Lattice-based weak curve fault attack on ECDSA

Weiqiong Cao[1,2], Hongsong Shi[1], Hua Chen[2], Wei Wei[1]

[1] China Information Technology Security Evaluation Center. Building 1, yard 8, Shangdi West Road, Haidian District, Beijing 100085, China.
[2] Trusted Computing and Information Assurance Laboratory, Institute of Software, Chinese Academy of Sciences, South Fourth Street 4#, ZhongGuanCun, Beijing 100190, China.
Email: caoweqion@163.com, hsshi@163.com

Abstract. This paper proposes a new lattice-based weak curve fault attack on ECDSA, which assumes that a continuous bits block of curve parameter a is disturbed randomly by fault injection. Firstly, the faulty a' can be deduced by a distinguisher of quadratic residue, from which a weak curve with order n' is derived. Secondly, under the assumption that there exists a solvable smooth small factor d in n', we obtain some reduced information of the nonce by solving the ECDLP constructed in a small subgroup with order d. Finally, based on the reduced information, a model of lattice attack can be constructed to recover the signature private key by solving special instances of closest vector problem(CVP) in lattice.
Compared with the previous weak curve fault attacks, our attack increases the success rate of fault injection sharply, since it is not required that the constructed instance of ECDLP with order n' is practically solvable. In addition, the application of lattice-based attack is further extended in our attack by relaxing the restriction on the information leakage of nonces in comparison with the traditional partially known information attacks. Moreover, when there is a general random scalar masking in ECDSA, our attack still works without the additional masked bits leakage. Finally, the experiments demonstrate that the practical rate of effective faulty a' is up to 94.9% when the bit length of d is greater than 8, and the corresponding lattice attack is also feasible practically.

Keywords: ECC, Weak Curve Attack, Fault Attack, Lattice Attack, ECDSA

1 Introduction

1.1 Background

Elliptic curve digital signature algorithm (ECDSA) is one of the most commonly used signature algorithms of Elliptic Curve Cryptography (ECC). It is mainly used for authentication in network communication protocols (e.g., TLS protocol), financial IC cards and smart keys and various embedded cryptographic devices. Side channel attack (SCA) and fault attack (FA) are the most popular physical attacks on ECDSA. In the models of SCA on ECDSA, by the detection on the leakage of nonces or other intermediate values during the signature generation, an adversary can recover the private key in signature by solving specific instances of shortest vector problem (SVP) or closest vector problem (CVP) in the lattice constructed by the leakages [14,20,21]. Therefore, how to obtain the bits-leakage information of nonces should be an initial step of SCA. For example, Brumley etc. in ESORICS 2011 [4] target wNAF scalar multiplication in OpenSSL, and obtain some leaked bits of nonces by timing attack. In addition, flush + reload attack [2,1] is used to obtain the leaked bits by employing the flaws of instruction cache and scheduling (of CPU), while power analysis [12] and template attack [10] can also be used based on the collected power traces.

1.2 Previous Works

Fault attacks on ECDSA have the same purpose with SCA, i.e., managing to obtain the leaked information of nonces so as to recover the private key by transforming it to a lattice problem. In the model, fault injections such as laser injection, electromagnetic injection or voltage glitch interference are induced during signature generation procedure, which makes some steps skipped or some intermediates faulty. In PKC 2005 [18], Nacache et al. introduced lattice-based fault attack on DSA. Some least significant bits of nonces are set to 0 by inducing voltage glitch, and the private key in DSA is recovered by solving a CVP in some lattice. In addition, Schmidt et al. in FDTC 2009 [24] introduced a new differential fault model where a point addition or doubling operation during scalar multiplication is skipped by fault injection. Thereby, some bits of the nonces can be obtained by differential analysis. After that, Nguyen et al. [22] summarized this kind of fault attacks, and called them lattice-based fault attack. Then, Cao et al. in ICISC 2015 [6] also introduced a random fault model targeting the y-coordinate of intermediate point during the calculation of scalar multiplication, which can tolerate more random faulty bits. To sum up, all the ultimate purpose of the attacks is to obtain some leaked bits of nonces for constructing the condition of lattice attack. The approaches of fault analysis mainly include two categories. The first one is based on the fact that fault injection [18,22] is induced directly toward the nonces during signature generation to make some bits known or fixed, and the second one assumes that fault injection is induced into the calculation of scalar multiplication for constructing differential distinguisher [24,6].

Besides, another weak curve fault attack [15] based on the faulty modulus p can also be applied on ECDSA. The attack assumes that fault injection can flip every bit of modulus p, and obtain a weak curve on which solving elliptic curve discrete logarithm problem (ECDLP) is computationally feasible. The solution can reveal some leaked information of nonce k, by which two faulty signatures can construct the model of lattice attack to recover the private key. However, the approach requires a strong fault model that only one bit of p is flipped and the faulty modulus p' is known for adversary. Moreover, in order to solve the

ECDLP, it requires all the prime factors $p_i(i = 1, ..., u)$ of p'(where $p' = \prod_{i=1}^{u} p_i^{e_i}$, $p_i < p_j$ for $1 \leq i < j \leq u$ and $e_i \in \mathbb{Z}$) to be relatively small, i.e., $\sqrt{p_u}$ steps is a feasible amount of computation. In addition, in order to mount lattice attack, the product $n'(= \prod_{i=1}^{u} n_i)$ of all the orders $n_i(i = 1, ..., u)$ of subgroups $\mathbb{Z}/p_i^{e_i}$ should satisfy $n' \geq n^{1/2}$, that is, the bit length of n' should be greater than half of the key length of ECDSA, which will restrain the fault injection.

1.3 Our contributions

In this paper, we propose a lattice-based weak curve fault attack on ECDSA by virtue of the partially solvable smoothness of the weak curve order n'. Firstly, by selecting a small factor d of n', we can construct an ECDLP on a small subgroup with order d, and thereby get some reduced information of the scalar(i.e., the nonce k) in ECDSA. Finally, based on the reduced information, we construct a new model of lattice attack to recover the private key in ECDSA signature.

– The fault model is based on the fact that a continuous bit block of the curve parameter a is disturbed by fault injection. The faulty a' is not required to be known, and can be determined by a specific quadratic residual distinguisher.
– Unlike the traditional weak curve attacks [15,3,8], our attack does not require that all the prime factors of the weak curve order n' are enough small such that it is computationally feasible to solve ECDLP. In addition, it is unnecessary that the bit length of n' is greater than $f/2$(where f is the key length of ECDSA) as in the reference [15]. Our attack just requires that there exist some small prime factors(not all the factors) in n', by which a computationally feasible ECDLP in a small subgroup can be constructed to obtain some reduced information of nonce k. The probability that there are some small prime factors in n' is much greater than that of all the prime factors being small(see Section 3.5). Hence, our attack increases the success rate of fault injection and reduces the computational complexity dramatically.
– Our proposed lattice attack is an extension of previous lattice attacks. Although there are no partial bits of nonces leaked as introduced in [14,20,21], we can construct a new model of lattice attack by virtue of the modulo-d reduced information of nonces obtained by the weak curve attack. When the modulus d is the format of 2^l, our model is equivalent to the model based on bit leakage of nonces.
– For ECDSA with random scalar masking, our attack is still feasible without any additional masked bits leakage. For example, if $k' = k + \lambda n$, where k is the real scalar and λ is a 64-bits random number, our attack can succeed by selecting bigger modulus d with high success rate of fault injection.

The remainder of this paper is organized as follows: Section 2 is the preliminaries where the overview of ECC, the traditional weak curve attack and some necessary backgrounds on lattice are introduced. Section 3 describes the concrete attack approach. Section 4 shows the experimental facets of the validity, including the rate of effective faulty parameter a's and the complexity of lattice attack. Finally, conclusion is given in Section 5.

2 Preliminaries

In this paper, we consider elliptic curves on prime field $\mathbb{F}_p$, where p is an odd prime.

2.1 Elliptic curve in $\mathbb{F}_p$

Generally, the Weierstrass equation of elliptic curves in $\mathbb{F}_p$ is given by

$$E(a,b) : y^2 = x^3 + ax + b \mod p,$$

where parameters $a, b \in \mathbb{F}_p$ satisfy $4a^3 + 27b^2 \neq 0$.

The group of rational points in elliptic curve $E(a,b)$ is defined by

$$\mathbf{G} = \left\{(x,y)|y^2 = x^3 + ax + b \bmod p, x, y, a, b \in \mathbb{F}_p\right\} \cup \{\mathcal{O}\},$$

where $\mathcal{O}$ is the infinite point.

Let G be an element in $\mathbf{G}$ with order n (which is usually a prime), $\langle G\rangle$ be the additive subgroup of $\mathbf{G}$ generated by G. If $P = (x,y) \in \langle G\rangle$, then the inverse element $-P \in \langle G\rangle$ of P is $(x,-y)$. For any integer $k \in \mathbb{Z}_n$, the calculation of $kG = G + G + \ldots + G$ (k times) is called the scalar multiplication in $E(a,b)$, and can be calculated using point doubling and addition operations.

Point Addition

If $P = (x_1,y_1) \in \langle G\rangle$, $Q = (x_2,y_2) \in \langle G\rangle$, and $P \neq \pm Q$, then $(x_3,y_3) = P+Q$ satisfies

$\begin{array}{l} x_3 = \lambda^2 - x_2 - x_1 \\ y_3 = \lambda(x_1 - x_3) - y_1 \end{array}$, where $\lambda = \frac{y_2-y_1}{x_2-x_1}$.

Point doubling

If $P = (x_1,y_1) \in \langle G\rangle$ and $P \neq -P$, then $(x_3,y_3) = 2P$ satisfies

$\begin{array}{l} x_3 = \lambda^2 - 2x_1 \\ y_3 = \lambda(x_1 - x_3) - y_1 \end{array}$, where $\lambda = \frac{3x_1^2+a}{2y_1}$.

An important notice is that the parameter b is not involved in the calculation of point doubling and addition. The order of $E(a,b)$, denoted by $\#E\left(a,b\right)$ can be calculated using SEA algorithm [11].

2.2 ECDSA digital signature algorithm

The ECDSA signature algorithm is described in Algorithm 1 with some less important details being abstracted away.

As shown in Algorithm 1, the randomly generated nonce k is involved in the calculation of scalar multiplication kG (step 3) and the calculation of s (step 6), which are exactly the targets of our attacks.

Algorithm 1 Signature generation of ECDSA

```
Input: The definition of a specific elliptic curve E$(a, b)$, a base point $G$ of the curve
    with order $n$, private key $d_A \in \mathbb{Z}_n$, message $m$.
Output: Signature pair $(r, s)$.
1: $e = H(m)$, where $H$ is a cryptographic hash function;
2: Generate $k$ randomly from $\mathbb{Z}_n$ ;
3: $Q(x_1, y_1) = kG$;
4: $r = x_1 \bmod n$;
5: if $r = 0$ then goto step 2;
6: $s = k^{-1}(e + d_A r) \bmod n$;
7: if $s = 0$ then goto step 2;
8: return $(r, s)$
```

2.3 Smoothness of weak curve order

The following definitions are required to better describe our approach. For all of them, let n be the order of point G in $E(a, b)$.

Definition 1. *Denote the prime factorization of n by $n = \prod_{i=1}^{u} q_i^{e_i}$, where $q_i \in \mathbb{N}$ is a prime factor of n, $e_i > 0$ denotes the degree of q_i in the factorization and $q_i < q_j$ for $1 \leq i < j \leq u$. For $y \in \mathbb{N}$, if the biggest prime factor q_u meets $q_u \leq y$, then n is called y-smooth.*

Definition 2. *The elliptic curve discrete logarithm problem (ECDLP) in $E(a, b)$ is defined as: given G, n and an element $Q \in \langle G \rangle$, compute the value $k \in \mathbb{Z}_n$ such that $Q = kG$.*

To our knowledge, the best known generic algorithm [29,23] in classical computer for solving ECDLP in arbitrary elliptic curves needs $O(\sqrt{q_u})$ group operations in computation. We call an ECDLP instance is practically solvable if its solving complexity is not bigger than a predefined constant PRAC_COMP. In this paper, we set PRAC_COMP$= 2^{64}$ group operations by considering currently achievable computing power of classical computers, which can be changed to adapt to the future development of computing technology.

Definition 3. *We call n practically* solvable smooth *(related to the group $\langle G \rangle$) if the ECDLP on $\langle G \rangle$ is practically solvable. We call n* partially solvable smooth *(related to the group $\langle G \rangle$) if there exists a factor $d = \prod_{i=1}^{\sigma} q_i^{e_i}$ of n such that the ECDLP on $\langle (n/d)G \rangle$(with order d) is practically solvable. Finally, we call n (practically)* non-solvable smooth *(related to the group $\langle G \rangle$) if n dose not meet the above two cases.*

2.4 Existing fault attacks to weak curves

In this section, we introduce the approach of traditional weak curve fault attack based on solvable smooth order [15,3], which also can be used for solving the ECDLP based on the partially solvable smooth order.

It is assumed that the y-coordinate of G is disturbed by fault injection, i.e., $G = (x_G, y_G)$ is changed into $G' = (x_G, y_{G'})$. Obviously, the faulty G' is not on the original curve $E(a, b)$. As stated in Section 2.1, since parameter b is not involved in the calculation of scalar multiplication, G' can be viewed on a new weak curve $E(a, b')$ and has a solvable smooth order $n' = \prod_{i=1}^{u} q_i^{e_i}$. Consequently, $Q' = kG'$ based on $E(a, b')$ is calculated, where $b' = y_{Q'}^2 - {x_{Q'}}^3 - ax_{Q'} = y_{G'}^2 - {x_{G'}}^3 - ax_{G'}$. Given Q', G' and n', the following approach can be used to solve the scalar k.

Firstly, the reduced value $k \bmod q_i$ can be obtained by solving the ECDLP $\frac{n'}{q_i}Q' = k\frac{n'}{q_i}G'(i = 1, ..., u)$ with Pollard-rho algorithm [29]. Next, the reduced value $k_i = k \bmod q_i^{e_i}(i = 1, ..., u)$ can be obtained by Pohlig-Hellman algorithm [23]. Finally, the modulo-n' reduced value $t = k \bmod n'$ can be obtained by Chinese remainder theorem(CRT). Hence, $k = t + \mu n'$, where $\mu \in \{0, ..., \lfloor n/n' \rfloor\}$. Guess all the possible values of μ and calculate the corresponding k. If $Q = kG$, then k is just the correct one.

The above approach shows that n' must be solvable smooth so as to solve ECDLP on $\langle G' \rangle$. Otherwise, the approach can not be applied in practice.

2.5 Lattice-based attack

Lattice attack is a key step in our approach. Here we will give some fundamental backgrounds about lattice attack.

Let $B = \{\boldsymbol{b_1}, \ldots, \boldsymbol{b_N}\} \subseteq \mathbb{R}^m$ be a series of N linearly independent vectors. The lattice generated by B is defined as

$$\mathcal{L}(B) = \left\{ \sum_{i=1}^{N} x_i \boldsymbol{b_i} : x_i \in \mathbb{Z} \right\},$$

where B is denoted as a basis for the lattice $\mathcal{L}(B)$, and we call the integers N and m as its rank and dimension respectively. If $m = N$, $\mathcal{L}$ is called full rank lattice with dimension N.

Shortest vector problem (SVP) and Closest vector problem (CVP) are two computational complexity problems crucial to lattice-based cryptography. The definitions of SVP and CVP are given as follows.

Definition 4. *[17] We give the definitions of SVP and CVP.*

(1) **Shortest Vector Problem (SVP)***: Given a basis of a lattice $\mathcal{L}$, find a lattice vector $\boldsymbol{v} \neq \boldsymbol{0}$ such that $\|\boldsymbol{v}\| \leq \|\boldsymbol{u}\|$ for any nonzero vector $\boldsymbol{u} \in \mathcal{L}$.*

(2) **Closest Vector Problem (CVP)***: Given a basis of a lattice $\mathcal{L}$ and a target vector $\boldsymbol{t} \in \mathbb{R}^m$, find a lattice vector $\boldsymbol{v}$ closest to the target $\boldsymbol{t}$, which means* $\mathrm{dist}(\boldsymbol{v}, \boldsymbol{t}) \leq \mathrm{dist}(\boldsymbol{u}, \boldsymbol{t})$ *for any vector $\boldsymbol{u} \in \mathcal{L}$, where* dist *denotes the Euclid norm of two points.*

For an N-dimensional approximate SVP, there exist polynomial-time basis reduction algorithms to output a short lattice vector when the approximate factor is large enough, among which the LLL algorithm[16] is the most typical one. Based on a series of optimizing technique[25,26,27], the BKZ-algorithm[7] has been the most practical algorithm to lattice basis reduction. Lemma 1 gives the approximate factor of the LLL algorithm.

Lemma 1. *[17,16] There exists a polynomial time algorithm on that input an integer basis B of lattice $\mathcal{L}$ outputs a nonzero lattice vector $\boldsymbol{x}$ meeting $\|\boldsymbol{x}\| \leq (2/\sqrt{3})^N \lambda_1(\mathcal{L})$, where N is the dimension of the lattice $\mathcal{L}$ and $\lambda_1(\mathcal{L})$ is the length of shortest vector in $\mathcal{L}$.*

For random lattices with dimension N, Gaussian heuristic gives a probable estimation on the length of shortest lattice vector in the sense of average.

Definition 5. *[19] Gaussian Heuristic: let $\mathcal{L}$ be a full rank lattice in $\mathbb{R}^N$, and C be a measurable subset of $\mathbb{R}^N$. The Gaussian Heuristic predicts that the number of points of $\mathcal{L} \cap C$ is roughly $\mathrm{vol}(C)/\mathrm{vol}(\mathcal{L})$, where vol denotes the volume or determinate.*

From Gaussian Heuristic, the Gaussian expected shortest length of an N-dimensional lattice $\mathcal{L}$ could be defined to be

$$\sigma(\mathcal{L}) = \sqrt{\frac{N}{2\pi e}} \mathrm{vol}(\mathcal{L})^{1/N}.$$

Generally, the actual shortest lattice vector is much easier to be found as the increment of the gap between the shortest length and the Gaussian heuristic. If it is significantly shorter than $\sigma(\mathcal{L})$, it shall be located in polynomial time by using LLL and related algorithms. Heuristically, assuming the lattice $\mathcal{L}$ behaves like random, if there exists a lattice vector whose distance from the target is much shorter than $\sigma(\mathcal{L})$, this lattice vector is expected to be the closest vector to the target. Accordingly, this special CVP instance usually could be solved by Babai algorithm or embedding-based SVP.

3 Lattice-based weak curve attack

In this section, we present the concrete approach of lattice-based weak curve attack on ECDSA. The whole attack consists of two steps: 1) Some reduced information of nonce is obtained by weak curve fault attack; 2) By virtue of the reduced information, a special instance of CVP can be constructed in a lattice to recover the private key.

3.1 Fault model

Generally, weak curve fault attack aims at the physical objects including RAM or PPROM, corresponding to the algorithmic objects including basis point G, curve parameter a and modulus p. The faults are classified as temporary fault, permanent fault and semi-permanent fault (i.e., after fault injection, the faulty data is kept until the device restores the correct one). In this paper, we mainly consider the random permanent fault or semi-permanent fault based on the fact that a continuous l-bits block of a is disturbed randomly by fault injection and its faulty starting location is random, where l is usually valued from $\{1, 8, 16, 24$ and $32\}$. It is assumed that an adversary induces one random permanent or semi-permanent fault into parameter a before signature generation such that a is transformed into a'(a' has continuous l bits different from a). Hence, the subsequent scalar multiplication is not $Q = kG$(step 3 in Algorithm 1) on $E(a, b)$ but $Q' = kG$ on a new weak curve $E(a', b') : x^3 + a'x + b' \bmod p$, where G is a point on $E(a', b')$ with an order n' and $b' = {y_G}^2 - {x_G}^3 - a'x_G \bmod p$. Finally, the results (r', s') of signature are output to adversary.

Note that our fault model has the following limitations: 1) parameter a must be involved in the calculation of scalar multiplication (except the case that a is usually substituted with $p-3$ when $a = p-3$); 2) There is no point verification checking whether the input point is on the original elliptic curve during the calculation of scalar multiplication. Otherwise, our attack will not work.

3.2 Weak curve fault attack

Assuming that the signatures on the weak curve can be executed, we will carry on the following steps to obtain some reduced information of nonces ks.

(1) Guess and determine a' and $x_{Q'}$

1. Set $\mathbb{T}$ as the set to store all the guessed values of a'.
2. Perform signature generation based on a', and obtain the faulty signature results (r', s').
3. The x-coordinate $x_{Q'} \bmod p$ of the faulty Q' can be deduced by $x_{Q'} = r' \bmod n$. In the standard curves, the cofactor $h(hn = \#E(a, b)$ is generally very small, such as $h = 1$ or 2. From Hasse theorem [13], we have $p+1-2\sqrt{p} < hn < p+1+2\sqrt{p}$. Hence, $x_{Q'} \bmod p$ has t possible values which is bounded by $h + 1$, depending on the concrete values of p and n.
4. Guess all the values of $a' \in \mathbb{T}$:
   Calculate $b' = {y_G}^2 - {x_G}^3 - a'x_G \bmod p$, substitute the t possible values of $x_{Q'}$ into the guessed curve $E(a', b')$ and obtain
   $$Y = {x_{Q'}}^3 - a'x_{Q'} + b' \bmod p.$$
   If Y is a quadratic residue, i.e., there exists a square root $y_{Q'}$ of Y, then keep the guessed value of a' in $\mathbb{T}$. Otherwise, eliminate the guessed value from $\mathbb{T}$.
5. If the number of $\mathbb{T}$ is greater than 1, then go to step 2; Otherwise, the only value in $\mathbb{T}$ is just the correct value of a'.

Since there is a continuous l-bits block in a disturbed and the starting position of the faulty bits is random, there are $(l_a - l + 1)2^l$ possible guessed values, where

l_a is the bit length of a. That is, the number of the initial $\mathbb{T}$ is $(l_a - l + 1)2^l$. The "quadratic residue" distinguisher can eliminate about half of the guessed values in each invocation. Hence, the total complexity is about $O((l_a - l + 1)2^{l+1})$, and the faulty values of $x_{Q'}$ can also be eliminated along with distinguishing.

(2) Obtain the reduced value of nonce k

1. Based on the known a', carry on the signature generation and get the results (r', s').
2. As mentioned above, $x_{Q'} \bmod p$ has t possible values from $x_{Q'} = r' \bmod n$. For each value of $x_{Q'} \bmod p$, calculate the corresponding $Y = {x_{Q'}}^3 - a'x_{Q'} + b' \bmod p$. If only one value of Ys is quadratic residue, then the corresponding $x_{Q'} \bmod p$ is the correct one; Otherwise, go to step 1 and re-carry on the signature generation.
3. Substitute the correct $x_{Q'}$ into curve $E(a', b')$, and obtain two possible points $(x_{Q'}, \pm\sqrt{Y})$ of Q'.
4. Let $Q'_1 = \left(x_{Q'}, \sqrt{Y}\right) = uG$. It is assumed that there exists a factor $d(= \prod_{i=1}^{\sigma} q_i^{e_i})$ in the order $n'(= \prod_{i=1}^{u} q_i^{e_i}, q_i < q_j$ for $1 \le i < j \le u)$ of G on $E(a', b')$ is solvable smooth(see the definitions in Section 2.3), i.e., n' is partially solvable smooth. Let $L = n'/d = \prod_{i=\sigma+1}^{u} q_i^{e_i}$. Consequently, in the subgroup $\langle LG \rangle$ with order d, the ECDLP
$$LQ'_1 = u(LG)$$
can be solved to obtain $u \bmod d$. If Q'_1 is the correct choice of Q', then $k = u \bmod d$; Otherwise, $k = d - u \bmod d$.

In the above analysis, it is unnecessary that n' is *solvable smooth*, as long as n' is partially solvable smooth.

Repeat the above steps $1 - 4$ for N times to obtain two possible reduced values of nonce k_i, i.e., $k_i = u_i \bmod d$ and $k_i = d - u_i \bmod d$. Hence, we have

$$k_i = c_i + \lambda_i d (i = 1, ..., N),$$

where $c_i = \begin{cases} u_i | y_{Q'} = Y \\ d - u_i | y_{Q'} = -Y \end{cases}$ and $0 < \lambda_i < n/d$.

3.3 Lattice attack based on partially solvable smooth order

As mentioned above, the traditional model of lattice attack [14,21] could not work since there is no leaked bits of k_is. However, similar to the lattice attack on ECMQV [5], we still can construct a new lattice attack model with the reduced information of k_is .

For $i = 1, ..., N$, substituting c_i into s_i, we can get

$$s_i(c_i + \lambda_i d) = e_i + r_i d_A, \tag{1}$$

where e_i is the hash value of message m_i and $0 < \lambda_i < n/d$.

The equation (1) can be transformed as

$$\lambda_i = {s_i}^{-1} d^{-1} r_i d_A - (d^{-1} c_i - {s_i}^{-1} d^{-1} e_i) \bmod n. \tag{2}$$

Let $A_i = {s_i}^{-1} d^{-1} r_i \bmod n$, $B_i = d^{-1}\left(c_i - {s_i}^{-1} e_i\right) + n/(2d)$, then there must exist $h_i \in \mathbb{Z}$ such that

$$|A_i d_A + h_i n - B_i| < n/(2d)(i = 1, ..., N). \tag{3}$$

We can construct a lattice $\mathcal{L}$ by the above inequations (3), whose basis matrix $\mathbf{M} \in \mathbb{Z}^{(N+1)\times(N+1)}$ is

$$\begin{pmatrix} n & 0 & \cdots & 0 \\ \vdots & \ddots & & \vdots \\ 0 & & n & 0 \\ A_1 & \cdots & A_N & 1/(2d) \end{pmatrix}.$$

The row vectors $\{\boldsymbol{b}_1, ..., \boldsymbol{b}_{N+1}\}$ of $\mathbf{M}$ is a basis of $\mathcal{L}$. Let the target vector $\boldsymbol{t} = (B_1, \ldots, B_N, 0) \in \mathbb{Z}^{N+1}$, and there exists a lattice vector $\boldsymbol{v} = \boldsymbol{x}\mathbf{M} = (A_1 d_A + h_1 n, \ldots, A_N d_A + h_N n, d_A/(2d)) \in \mathcal{L}$ with coordinate vector $\boldsymbol{x} = (h_1, \ldots, h_N, d_A) \in \mathbb{Z}^{N+1}$. From inequations (3), we have

$$\|\boldsymbol{v} - \boldsymbol{t}\| < \sqrt{N+1}n/(2d), \tag{4}$$

Heuristically, we assume $\mathcal{L}$ is a random lattice. As introduced in Section 2.5, if $\|\boldsymbol{v} - \boldsymbol{t}\|$ is much less than $\sigma(\mathcal{L})$ $(= \sqrt{\frac{N+1}{2\pi e}}\mathrm{vol}(\mathcal{L})^{\frac{1}{N+1}})$, we expect that $\boldsymbol{v}$ is the closest vector to $\boldsymbol{t}$ in $\mathcal{L}$, where $\mathrm{vol}(\mathcal{L}) = \det(\mathbf{M}) = n^N/(2d)$. Hence, it is required

$$\|\boldsymbol{v} - \boldsymbol{t}\| < \sqrt{N+1}n/(2d) \ll \sqrt{\frac{N+1}{2\pi e}}\left(n^N/(2d)\right)^{\frac{1}{N+1}}. \tag{5}$$

Let the bit length of ECDSA be denoted by $f = \lceil \log n \rceil$ and the bit length of d be denoted by $l_d = \lceil \log d \rceil$. If $N > \frac{f + \log\sqrt{2\pi e}}{l_d + 1 - \log\sqrt{2\pi e}}$ and $l_d > \log\sqrt{2\pi e} - 1$, then the above inequality can be viewed as a special instance of CVP in lattice $\mathcal{L}$ heuristically.

Consequently, the vector $\boldsymbol{v}$ can be determined by solving the instance of CVP and thereby to recover the private key d_A. If $P_A = d_A G$(P_A is the public key of signature), the attack is successful.

In addition, the inequality (3) is equivalent to

$$|A_i d_A + h_i n - B_i| < n/2^{l_d}(i = 1, ..., N), \tag{6}$$

which is a hidden number problem(HNP)[20,21]. By the same way, it can be transformed into a CVP to recover d_A.

It is noted that the above lattice attack can recover d_A successfully only when all the values of $c_i(i = 1, ...N)$ are correct. Nevertheless, there are two solutions to c_i in the weak curve attack and it is not sure which one is the correct value. Therefore, it is necessary to enumerate all the possible values of c_i to carry on lattice attack. The maximum time complexity of the attack is $2^N T$, where T is the time required for one-time lattice attack. The needed number N of faulty signatures depends on the size of factor d. The larger d is, the smaller N and T are. Generally, In order to ensure the absolute success rate of lattice attack, d should not be too small. For example, d is generally recommended to satisfy $l_d \geq 8$ such that $N \approx 45$ for 256-bits ECDSA.

3.4 Attack on ECDSA with scalar masking

If $d = 2^l$ ($l \in \mathbb{Z} > 0$), which means the least significant l bits of nonces are known, the above lattice attack is equivalent to the lattice attack described in [14,20,21]. Therefore, our lattice attack extends the traditional model of lattice attack with bit leakage, and is still feasible for ECDSA with general scalar masking countermeasure.

If the nonce k_i in signature generation is masked with a random number α_i, then the masked nonce k'_i is equal to $k_i + \alpha_i n (i = 1, ..., N)$ and we have

$$Q = k'_i G, \qquad s_i = {k_i}^{-1}(e_i + r_i d_A) = {k'_i}^{-1}(e_i + r_i d_A) \mod n,$$

where the bit length of α_i is denoted by l_{α_i}.

Accordingly, the reduced information derived by weak curve fault attack is $k'_i = c_i + \lambda_i d$, where $\lambda_i < 2^{f + l_{\alpha_i} - l_d}$. Similarly, substitute the reduced information into s_i and mount lattice attack. If $l_d > \log\sqrt{2\pi e} + l_{\alpha_i} - 1$ and $N > \frac{f + \log\sqrt{2\pi e}}{l_d - l_{\alpha_i} + 1 - \log\sqrt{2\pi e}}$, the private key d_A still can be recovered by constructing a CVP. In practice, α_i is usually a 32 or 64-bits random number in view of performance. For the traditional model of lattice attack with bit leakage, it means α_i bits of k'_i is needed to be acquired for an adversary, which add the difficulty of the attack extremely. However, for our attack, if $l_{\alpha_i} = 64$, the experimental success rate of fault injection is still up to 66% since l_d is recommended as 72 (see Section 4). Obviously, it is also practical for fault injection.

3.5 Comparison with other fault attacks

Compared with the weak curve fault attack proposed in [15,3,8], our attack has higher success rate of fault injection. Firstly, it is not required to know the faulty parameter a'. The number of continuous faulty bits can be up to PRAC_COMP$(= 2^{64})$ and its starting position is random, which reduces the requirements of fault injection. Secondly, the generated weak curve order is not required to be solvable smooth. In view of the influence of different smoothness of n' on our attack, we will discuss the density of its smoothness as follows.

The density of smooth numbers

Let z be an integer with prime factorization $z = \prod_{i=1}^{u} p_i^{e_i}$. We say z is smooth with respect to an integer x if $\underset{1 \leq i \leq u}{max}\{p_i\} \leq x$ as mentioned in Section 2.3.

We denote by $\psi(x, y)$ the number of integers $z \leq y$ such that z is x-smooth. In [9], a result on the bound of $\psi(x, y)$ shows smooth numbers with suitable x, y are relatively common to meet. Specifically, let ϵ be an arbitrary positive constant, then uniformly for $y \geq 10$ and $x \geq (\ln y)^{1+\epsilon}$, we have

$$\psi(x, y)/y = e^{-(1+o(1))u \ln u} \qquad \text{as } y \to \infty,$$

where $u = \ln y / \ln x$ and e is the natural number. Note that for a fixed y, the density of smooth numbers (i.e., $\psi(x, y)/y$) is an increasing function with respect to the bound x of factors. In particular, we can roughly estimate that $\psi(2^{256}, 2^{80})/2^{256} = 0.024$ and $\psi(2^{256}, 2^{128})/2^{256} = 0.25$. It means, given a fixed y, smooth numbers in the scope $[1, \ldots, y]$ with at least one large factor could be much more frequent than those with only smaller factors.

In this paper, we assume n', *i.e.*, the order of G' on the fault induced weak elliptic curve $E(a', b')$, is a smooth number. Note the action of the generation of n' may not be uniformly random, since the injected fault is only considered to impact very limited bits of the parameter a of the curve. (Or in other words, a' is not randomly selected to generate the new curve). Similar to the heuristic assumption given in [9], we assume that the probability that n's are sampled in this method is comparable to the density $\psi(x, y)/y$. As shown in Figure 1, the density(y-coordinate) for x-smooth n'(x-coordinate) in 256-bits ECDSA is depicted. When n' is 2^{247}-smooth, the probability, i.e., density is about 96.3%, which is much greater than 25% for 2^{128}-smooth n' required in the previous weak curve attacks [15,3,8]. For this 2^{247}-smooth case, there could be small factor $d \geq 2^8$ such that our attack is feasible. And the practical experiment in Section 4 also supports this assumption.

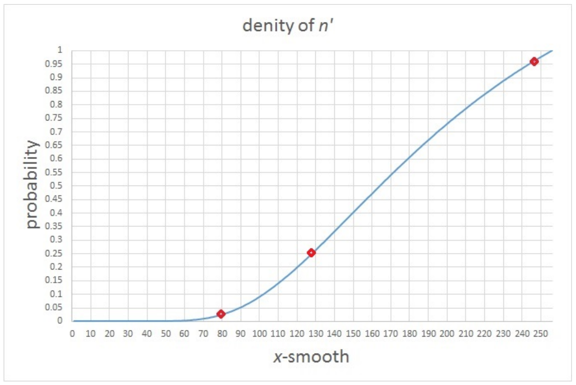

Fig. 1. The density for x-smooth n'

In conclusion, There could exist large prime factor in n' with high probability such that the traditional weak curve attacks [15,3,8] are infeasible. However, if n'

is partially solvable smooth which has absolutely higher probability than those solvable smooth orders, our attack still work.

In addition, our model of lattice attack is simpler and more feasible, except the needed number N of faulty signatures is greater than the one in [15]. Table 1 lists the detailed differences between our attack and the attacks in [15,3,8]. Our attack has less restrictions than the previous attacks and increases the success probability of fault injection extremely.

Table 1. Comparison of our attack with the fault attack in [15] for 256-bits ECDSA

Items \ Attacks	Our attack	Attack in [15]	Attacks in [3,8]
solvable smooth order	✗	✓	✓
success probability of fault injection	$\gg 25\%$	25%	25%
given exact faulty value	✗	✓	✓
tolerant faulty bits	PRAC_COMP	1	—
needed signature number N	$N > \frac{f+\log\sqrt{2\pi e}}{l_d+1-\log\sqrt{2\pi e}}$	2	—

4 Experimental validation of feasibility

In this section, the feasibility of our attack is verified by simulations. First of all, the proportions of the effective faulty a's are counted out by the simulations of fault injection. Secondly, the lattice attack is implemented by virtue of the faulty signatures derived from the effective a's. The experiments are conducted in a commonly used computer with 8-core CPU 3.4GHz, 8G memory and Windows7 OS. The weak curve order n' derived form faulty a' is calculated by the SEA algorithm implemented in miracl library(C/C++), and the constructed CVP in lattice is solved by the BKZ algorithm implemented in NTL library [28] with block 10.

In the first experiment, two kinds of 256-bits curve parameters based on prime field $\mathbb{F}_p$ recommended in FIPS and SM2 digital signature algorithm(hereafter called SM2) are selected to simulate the fault injection. Moreover, two types of fault injection are also simulated, including single-bit flipped fault and 16-bits random fault. The single-bit flipped fault is simulated to flip every bit of a AND results in 256 kinds of cases. The 16-bits random fault is simulated to XOR a 16-bits random number β with any continuous 16-bits block of parameter a, i.e., $a' = a \oplus \beta 2^l$, where $l(0 < l < f)$ is also random. The simulations are also carried on for 256 times. Hence, there are four faulty simulation cases(i.e., two types of fault injection for two kinds of curves): 1) single-bit flipped fault of a on the curves recommended by FIPS and SM2; 2) 16-bits random fault of a on the curves recommended by FIPS and SM2.

Table 2 lists the proportion γ of the partially solvable smooth n's(i.e., effective a's) when l_d is greater than $8, 16, 32, 40, 64, 72$ and 128. It is noted that the factor d is required to be solvable 2^{128}-smooth(i.e., the constructed ECDLP is practically solvable) in experiment. As shown in Table 2, when $l_d \geq 8$, the proportion is greater than 94.9% for the four faulty cases. In order to ensure 100% experimental success rate of the consequent lattice attack, the needed dimension N of the constructed lattice is about 45 in experiments. Obviously, the complexity of enumerating the correct c_i in lattice attack is computationally feasible. In addition, to speed up enumerating, the case $l_d \geq 16$ is generally selected in experiments, whose proportion is also up to 92.6% at least. The results show that the success rate of fault injection is significantly high.

Table 2. The proportion of the effective a's in the four cases

Fault type	d is 2^{128}-solvable smooth						
	$l_d \geq 8$	$l_d \geq 16$	$l_d \geq 32$	$l_d \geq 40$	$l_d \geq 64$	$l_d \geq 72$	$l_d \geq 128$
	γ	γ	γ	γ	γ	γ	γ
single-bit flipped fault(FIPS)	96.5%	92.6%	85.2%	81.6%	69.9%	66%	34.8%
16-bits random fault(FIPS)	94.9%	92.6%	85.6%	82.8%	74.2%	68.4%	32.4%
single-bit flipped fault(SM2)	96.5%	94.1%	83.1%	80.7%	70.1%	66.9%	34.7%
16-bits random fault(SM2)	98.1%	93.4%	86.3%	83.2%	71.1%	66.8%	32%

Moreover, Figure 2 depicts the proportion γ(y-coordinate) of the effective faulty a's when $l_d > X$ (x-coordinate). No matter whether the curve belongs to FIPS or SM2, and the fault type is single-bit flipped fault or random fault, all the proportions of the four cases are roughly similar with the density mentioned in Section 3.1. Hence, our weak curve fault attack is effective for most of ECC signatures based on prime field.

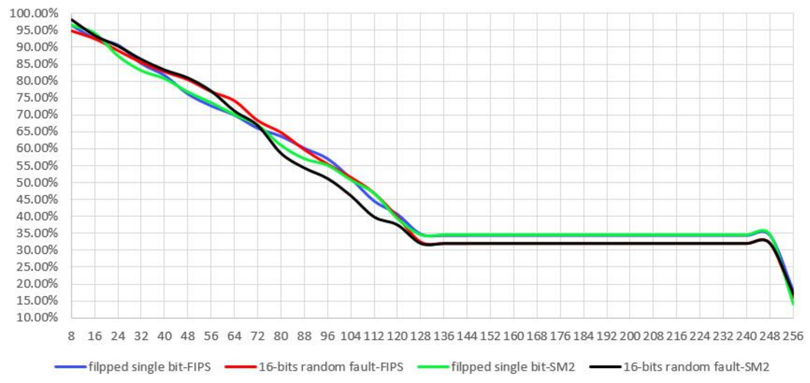

Fig. 2. The proportion γ of effective a' when $l_d > X$

Finally, based on the faulty a' in SM2 signature, select the ds with different bit length, and carry on the corresponding lattice attacks. As shown in Table 3,

N is the number of signatures required to achieve 100% success rate of lattice attack, T is the time of each lattice attack (8-core CPU 3.4GHz, 8G memory and BKZ with block 10) and O is the maximum complexity of the attack including the enumerating. From Table 3, $l_d = 16$ with 92.6% high success rate of fault injection and $2^{19}T$ time complexity is the optimal selection in our experiments.

Table 3. The number of faulty signatures and complexity for lattice attack

Items	$l_d = 8$	$l_d = 9$	$l_d = 16$	$l_d = 32$	$l_d = 64$
N	45	40	19	9	5
$T(s)$	≈ 5.788	≈ 3.675	≈ 0.255	≈ 0.021	≈ 0.005
O	$2^{45}T$	$2^{40}T$	$2^{19}T$	$2^{9}T$	$2^{5}T$

5 Conclusion

We propose a new lattice-based weak curve fault attack on ECDSA which combines the advantages of weak curve fault attack and lattice attack. The order n' of the weak curve generated by faulty a' is not required to be solvable smooth, and the reduced information of nonces is obtained by solving the ECDLP constructed in a small subgroup, by which a new model of lattice attack is constructed to recover the private key. For the single-bit fault or 16-bits random fault, the success rate of fault injection that there exists a solvable smooth factor d of n' to satisfy $l_d \geq 8$ can be as high as 94.9%. In addition, for ECDSA with w-bits scalar masking, our attack still work with high success rate of fault injection by selecting an appropriate d satisfying $l_d - w > \log \sqrt{2\pi e} - 1$.

Further Work. Our attack is based on the disturbed parameter a. However, if a is not involved in the calculation of point doubling and addition, e.g., $a = p - 3$, or there is the countermeasure of point verification, our attack will not work. Hence, it is worthy of further study on whether there are more effective fault vulnerabilities which can be exploited for carrying on weak curve attack and lattice attack, modulus p, for example.

References

1. D. F. Aranha, F. R. Novaes, A. Takahashi, M. Tibouchi, and Y. Yarom. Ladderleak: Breaking ECDSA with less than one bit of nonce leakage. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 225–242, 2020.
2. N. Benger, J. Van de Pol, N. P. Smart, and Y. Yarom. ooh aah... just a little bit: A small amount of side channel can go a long way. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 75–92. Springer, 2014.
3. I. Biehl, B. Meyer, and V. Müller. Differential Fault Attacks on Elliptic Curve Cryptosystems. In *Advances in Cryptology-CRYPTO 2000*, pages 131–146. Springer, 2000.

4. B. B. Brumley and N. Tuveri. Remote timing attacks are still practical. In *European Symposium on Research in Computer Security*, pages 355–371. Springer, 2011.
5. W. Cao, H. Chen, J. Feng, L. Fan, and W. Wu. Lattice-based fault attacks against ecmqv. In *International Workshop on Constructive Side-Channel Analysis and Secure Design*, pages 101–116. Springer, 2018.
6. W. Cao, J. Feng, H. Chen, S. Zhu, W. Wu, X. Han, and X. Zheng. Two lattice-based differential fault attacks against ECDSA with wNAF algorithm. In *ICISC 2015*, pages 297–313. Springer, 2015.
7. Y. Chen and P. Q. Nguyen. BKZ 2.0: better lattice security estimates. In *Advances in Cryptology–ASIACRYPT 2011*, pages 1–20. Springer, 2011.
8. M. Ciet and M. Joye. Elliptic curve cryptosystems in the presence of permanent and transient faults. *Designs Codes Cryptography*, 36(1):33–43, 2005.
9. D. Coppersmith, A. M. Odlzyko, and R. Schroeppel. Discrete logarithms in GF(p). *Algorithmica*, 1(1-4):1–15, 1986.
10. E. De Mulder, M. Hutter, M. E. Marson, and P. Pearson. Using bleichenbachers solution to the hidden number problem to attack nonce leaks in 384-bit ECDSA: extended version. *Journal of cryptographic engineering*, 4(1):33–45, 2014.
11. N. D. Elkies et al. Elliptic and modular curves over finite fields and related computational issues. *AMS IP STUDIES IN ADVANCED MATHEMATICS*, 7:21–76, 1998.
12. D. Genkin, L. Pachmanov, I. Pipman, E. Tromer, and Y. Yarom. ECDSA key extraction from mobile devices via nonintrusive physical side channels. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1626–1638, 2016.
13. D. Hankerson, A. J. Menezes, and S. Vanstone. *Guide to elliptic curve cryptography.* Springer, 2004.
14. N. A. Howgrave-Graham and N. P. Smart. Lattice attacks on digital signature schemes. *Designs, Codes and Cryptography*, 23(3):283–290, 2001.
15. T. Kim and M. Tibouchi. Bit-flip faults on elliptic curve base fields, revisited. In *International Conference on Applied Cryptography and Network Security*, pages 163–180. Springer, 2014.
16. A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.
17. D. Micciancio and S. Goldwasser. *Complexity of lattice problems: a cryptographic perspective*, volume 671. Springer, 2002.
18. D. Naccache, P. Q. Nguyên, M. Tunstall, and C. Whelan. Experimenting with Faults, Lattices and the DSA. In *International Workshop on Public Key Cryptography*, pages 16–28. Springer, 2005.
19. P. Q. Nguyen. *Hermite's Constant and Lattice Algorithms*, pages 19–69. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.
20. P. Q. Nguyen and I. E. Shparlinski. The insecurity of the digital signature algorithm with partially known nonces. *Journal of Cryptology*, 15(3), 2002.
21. P. Q. Nguyen and I. E. Shparlinski. The insecurity of the elliptic curve digital signature algorithm with partially known nonces. *Designs, codes and cryptography*, 30(2):201–217, 2003.
22. P. Q. Nguyen and M. Tibouchi. *Lattice-Based Fault Attacks on Signatures.* 2013.
23. S. Pohlig and M. Hellman. An improved algorithm for computing logarithms over gf (p) and its cryptographic significance (corresp.). *IEEE Transactions on information Theory*, 24(1):106–110, 1978.
24. J. Schmidt and M. Medwed. A Fault Attack on ECDSA. In *Fault Diagnosis and Tolerance in Cryptography (FDTC), 2009 Workshop on*, pages 93–99. IEEE, 2009.

25. C. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical Computer Science*, 53(2):201 – 224, 1987.
26. C.-P. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Mathematical programming*, 66(1-3):181–199, 1994.
27. C.-P. Schnorr and H. H. Hörner. Attacking the chor-rivest cryptosystem by improved lattice reduction. In *Advances in Cryptology Eurocrypt 1995*, pages 1–12. Springer, 1995.
28. V. Shoup. Number Theory C++ Library (NTL) version 9.6.4. `http://www.shoup.net/ntl/`, 2016.
29. P. C. Van Oorschot and M. J. Wiener. Parallel collision search with cryptanalytic applications. *Journal of cryptology*, 12(1):1–28, 1999.