Attacking (EC)DSA with partially known multiples of nonces

Marios Adamoudis[a], Konstantinos A. Draziotis[b], Dimitrios Poulakis[a]

[a] *Department of Mathematics*
Aristotle University of Thessaloniki
54124 Thessaloniki, Greece
[b] *Department of Informatics*
Aristotle University of Thessaloniki
54124 Thessaloniki, Greece

Abstract

In this paper, we improve the theoretical background of the attacks on the DSA schemes given in [1, 29], and we present some new more practical attacks.

Keywords:

2020 MSC: : 94A60, 11T71, 11Y16.
Public Key Cryptography, Digital Signatures, Digital Signature Algorithm, Elliptic Curve Digital Signature Algorithm, Closest Vector Problem, Discrete Logarithm,, Lattices, LLL algorith, BKZ algorithm, Closest Vector Problem, Babai's Nearest Plane Algorithm

1. Introduction

In August 1991, the U.S. government's National Institute of Standards and Technology (NIST) proposed the Digital Signature Algorithm (DSA) for digital signatures [24, 21]. This algorithm has become a standard [11] and has been called Digital Signature Standard (DSS). In 1998, an elliptic curve analogue called Elliptic Curve Digital Signature Algorithm (ECDSA) was proposed and standardized, see [19].

In the first Subsection we present DSA and ECDSA. Then, in Subsection 1.2 we provide some necessary bibliographic references and in Subsection 1.3 we present the contribution of our paper.

1.1. The DSA and ECDSA Schemes

First, we provide a brief outline of DSA. The signer chooses a prime p of size between 1024 and 3072 bits with increments of 1024, as recommended in

Email addresses: aamarios@math.auth.gr (Marios Adamoudis), drazioti@csd.auth.gr (Konstantinos A. Draziotis), poulakis@math.auth.gr (Dimitrios Poulakis)

FIPS 186-3 [11, page 15]. In addition, the signer selects a prime q of size 160, 224 or 256 bits, with $q|p-1$, Further, the signer considers a generator g of the unique cyclic subgroup of $\mathbb{F}_p^*$ of order q. Moreover, the signer selects randomly an integer a from the set $\{1,\dots,q-1\}$ and computes $R = g^a \bmod p$. The public key of the signer is (p,q,g,R) and the private key is a. The signer also publishes a secure hash function $h : \{0,1\}^* \to \{0,\dots,q-1\}$. To sign a message $m \in \{0,1\}^*$, the signer picks a random $k \in \{1,\dots,q-1\}$ which is the ephemeral key (also, is called *nonce*). Finally, the signer computes $r = (g^k \mod p) \mod q$ and $s = k^{-1}(h(m)+ar) \mod q$. The signature of a message m is the pair (r,s), and it is valid if and only if, satisfies the following equality :

$$r = ((g^{s^{-1}h(m) \text{mod} q} R^{s^{-1}r \bmod q}) \mod p) \mod q.$$

This final equality, have to be checked by the verifier.

For the ECDSA the signer initially selects an elliptic curve E over the finite field $\mathbb{F}_p$ and a point $P \in E(\mathbb{F}_p)$, with order a prime q of size at least 160 bits. According to FIPS 186-3, the binary length of the prime p must be in the set $\{160, 224, 256, 512\}$. Furthermore, for some randomly chosen $a \in \{1,\dots,q-1\}$ the signer computes the point $Q = aP$. The public key is (E,p,q,P,Q) and the private key is a. The signer also publishes a secure hash function $h : \{0,1\}^* \to \{0,\dots,q-1\}$. As previous, the signer in order to sign a message m selects randomly $k \in \{1,\dots,q-1\}$ (the nonce) and computes $kP = (x,y)$ (where x and y are regarded as integers between 0 and $p-1$). The signature of the message m is the pair (r,s) :

$$r = x \bmod q$$

and

$$s = k^{-1}(h(m) + ar) \bmod q.$$

For the verification process, the verifier executes the following calculations :

$$u_1 = s^{-1}h(m) \bmod q, \quad u_2 = s^{-1}r \bmod q, \quad u_1P + u_2Q = (x_0, y_0).$$

The verifier accepts the signature if and only if $r = x_0 \mod q$.

The security of the two systems is relied on the assumption that, the only way to forge the signature, is to recover either the secret key a, or the ephemeral key k. In the latter case is very easy to compute a. Accordingly, the parameters of these systems were chosen in such a way, that the computation of discrete logarithms is computationally infeasible.

1.2. Previous Work

Several papers in the bibliography have used the equivalence $s = k^{-1}(h(m) + ar) \bmod q$, to exploit (EC)DSA scheme. By picking many signatures (r_i, s_i), someone can construct a lattice and then lattice reduction techniques can be applied. This line of research started with Nigel Smart and Howgrave Graham, in [18]. Below, we list some papers which presents attacks based on the previous idea.

The case where the random parameters for DSA are generated using a Linear Congruential pseudorandom number Generator (LCG), was studied in [2]. It was proved that, the combination of the DSA "signature equations" with the LCG, leads to a system of equations which reveals the secret key by using Babai's CVP approximation algorithm.

The first rigorous lattice attack was given in [25]. The authors managed to reduce the security of DSA to the Hidden Number Problem (HNP), which can further be reduced to an approximation Closest Vector Problem CVP_γ, to a specific lattice. As a result, the secret key a can be computed deterministically in polynomial time complexity. What is more, in [26] the previous attack was adapted to the elliptic curve variant, ECDSA.

In [3] another attack was presented. One signature is used to compute two short vectors of a three-dimensional lattice by applying LLL reduction. The pair (a, k) was proved to be in the intersection of two straight lines constructed by the previous short vectors.

A variant of HNP was provided in [17]. In this paper a practically attack was presented which affects a specific version of OpenSSL [27]. Furthermore, the implementation of ECDSA in OpenSSL was also attacked in [5, 6]. Moreover, a further improvement was presented in [23], where the authors managed to find the secret key of the bitcoin elliptic curve secp256k1, using only 200 signatures.

Another attack was presented in [28] which combines the LLL algorithm and two algorithms for the computation of the integral points on two family of conics. The attack is effective, if one message is available and at least the elements of one of the sets $\{a, k^{-1} \bmod q\}$, $\{k, a^{-1} \bmod q\}$ and $\{a^{-1} \bmod q, k^{-1} \bmod q\}$ are sufficiently small.

In [8], a two dimensional lattice $\mathcal{L}$ was used. Then, the Gauss-Lagrange Lattice Reduction algorithm was applied to provide a basis of L, formed by the two successive minima. These two vectors, correspond to two straight lines intersecting at the point (a, k). If a and k are sufficiently small, then it was proved that (a, k) can be computed in polynomial time. Similar attacks hold for the pairs $(k^{-1} \bmod q, k^{-1}a \bmod q)$ and $(a^{-1} \bmod q, a^{-1}k \bmod q)$.

The attacks presented in [10, 15], assume that there are equalities between δ bits of the unknown ephemeral keys. In fact, this implicit information should be extracted by constructing a lattice which contains a very short lattice vector such that, its components yield the secret key. When the ephemeral keys share enough bits, this vector is small enough and so it can be computed by the LLL.

In [9], a Coppersmith type attack was presented. It was proved that in case where a and k satisfy a certain inequality, the secret key a can be efficiently computed.

The attack described in [29] is based on the construction of a system of linear congruences using some signatures. If this system has at most one solution below a certain bound, then the author proved that the secret key can be computed efficiently. This result was further improved in [1].

Finally, in [20], a probabilistic attack based on enumeration techniques was presented. The authors reveal the secret key, if two bits of 100 ephemeral keys are known. The attack first reduces the problem of finding the secret key to a

HNP and then, reduces HNP to a suitable Bounded Distance Decoding problem. This last problem was attacked by advanced lattice enumeration algorithms.

1.3. Our Contribution

In the present work we also consider lattice based attacks applied to (EC)DSA. We provide improvements of the results presented in [1] and [29]. Actually, we consider the system of linear congruences of [1] and we improve the upper bound under which the system has at most one solution. Propositions 4.1 and 4.3 improve Proposition 2 of [1]. In addition, we update our basic deterministic attack provided in [1] (see Table 1). Moreover, a heuristic improvement is provided, see Section 6. In fact, assuming the existence of a suitable oracle (which is more weak than knowing specific number of contiguous bits of the ephemeral keys) we suggest a new heuristic attack. As an illustration, we break secp161k1 (see [31]) under the assumption that one specific multiple of an ephemeral key has 161 bits.

1.4. The Structure of the Paper.

The paper is organized as follows. In Section 2 we recall some basic facts concerning lattices. In Section 3 we continue with the proof of some auxiliary results, which we shall need for the presentation of our attacks. Section 4 is devoted to the construction of the DSA-system, which is actually a linear system over a prime finite field. Our attacks, are presented in Sections 5 and 6. What is more, some experimental results are presented in Section 7. Finally, in the last section we provide some concluding remarks.

2. Background on Lattices

In the current Section, we recall some well-known facts about lattices.

Let $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n$ be linearly independent vectors of $\mathbb{R}^m$. The set

$$\mathcal{L} = \left\{ \sum_{j=1}^{n} \alpha_j \mathbf{b}_j : \alpha_j \in \mathbb{Z}, 1 \le j \le n \right\}$$

is called a *lattice* and the finite vector set $\mathcal{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is called a basis of the lattice $\mathcal{L}$. All the bases of $\mathcal{L}$ have the same number of elements n which is called *dimension* or *rank* of $\mathcal{L}$. If $n = m$, then the lattice $\mathcal{L}$ is said to have *full rank*. We denote by M the $n \times m$-matrix having as rows the vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$. If $\mathcal{L}$ has full rank, then the *volume* of the lattice $\mathcal{L}$ is defined to be the positive number $|\det M|$. The volume, as well as the rank, are independent of the basis $\mathcal{B}$. It is denoted by $vol(\mathcal{L})$ or $\det \mathcal{L}$ (see also [12]). If $\mathbf{v} \in \mathbb{R}^m$, then $\|\mathbf{v}\|$ denotes, as usually, the Euclidean norm of $\mathbf{v}$. Further, we denote by $LLL(M)$, the application of the well-known LLL-algorithm on the rows of M and by $\lambda_1(\mathcal{L})$ we denote the least of the lengths of vectors of $\mathcal{L} - \{\mathbf{0}\}$. Finally, if $\mathbf{t} \in \mathbb{R}^m$, then with $dist(\mathcal{L}, \mathbf{t})$ we denote $\min\{\|\mathbf{v} - \mathbf{t}\| : \mathbf{v} \in \mathcal{L}\}$.

We define the approximate Closest Vector Problem $CVP_{\gamma_n}(\mathcal{L})$ (for some $\gamma_n \geq 1$) as follows: Given a lattice $\mathcal{L} \subset \mathbb{Z}^m$ of rank n and a vector $\mathbf{t} \in \mathbb{R}^m$, find a vector $\mathbf{u} \in \mathcal{L}$ such that, for every $\mathbf{u}' \in \mathcal{L}$ we have:

$$\|\mathbf{u} - \mathbf{t}\| \leq \gamma_n \|\mathbf{u}' - \mathbf{t}\|.$$

We say that we have a CVP oracle, if we have an efficient probabilistic algorithm that solves CVP_{γ_n}, for $\gamma_n = 1$. To solve CVP_{γ_n}, we usually use Babai's algorithm [12, Chapter 18] (which has polynomial running time). In fact, combining this algorithm with the LLL algorithm, we solve $\text{CVP}_{\gamma_n}(\mathcal{L})$ for some lattice $\mathcal{L} \subset \mathbb{Z}^m$ having $\gamma_n = 2^{n/2}$ and $n = rank(\mathcal{L})$, in polynomial time. Below, we provide a pseudocode for Babai's Nearest Plane Algorithm.

Babai's Nearest plane Algorithm:

INPUT: A $n \times m$-matrix M with rows the vectors of a basis $\mathcal{B} = \{\mathbf{b}_i\}_{1 \leq i \leq n}$ $\subset \mathbb{Z}^m$ of the lattice $\mathcal{L}$ and a vector $\mathbf{t} \in \mathbb{R}^m$
OUTPUT: $\mathbf{x} \in \mathcal{L}$ such that $||\mathbf{x} - \mathbf{t}|| \leq 2^{n/2} dist(\mathcal{L}, \mathbf{t})$.

1. $M^* = \{(\mathbf{b}_j^*)_j\} \leftarrow GSO(M)$ # GSO : Gram-Schmidt Orthogonalization
2. $\mathbf{b} \leftarrow \mathbf{t}$
3. For $j = n$ to 1
4. $\quad c_j \leftarrow \left\lfloor \frac{\mathbf{b} \cdot \mathbf{b}_j^*}{||\mathbf{b}_j^*||^2} \right\rceil$ $\quad \# \lfloor x \rceil = \lfloor x + 0.5 \rfloor$
5. $\quad \mathbf{b} \leftarrow \mathbf{b} - c_j \mathbf{b}_j$
6. Return $\mathbf{t} - \mathbf{b}$.

If the rank of $\mathcal{L}$ is "quite" small, then we can use as a CVP oracle the deterministic algorithm of Micciancio-Voulgaris [22].

3. Auxiliary Results

In this section we provide two fundamental results, necessary for the description of our attacks.

Proposition 3.1. *Let n, q and A_j be positive integers satisfying*

$$\frac{q^{\frac{j}{n+1} + f_q(n)}}{2} < A_j < \frac{q^{\frac{j}{n+1} + f_q(n)}}{1.5} \quad (j = 1, \ldots, n), \tag{1}$$

where $f_q(n)$ is a positive real number such that

$$f_q(n) < \frac{1}{n+1} \tag{2}$$

and

$$\frac{q^{1+2f_q(n)}}{1.5} < q - \frac{1}{2}\, q^{\frac{n}{n+1} + f_q(n)} \tag{3}$$

Let $\mathcal{L}$ be the lattice generated by the vectors

$$\mathbf{b}_0 = (-1, A_1, \dots, A_n), \mathbf{b}_1 = (0, q, 0, \dots, 0), \dots, \mathbf{b}_n = (0, \dots, 0, q).$$

Then, for all nonzero $\mathbf{v} \in \mathcal{L}$, we have:

$$\|\mathbf{v}\| > \frac{1}{2}\, q^{\frac{n}{n+1} + f_q(n)}.$$

Proof. See [1, Proposition 3.1] □

The following two remarks show us how we can choose the quantity $f_q(n)$.

Remark 3.2. The quantity $f_q(n)$ has been choosen in order to satisfy the inequalities (2) and (3). The second inequality holds if and only if we have:

$$4q^{2f_q(n)} + 3q^{-\frac{1}{n+1}} q^{f_q(n)} - 6 < 0,$$

which is equivalent to the following inequality:

$$q^{f_q(n)} < \frac{-3q^{-\frac{1}{n+1}} + \sqrt{96 + 9q^{-\frac{2}{n+1}}}}{8}.$$

Accordingly, (3) holds if and only if we have:

$$f_q(n) < \frac{\ln\left(-3q^{-\frac{1}{n+1}} + \sqrt{96 + 9q^{-\frac{2}{n+1}}}\right) - \ln 8}{\ln q}. \tag{4}$$

In which case the quantity $f_q(n)$ has to satisfy the following inequality:

$$f_q(n) < \min\left\{\frac{1}{n+1}, \frac{\ln\left(-3q^{-\frac{1}{n+1}} + \sqrt{96 + 9q^{-\frac{2}{n+1}}}\right) - \ln 8}{\ln q}\right\}.$$

Remark 3.3. The integers A_j $(j = 1, \dots, n)$ satisfy the inequality

$$\frac{q^{\frac{j}{n+1} + f_q(n)}}{2} < A_j < \frac{q^{\frac{j}{n+1} + f_q(n)}}{1.5}.$$

If the interval $\left[q^{\frac{j}{n+1} + f_q(n)}/2, q^{\frac{j}{n+1} + f_q(n)}/1.5\right]$ has length > 1, equivalently $q^{\frac{j}{n+1} + f_q(n)} > 6$, then it contains an integer. Therefore, if

$$f_q(n) > \frac{\ln 6}{\ln q} - \frac{1}{n+1},$$

then all the above intervals contain an integer.

Next, let us recall Hermite's Theorem which provides an estimate for the smallest vector of a lattice.

Proposition 3.4. *(Hermite's theorem). Let n be a positive integer. There is a constant $\gamma_n \in (0, n]$ such that, for every full rank lattice $\mathcal{L} \subset \mathbb{R}^n$ we have:*

$$\lambda_1(\mathcal{L}) \leq \sqrt{\gamma_n}\, (\det \mathcal{L})^{\frac{1}{n}}.$$

Proof. See [13, Theorem 1.5]. □

The quantity γ_n is called Hermite's constant. The exact value for γ_n is known only for $1 \leq n \leq 8$ and for $n = 24$. Furthermore, an asymptotic bound is given in [7, Chapter 1, p. 20]. In our case we have $\lambda_1(\mathcal{L}) \leq \sqrt{n+1}\, q^{\frac{n}{n+1}}$, and so Proposition 3.1 yields:

$$\frac{1}{2}\, q^{\frac{n}{n+1}+f_q(n)} \leq \lambda_1(\mathcal{L}) \leq \sqrt{n+1}\, q^{\frac{n}{n+1}}.$$

It follows:

$$f_q(n) \leq \frac{\ln(2\sqrt{n+1})}{\ln q}. \tag{5}$$

It is easily seen that, the bound of the inequality (5) is larger than that of (4). Therefore, Hermite's result does not add any restriction on the choice of $f_q(n)$ and so, for $f_q(n)$ we can take any real number satisfying the inequalities of Remarks 3.2 and 3.3.

The following Proposition improves in some sense Proposition 3.1. For instance in Proposition 3.1, n can not be freely chosen. In the following Proposition, we fix A_j's to some suitable values, and this allows us to consider larger values of the number of signatures n.

Proposition 3.5. *Let n and q be positive integers. Set $A_j = \lfloor Cq^{j/(n+1)+g_q(n)} \rfloor + 1$ $(j = 1, \ldots, n)$, where C and $g_q(n) \in (0, 1)$. Furthermore, we assume that*

$$g_q(n) < \frac{1}{n+1} \tag{6}$$

and, for $j = 1, \ldots, n-1$,

$$\max\left\{q^{j/(n+1)} A_{n+1-j}, Cq^{n/(n+1)+g_q(n)} A_1\right\} < q - Cq^{n/(n+1)+g_q(n)}. \tag{7}$$

Denote by $\mathcal{L}$ the lattice generated by the vectors

$$\mathbf{b}_0 = (-1, A_1, \ldots, A_n), \mathbf{b}_1 = (0, q, 0, \ldots, 0), \ldots, \mathbf{b}_n = (0, \ldots, 0, q).$$

Then, for all nonzero $\mathbf{v} \in \mathcal{L}$, we have:

$$\|\mathbf{v}\| > Cq^{\frac{n}{n+1}+g_q(n)}.$$

Proof. Assume that there is a nonzero vector $\mathbf{v} \in \mathcal{L}$ such that,

$$\|\mathbf{v}\| \leq Cq^{n/(n+1)+g_q(n)}.$$

Then, from inequality (6) and the fact that $C \in (0,1)$ we get $\|\mathbf{v}\| < q$. Also $\mathbf{v} \in \mathcal{L}$, so there are integers $x_0, \ldots, x_n$ such that

$$\mathbf{v} = x_0\mathbf{b}_0 + \cdots + x_n\mathbf{b}_n = (-x_0, x_0A_1 + x_1q, \ldots, x_0A_n + x_nq).$$

Then, we have,

$$|x_0|, |x_0A_j + x_jq| \leq Cq^{n/(n+1)+g_q(n)} \quad (j = 1, \ldots, n).$$

If $x_0 = 0$, then $\mathbf{v} = (0, x_1q, \ldots, x_nq)$ and so, $\|\mathbf{v}\| \geq q$ which is a contradiction, and so $x_0 \neq 0$.

Since $1 \leq |x_0| \leq Cq^{n/(n+1)+g_q(n)}$, we consider the following two cases.

(**i**) Assume that,

$$q^{(k-1)/(n+1)} < |x_0| < q^{k/(n+1)},$$

for some $k \in \{1, 2, \ldots, n-1\}$. Thus, we get :

$$Cq^{n/(n+1)+g_q(n)} < |x_0|A_{n+1-k} < q^{k/(n+1)}A_{n+1-k}.$$

On the other hand, inequality (7) yields,

$$q^{k/(n+1)}A_{n+1-k} < q - Cq^{n/(n+1)+g_q(n)}.$$

Combining the two previous inequalities we obtain,

$$Cq^{n/(n+1)+g_q(n)} < |x_0|A_{n+1-k} < q - Cq^{n/(n+1)+g_q(n)} \tag{8}$$

If $x_0A_{n+1-k} + qx_{n+1-k} = 0$, then we get

$$0 \neq |x_0|A_{n+1-k} = q|x_{n+1-k}| > q,$$

which contradicts with inequality (8). Assume that, $x_{n+1-k} \neq 0$. Since $\|\mathbf{v}\| \geq |x_0A_{n+1-k} + qx_{n+1-k}|$, we get

$$\|\mathbf{v}\| \geq \big||x_0|A_{n+1-k} - q|x_{n+1-k}|\big| \geq q|x_{n+1-k}| - |x_0|A_{n+1-k}. \tag{9}$$

So,

$$\|\mathbf{v}\| \geq q - |x_0|A_{n+1-k}.$$

Then, using the right part of inequality (8), we get,

$$\|\mathbf{v}\| > Cq^{n/(n+1)+g_q(n)}$$

which is a contradiction. Thus, we have $x_{n+1-k} = 0$. Then, inequality (9) yields,

$$\|\mathbf{v}\| \geq |x_0|A_{n+1-k} > Cq^{n/(n+1)+g_q(n)}$$

which is again a contradiction.

(**ii**) We assume now that,

$$q^{(n-1)/(n+1)} < |x_0| < Cq^{n/(n+1)+g_q(n)}.$$

So,
$$Cq^{n/(n+1)+g_q(n)} < |x_0|A_1 < Cq^{n/(n+1)+g_q(n)}A_1.$$
By (7), we get,
$$Cq^{n/(n+1)+g_q(n)}A_1 < q - Cq^{n/(n+1)+g_q(n)}.$$
Combining the two previous inequalities we obtain,
$$Cq^{n/(n+1)+g_q(n)} < |x_0|A_1 < q - Cq^{n/(n+1)+g_q(n)},$$
which is relation (8) (for $k = n$). Accordingly, we proceed as previously but we set $k = n$, and we finally get a contradiction. The Proposition follows. □

Remark 3.6. It is easily seen that
$$Cq^{g_q(n)} < \frac{q^{1/(n+1)}}{1 + q^{1/(n+1)}}.$$

4. A System of Linear Congruences

In this Section we give two results which yield sufficient conditions for the success of our attacks.

Proposition 4.1. *Let q and A_i, B_i $(i = 1, \ldots, n)$ and $f_q(n)$ as in Proposition 3.1. Set*
$$M_{n,q} = \frac{1}{4}\, q^{\frac{n}{n+1}+f_q(n)}.$$
Then, the system of congruences
$$y_i + A_i x + B_i \equiv 0 \pmod{q} \quad (i = 1, \ldots, n) \tag{10}$$
has at most one solution $\mathbf{v} = (x, y_1, \ldots, y_n)$ having
$$\|\mathbf{v} - \mathbf{e}\| < M_{n,q},$$
for some $\mathbf{e} \in \mathbb{R}^{n+1}$. If such a solution exists we can find it using a CVP oracle.

Proof. Let $\mathbf{v} = (x, y_1, \ldots, y_n)$ be a solution of the system with
$$\|\mathbf{v} - \mathbf{e}\| < M_{n,q}.$$
Let $\mathcal{L}$ be the lattice spanned by the rows of the $(n+1) \times (n+1)$ matrix
$$\begin{bmatrix} -1 & A_1 & A_2 & \ldots & A_n \\ 0 & q & 0 & \ldots & 0 \\ 0 & 0 & q & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & q \end{bmatrix} \tag{11}$$

and set $\mathbf{b}_{old} = (0, B_1, \ldots, B_n)$ and $\mathbf{b}_{new} = \mathbf{b}_{old} + \mathbf{e}$. Since $y_i + A_i x + B_i \equiv 0 \pmod q$ there is a $z_i \in \mathbb{Z}$, such that $y_i + B_i = -A_i x + z_i q$. Let

$$\mathbf{u} = \mathbf{v} + \mathbf{b}_{old} = (x, y_1 + B_1, \ldots, y_n + B_n).$$

Then $\mathbf{u} = (x, -A_1 x + z_1 q, \ldots, -A_n x + z_n q)$ belongs to $\mathcal{L}$ and we have

$$\|\mathbf{u} - \mathbf{b}_{new}\| = \|\mathbf{v} - \mathbf{e}\| < M_{n,q}.$$

Alternatively, using the CVP-oracle with input the lattice $\mathcal{L}$ and target vector $\mathbf{b}_{new}$ outputs a vector $\mathbf{w}$ such that,

$$\|\mathbf{w} - \mathbf{b}_{new}\| \leq \|\mathbf{u} - \mathbf{b}_{new}\| < M_{n,q}. \tag{12}$$

Thus we get,

$$\|\mathbf{w} - \mathbf{u}\| \leq \|\mathbf{w} - \mathbf{b}_{new}\| + \|\mathbf{b}_{new} - \mathbf{u}\| < \frac{1}{2} q^{\frac{n}{n+1} + f_q(n)}.$$

Since $\mathbf{w} - \mathbf{u} \in \mathcal{L}$, Proposition 3.1 implies $\mathbf{w} = \mathbf{u}$. So the CVP oracle outputs the vector $\mathbf{w}$ and so we can compute

$$\mathbf{v} = \mathbf{w} - \mathbf{b}_{new} + \mathbf{e} = \mathbf{w} - \mathbf{b}_{old}. \tag{13}$$

□

Remark 4.2. (**i**) Taking $\mathbf{e} = \mathbf{0}$, we get Proposition 2 of [1], which is an improvement of Theorem 3.1 of [29].
(**ii**) If $\mathbf{u} \in \mathcal{L}$, then the entries of the vector $\mathbf{u} - \mathbf{b}_{old}$ satisfy the system (10). Indeed, since $\mathbf{u} \in \mathcal{L}$, there are integers $l_0, \ldots, l_n$ such that $\mathbf{u} = l_0 \mathbf{b_0} + \cdots + l_n \mathbf{b_n}$, and so we get:

$$\mathbf{u} - \mathbf{b}_{old} = (-l_0, l_0 A_1 + l_1 q - B_1, \ldots, l_0 A_n + l_n q - B_n) = (x, y_1, \ldots, y_n).$$

Thus, we obtain $y_i + A_i x + B_i = l_i q \equiv 0 \pmod q$ $(i = 1, \ldots, n)$.
(**iii**) Note that if a CVP oracle finds a vector $\mathbf{w} \in \mathcal{L}$ such that,

$$\|\mathbf{w} - \mathbf{b}_{new}\| < \frac{1}{4} q^{\frac{n}{n+1} + f_q(n)} = M_{n,q},$$

this does not imply that $\mathbf{w} = \mathbf{u}$. Indeed, it may occur instead of inequality (12), to have the inequality :

$$\|\mathbf{w} - \mathbf{b}_{new}\| < M_{n,q} \leq \|\mathbf{u} - \mathbf{b}_{new}\|.$$

When we apply our attack, we do not know if inequality (12) holds. Therefore, it is make sense to check if the following inequality,

$$\|\mathbf{w} - \mathbf{b}_{new}\| < M_{n,q}$$

holds. However, there are instances that satisfy the previous inequality, but they fail to find the secret key.

Using Proposition 3.5 we obtain the following alternative result to the previous Proposition.

Proposition 4.3. *Let q, A_i, B_i $(i = 1, \dots, n)$, C and $g_q(n)$ be as in Proposition 3.5. Set*

$$N_{n,q,C} = \frac{C}{2}\, q^{\frac{n}{n+1} + g_q(n)}.$$

Then, the system of congruences

$$y_i + A_i x + B_i \equiv 0 \pmod{q}$$

has at most one solution $\mathbf{v} = (x, y_1, \dots, y_n)$ having

$$\|\mathbf{v} - \mathbf{e}\| < N_{n,q,C},$$

for some $\mathbf{e} \in \mathbb{R}^{n+1}$. If such a solution exists, then we can find it using a CVP oracle.

5. Babai's Attack

This section is devoted to the description of an attack to (EC)DSA scheme based on Babai's algorithm which can be made rigorous if a plausible condition holds. First, we provide an auxiliary construction of a linear system of congruences, crucial to our attack.

5.1. Construction of a linear system

Let m_i be messages signed with (EC)DSA system and (r_i, s_i) their signatures $(i = 1, \dots, n)$, respectively. Then, there are $k_i \in \{1, \dots, q-1\}$ such that $r_i = (g^{k_i} \bmod p) \bmod q$ (resp. $r_i = x_i \bmod q$ and $k_i P = (x_i, y_i)$) and $s_i = k_i^{-1}(h(m_i) + a r_i) \bmod q$. It follows that

$$k_i + C_i a + D_i \equiv 0 \pmod{q} \quad (i = 1, \dots, n),$$

where $C_i = -r_i s_i^{-1} \bmod q$ and $D_i = -s_i^{-1} h(m_i) \bmod q$. Multiplying both sides by $C_i^{-1} \bmod q$, we get:

$$C_i^{-1} k_i + a + C_i^{-1} D_i \equiv 0 \,(\mathrm{mod}\, q).$$

We choose $f_q(n)$ and integers A_i satisfying the hypothesis of Proposition 3.1. We multiply by A_i both sides of the above congruence and we get:

$$A_i C_i^{-1} k_i + A_i a + A_i C_i^{-1} D_i \equiv 0 \,(\mathrm{mod}\, q).$$

Set $B_i = A_i C_i^{-1} D_i \bmod q$ $(i = 1, \dots, n)$. We call the linear system

$$y_i + A_i x + B_i \equiv 0 \,(\mathrm{mod}\, q) \quad (i = 1, \dots, n), \tag{14}$$

the *DSA-system* associated to n, A_i and m_i. The vector

$$(a, A_1 C_1^{-1} k_1 \bmod q, \dots, A_n C_n^{-1} k_n \bmod q)$$

satisfies the above system. We call the integer $k_i' = A_i C_i^{-1} k_i \bmod q$ the *derivative ephemeral key* corresponding to the ephemeral key k_i.

5.2. The Attack

Suppose that a public key (p, q, g, R) of a DSA scheme or a public key (E, p, q, P, Q) of a ECDSA scheme is given.

Babai's Attack

Input: l signed messages m_i corresponding to the above public key and (r_i, s_i) their signatures $(i = 1, \ldots, l)$.

Output: The secret key or Fail.

1. Choose n with $0 < n \leq l$ and $f_q(n)$ satisfying the hypothesis of Proposition 3.1, and such that for every $i = 1, \ldots, n$, the interval

$$I_i = \left(\frac{q^{\frac{i}{n+1} + f_q(n)}}{2}, \frac{q^{\frac{i}{n+1} + f_q(n)}}{1.5} \right)$$

contains an integer. If such n does not exist, return fail. Otherwise, go to the next step.

2. Choose randomly A_i from I_i.

3. Construct the DSA-system

$$y_i + A_i x + B_i \equiv 0 \, (\bmod \, q) \quad (i = 1, \ldots, n)$$

associated to n, m_i and A_i.

4. Choose $\mathbf{e} \in \mathbb{R}^{n+1}$.

5. Set $\mathbf{b} = (0, B_1, \ldots, B_n)$ and construct the lattice $\mathcal{L}$, generated by the rows of the matrix M as in (11).

6. Compute $B = LLL(M)$.

7. Apply Babai's Nearest Plane Algorithm in the rows of matrix B with target vector $\mathbf{b} + \mathbf{e}$ and let $\mathbf{s}$ be the output.

8. If the first coordinate s_1 of $\mathbf{s}$ satisfies either $g^{s_1} = R$ in $\mathbb{F}_p^*$, (respectively $Q = s_1 P$ in $E(\mathbb{F}_p)$) or $g^{s_1 \pm 1} = R$ (respectively $Q = (s_1 \pm 1)P$) return s_1, else return fail.

Remark 5.1. (**i**) In step 2, we can choose A_j as in Proposition 3.5. That is, $A_j = \lfloor C q^{j/(n+1) + g_q(n)} \rfloor + 1$ $(j = 1, \ldots, n)$.

(**ii**) In step 6, we can apply BKZ instead of applying LLL, in order to get a more reduced basis.

The following Proposition provides a sufficient condition for the success of the previous algorithm.

Proposition 5.2. *Set $\Omega = M_{n,q}$ or $N_{n,q,C}$ (for the definitions of the constants see Propositions 4.1 and 4.3, respectively). If*

$$\|(a, k_1', \ldots, k_n') - \mathbf{e}\| < \Omega,$$

then a CVP oracle implies that the output of the above algorithm s_1 is the secret key a.

Proof. Let $\Omega = M_{n,k}$. Then, we consider quantities A_i defined as in Proposition 3.1 and we construct the DSA-system associated to n, A_i and m_i $(i = 1, \ldots, n)$. The system has as a solution the vector $(a, k'_1, \ldots, k'_n)$. By Proposition 4.1, the previous DSA-system has at most one solution $\mathbf{v}$ satisfying

$$\|\mathbf{v} - \mathbf{e}\| < \Omega,$$

and $\mathbf{v}$ can be found by using a CVP oracle. Hence, this oracle implies that $s_1 = a$ and so, the above algorithm provides the secret key a. If $\Omega = N_{n,q,C}$, then we consider quantities A_i defined as in Proposition 4.3 and we construct the DSA-system associated to n, A_i and m_i $(i = 1, \ldots, n)$. The vector $(a, k'_1, \ldots, k'_n)$ is a solution of this system and so, as previously, Proposition 4.3 implies that $s_1 = a$, since this solution is unique. □

Remark 5.3. In the algorithm Babai's Attack we use as a CVP oracle the Babai Nearest Plane Algorithm. Since this algorithm solve only CVP_{γ_n}, in many cases it does not provide us with the solution of CVP but with a vector close to it. Thus, in the above algorithm, in order to find the secret key, we ask the verification not only of the equality $g^{s_1} = R$ (respectively $Q = s_1 P$), but also of the equalities $g^{s_1 \pm 1} = R$ (respectively $Q = (s_1 \pm 1)P$).

By Proposition 3.1 (respectively Proposition 3.5), we have $\lambda_1(\mathcal{L}) > 2M_{n,q}$ (respectively $\lambda_1(\mathcal{L}) > 2N_{n,q,C}$). So, in case that we know the quantity $\lambda_1(\mathcal{L})$, the bound Ω of Proposition 5.2 can be replaced by $\lambda_1(\mathcal{L})/2$. Recall however that, the Gaussian heuristic implies that $\lambda_1(\mathcal{L}) \approx \sqrt{(n+1)/2\pi e}\, q^{n/n+1}$. On the other hand, we observe that the inequality of Proposition 5.2 is only sufficient. Thus, it is possible to find the secret key a, even if the inequality is not being satisfied.

Note that, there is a variant of CVP which is called BDD : Bounded Distance Decoding problem. In the latter problem we search for lattice vectors $\mathbf{u}$ such that,

$$\|\mathbf{u} - \mathbf{t}\| \leq \lambda_1(\mathcal{L})/2.$$

Moreover, there are enumeration algorithms which compute all the lattice vectors within distance R from the target vector, see for example [14, 16]. But, these algorithms do not run in polynomial time with respect to the rank of the lattice.

For the selection of the vector $\mathbf{e}$, we "guess" a vector $\mathbf{e}$ such that the length of $(a, y_1, \ldots, y_n) - \mathbf{e}$ is less than the quantity $\lambda_1(\mathcal{L})/2$. This is not an easy task. We deal with this issue in the next section.

5.3. Some Variants of the Attack

As in [29], we can transform our congruences and obtain new systems of congruences in order to apply our attack. More precisely, multiplying by $a^{-1} \mod q$ the congruence

$$k_j + C_j a + D_j \equiv 0 \pmod{q} \quad (j = 1, \ldots, n),$$

we get

$$k_j a^{-1} + C_j + D_j a^{-1} \equiv 0 \pmod q \quad (j = 1, \ldots, n).$$

Thus, replacing (C_j, D_j) by (D_j, C_j) and a by a^{-1}, we obtain a variant of our attack under which it is possible to compute the inverse $a^{-1} \bmod q$ and so a.

Suppose now that $n \geq 2$. So, we can eliminate a among the congruences

$$k_j + C_j a + D_j \equiv 0 \pmod q \quad (j = 1, \ldots, n)$$

and we easily get the congruences

$$k_j + \tilde{C}_j k_n + \tilde{D}_j \equiv 0 \pmod q \quad (j = 1, \ldots, n-1).$$

Where $\tilde{C}_j = -C_j C_n^{-1} \bmod q$ and $\tilde{D}_j = -C_j C_n^{-1} D_n + D_j \bmod q$. Replacing in our attack (C_j, D_j) by $(\tilde{C}_j, \tilde{D}_j)$ we get another variant which is possible to provide us k_n and so a.

Finally, multiplying by k_n^{-1} the congruences

$$k_j + \tilde{C}_j k_n + \tilde{D}_j \equiv 0 \pmod q \quad (j = 1, \ldots, n-1)$$

we obtain

$$k_j k_n^{-1} + \tilde{C}_j + \tilde{D}_j k_n^{-1} \equiv 0 \pmod q \quad (j = 1, \ldots, n-1).$$

So, we have another attack which is possible to provide k_n^{-1}, and so a.

6. Heuristic Attack

We start with two definitions. These will be used in our attacks.

Definition 6.1. Let q be a prime with binary length ℓ bits and $x, c \in \mathbb{Z}_q$. Let $\mathcal{A}$ be a probabilistic polynomial algorithm which accepts (c, x, ℓ, PK), where PK is the public key of (EC)DSA-scheme, and returns

0, if the binary length of $cx \bmod q$ is ℓ bits,
1, if the binary length of $cx \bmod q$ is $\ell - 1$ bits,
2, if the binary length of $cx \bmod q$ is $< \ell - 1$ bits.

We call such an oracle *a length DSA oracle.*

In addition, we consider the following type of (binary) oracle.

Definition 6.2. Let $\mathcal{B}$ be a probabilistic polynomial algorithm which accepts a triple (x, ℓ, PK), where PK is the public key of (EC)DSA-scheme and $x \in \mathbb{Z}_q$. Then $\mathcal{B}$ returns True, if the binary length of $q - x$ is $\ell - 1$ bits, and False, otherwise. We call such an oracle *a binary length DSA oracle.*

We remark that, in the following algorithm (see next subsection) we implement the previous oracles as in the definition, i.e. as deterministic algorithms. Although, a cryptanalyst calls an oracle, either $\mathcal{A}$ or $\mathcal{B}$, with the public key and a message m (or the signature (r, s)) and gets the right information for the derivative key. For instance, the cryptanalyst calls $\mathcal{A}$ and gets 0, if the derivative key has binary length ℓ.

In the previous oracles, the value of ℓ and the public key PK are fixed, so formally the real input is x in the case of $\mathcal{B}$ and x, c in the case of $\mathcal{A}$. Usually, $\ell \in \{160, 224, 256\}$. In our case we shall choose the constant $c = A_i C_i^{-1} \mod q$ (see Subsection 5.1). Accordingly, we use oracle $\mathcal{A}$ for determining a bound for the length of the derivative ephemeral keys. Observe that, having such an oracle is more weak than knowing the MSB of the derivative keys and the secret key. We shall use oracle $\mathcal{B}$, when the derivative ephemeral keys have binary length ℓ bits.

6.1. *Conditional Babai's Attack*

Let PK be a (EC)DSA public key and ℓ be the binary length of q. We assume that we have a length and a binary length DSA oracle, $\mathcal{A}$ and $\mathcal{B}$, respectively. We fix ℓ and PK. Given $x, c \in \mathbb{Z}_q$, we denote by $\mathcal{A}(cx)$ and $\mathcal{B}(x)$ the outputs of $\mathcal{A}$ and $\mathcal{B}$ at cx and x, respectively.

Conditional Babai's Attack
Input: l signed messages m_i corresponding to a (EC)DSA public key and their signatures (r_i, s_i) $(i = 1, \dots, l)$.
Output: The secret key or Fail.
1. Choose n with $0 < n \leq l$ and $f_q(n)$ satisfying the hypothesis of Proposition 3.1, and such that for every $i = 1, \dots, n$, the interval

$$I_i = \left(\frac{q^{\frac{i}{n+1}+f_q(n)}}{2}, \frac{q^{\frac{i}{n+1}+f_q(n)}}{1.5} \right)$$

contains an integer. If such n do not exist, return fail. Otherwise, go to the next step.
3. Choose randomly A_i from I_i.
4. Let k'_i as previously the derivative ephemeral key corresponding to the nonce k_i. Construct the DSA-system as follows:
For $i = 1, \dots, n$,

4a. if $\mathcal{A}(k'_i) = 0$, **then**
if $\mathcal{B}(k'_i) =$ True, consider the congruence,

$$(-y_i) + (-A_i)x + (-B_i) \equiv 0 \,(\mathrm{mod}\, q).$$

else, consider the congruence,

$$(2^{\ell-2} - y_i) + (-A_i)x + (-2^{\ell-2} - B_i) \equiv 0 \,(\mathrm{mod}\, q).$$

4b. if $\mathcal{A}(k'_i) = 1$, **then** do not modify the $i-$ equation.
4c. if $\mathcal{A}(k'_i) = 2$, **then** consider the congruence,

$$(2^{\ell-2} + y_i) + A_i x + (-2^{\ell-2} + B_i) \equiv 0 \,(\mathrm{mod}\, q).$$

4d. Let $A'_1, \dots, A'_n$ and $B'_1, \dots, B'_n$ be the coefficients of variable x and the constant terms, respectively, of the congruences constructed in steps **4a**, **4b** and **4c**. Thus we have the following system:

$$y_i + A'_i x + B'_i \equiv 0 \pmod{q} \quad (i = 1, \dots, n).$$

5. Consider the square matrix M having as rows $(-1, A'_1, \dots, A'_n)$, $(0, q, 0, \dots, 0)$,..., $(0, \dots, 0, q)$ and denote by $\mathcal{L}$ the lattice generated by the rows of M. Further, set $\mathbf{b} = (0, B'_1, \dots, B'_n)$ and $\mathbf{e} = (2^{\ell-2} + 2^{\ell-3}, \dots, 2^{\ell-2} + 2^{\ell-3})$.
6. Compute $B = LLL(M)$.
7. Apply Babai's Nearest Plane Algorithm in the rows of matrix B with target vector $\mathbf{b} + \mathbf{e}$ and let $\mathbf{s}$ be its output.
8. If the first coordinate s_1 of $\mathbf{s}$ satisfies $g^{s_1} = R$, (respectively $Q = s_1 P$) in $\mathbb{F}_p^*$, return s_1, else return fail.

6.2. The case $\mathcal{A}(k'_i) = 0$.

Assume without loss of generality that $\ell = 160$. We consider the following assumption:
Assumption-1. All the derivative ephemeral keys have 160 *bits.*
Then, we can exploit the fact that $q - a$ and $q - k'_i$ have at most 159 bits. So by adding and subtracting to the DSA-system the number 2^{158}, we balance the new solution set to 159 bits (except maybe the first entry concerning to the secret key a). In this way, we can choose $\mathbf{e} = (2^{158} + 2^{157}, \dots, 2^{158} + 2^{157})$. So, in this case, we modify step **4** of the previous algorithm.

4. Let k'_i be as previously the derivative ephemeral key corresponding to the nonce k_i. Construct the DSA-system as follows:
4a. **if** $\mathcal{B}(k'_i) =$ True, consider the congruence,

$$(-y_i) + A_i(-x) + (-B_i) \equiv 0 \pmod{q}.$$

else, consider the congruence,

$$(2^{\ell-2} - y_i) + A_i(-x) + (-2^{\ell-2} - B_i) \equiv 0 \pmod{q}.$$

4b. Let B'_i $(i = 1, \dots, n)$ be the constant terms of the congruences of the system constructed in **4a**-step. Thus we have the following system:

$$y_i + A_i x + B'_i \equiv 0 \pmod{q} \quad (i = 1, \dots, n).$$

We applied the previous attack which has success rate 83% for q with 160 bits, and $n = 204$ signatures, and 70% if q has 256 bits and we have $n = 300$ signatures[1] (see also section 7 for the details of the experiments).

[1] `https://github.com/drazioti/python_scripts/blob/master/paper_dsa/improved_attack.py`

Suppose now that $q = 2^{159} + 2^{\alpha} + \cdots + 1$, with $\alpha \leq 156$. Since $k_i' < q$ and the size of k_i' is 160 bits, we have $k_i' = 2^{159} + \varepsilon_{156} 2^{156} + \cdots + \varepsilon_0 2^0$, where $\varepsilon_i \in \{0, 1\}$. Thus, the size of $q - k_i'$ is at most 157 bits. So, in a DSA system, we first multiply with -1 and then, we add and subtract $2^{\alpha+1}$ (instead of 2^{158}). So we deal with shorter keys than 159 bits, as we have suggested in the previous algorithm.

Remark 6.3. (**i**) Note that we do not use our oracle $\mathcal{B}$ for the secret key a, but only for the derivative ephemeral keys.
(**ii**) The attack 6.2 is very close to the heuristic attack proposed in [1]. The former attack uses secret keys having ≤ 159 bits, but the latter uses 160 bits. Also, we have a weaker assumption than in [1]. Oracle $\mathcal{B}$ can work even if we do not know the exact binary length of $q - k_i'$, where k_i' are the derivative ephemeral keys.

6.3. An application

For instance the following ECDSA systems (recommended by Standards for Efficient Cryptography Group (SECG)) can easily be broken under *Assumption-1*, since they use a parameter q of the previous form.

(**i**) The elliptic curve secp160k1 : $y^2 = x^3 + 7$, see [31], is defined over the prime finite field $\mathbb{F}_p$, where

$$p = 2^{160} - 2^{32} - 2^{14} - 2^{12} - 2^9 - 2^8 - 2^7 - 2^3 - 2^2 - 1,$$

and the base point has order:

$$\begin{aligned} q &= 1461501637330902918203686915170869725397159163571 \\ &= 2^{160} + 2^{80} + \cdots + 2^5 + 2^4 + 2 + 1, \end{aligned}$$

which is a 161 bits prime. The order of the elliptic curve is

$$|E(\mathbb{F}_p)| = 1461501637330902918203686915170869725397159163571$$

and the cofactor $h = |E(\mathbb{F}_p)|/q = 1$. This curve is used in `TinyECC`[2] which has applications in sensor networks. The ECDSA scheme with the previous parameter is vulnerable since $q = 2^{160} + q'$ with $q' \ll 2^{157}$. This curve is supported by OpenSSL (ver. 1.1.0j, 20 Nov 2018)[3]. Albeit in this specific curve we can do even better. Since the secret and all the derivative ephemeral keys are 161 bits then $q - a$ and $q - k_i'$ are at most 81 bits. Set $e_i = e_{i,0} 2^{80} + e_{i,1} 2^{79} + e_{i,2} 2^{78}$ $(i = 1, 2)$ $e_{i,0}, e_{i,1}, e_{i,2} \in \{0, 1\}$ and $\mathbf{e} = (e_1, e_2)$. Then, there is such a vector $\mathbf{e}$ satisfying

$$\|\mathbf{v} - \mathbf{e}\| < 2^{78} < \frac{1}{4}\, q^{1/2 + f_q(n)}.$$

[2] `http://discovery.csc.ncsu.edu/software/TinyECC/`
[3] We made the check with the (Linux) command `openssl ecparam -list_curves`

Thus Proposition 5.1 implies that $q-a$ can be computed and hence a. The number of the above vectors $\mathbf{e}$ is 64. Furthermore, since the dimension of the involved lattice is 2 (set $n=1$ to the matrix (11)) we can use as a CVP oracle the Micciancio - Voulgaris algorithm [22]. Therefore, if the secret and a derivative ephemeral key, are 161 bits, then using only one signature we compute the secret key a in polynomial time.

(**ii**) The elliptic curve secp224k1 : $y^2 = x^3+5$, see [31], is defined over the prime finite field $\mathbb{F}_p$, where

$$p = 2^{224} - 2^{32} - 2^{12} - 2^{11} - 2^{9} - 2^{7} - 2^{4} - 2 - 1,$$

and the base point has order:

$$\begin{aligned} q &= 26959946667150639794667015087019640346510327083120074548994958668279 \\ &= 2^{224} + 2^{112} + 2^{111} + \cdots + 2^{4} + 2^{2} + 2 + 1, \end{aligned}$$

which is a prime of 225 bits, having the form $q = 2^{224} + q'$ with $q' \ll 2^{222}$. So, this curve is vulnerable to the previous attack. Furthermore, it is also supported by OpenSSL.

7. Experimental Results

For the following experiments we used the computer algebra system Sagemath [30] in a Linux PC with I3 Intel CPU and 16GB memory.

7.1. *Conditional Babai's attack*

We applied the algorithm[4] Conditional Babai's attack of Section 6. So, we assume that we have the two oracles $\mathcal{A}$ and $\mathcal{B}$. The systems we generated have solutions $\mathbf{s}$, such that

$$\mathbf{s} \in \{2^{159}, \ldots, 2^{160}-1\} \times \mathbb{Z}_q^n.$$

That is, the secret key has 160 bits and the derivative ephemeral keys are $< q$. So, the solutions do not have any constraints. For preprocessing we used BKZ-70 and for each instance we considered a different prime number q. We also choose,

$$f_q(n) = \min\left\{ \frac{1}{n+1}, \frac{\ln\left(-3q^{-\frac{1}{n+1}} + \sqrt{96 + 9q^{-\frac{2}{n+1}}}\right) - \ln 8}{\ln q} \right\} - 10^{-10}.$$

We generated 100 random DSA systems with $n = 204$ and we found 64 secret keys. On average the wall time per example, was approximately 1 min.

[4] The code can be found in `https://github.com/drazioti/python_scripts/tree/master/paper_dsa`

All the derivative ephemeral keys have 160 *bits.*
We tested the attack of Subsection 6.2 for primes q having 160 bits and $n = 204$ signatures. Our algorithm in 100 random instances found the secret keys in 83 instances. The (wall) time execution per example was about 2 minutes (this time is dominated by the preprocessing step). So, having only a binary length oracle we can find the secret key.

Although the following reasonable question arises; *how many signatures do we need to collect, to get* 204 *signatures*[5] *such that, their derivative ephemeral keys have exactly* 160 *bits, assuming that all the ephemeral keys have* 160 *bits?* Experimentally[6] we noticed that the answer depends on the form of the prime number q. Let DSA_1, DSA_2 two DSA schemes corresponding to public parameters q_1, q_2. Assume that q_1, q_2 belong to the interval $[2^{159}, 2^{160} - 1]$ and $q_1 > q_2$. Then, the derivative keys of DSA_1 have more often (on average) 160 bits that in DSA_2. This is depicted more clearly in Figure 1. We executed three experiments. Each experiment was described with a different color and it corresponds to a different form of the prime q. For instance, the green line concerns a prime of the form : $2^{159} + 2^{158} + q'$ where q' was picked randomly from the real interval $(0, 2^{157})$. The red line corresponds to a prime of the form $2^{159} + 2^{158} + 2^{157} + q'$ and similar for the blue line. For each q we considered 2000 (EC)DSA derivative ephemeral keys. We note that, *dense* primes q seems more vulnerable to this attack. For instance such a *dense* q is the one used in the bitcoin curve $y^2 = x^3 + 7$, where $q = 2^{255} + 2^{254} + \cdots + 2^{129} + q'$.

Finally, this attack improves the success rate of the heuristic attack provided in [1, Section 5]. The latter deploys an attack using a secret key with at most 160 bits and all the derivative ephemeral keys have at most 159 bits. I.e.

$$\mathbf{s} \in \{2^{\alpha-1}, \ldots, 2^{\alpha} - 1\} \times \{2^{\beta-1}, \ldots, 2^{\beta} - 1\}^n \ (\alpha \leq 160,\ \beta \leq 159).$$

In the present paper we did not consider any constraint in the solution, but we assumed that we have a binary length oracle. This decides if a $q - k_i'$ (where k_i' is a derivative ephemeral key) has 159 bits or not.

Further, we executed the same experiment, but for primes q having 256 bits. So we assumed that all the derivative ephemeral keys have 256 bits. This is the case where the prime p of DSA has 2048 or 3072 bits (see [11, Section 4.2]). In this case we used again BKZ with blocksize 70 and having $n = 300$ signatures, we found 70 secret keys for 100 random DSA-systems. The average wall time per experiment was 5 minutes.

7.2. Babai's Attack.

Finally, in the following table (Table 1) we provide an update of [1, Table 2]. We applied the attack given by the algorithm, Babai's attack, of Subsection

[5]There is nothing special about $n = 204$. This number of signatures was chosen experimentally, in order to optimize the results of our experiments.

[6]`https://github.com/drazioti/python_scripts/blob/master/paper_dsa/experiments.py`

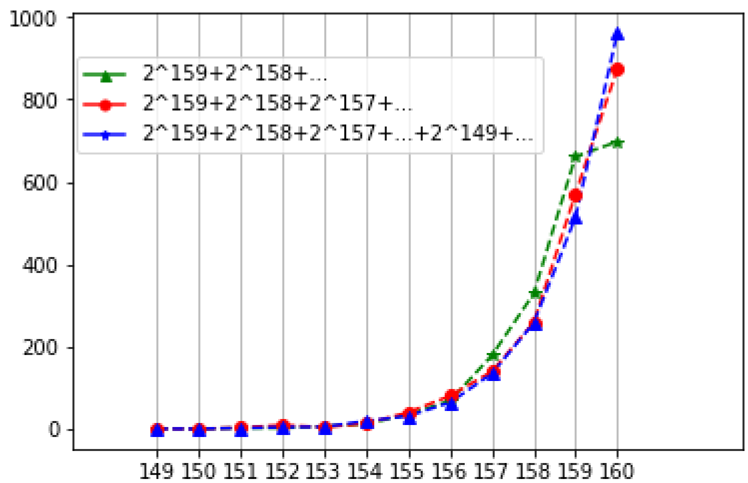

Figure 1: The $x-$axis shows the number of bits of the derivative keys and the $y-$axis shows the quantity of the derivative ephemeral keys having specific number of bits.

5.2. For all the rows except the last one, we set $n = 206$ and $g_q(n) = \frac{\ln(n+1)}{2n\ln q}$, $\mathbf{e} = \mathbf{0}$ and $C = 55/100$ (see Remark 5.1). Note that, the choice of $g_q(n)$ satisfies inequalities (6) and (7). We generated 100 random DSA systems for each row of Table 1, with specific binary length of derivative keys and the secret key. The pair (α, β) at the first column, means that we pick the secret key with α bits and the derivative ephemeral keys to have β bits (and we have fixed a prime q of 160 bits). For preprocessing we used LLL algorithm (instead of BKZ in [1]). The second column contains the percentage that Babai's attack succeeds in finding the solution i.e. the secret key. Note that, all solutions $\mathbf{y} = (y_i)_i$ that Babai's attack provides us, satisfy either $y_1 = s$ or $y_1 = s \pm 1$ (s is the secret key). For the last row we set $n = 215$, $\mathbf{e} = (2^{158} + 2^{157}, \ldots, 2^{158} + 2^{157})$ and C, g_q as previously.

bits:(Skey, Der.Ep.keys)	suc.rate
$(158, 157)$	100%
$(158, 155)$	100%
$(157, 157)$	100%
$(157, 156)$	100%
$(160, 159)$	70%

Table 1:

8. Conclusion

In the present work we consider extensions of the results presented in [1, 28]. In fact, we provide two heuristic attacks, namely Babais's Attack and Conditional Babai's attack. The first attack is an improvement of the paper [1]. The improvement is clear in Table 1.

The second attack is a new one. For instance, Conditional Babai's attack can be applied if we know the binary lengths of some multiples of the ephemeral keys. In fact, using our oracle, in the case where q has 160 bits, then 204 ephemeral keys are enough to find the secret key with success 83%, and 70% if q has 256 bits. What is more, the heuristic attack is supported by many theoretic evidences. On the other hand, in some real world cases the rigorous attack may be applied. Here with the word *rigorous* attack we mean any attack that satisfies the assumptions of Proposition 4.1. For instance, if we know the length of a (specific) multiple of an ephemeral key of the curve secp161k1, then we easily calculate rigorously and in polynomial time the secret key. Similar for the curve secp224k1.

The main reason that the attacks (and so the experiments) succeed so often even they do not satisfy the requirements of the theory, is the use of the two oracles, $\mathcal{A}$ and $\mathcal{B}$. These provide us with good predictions for the length of the derivative ephemeral keys. Consequently, the auxiliary vector $\mathbf{e}$ can be guessed well enough. In other words, if $\mathbf{e}$ is close to the solution of the DSA system (where with close we mean that their distance is close to $M_{n,q}$), then Babai's nearest plane algorithm works as a CVP-oracle and reveals the solution of the DSA system and from this we get the secret key. Finally, the attacks have been tested on instances that have already passed the two oracles $\mathcal{A}$ and $\mathcal{B}$ or only the oracle $\mathcal{B}$.

A drawback of our method is that we can not use the fault attacks to (EC)DSA, since they simulate an oracle that outputs some contiguous bits of the ephemeral keys, and not some multiples of them.

Acknowledgment

The authors sincerely thank the anonymous reviewers for their helpful suggestions.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

[1] M. Adamoudis, K. A. Draziotis and D. Poulakis, Enhancing an attack to DSA schemes, CAI 2019, p. 13–25. LNCS **11545**, Springer 2019.

[2] M. Bellare, S. Goldwasser and D. Micciancio, "Pseudo-random" number generation within cryptographic algorithms: the DSS case. In *Proc. of Crypto '97,* LNCS **1294**, IACR, Palo Alto, CA. Springer-Verlag, Berlin 1997.

[3] I. F. Blake and T. Garefalakis, On the security of the digital signature algorithm, *Des. Codes Cryptogr.,* **26**, no. 1-3 p. 87–96, 2002.

[4] Dan Boneh and R. Venkatesan, Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes. CRYPTO 1996. LNCS, vol. **1109**, p. 129–142. Springer, Heidelberg (1996).

[5] Billy Bob Brumley and Risto M. Hakala. Cache-timing template attacks. ASIACRYPT 2009, LNCS **5912**, p. 667–684, Springer-Verlag, 2009.

[6] Billy Bob Brumley and Nicola Tuveri, Remote timing attacks are still practical. ESORICS 2011, LNCS **6879**, p. 355–371, Springer-Verlag, 2011.

[7] J. Conway and N. Sloane, *Sphere Packings, Lattices and Groups.* Springer, 1998. Third edition.

[8] K. A. Draziotis and D. Poulakis, Lattice attacks on DSA schemes based on Lagrange's algorithm. 5th international Conference on Algebraic Informatics, CAI 2013. LNCS **8080**, p. 119-131, Springer 2013.

[9] K. A. Draziotis, (EC)DSA lattice attacks based on Coppersmith's method, Information Processing Letters **116(8)**, Elsevier (2016), p. 541–545.

[10] J. -L. Faugère, C. Goyet, and G. Renault, Attacking (EC)DSA Given Only an Implicit Hint, Selected Area of Cryptography, LNCS **7707**, p. 252–274, Springer-Verlag, Berlin - Heidelberg 2013.

[11] FIPS PUB 186-3, Federal Information Processing Standards Publication, Digital Signature Standard (DSS).

[12] S. Galbraith, Mathematics of Public key Cryptography, Cambridge university press, 2012.

[13] S. Goldwasser and D. Micciancio, Complexity of Lattice Problems: A Cryptographic Perspective, Springer, 2002.

[14] G. Harnot, X. Pujol and D. Stéhle, Algorithms for the Shortest and Closest Lattice Vector Problems. In: Chee Y.M. et al. (eds) Coding and Cryptology. IWCC 2011. LNCS **6639**. Springer (2011), Berlin, Heidelberg.

[15] A.I. Gomez,, D. Gomez-Perez & G. A. Renault, A probabilistic analysis on a lattice attack against DSA. Des. Codes Cryptogr. **87**, 2469–2488 (2019). `https://doi.org/10.1007/s10623-019-00633-w`.

[16] G. Harnot and D. Stéhle, Improved Analysis of Kannan's Shortest Lattice Vector Algorithm. In: Menezes A. (eds) Advances in Cryptology - CRYPTO 2007. CRYPTO 2007. LNCS **4622**. Springer (2007), Berlin, Heidelberg.

[17] M. Hlavac and T. Rosa, Extended hidden number problem and its cryptanalytic applications. SAC 2006, LNCS **4356**, p. 114–133, Springer (2006).

[18] N. A. Howgrave-Graham and N. P. Smart, Lattice Attacks on Digital Signature Schemes, *Des. Codes Cryptogr.* **23**, p. 283–290, 2001.

[19] D. Johnson, A. J. Menezes and S. A. Vanstone, The elliptic curve digital signature algorithm (ECDSA), *Intern. J. of Information Security,* **1**, p. 36–63, 2001.

[20] M. Liu and P. Q. Nguyen, Solving BDD by Enumeration: An Update, CT-RSA 2013, LNCS **7779**.

[21] A. J. Menezes, P. C. van Oorschot and S. A. Vanstone, *Handbook of Applied Cryptography,* CRC Press, Boca Raton, Florida, 1997.

[22] D. Micciancio and P. Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. *In Proc. of STOC, ACM*, p. 351-358, 2010.

[23] David Naccache, Phong Q. Nguyen, Michael Tunstall, and Claire Whealan. Experimenting with faults, lattices and the DSA. In Serge Vaudenay, editor, Public Key Cryptography, LNCS **3386**, p.s 16–28, Springer, 2005.

[24] National Institute of Standards and Technology (NIST). *FIPS Publication* 186*: Digital Signatur Standard.* May 1994.

[25] P. Q. Nguyen and I. E. Shparlinski, The Insecurity of the Digital Signature Algorithm with Partially Known Nonces, *J. Cryptology,* **15** p. 151-176, 2002.

[26] P. Q. Nguyen and I. E. Shparlinski, The Insecurity of the Elliptic Curve Digital Signature Algorithm with Partially Known Nonces, *Des. Codes Cryptogr.* **30**, p. 201-217, 2003.

[27] OpenSSL. http://www.openssl.org

[28] D. Poulakis, Some Lattice Attacks on DSA and ECDSA, *Applicable Algebra in Engineering, Communication and Computing,* **22**, p. 347-358, 2011.

[29] D. Poulakis, New lattice attacks on DSA schemes, *J. Math. Cryptol.* **10 (2)**, p. 135–144, 2016.

[30] Sage Mathematics Software, The Sage Development Team (version 8.1). http://www.sagemath.org.

[31] Standards for Efficient Cryptography, Certicom Research, Version 1, 2000, `https://www.secg.org/SEC2-Ver-1.0.pdf`