International Journal of Computer Applications (0975 - 8887)
Volume 140 - No.6, April 2016

Differential Fault Attacks and Countermeasures in Elliptic Curve Cryptography

Anissa Sghaier
Faculty of Sciences, EμE Lab,
University of Monastir 5019, Tunisia

Medien Zeghid
Higher Institute of Applied Sciences
and Technology of Sousse 4003, Tunisia
University of Monastir, EμE Lab

Mohsen Machhout
Faculty of Sciences, EμE Lab,
University of Monastir 5019, Tunisia

ABSTRACT

In asymetric cryptography, Elliptic Curve Cryptography (ECC) is the fastest in term of computation and the strongest in term of security. It can be used in message encryption/decryption, digital signature or key exchange. ECC can be implemented in hard over binary field $GF(2^n)$ or in soft over prime field $GF(p)$. This paper presents an efficient software implementation of ECC scalar multiplication over prime field using GNU Multiple Precision (GMP) Library. The differential fault attacks (DFA) on the RSA cryptosystem can be extended to ECC one by inserting bit errors during computation. In this paper, a "No Correctness Check for Input Points" (NCCIP) attacks was applied and a countermeasures was given to protect ECC cryptosystem against it. Software implementation in C language, using GMP library, was simulated on Intel(R) Core(TM) i3 CPU M380 @ 2.87GHz(3 GB RAM, 32-bit architecture).

Keywords

ECC, scalar multiplication, DFA, GMP Library, El-gamel Scheme

1. INTRODUCTION

RSA is the most asymmetric scheme used in the classical protocol for both encryption and decryption. But the confidentiality of the cryptosystem depends on the security of the private key. In this way, increasing the key length can not be a solution in some cases because systems can suffer from luck of memory. The current security trends are increasing the security mechanisms complexity and decreasing key/operand size. For this reason, researches moved to asymmetric scheme providing a high level of security with the minimum key size such as Elliptic/Hyperelliptic cryptography and Pairing computation based on algebraic curves. Protocols based on these schemes are Elgamel encryption, and DiffieHellman (DH) key exchange. DH key exchange is how to create a key together without be saved, never transmitted, and never made visible anywhere, which represent an important advantages to be used. The hardness of the algebraic curves is the hardness to solve discrete logarithm problem (DLP); let G be a finite cyclic group, and g a generator of G, let take x such as: $x \in G$, then find natural number t, such as $gt = x$, if t exists [1].
Researches prove that modular arithmetic operations in binary fields curves are cheaper to be implement in hardware than prime field curves, contrarily to software implementation, fast integer multiplication is less efficient in hard. Thus, modular arithmetic operations should be well studied in order to find an efficient implementation, authors in [2] and [3] studied arithmetic operation over finite fields. Many software implementation was ben proposed [4, 5, 6]. In this paper, a software implementation of scalar multiplication over prime elliptic curves using GMP Library will be firstly presented, which is a portable library written in C, with some optimized routines written in assembly code, specialized for different processors, more information is given in [7]. It's suitable for integer arithmetic using big operands size. In this work, operand dimension limit is $2^{32} - 1$ bits on 32-bit machines. The GMP library is useful for doing multiple precision arithmetic quickly, for this reason it was used, because ECC is based on point addition/doubling which require fast integer multiplication [8, 9, 10].
This paper also, focus on DFA attack in El-Gamal cryptosystem. DFA has been successful in attacking RSA cryptosystems on smart card [11]. One of the DFA attack, which is NCCIP, will be presented and applied . Attacks will continue to become more innovative and sophisticated, so proposed cryptosystem have to be well protected. A technique used to resist to DFAs attacks is to make the software implementation more robust by reducing the weakness of the software. This paper will also provide countermeasure to NCCIP attack, ECC cryptosystem have to check in the first way if input points belong to the selected elliptic curve.
The present paper is organized as follows. Section 2 gives briefly mathematical background of Elliptic Curves. Within section 3, the scalar multiplication algorithm will be presented. Section 4 presents the software implementation of ECC using El-Gamal protocol. DFA attacks will be applied to the proposed ECC implementation in section 5. To protect the proposed software implementation, countermeasure have to be applied, that will be presented in section 6, then practical remarks will be given. Finally, a conclusion will be given in section 7.

2. ELLIPTIC CURVES MATHEMATICAL BACKGROUND

In this section, Elliptic Curves over finite prime field $\mathbb{F}_p$ will be briefly introduced and the formula of point addition and point doubling will be given. An Elliptic Curve has the general following form:

$$E(\mathbb{F}_q) = (x, y) \in F_q \times F_q$$

such as

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \cup O$$

2.1 Elliptic Curves Over Finite Prime Field $\mathbb{F}_p$

Let's take an odd prime number p and a finite prime field $\mathbb{F}_p$ which contains p elements. Each odd prime number p have only one finite prime field $\mathbb{F}_p$, however, the field elements representation can vary [8]. If $a, b \in \mathbb{F}_p$ and $4a^3 + 27b^2 \neq 0(modp)$ then the elliptic curve over $\mathbb{F}_p$ have the following set of points [12]:

$$E(\mathbb{F}_p) = (x, y) \in \mathbb{F}_p^2 | y^2 = x^3 + ax + b(modp)$$

$$4a^3 + 27b^2 \neq 0(modp) \wedge a, b, y, x \in \mathbb{F}_p \cup\ O$$

where O is the point at infinity.

The number of elements $\#E(\mathbb{F}_p)$, in $E(\mathbb{F}_p)$ is equal to the number of points of elliptic curve over $\mathbb{F}_p$. According to the Hasse Theorem $\#E(\mathbb{F}_p)$ belongs to the interval [8]:

$$p + 1 - 2\sqrt{p} \leq \#E(\mathbb{F}_p) \leq p + 1 + 2\sqrt{p},$$

The number of elements $\#E(\mathbb{F}_p)$ in $E(\mathbb{F}_p)$ is called the order of group.

2.2 Addition and Doubling Operation in E(Fp)

In this section, the algebraic formula of point addition and point doubling will be presented.
Let $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2) \in E(\mathbb{F}_p)$

(1) $P_1 + P_2 = (x_1, y_1) + (x_2, y_2) = (x_2, y_2) + (x_1, y_1) = (x_3, y_3) = P_3$
- $x_3 = \lambda^2 - x1 - x_2(modp)$
- $y_3 = \lambda(x_1 - x_3) - y_1(modp)$
- $\lambda = (\frac{y_2 - y_1}{x_2 - x_1}(modp)$

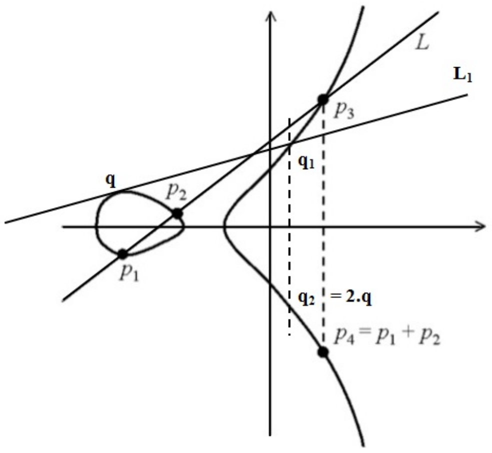

Fig. 1. EC point Addition and Doubling

(2) $2 \times P_1 = (x_1, y_1) + (x_1, y_1) = P_4$
- $x_4 = \lambda_2^2 - 2x_1(modp)$
- $y_4 = \lambda_2(x_1 - x_4) - y_1(modp)$
- $\lambda_2 = \frac{3x_1^2 + a}{2y_1}(modp)$

As it's mentioned in Figure 1, to add 2 distinct points, p_1 and p_2, the straight line defined by these points should be drawn, its intersection with the curve E gives the point p_3, then the negation of this point is the result of point addition which is p_4. For point doubling, the tangent to the curve E, at q gives the point q_1, its negation q_2 is the final result.

3. MONTGOMERY SCALAR MULTIPLICATION

Montgomery [13] have well studied the scalar multiplication and gave an important approach. Let's $\mathbb{G}$ a cyclic subgroup of $E(K)$ of prime order n generated by a point P.

$$\mathbb{G} =< P >= O, P, [2]P, ..., [n-1]P \subseteq E(K)$$

Let's $k \in \mathbb{Z}$, and $P, Q \in \mathbb{G}$ so :

$$Q = [k]P = \underbrace{P + P + \ldots + P}_{k \text{ times}}$$

Algorithm 1 give the Montgomery Add and Double method which was developed for fast scalar multiplication on EC:

Algorithm 1 : Montgomery Double and Add algorithm over $\mathbb{F}$

```
Input: $P(x, y) \in E(F_p), k >= 1$
Output: $Q = kP$
Set $S_0 = 0$;$S_1 = P$
if $k > 0$ then
  if $k = 1(mod2)$ then
    $S_0 = S_0 + S_1$
    $S_1 = 2S_1$
    $k = k/2$
  end if
end if
return $(Q = S_1 = kP)$
```

Figure 2 presents ECC hierarchy, thus, scalar multiplication is based in two main operations: point addition and point doubling. These operations need arithmetic operations (multiplication, addition and squaring). An optimized implementation is based on an optimized arithmetic operations modules.

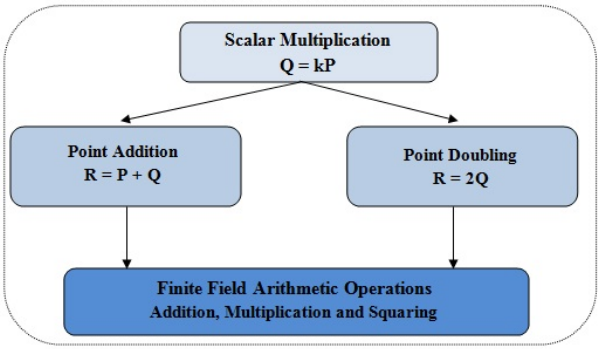
Scalar Multiplication
Q = kP
Point Addition
R = P + Q
Point Doubling
R = 2Q
Finite Field Arithmetic Operations
Addition, Multiplication and Squaring

Fig. 2. Elliptic Curve Hierarchy

It is possible to calculate Q=[k]P, but it's difficult to find k knowing the points P and Q. The security of ECC cryptosystems is based on the difficulty to solve this problem called Elliptic Curve Discrete Logarithm Problem (ECDLP).

4. SOFTWARE IMPLEMENTATION OF ECC

In this section, ECC software implementation design and its simulation using Intel(R) Core(TM) i3 CPU M380 @ 2.87GHz processor (3 GB RAM, 32-bit architecture) will be presented. Implementation was done in C language using GNU Multiple Precision (GMP) library which support large integers, for this reason that the main target applications for it are cryptography applications and algebra systems. The advantage for GMP increases with the operand sizes for many operations, since GMP uses asymptotically faster algorithms [14]. In [15], M. Welschenbach compared between the FLINT/C and GMP functions. He find that the GMP multiplication and division are faster by 30% and 40% than the corresponding FLINT/C functions. He showed that the GMP library is the fastest of the available libraries for large-integer arithmetic, Table 1 compares Multiplication operation in C, and using GMP Library which is the fastest. GMP developers prove that the GMP library achieved a speed of a factor of two comparing with other library.
”Double and Add” method for scalar multiplication and El-Gamal public key Encryption/Decryption Scheme based on DH was been used. Figure 3 presents an example of key exchange using Diffie-Hellman protocol.
Alice and Bob would like to communicate secret messages together. They should fix system parameters; they agree on an elliptic curve E over $\mathbb{F}_{q}$ with $q = p^n$ is large (and p is prime), a group (G, +, O) and a point $P \in E$ generator of the group and of order p. For the key generation, Alice chooses random $n \in [1, q-1]$ and calculates $K_n = n \times P$. Bob also chooses random $s \in [1, q-1]$ and calculates $K_s = s \times P$. Then Alice and Bob exchange their keys K_n and K_s. Alice computes $K = n \times K_s$, and Bob calculates $K = s \times K_n$. So, the shared secret key $K = n \times s \times P$.

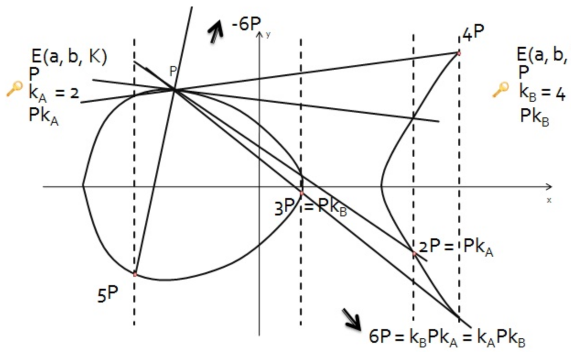

Fig. 3. Diffie-Hellman key exchange

Then, if Alice chooses a massage m to be sent to Bob, so the message encryption and decryption is as follow:
Encryption of a message m:

—Pick $k \in [1, q-1]$.
—Compute the points $kP = (x_1, y_1)$ and $kQ = (x_2, y_2)$, and $c = x_2 + m$.
—The ciphertext is (x_1, y_1, c).

Decryption:

—Compute $(x'_2, y'_2) = d(x_1, y_1)$ and $m = c - x'_2$.

Figure 4 shows the timing of El-Gamal protocol implemented in this work, the first remark is that when the key size increase, the necessary time for Encryption/Decryption increase slowly.

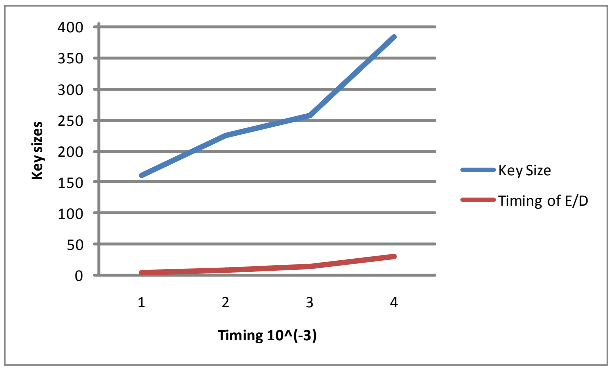

Fig. 4. The timings of El-Gamal Encryption/Decryption

In the next section, software implementations results will be presented and discussed.

5. RESULTS AND COMPARISON

In this section implementation results will be presented and compared with the state of the art software implementations. Table 2 summarize the timing of scalar multiplication of different software implementation.
Comparing this work to those in [16] and [17] over $\mathbb{F}_{2^{163}}$, there is a gain of 11.32% and 38.26% respectively. In addition, results of the proposed software implementation over $\mathbb{F}_{2^{233}}$ outperforms those in [16] and [17] by a 11.87% and 30.15% respectively. Performance of [16] and [17] decreases by about 10.92%and 27.49% respectively. Differential Fault attack will be applied, in the next section, to the proposed implementation in order to get the private key.

6. FAULT INJECTION ATTACKS

An Elliptic curve E is given by a nonsingular Weierstrass equation:

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6,$$

The coefficients a_i are in a field K and E(K) denotes the set of all solutions $(x, y) \in K \times K$, together with the point at infinity O. Biehl et al. [18] note that when calculating a scalar multiplication, the coefficient a_6 is not used. By changing it, from the curve E, another curve E’ is found such as:

$$E' : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a'_6$$

The result of the scalar multiplication is identical. If a cryptosystem receives a point P’(x’,y’) with $x', y' \in K$ but $P' \notin E$, then the scalar multiplication [k]P is calculated on the curve E’ defined above with $a'_6 = y'^2 + a_1x'y' + a_3y' - x'^3 - a_2x'^2 - a_4x'$ instead of the original curve E. But the point P’ is not arbitrary chosen, it’s chosen such that the cardinal points of E’ group has a small factor r and such that P’ is of order r. If r is relatively small, the attacker can solve the discrete logarithm problem in the subgroup of order r and find $k_r = kmodr$. Repeating the fault injection, the attacker gets several k_r for different r and, thanks to the Chinese remainder theorem, he can find the secret key k. In particular, an attack is possible by injecting any fault on the coordinates x or y point P. With stronger assumptions, the attacker can even find the secret k having injected any fault on the two coordinates.

6.1 Chinese Remainder Theorem

In this section, Chinese remainder theorem will be briefly presented.
Given a set of simultaneous congruences:

$$n = n_i(mod m_i)$$

The problem can be stated as finding n, given its remainders of division by several numbers $m_1, m_2, ..., m_k$:

$$\begin{cases} n = n_1 \ (mod \ m_1) \\ n = n_2 \ (mod \ m_2) \\ n = n_3 \ (mod \ m_3) \\ \quad . \\ \quad . \\ \quad . \\ n = n_r \ (mod \ m_r) \end{cases}$$

For $i = 1, ..., r$, all the m_i are pairwise relatively prime, the solution of the set of congruences is

$$x = (n_1 \times b_1 \times (M/(m_1)) + ... + n_r \times b_r \times (M/(m_r)))(modM)$$

where: $M = m_1 \times m_2 \times ... \times m_r$ and $b_i \times (M/m_i) = 1(modm_i)$.

6.2 ECC attack of the proposed software implementation

In this section, an example to apply NCCIP attack on the proposed software implementation will be detailed. Let's take an elliptic curve E over $\mathbb{F}_p$ having the following form:

$$y^2 = x^3 + a_4x + a_6mod5$$

, with $a_1 = a_2 = a_3 = 0$ and $a_4 \neq 0$.
The correspondent equation of a'_6 is: $a'_6 = y'^2 - x'^3 - a_4x'$.
Let Alice and Bob fix a point $P(0, 4) \in E$ and choose their private keys respectively: a=2 and b=3.
Alice and Bob calculate respectively: $P_a = a \times P$ and $P_b = b \times P$.
After exchange keys every one find that the shared key is: $K = b \times P_a = a \times P_b = a \times b \times P$ In this case $K = (2, 1)$.
The NCCIP attack consists on modified the input points coordinates, here only x-coordinate will be modified.

(1) $P \to G_1 = (1, 4)$, when replacing x' by 1 and y' by 4, then $a'_6 = 4$, the G_1 order is $r_1 = 3$. The new curve have the following equation:

$$E'_1 : y_1^2 = x_1^3 + x_1 + 4mod5$$

In this case: K= (0,0).

(2) $P \to G_2 = (2, 4)$, when replacing x' by 2 and y' by 4, then $a'_6 = 1$, the G_2 order is $r_2 = 3$. The new curve have the following equation:

$$E'_1 : y_1^2 = x_1^3 + x_1 + 1mod5$$

In this case: K= (0,0).

(3) $P \to G_3 = (3, 4)$, when replacing x' by 3 and y' by 4, then $a'_6 = 1$, the G_3 order is $r_3 = 5$. The new curve have the following equation:

$$E'_1 : y_1^2 = x_1^3 + x_1 + 1mod5$$

In this case: K= (2,4).

(4) $P \to G_4 = (4, 4)$, when replacing x' by 4 and y' by 4, then $a'_6 = 3$, the G_4 order is $r_4 = 2$. The new curve have the following equation:

$$E'_1 : y_1^2 = x_1^3 + x_1 + 3mod5$$

In this case: K= (1,0).

The obtained values can be used, here $r_1 = r_2$, and r_2, r_3 and r_4 are pairwise prime. Now, the Chinese reminder theorem will be applied :

$$\left.\begin{aligned} K_{r_2} = Kmodr_2 \\ K_{r_3} = Kmodr_3 \\ K_{r_4} = Kmodr_4 \end{aligned}\right\} \Rightarrow \left.\begin{aligned} (1,0) = kmod2 \\ (0,0) = kmod3 \\ (2,4) = kmod5 \end{aligned}\right\} \Rightarrow k = (2,1)$$

Finally, the key k is found which is (2,1).

7. COUNTERMEASURES

As the different cryptographic schemes, ECC implementation is susceptible to software attacks. This section discusses the potential countermeasures of software DFA attacks, specially NCCIP attack where an adversary attempts to change input point coordinates. For this reason, DFA attacks have to be taken into account, when implementing ECC in software and the appropriate countermeasures should be added. To avoid NCCIP attack in scalar multiplication, point P must be a valid point on the curve, as it is advised in the protocols of most ECC. This software countermeasures against DFA are applied at proposed source code, and different tests was made to find the key k, but with a fail. Experimental results illustrate the effectiveness of this approach on ECC software implementations.

8. CONCLUSION AND FUTURE WORK

In this work, an efficient software implementation of the scalar multiplication over prime field elliptic curves was proposed. One of the DFA attacks (NCCIP) was been studied. Thus, without protecting ECC software implementation there are a big risk that an adversary steal the key. Proposed implementation was analyzed and the appropriate countermeasures against DFA was applied.
A possible future work is to inject errors in the base field of elliptic curve, in prime p of $\mathbb{F}_p$ if a big curve characteristic is used. In addition, errors can be applied to curve parameters: a_1, a_2, a_3 or a_4.

9. REFERENCES

[1] D. R. Stinson, *Cryptography Theory And Practice*. 3th edition, Chapman and Hall/CRC, 2006.

[2] P. Longa and A. Miri, *New Composite Operations and Precomputation Scheme for Elliptic Curve Cryptosystems over Prime Fields*. In Ronald Cramer, editor, Public Key Cryptography, 11th International Workshop PKC 2008, volume 4939 of Lecture Notes in Computer Science, pp. 229-247. Springer, 2008.

[3] A. Sakthivel and R. Nedunchezhian, *Analyzing the Point Multiplication Operation of Elliptic Curve Cryptosystem over Prime Field for Parallel Processing*. The International Arab Journal of Information Technology, Vol. 11, No. 4, 2014.

[4] M. K. Brown, D. Hankerson, J. Lopez and A. Menezes, *Software implementation of the NIST elliptic curves over prime fields*. Proc. CT-RSA 2001. LNCS 2020, pp. 250-265. Springer, 2001.

[5] M. Bluhm and S. Gueron, *Fast Software Implementation of Binary Elliptic Curve Cryptography*.Cryptology ePrint Archive, 2013.

[6] J. M. Robert, *Software Implementation of Parallelized ECSM over Binary and Prime Fields*. Inscrypt: Information Security and Cryptology, 2014.

[7] T. Grandlund, *A software library for arbitrary precision integers*. `https://gmplib.org/gmp-man-6.0.0a.pdf`

[8] D. R.L. Brown, *Elliptic Curve Cryptography*. Certicom Corp 2009.

[9] H. Cohen and G. Frey, *Handbook of elliptic and hyperelliptic curve cryptography*.Chapman and Hall/CRC, 2006.

[10] C. Costello, P. Longa, and M. Naehrig, *A brief discussion on selecting new elliptic curves*, 2015.

[11] C. Giraud, *Fault resistant RSA implementation*. Second Workshop on Fault Detection and Tolerance in Cryptography (Edinburgh, UK) September 2, pp. 142-151, 2005.

[12] P. Novotney, *Weak Curves In Elliptic Curve Cryptography*. 2010. `http://ftp.mpir.org/edu/2010/414/projects/novotney.pdf`

[13] P. L. Montgomery, *Speeding the Pollard and Elliptic Curve Methods of Factorization*. Mathematics of Computation, pp. 243-264, 1987.

[14] `https://gmplib.org/`

[15] M. Welschenbach, *Cryptography in C and C++-Apress*. 2005.

[16] K. Chatterjee, A. De and D. Gupta, *Software Implementation of Curve based Cryptography for Constrained Devices*. International Journal of Computer Applications (0975-8887), vol. 24-No.5, June 2011.

[17] K. Bhardwaj and S. Chaudhary, *Implementation of Elliptic Curve Cryptography in 'C'*. International Journal on Emerging Technologies 3(2): 38-51, 2012.

[18] I. Biehl, B. Meyer and V. Muller, *Differential Fault Attacks on Elliptic Curve Cryptosystems*. CRYPTO-2000, 2000.

Table 1. Calculation times for multiplication operation in GMP Library and C.

	128	256	512	768	1024
Multiplication in C	1.5×10^{-6}	2.2×10^{-6}	4.6×10^{-6}	9.1×10^{-6}	1.4×10^{-5}
Multiplication in GMP	1.7×10^{-7}	5.5×10^{-7}	1.8×10^{-6}	3.7×10^{-6}	8.1×10^{-6}

Table 2. Timing of scalar multiplication operations in Elliptic Curve over different $\mathbb{F}_q$

Design	Field	Time (ms)
This work	$\mathbb{F}_{2^{163}}$	3.76
	$\mathbb{F}_{2^{233}}$	7.67
	$\mathbb{F}_{2^{283}}$	11.21
[16]	$\mathbb{F}_{2^{163}}$	4.24
	$\mathbb{F}_{2^{233}}$	8.61
	$\mathbb{F}_{2^{283}}$	12.72
[17]	$\mathbb{F}_{2^{163}}$	6.09
	$\mathbb{F}_{2^{233}}$	10.98
	$\mathbb{F}_{2^{283}}$	15.46