Binary Edwards Curves for Intrinsically Secure ECC Implementations for the IoT

Antoine Loiseau[1,2] and Jacques J. A. Fournier[3]
[1]*CEA-Tech, Gardanne, France*
[2]*Mines Saint Etienne, Gardanne, France*
[3]*Univ. Grenoble Alpes, CEA Leti, DSYS/LSOSP, F-38000 Grenoble, France*

Keywords: IoT, Elliptic Curves Cryptography, Binary Edwards Curves.

Abstract: Even if recent advances in public key cryptography tend to focus on algorithms able to survive the post quantum era, at present, there is an urgent need to propose fast, low power and securely implemented cryptography to address the immediate security challenges of the IoT. In this document, we present a new set of Binary Edwards Curves which have been defined to achieve the highest security levels (up to 284-bit security level) and whose parameters have been defined to fit IoT devices embedding 32-bit general purpose processors. We optimized the choice of the point generator with the w-coordinate to save a multiplication in the addition and doubling formulae. We manage to compute one step of the Montgomery Ladder in 4 multiplications and 4 squares. On top of the performance benefits, cryptography over such curves have some intrinsic security properties against physical attacks.

1 INTRODUCTION

Security and privacy have already been identified as vital issues of the "Internet of Things" (IoT) and as key to its adoption and deployment (Skarmeta and Moreno, 2014) (Rubens, 2014). End-to-end security has to be enforced, which means that the end-nodes of an IoT infrastructure shall also embed security features and applications. Asymmetric cryptography is one of such security tools and given the size, power and performance constraints on IoT end-nodes, Elliptic Curves Cryptography (ECC)(Koblitz, 1987) (Miller, 1986) is by far the best fit for such use cases and this despite the recently growing fear of quantum computers: since the latter are thought to be able one day to efficiently solve the Discrete Logarithm Problem (DLP), post-quantum cryptosystems are currently being studied. However, these cryptosystems as they exist today, cannot be used in the constrained IoT nodes due to the large keys and long computing times. ECC is still, in our opinion, the best short and mid term answer to address the immediate security needs for the IoT.

The main standardization effort for ECC has been through the FIPS 186-4 (Information Technology Laboratory, 2013) by the NIST. Alternative approaches have been proposed like (Brainpool, 2005)(Bernstein et al., 2011). The latter pieces of work open a new vision in the landscape of applied ECC. The so-called "NIST curves" were proposed in 1999 where "low power devices" were not as common as today and hence these curves were not designed for today's IoT constraints. Elliptic Curves usually used for cryptography are defined by the Weierstrass equation $y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$ and defined over a prime field with large prime characteristic or over binary extension fields of characteristic 2. However, the arithmetics of the Weiesrtrass form are not adapted to the low resource constraints of IoT devices with the additional constraint of being resistant to side channel attacks and fault attacks (Fan et al., 2010b). Adding countermeasures against these attacks severely downgrade the performance and efficiency of such ECC implementations, which can be a heavy drawback for the IoT end-nodes. This is why it would be more interesting to look for curves that could avoid such weaknesses. For example curves like Curve25519, (Bernstein et al., 2011) (Edwards, 2007) have a complete group law providing one same formula for point addition and point doubling. Another important aspect for an ECC implementation is the choice between curves over $\mathbb{F}_p$ or $\mathbb{F}_{2^m}$. Curves over large prime characteristics are usually preferred to those over binary fields due to claims that there may

Loiseau, A. and Fournier, J.
Binary Edwards Curves for Intrinsically Secure ECC Implementations for the IoT.
DOI: 10.5220/0006831504590465
In *Proceedings of the 15th International Joint Conference on e-Business and Telecommunications (ICETE 2018) - Volume 2: SECRYPT*, pages 459-465
ISBN: 978-989-758-319-3

be sub-exponential algorithms for solving the discrete logarithm problem for elliptic curves on binary fields. However even if the best optimistic conjectures of the DLP claim to have a complexity of $L_{2^n}(\frac{2}{3})$[1] or even $L_{2^n}(\frac{1}{2})$, (Galbraith and Gaudry, 2016) recalls that they are still "hypotheses" and that recent experiments *"raise the doubt about the subexponentiality claims"*. One of the main performance and security (Fouque et al., 2008b) advantages of arithmetics in binary fields is the carry-less operation involved.

Our approach consists in investigating about Binary Edwards Curves (BEC) (Bernstein et al., 2008) which, given the constraints of IoT devices, are one of the best choices for implementing ECC resistant to physical attacks: they are fast, have a complete group law and the underlying carry-less arithmetics would require low power and compact dedicated hardware accelerators. So far, only one such BEC has been proposed in the literature, which does not even offer what is the admitted reference of "128-bit security level". Hence we generated BECs, with higher security levels (112-bit to 284-bit) and with parameters tailored to fit the hardware constraints of IoT devices (working on data paths of 32 bits or 64 bits).

The rest of this paper details each step used for generating those curves. First Section 2 presents the concept of friendly polynomials and the associated advantages. Then Section 3 introduces the BECs with a focus on those curves having efficient arithmetics and fulfilling the security requirements 3.2. Section 4 details how we built the generator of the elliptic curve subgroup used aiming at performance gains. Section 5 provides the performance results and summarizes the expected intrinsic security properties against physical attacks. Some concluding remarks and perspectives are finally provided in the last section of this paper.

2 OPTIMAL FINITE BINARY FIELDS

The first step for generating a new set of elliptic curves for cryptography is to build "friendly" (i.e. for which calculations can be done rapidly) finite fields with an optimal arithmetic (i.e. for which some calculation "tricks" can be used for fast and compact implementations) for the targeted IoT devices. Arithmetics over elliptic curves depend on the representation of the base finite field which in turn depends on the choice of the modulus, represented under a polynomial form, used to define the finite field.

[1] $L_q(a) = e^{c\log(q)^a \log\log(q)^{1-a}}$

A choice usually made for polynomials in binary fields $\mathbb{F}_{2^n}$ is a trinomial or a pentanomial of the type:

$$p(x) = x^n + x^a + 1, \qquad n > a > 0$$
$$p(x) = x^n + x^a + x^b + x^c + 1, \quad n > a > b > c > 0$$

As suggested in (Scott, 2007), the construction of the binary field is important for the efficiency of the implementation and this depends on the architecture used. Let w be the length in terms of bits of the registers of the targeted architecture, Scott then defines a *Lucky Trinomial* with $n - a \equiv 0 \mod w$; a *Lucky Pentanomial* with $n - a \equiv 0 \mod w$, $n - b \equiv 0 \mod w$ and $n - c \equiv 0 \mod w$; and a *Fortunate Pentanomial* such that at least two of these equations are verified : $n - a \equiv 0 \mod w$, $n - b \equiv 0 \mod w$ and $n - c \equiv 0 \mod w$.

With a simple script in Sage (The Sage Developers, 2017) we searched all lucky trinomials and pentanomials for fields of bit lengths between 163 and 571: we found only 6 lucky trinomials compliant with a 32-bit architecture (except for $\mathbb{F}_{2^{257}}$ kept as an intermediate field between $\mathbb{F}_{2^{223}}$ and $\mathbb{F}_{2^{313}}$). As for pentanomials fitted for 32-bit architectures, we have a larger number of them over several finite fields. We also realised that most of those polynomials would also be compatible with 64-bit platforms, except for the trinomials of $\mathbb{F}_{2^{479}}$ and $\mathbb{F}_{2^{521}}$. So we did the additional effort of looking for polynomials compatible with 64-bit data widths for the latter levels of security, hence the two additional pentanomials in $\mathbb{F}_{2^{487}}$ and $\mathbb{F}_{2^{569}}$. The list of generated polynomials is as follows:

$$\mathbb{F}_{2^{223}} : x^{223} + x^{159} + 1$$
$$\mathbb{F}_{2^{257}} : x^{257} + x^{65} + 1$$
$$\mathbb{F}_{2^{313}} : x^{313} + x^{121} + 1$$
$$\mathbb{F}_{2^{431}} : x^{431} + x^{303} + x^{239} + x^{111} + 1$$
$$\mathbb{F}_{2^{479}} : x^{479} + x^{255} + 1$$
$$\mathbb{F}_{2^{487}} : x^{487} + x^{295} + x^{167} + x^{39} + 1$$
$$\mathbb{F}_{2^{521}} : x^{521} + x^{489} + 1$$
$$\mathbb{F}_{2^{569}} : x^{569} + x^{441} + x^{313} + x^{121} + 1$$

3 BINARY EDWARDS CURVES

Among the recently proposed models of elliptic curves for cryptography, Binary Edwards Curves (BEC) exhibit several interesting properties for IoT applications. First, these curves favour implementations that will be intrinsically safe against a set of side channel and fault attacks (see Section 5). Second, these curves allow an efficient arithmetic when

using the correct coordinates' representations and fast arithmetics in binary fields.

In (Edwards, 2007), Harold Edwards introduced a new model for elliptic curves allowing efficient computations with a complete group law. In (Bernstein et al., 2008) (Bernstein, 2009), Bernstein *et al.* constructed an equivalent to Edwards curves onto binary fields.

Definition 1. *Binary Edwards Curves (BEC): Let K be a field of characteristic 2 and let $d_1, d_2 \in K$ with $d_1 \neq 0$ and $d_2 \neq d_1^2 + d_1$. The Binary Edwards Curve, with coefficients d_1 and d_2, is the affine equation:*

$$E_{d_1,d_2} : d_1(x+y) + d_2(x^2+y^2) = xy + xy(x+y) + x^2y^2$$

This curve is symmetric and we have $\forall P \in E(K), P = (x,y) \Rightarrow -P = (y,x)$. We define the neutral point of the group law by $(0,0)$.

Definition 2. *Projective Closure: The projective closure of the BEC E_{d_1,d_2} is:*

$$d_1(X+Y)Z^3 + d_2(X^2+Y^2)Z^2 = XYZ^2 + XY(X+Y)Z + X^2Y^2$$

The last important aspect of this construction is that each BEC is birationally equivalent to a Weierstrass form.

3.1 Arithmetics on BEC

There are special cases where the arithmetics associated with the BEC can be optimised. For the case where $d_1 = d_2$ we have the following equation for a BEC:

$$d(x+y+x^2+y^2) = xy + xy(x+y) + x^2y^2 \qquad (1)$$

with the condition $\nexists t \in K \Rightarrow d = t^2 + t$. In the following sections, we consider this particular case of BECs.

Addition and Doubling: Efficient arithmetics on elliptic curves depend on the choice for the coordinates' representation. In the case of BECs, the projective w-coordinate with differential additions and doublings (Bernstein et al., 2008) is the fastest way for doing points addition and point doubling. The idea of the differential w-coordinate is to represent a point $P(x,y)$ of the curve by $w(P) = x+y$. This kind of coordinates is well suited for implementing the Montgomery ladder (Joye and Yen, 2002) to compute the exponentiation of a point. The Montgomery ladder computes $w(kP)$ and $w(kP+P)$ with an addition and a doubling at each step. For the addition and doubling we use a scalar $w_1 = w(Q-P)$ which is constant at each step of the ladder. Using the Co-Z trick (Kim et al., 2014)(Koziel et al., 2015) yields Algorithm 1 for adding and doubling operations taking 5 multiplications, 4 squares and 1 multiplication by d.

Algorithm 1: w-coordinates Adding and Doubling revisited with the Co-Z trick.

Require: $W_2, W_3, Z, \frac{1}{w_1}$
1: $C \leftarrow (W_2 + W_3)^2$
2: $D \leftarrow Z^2$
3: $E \leftarrow \frac{1}{w_1} C$
4: $U \leftarrow E + C$
5: $V \leftarrow E + D$
6: $S \leftarrow (W_2(Z + W_2))^2$
7: $T \leftarrow S + dD^2$
8: $W_4 \leftarrow UT$
9: $W_5 \leftarrow VS$
10: $Z' \leftarrow VT$
11: **return** W_4, W_5, Z'

3.2 Generating Secure BECs

We implemented methods and tools to generate and propose a set of BECs with different field sizes for different security levels in order to match a targeted device's requirements and the best security/efficiency (power and performance) trade-offs for a minimum security level of 128 bits. In (Bernstein, 2009) the authors also enumerate the following security requirements for a BEC (and ECC in general):

- *Prime degree of the finite field extension:* A large prime degree has to be chosen due to the birthday paradox and has to provide a sufficient level security (Gaudry et al., 2002).
- *Number of points:* The curve E shall have a cardinal of $2^c p$ where p is a large prime and c shall be small (1, 2 or 3).
- *Number of points on the Twist:* We have the same requirement on the number of points of the twist of the curve E in order to ensure resistance against some fault attacks on the original curve.
- *j-invariant:* The j-invariant of the BEC, $1/d^8$, shall generate the extension field.
- *Avoiding small discriminants:* The discriminant $\Delta_E = \mathrm{Tr}(E)^2 - 4q$ shall be divisible by a large prime exactly once.
- *Avoiding pairing attack:* The multiplicative order k of 2^m modulo the large prime of $|E|$ shall be large. We have the same condition with the prime of the twist. We used an embedding degree greater than $\frac{l-1}{100}$ as proposed in (Brainpool, 2005).

Based on these requirements, we can check, for each d, if a given curve is secure. We implemented the AGM method (Cohen et al., 2006) to perform the point counting. Moreover, to avoid counting the

Table 1: Performances of a kP operation without countermeasures over new BECs on a RISC-V processor at 100 MHz.

Curves	w-coor., optimal generator	w-coor., random generator	proj. coor., random generator
BEC223	32 ms	39 ms	137 ms
BEC257	46 ms	57 ms	201 ms
BEC313	79 ms	96 ms	342 ms
BEC431	188 ms	231 ms	821 ms
BEC479	242 ms	299 ms	1064 ms
BEC487	264 ms	326 ms	1156 ms
BEC521	316 ms	390 ms	1388 ms
BEC569	396 ms	489 ms	1744 ms

Table 2: SCA and Fault Attacks against BEC-based ECC.

Physical Attacks	Intrinsic Resistance		Remaining vulnerability
	Due to choice of parameters of BEC	Due to implementation done	Additional countermeasure implemented
Timming Attack (Kocher, 1996)	Unified arithmetics	Use of Montgomery ladder	-
SPA (Fan et al., 2010b)		Constant time programming	
DPA (Kocher et al., 1999)	-	-	Randomization
Template Attack (Herbst and Medwed, 2008)			of coordinates (Coron, 1999)
Relative Doubling Attack (Yen et al., 2005)	-	-	Blinded scalar (Coron, 1999)
RPA/ZPA (Akishita and Takagi, 2003)	w-coordinates arithmetics	Direct implementation of the generator	-
Carry-based Attack (Fouque et al., 2008b)	Binary curves chosen	-	-
Horizontal Attack (Clavier et al., 2010)	-	-	Attack model to be tested
Safe error (Fan et al., 2010b)	-	Use of Montgomery ladder	-
Invalid point analysis (Biehl et al., 2000)	-	-	Verify that point in on the curve
Invalid curve analysis (Ciet and Joye, 2005)	Curves' parameters on Twist for eg	Direct implementation of curve parameter	-
Twist Attack (Fouque et al., 2008a)	Curves' parameter for Twist	-	-
Differential Fault Attack (Biehl et al., 2000)	-	-	Blinded scalar (Coron, 1999)

number of points on the twist, we use the relationship between a curve E and its twist E^{tw} such that $|E|+|E^{tw}| = 2q+2$ to determine the number of points on the twist. We implemented this curve generation algorithm in C language and based on the FLINT library (FLINT, 2015). Curves with different sizes have been generated as given in Appendix (parameters for $m = 223$ and $m = 569$ have been removed due to page limitations).

4 OPTIMAL GROUP GENERATOR

In our case we have curves with a number of points $n = 2^c p$ with $c = 1, 2$ or 3 and p a large prime. So, we look for a point of the curve of order p. In the "Add and Double" formula of the BEC (Algorithm 1) we have a multiplication by the scalar $\frac{1}{w_1}$ which is constant due to the property of the Montgomery Ladder. We hence looked for a generator G with a $\frac{1}{w(G)}$ having a small Hamming Weight, hence saving 1 multiplication for each addition and doubling step. If we consider $w = x + y$, then from the equation 1 we have a new equation of BEC:

$$d(w+w^2) = x^4 + (1+w+w^2)x^2 + (w+w^2)x \quad (2)$$

So, once we choose an inverse of w with a small Hamming Weight, we can calculate the x coordinate of G from the equation 2 and its y coordinate from the equation 1.

5 PERFORMANCES & SECURITY

We implemented this new set of elliptic curves on a RISC V core running at 100 MHz. No special instruction has been added to the RISC V. We implemented the differential *w*-coordinates system with two cases, one with the optimized generator and the other with a random generator. We added the projective coordinates system for an interesting comparison and to show how the *w*-coordinates could be useful. For the finite field multiplication in $GF(2^n)$ we use a López-Dahab multiplication (Aranha et al., 2010). The exponentiation computation is performed by using a Powering Montgomery Ladder (Joye and Yen, 2002). All functions for the finite field arithmetics are written in Assembly Language and functions for ECC arithmetics in C language. The performances are given in Table 1. These performances are for a computation of kP without additional countermeasures (we are theoretically secure against against a set of side channel attacks and fault attacks). We can compare with performances of MbedTLS on an ARM M3 running at 96MHZ presented in (Tschofenig et al., 2015) (one of the rare public data available). For a security level of 128 bits, an ECDSA signature (where the main operation is the kP operation) takes 122 ms with the standard secp256r1 curve, which is to be compared to 46 ms (for the kP operation) in our case. Note that the performances in (Tschofenig et al., 2015) are given for a sliding window implementation, which is still vulnerable to side channel and fault attacks. Table 2 summarizes the theoretical advantages of the BEC model with respect to Side Channel Attacks (SCA) and Fault Attacks (FA). We hard-coded the d parameter of the curve and the $\frac{1}{w}$ parameter of the point generator to avoid any attack of invalid curve analysis and invalid point analysis.

6 CONCLUSION

In this paper, we propose a new set of BECs which might be efficient, secure and low power alternatives for implementing ECC in IoT devices. The optimization of the *w*-coordinate of the point generator allows a speed up if these coordinates are used with the Montgomery Ladder. This new set of curves covers different security levels to offer the possibility of making the best trade-off between security and execution time, bearing in mind that those BECs have some intrinsic protections against a set of side channel and fault attacks. Future work will consist in validating the expected "physical security" properties in practice. As the BEC model allows a complete compliance with the Weierstrass model, we shall also study how the use of BEC curves can be compliant, in practice, with already deployed implementations of ECDSA on Weierstrass curves. We shall also work on the implementation of a dedicated hardware accelerator in order to take advantage of the carry-less operations.

REFERENCES

Akishita, T. and Takagi, T. (2003). Zero-value point attacks on elliptic curve cryptosystem. In *International Conference on Information Security*, pages 218–233.

Aranha, D., Dahab, R., López, J., and Oliveira, L. (2010). Efficient implementation of elliptic curve cryptography in wireless sensors. 4(2):169–187.

Bernstein, D. (2009). Batch binary Edwards. In *Lecture Notes in Computer Science*, volume 5677 LNCS, pages 317–336.

Bernstein, D., Duif, N., Lange, T., Schwabe, P., and Yang, B.-Y. (2011). High-speed high-security signatures. LNCS 6917 :124–142.

Bernstein, D., Lange, T., and Rezaeian Farashahi, R. (2008). Binary edwards curves. LNCS 5154, pages 244–265.

Biehl, I., Meyer, B., and Mller, V. (2000). Differential fault attacks on elliptic curve cryptosystems. In *Annual International Cryptology Conference*, pages 131–146.

Brainpool (2005). ECC Brainpool Standard Curves and Curves Generation v1.0. http://www.ecc-brainpool.org/.

Ciet, M. and Joye, M. (2005). Elliptic curve cryptosystems in the presence of permanent and transient faults. *Designs, codes and cryptography*, 36(1):33–43.

Clavier, C., Feix, B., Gagnerot, G., Roussellet, M., and Verneuil, V. (2010). Horizontal correlation analysis on exponentiation. In *International Conference on Information and Communications Security*, pages 46–61.

Cohen, H., Frey, G., Avanzi, R., Doche, C., Lange, T., Nguyen, K., and Vercauteren, F. (2006). *Handbook of elliptic and hyperelliptic curve cryptography*. Discrete mathematics and its applications. Chapman & Hall/CRC.

Coron, J.-S. (1999). Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems. In CHES, vol. 1717, pages 292–302. LNCS.

Edwards, H. (2007). A normal form for elliptic curves. In *Bulletin of the American Mathematical Society*, pages 393–422.

Fan, J., Guo, X., Mulder, E. D., Schaumont, P., Preneel, B., and Verbauwhede, I. (2010b). State-of-the-art of secure ECC implementations: a survey on known side-channel attacks and countermeasures. In *HOST 2010*, pages 76–87.

FLINT (2015). FLINT: Fast Library for Number Theory. http://flintlib.org/.

Fouque, P.-A., Lercier, R., Ral, D., and Valette, F. (2008a). Fault attack on elliptic curve Montgomery ladder implementation. In *FDTC'08*, pages 92–98. IEEE.

Fouque, P.-A., Réal, D., Valette, F., and Drissi, M. (2008b). *The Carry Leakage on the Randomized Exponent Countermeasure*, pages 198–213. Springer Berlin Heidelberg, Berlin, Heidelberg.

Galbraith, S. and Gaudry, P. (2016). Recent progress on the elliptic curve discrete logarithm problem. *Designs, Codes, and Cryptography*, 78(1):51–72.

Gaudry, P., Hess, F., and Smart, N. (2002). Constructive and destructive facets of weil descent on elliptic curves. *Journal of Cryptology*, 15(1):19–46.

Herbst, C. and Medwed, M. (2008). Using templates to attack masked montgomery ladder implementations of modular exponentiation. In *International Workshop on Information Security Applications*, pages 1–13. Springer.

Information Technology Laboratory (2013). Digital Signature Standard (DSS). Technical Report NIST FIPS 186-4, NIST.

Joye, M. and Yen, S.-M. (2002). The Montgomery powering ladder. In *CHES*, pages 291–302.

Kim, K., Lee, C., Negre, C., and Negre, C. (2014). Binary edwards curves revisited. *Lecture Notes in Computer Science*, 8885:393–408.

Koblitz, N. (1987). Elliptic curve cryptosystems. *Mathematics of Computation*, 48(177):243–264.

Kocher, P. C. (1996). Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In *CRYPTO'96*, LNCS, pages 104–113.

Kocher, P. C., Jaffe, J., and Jun, B. (1999). Differential Power Analysis. In *CRYPTO'99*, number 1666 in LNCS, pages 388–397.

Koziel, B., Azarderakhsh, R., and Mozaffari-Kermani, M. (2015). Low-resource and fast binary edwards curves cryptography. *LNCS*, 9462:347–369.

Miller, V. (1986). Use of Elliptic-Curves in Cryptography. *Lecture Notes in Computer Science*, 218:417–426.

Rubens, P. (2014). Internet of Things a Potential Security Disaster.

Scott, M. (2007). Optimal Irreducible Polynomials for $GF(2^m)$ Arithmetic. In *IACR Cryptology ePrint Archive*, volume 2007, page 192.

Skarmeta, A. and Moreno, M. (2014). Internet of Things. In Jonker, W. and Petkovic, M., editors, *Secure Data Management*, LNCS, pages 48–53.

The Sage Developers (2017). *SageMath, the Sage Mathematics Software System (Version x.y.z)*. `http://www.sagemath.org`.

Tschofenig, H., Pegourie-Gonnard, M., and Vincent, H. (2015). Performance of State-of-the-Art Cryptography on ARM-based Microprocessors. In *NIST Lightweight Cryptography Workshop 2015 Session VII: Implementations & Performane*.

Yen, S.-M., Ko, L.-C., Moon, S., and Ha, J. (2005). Relative doubling attack against montgomery ladder. In *International Conference on Information Security and Cryptology*, pages 117–128. Springer.

APPENDIX

m	is the size of the extension of $\mathbb{F}_2$
f	is the modulus defining the representation of the finite field
d	is the parameter of the curve in BEC form, equation 1
G_x, G_y and $G_{1/w}$	are the coordinates of a friendly (32-bit) generator with $G_{1/w} = \frac{1}{w}$ where $w = G_x + G_y$
n	is the number of points of the curve
p	is the prime order of the sub-group generated by G
h	is the cofactor = 4

$m =$ 257; $f = x^{257} + x^{65} + 1$; $d = t^{65} + t^{31} + t^{14} + 1$

$G_x =$ 16b46e24aa4b12ab2289fcd3417615387810f083f43419d8c
ae38ad9ac640d960

$G_y =$ 16b46e24aa4b12aba289fcd3417615383810f083f43419d8e
ae38ad9ac640d968

$G_{1/w} =$ 10000000000000000000000000000000000000
000000000000

$n =$ 231584178474632390847141970017375815706332616967
362709021140632923291797618908

$p =$ 57896044618658097711785492504343953926583154241 8
406772552851582308229494047 27

$m =$ 313; $f = x^{313} + x^{121} + 1$; $d = t^{38} + t^{33} + t^{28} + 1$

$G_x =$ 15c67e3024c7c27466e72a3391256e9a729fc158092053d89
087c0f38408b214b0ade57363ea938

$G_y =$ 15c67e3024c7c27446e72a3391256e9a529fc158092053d8b
087c0f38408b214b0ade57363ea938

$G_{1/w} =$ 10000000000000001

$n =$ 166873987181321100187111070794496258953336290808
161456226545492179886000188954062240976633721 2

$p =$ 417184967953302750467777676986240647383340727020
403640566636373044971500047238515560774415843 03

$m =$ 431; $f = x^{431} + x^{303} + x^{239} + x^{111} + 1$; $d = t^{83} + t^{66} + t^{17} + 1$

$G_x =$ 4e1765c1f2f6140db17d5ef2f14c59a38a93e5b65ba9acca54
7bf2cc34f3d55bd85ccf4daeaf7ca1becaa8ee877b01f8d8ac
ae12b210

$G_y =$ 4e1765c1f2f6140d317d5ef2f14c59a30a93e5b65ba9acca54
7bf2cc34f3d55b585ccf4daeaf7ca13ecaa8ee877b01f8d8ac
ae12b210

$G_{1/w} =$ 10000000000000001

$n =$ 554533938824162971915682836828616740687287415075
163315034095916131188222853620417205051641025857
549800325003969881916422248862069 2

$p =$ 138633484706040742978920709207154185171821853768
790828758523979032797055713405104301262910256464
387450081250992470479105562215517 3

$m =$ 479; $f = x^{479} + x^{255} + 1$; $d = t^{73} + t^{29} + t^3 + 1$

$G_x =$ 7bdd9f19e11e888e80d7c093092d208b4fe996e8fcbdffa28c
c90173ece2c43673f1372e975ba9dcd3a06332abf15dbe9b67
9f6c63e30b884ab93272

$G_y =$ 3bdd9f19a11e888e40d7c093492d208b8fe996e8bcbdffa24c
c90173ece2c436f3f1372e175ba9dc53a063322bf15dbe1b67
9f6ce3e30b88cab93272

$G_{1/w} =$ 10000000000000001

$n =$ 156087427515799611569079861489658315287429907133 2
485575429578479812685869415448019717954458818867 6
304693469803241139597788961643097959459945585 6

$p =$ 390218568789499028922699653724145788218574767833 1
213938573946199531714673538620049294886147047169 0
761733674508102848994472404107744898649863958 9

$m =$ 487; $f = x^{487} + x^{295} + x^{167} + x^{39} + 1$; $d = t^{69} + t^{33} + t^{15} + 1$

$G_x =$ 339b843c53c409543f396d39e57efde813f06e3099735004b
999b15776a75a4c3a22dcaf1e91e261fe479b89a64d651039
28195d727bd3d157735b2071

$G_y =$ 339b843c53c40954bf396d39e57efde893f06e30997350043
999b15776a75a4c3a22dcaf1e91e261fe479b89a64d6510b9
28195d727bd3d1d7735b2071

$G_{1/w} =$ 10000000000000001

$n =$ 399583814440447005616844445413525287135820562261 1
163073099720908320475826018186213822669385452154 4
466825345498675533707766156971976964541332597784 4

$p =$ 998959536101117514042111113533813217839551405652 7
907682749302270801189565045465534556673463630386 1
167063363746688834269415392429942411353331494461

$m =$	521; $f = x^{521} + x^{489} + 1$; $d = t^{66} + t^{29} + t^{28} + 1$
$G_x =$	16b369b497b805e6199a342909aa4608cdcecb10e0988ba73 eb1f1186039c8b1f6d2a9db39b1302d29d9d449b9aa459cc5 d6bbb4e33a1eb8fcc056ce724cde5aaa8
$G_y =$	16b369b4b7b805e6199a342909aa4608cdcecb10e0988ba73 eb1f1186039c8b1f6d2a9db39b1302d29d9d449b9aa459cc5 d6bbb4e33a1eb8fcc056ce724cde5aaa8
$G_{1/w} =$	100000001
$n =$	68647976601306097149819007990813932172694353001433054093944634591855431833976570894395032898316224396265394341977861311221626406898579780051322403286027822 04
$p =$	17161994150326524287454751997703483043173588250358263523486158647963857958494142723598758224579056099066348585494465327805406601724644945012830600821506955 51