of L and we call $g(L) = \lambda_2(L)/\lambda_1(L)$ the gap of L.*

Theorem 5 (LLL [LLL82]). *Let L be a d-dimensional lattice of $\mathbb{Z}^n$ given by a basis $(\mathbf{b_1},\ldots,\mathbf{b_d})$. Then LLL algorithm computes a reduced basis $(\mathbf{v_1},\ldots,\mathbf{v_d})$ that approximates the shortest vector of L within an exponential factor:*

$$\|\mathbf{v_1}\| \leq 2^{\frac{d-1}{4}} \operatorname{Vol}(L)^{\frac{1}{d}}$$

The running time of Nguyen and Stehlé's version is $\mathscr{O}(d^5(d+\log B)\log B)$ where $B = \max_i(\|\mathbf{b_i}\|)$, see [NS05].

The time complexity of computing a shortest vector of L (which is a NP-Hard problem [Ajt98]) is denoted here by $\mathscr{C}(d,B)$.

Theorem 6 (Gaussian heuristic [Ajt06]). *Let L be a random d-dimensional lattice of $\mathbb{Z}^n$. Then, with overwhelming probability, all the minima of L are asymptotically close to:*

$$\sqrt{\frac{d}{2\pi e}} \operatorname{Vol}(L)^{\frac{1}{d}}$$

Thus, it is common practice to assume that if a vector $v \in L$ is shorter than the Gaussian heuristic $\lambda_1(L) \approx \sqrt{\frac{d}{2\pi e}} \operatorname{Vol}(L)^{\frac{1}{d}}$ applied to the d-dimensional lattice L then it is the shortest vector of L. Moreover, when the gap of L is high enough, this vector can be found in an LLL-reduced basis of L. For lattices proposed in this article, this essential and common assumption is confirmed by experimental results of section 5 and seems to be true in practice.

B Proof of Theorem 4

Proof. Let an integer $k \geq N - \delta$. We use a weighted Euclidean inner product such that each component of the seek vector v_0 have the same size k (i.e. the i-th weight is equal to $k - \lceil \log_2(v_{0,i}) \rceil$).

a. With the exhaustive search method: first note that the dimension of the lattice L defined by (15) is

$$\dim(L) = (l+1)(n-1)+2$$

A close approximation of the norm of vector v_0 (16) is then computed:

$$\|v_0\|^2 = \sum_{i=1}^{\dim(L)} v_{0,i}^2 2^{2(k-\lceil \log_2(v_{0,i}) \rceil)} \leq \sum_{i=1}^{\dim(L)} 2^{2k} = 2^{2k}((l+1)(n-1)+2)$$

Next, the volume of the lattice (15) is

$$\operatorname{Vol}(L) = 2^k 2^{k-(N-\delta)} (q 2^{k-p_1})^{(n-1)} \prod_{j=1}^{l} 2^{(k-(p_{j+1}-t_j))(n-1)}$$

but, we have

$$\begin{aligned}
&(k-p_1)(n-1) + \textstyle\sum_{j=1}^{l}(k-(p_{j+1}-t_j))(n-1) \\
&= (n-1)(k-p_1+\textstyle\sum_{j=1}^{l}(k-p_{j+1}+p_j+\delta_j)) \\
&= (n-1)(k-p_1+lk-\textstyle\sum_{j=1}^{l} p_{j+1}+\sum_{j=1}^{l} p_j+\sum_{j=1}^{l}\delta_j) \\
&= (n-1)(k(l+1)-p_{l+1}+\delta) \\
&= (n-1)(k(l+1)-N+\delta)
\end{aligned}$$

then

$$\operatorname{Vol}(L) = q^{n-1} 2^k 2^{k-(N-\delta)} 2^{(n-1)(k(l+1)-N+\delta)} \geq 2^{k((l+1)(n-1)+2)+n(\delta-1)-N+1}$$

Using the Gaussian heuristic assumption, we have

$$2^{2k+\log_2(\dim(L))} \leq \frac{\dim(L)}{2\pi e} (2^{k\dim(L)+n(\delta-1)-N+1})^{\frac{2}{\dim(L)}}$$

which is equivalent to

$$\delta \geq \frac{N+(n-1)}{n} + (1+\log_2(\pi e))\frac{\dim(L)}{2n}$$

b. Same computation with the lattice L' obtained by removing the second column of (15). The dimension of L' is equal to $(l+1)(n-1)+1$. Then a close approximation of the norm of vector v'_0 is

$$\|v'_0\|^2 \leq 2^{2k}((l+1)(n-1)+1)$$

Next, the volume of L' is

$$\mathrm{Vol}(L') = 2^k q^{n-2}(2^{k-p_1})^{(n-1)} \prod_{j=1}^{l} 2^{(k-(p_{j+1}-t_j))(n-1)} \geq 2^{k((l+1)(n-1)+1)+(n-1)\delta-N-n+2}$$

Using the Gaussian heuristic assumption, we have

$$2^{2k}\dim(L') \leq \frac{\dim(L')}{2\pi e}(2^{k\dim(L')+(n-1)\delta-N-n+2})^{\frac{2}{\dim(L')}}$$

which is equivalent to

$$\delta \geq \frac{N+(n-2)}{n-1} + (1+\log_2(\pi e))\frac{\dim(L')}{2(n-1)}$$

□