Math-Net.Ru

Общероссийский математический портал

Е. К. Alekseev, V. D. Nikolaev, S. V. Smyshlyaev, О свойствах стойкости эллиптических кривых в российских стандартах, *Матем. вопр. криптогр.*, 2018, том 9, выпуск 3, 5–32

DOI: <https://doi.org/10.4213/mvk260>

Использование Общероссийского математического портала Math-Net.Ru подразумевает, что вы прочитали и согласны с пользовательским соглашением

<http://www.mathnet.ru/rus/agreement>

Параметры загрузки:

IP: 54.198.67.100

26 апреля 2023 г., 02:29:09

**On the security properties
of Russian standardized elliptic curves¹**

E. K. Alekseev, V. D. Nikolaev, S. V. Smyshlyaev

CryptoPro LLC, Moscow

Получено 06.11.2017

Abstract. In the last two decades elliptic curves have become a necessary part of numerous cryptographic primitives and protocols. Hence it is extremely important to use the elliptic curves that do not weaken the security of such protocols. We investigate the elliptic curves used with GOST R 34.10-2001, GOST R 34.10-2012 and the accompanying algorithms, their security properties and generation process.

Key words: elliptic curve, GOST R 34.10-2001, GOST R 34.10-2012, Pollard ρ -method, R 50.1.114-2016

О свойствах стойкости эллиптических кривых в российских стандартах

Е. К. Алексеев, В. Д. Николаев, С. В. Смышляев

ООО «КРИПТО-ПРО», Москва

Аннотация. На протяжении последних двух десятилетий эллиптические кривые стали неотъемлемой частью различных криптографических примитивов и протоколов. В связи с этим особенно важно использовать эллиптические кривые, которые не ослабляют стойкость таких протоколов. В работе исследуется стойкость эллиптических кривых, определяемых российскими рекомендациями по стандартизации, относительно известных вычислительных задач в группе точек эллиптической кривой. В частности, показано, что атака, которую предложили Пети, Костерс и Мессинг, не применима к стандартизированным в Российской Федерации эллиптическим кривым. Описан процесс выработки указанных кривых, основанный на использовании криптографических функций хэширования и позволяющий гарантировать псевдослучайность получаемых значений.

Ключевые слова: эллиптическая кривая, ГОСТ Р 34.10-2001, ГОСТ Р 34.10-2012, метод Полларда, Р 50.1.114-2016

Citation: *Mathematical Aspects of Cryptography*, 2018, v. 9, № 3, pp. 5–32 (Russian)

© Академия криптографии Российской Федерации, 2018 г.

¹ The article was submitted by the Organizing Committee of the Symposium CTCrypt'2017.

Introduction

Russian national cryptographic standards employ elliptic curve cryptography since 2001, when the national digital signature standard GOST R 34.10-2001 [1] was adopted. Since this standard provided no set of elliptic curves except for the test usage, there was a need to generate standardized curves which provided reasonable implementation speed and safety as well as compatibility of different implementations. Thus, three 256-bit curves were generated and defined in [2].

In 2012 new national digital signature standard GOST R 34.10-2012 was adopted ([3]). This standard introduces 256- and 512-bit signature schemes. Since GOST R 34.10-2001 introduced only a 256-bit version of the scheme, it was necessary to create a set of 512-bit elliptic curves. The other problem was generation of elliptic curves that provided higher performance of high-level protocols. The perspectives for the use of curves of different types (including twisted Edwards curves and Montgomery curves) with the national standard GOST R 34.10-2012 and the algorithms accompanying the usage of the GOST R 34.11-2012 and GOST R 34.10-2012 national standards (defined in [4], [5]) have been studied in several papers including [6].

Four new elliptic curves (including two twisted Edwards curves) were adopted by the Technical committee for standardization «Cryptography and security mechanisms» as a standardization recommendation [7].

In this paper we are going to describe the process of generation of Russian standardized curves and provide some security considerations.

Notation

In this paper we use the following notation.

- p — characteristic of a finite field on which an elliptic curve is defined,
- a, b — short Weierstrass equation coefficients ($y^2 = x^3 + ax + b$),
- ε, δ — twisted Edwards curve equation coefficients ($\varepsilon u^2 + v^2 = 1 + \delta u^2 v^2$),
- $E(a, b)$ — the group of points of the elliptic curve defined by short Weierstrass equation with coefficients a and b ,
- $E(\varepsilon, \delta)$ — the group of points of the elliptic curve defined by a twisted Edwards curve equation with coefficients ε and δ ,
- m — the order of group of points of the elliptic curve,
- q — a prime number,
the order of the prime subgroup of the group of points of elliptic curve,

- P — a base point of the prime subgroup of the group of points of the elliptic curve,
- O — neutral element of group of points of the elliptic curve,
- m' — the order of the group of non-trivial quadratic twist points,
- q' — a prime number, the order of the prime subgroup of the group of non-trivial quadratic twist points.

By $\oplus$ we denote elliptic curve point addition. By $[a]P$ we denote the scalar point multiplication (point P is here multiplied by the integer a).

Curve generation

All standardized elliptic curves observed in this paper were generated with accordance to the verifiable pseudo-randomness principle. This curve generation principle allows to show that there are no properties of the elliptic curve known only by its developers. This property is met by selecting curve parameters as an output of a «one-way» function applied to a random seed. The knowledge of this random argument proves the fact that none of the curve parameters could be directly manipulated.

The curves which may be written in twisted Edwards form were selected by generating δ value. The GOST R 34.11-2012 hash function (Streebog, [8]) was used as a «one-way» function. As only 512-bit variant of the function was used, last 32 bytes of the output were taken when generating 256-bit curves. The obtained values were then taken modulo p . The random seeds and the resulting values of δ of the standardized curves are presented in the Appendix C.

The Weierstrass curves were generated by selecting a verifiable pseudo-random $k = a^3/b^2$ value and assuming $a = -3 \bmod p$ (such a value of a allows to create efficient implementations of elliptic curve arithmetic [9]). The method of selecting the value k is similar.

All base point of the standardized curves were selected by iterating abscissa (x for short Weierstrass form and u for twisted Edwards form) beginning with 0 till the point was in a subgroup of order q .

The use of such a generation procedure guarantees that it is impossible for developers to add vulnerabilities into resulting elliptic curves. We would like to mention that a so called «BADA55 curves» attack ([10]) does not violate the last statement since it requires the existence of a large publicly unknown class of weak curves. We suspect it is very unlikely that such a class could stay undiscovered for decades.

Cryptographic properties of the Russian standardized elliptic curves

Common computational problems on elliptic curves

In this paper we consider the following computational problems on the elliptic curve points groups.

- Discrete logarithm problem (DLP)
- Computational Diffie-Hellman problem (CDH)
- Decisional Diffie-Hellman problem (DDH)
- Discrete semilogarithm problem (DSLP)

It was shown in [11] that intractability of DSLP is a sufficient criterion of GMR-security (existential unforgeability under adaptive chosen message attack) of the digital schemes defined in [1] and [3] in the tamper-proof device model. The fact that the intractability of DDH is a sufficient condition of the VKO key agreement scheme (defined in [4], [5]), was shown in [12], while the intractability of the CDH problem is here the necessary condition.

It is easy to see that the following reductions are correct:

$$\text{DDH} \rightarrow \text{CDH} \rightarrow \text{DLP} \leftarrow \text{DSLP}$$

This statement implies that intractability of DLP is a necessary condition of the intractability of all the computational problems observed and therefore of the safety of all the cryptographic primitives and protocols discussed.

Generic algorithms

Here we show that the common algorithms which do not rely on inner structure of elliptic points groups but on some generic group assumptions only, do not violate the security of cryptographic primitives and protocols using Russian standardized curves. The conditions under which such algorithms work are described by the generic group model (see [13]). This paper also provides the results on the lower bounds of complexity of solving DDH, CDH and DLP. The same result for a similar group model was shown in [14]. These results imply that the best generic algorithm that solves the mentioned problems is Pollard's ρ -method ([15]) which has square-root complexity.

DSLP may be easily solved when one can efficiently solve DLP. Currently there are no known results showing that this problem may be solved in the other way. This implies that Pollard's ρ -method remains the best method that solves DSLP.

According to the requirements of [3] the order q of the elliptic curve points subgroup used in the standardized digital signature schemes should be at least 2^{254} for the 256-bit fields and at least 2^{508} for the 512-bit fields. Since these requirements are met for all Russian standardized curves, it is computationally infeasible to mount a successful attack based on Pollard's ρ -method.

Specific algorithms

The complexity of the computational problems discussed above is estimated on the base of known results. The report [16] contains a summary based on observation of a major amount of publications on the complexity of these problems. It concludes with the following statements.

- The fastest known algorithm that solves DLP is the Pollard's ρ -method.
- The fastest known algorithm that solves CDH is the computation of the discrete logarithm of one of public keys.
- The fastest known algorithm that solves DDH is the computation of the discrete logarithm of one of public keys.

It is also mentioned that these three problems may be solved more effectively than using Pollard's ρ -method in the groups having some effectively computable isomorphisms (so called «pairing groups»).

There are no known results on the methods of solving DSLP which had lower complexity than computing corresponding discrete logarithm. Thus, Pollard's ρ -method remains the most effective method for solving DSLP.

All the studied elliptic curves were generated according to the requirements of the national standard [3].

- $2^{254} < q < 2^{256}$ for the 256-bit curves and $2^{508} < q < 2^{512}$ for the 512-bit curves. All the generated curves satisfy this condition.
- P belongs to the curve, $P \neq O$ and $[q]P = O$. This condition is also satisfied for all curves.
- $4a^3/b^2 + 27 \neq 0 \pmod p$. If this inequality does not hold, the discriminant of the curve is equal to 0. This means the curve is not smooth and therefore elliptic. All the studied curves satisfy this condition.
- $b \neq 0 \pmod p$. If this condition is violated then j -invariant of the curve equals 1728 what violates the requirements of [3]. All the studied curves satisfy this condition.
- $a^3/b^2 \neq 0 \pmod p$. If this condition is violated, then $a = 0$. This implies that j -invariant is equal to 0 thus violating the requirements of [3]. All the studied curves satisfy this condition.

Further we study the properties of the elliptic curves which could decrease the complexity of the main computational problems on the groups of elliptic curve points and thus lead to vulnerabilities in cryptographic primitives and protocols using such curves. We are going to show that Russian standardized curves do not have these properties.

There are several properties of groups of elliptic curve points which may lead to weaknesses when violated. Here we base on the recommendations [17].

Condition 1. $(m, p) = 1$.

According to [18] for the groups of elliptic curve points that do not satisfy this criterion it is possible to construct an effective isomorphism (for which evaluation of a value takes $O(\ln p)$ time) to the additive group of F_p , therefore one can easily solve DLP in such groups in polynomial time.

It's easy to see that all the studied curves satisfy this condition.

Condition 2. $\text{ord } p \text{ in } \mathbb{Z}_q^*$ should not be small.

There exist algorithms (e. g. [19]) which solve DLP in prime subgroups of the elliptic curve points groups based on embedding of these groups into finite fields multiplicative subgroups. In order to increase the complexity of computing these embeddings (what implies high complexity of the algorithms solving DLP) one should check that $\text{ord } p \text{ in } \mathbb{Z}_q^*$ is large enough. We recommend to select curves in such a way that inequality

$$\text{ord } p > \frac{q-1}{100}$$

holds.

In order to calculate the value of $\text{ord } p$ effectively one has to factor $q-1$. The factorings of $q-1$ for the curves observed and the values of $\text{ord } p \text{ in } \mathbb{Z}_q^*$ may be found in Appendix D. It easy to see that these values satisfy the condition. This implies that it is not possible to compute embeddings effectively in a way they could be used to construct a DLP solving algorithm that had lower complexity than the Pollard's ρ -method.

Condition 3. *Complex multiplication discriminant should not be «small».*

According to SafeCurves (see [17]) Pollard's ρ -method may be made more efficient for the curves having small complex multiplication discriminant D . SafeCurves propose to use elliptic curves for which inequality $|D| > 2^{100}$ holds.

The values of discriminants are given in Appendix E. One can see that this condition is satisfied for all the examined curves except for the curve id-GostR3410-2001-CryptoPro-B-ParamSet. This does not make this curve practically vulnerable, however.

Condition 4. *The largest prime subgroup of the non-trivial quadratic twist points group should be secure.*

SafeCurves [17] proposes that the curve should be twist-secure in order to prevent attacks on scalar point multiplication using one coordinate formulae like Montgomery ladder. The orders of the subgroups of twisted groups of points of elliptic curves were estimated and checked to be secure against common attacks (these parameters may be found in Appendix F). These checks include security against attacks based on Pollard ρ -method and small embedding degrees.

These requirements are not satisfied for the curves GostR3410-2001-CryptoPro-A-ParamSet, GostR3410-2001-CryptoPro-C-ParamSet and id-tc26-gost-3410-12-512-paramSetB. These curves, however, cannot be transformed into Montgomery form so the attacks based on the use of the Montgomery ladder are not applicable. Nevertheless, we highly recommend to add checks that a point belongs to a curve in every implementation employing one coordinate formulae.

Condition 5. *The F_p^* group should not have many small subgroups.*

In 2016 Petit, Kesters and Messeng [20] had proposed an algorithm that solves DLP in the groups of elliptic curve points defined over finite fields of large characteristic. This algorithm is based on the factor base algorithm. Though the authors did not provide exact complexity estimations, they supposed that curves defined over fields with characteristic p , where $p - 1$ could be factorized into small primes, may be vulnerable to such an attack.

Here we try to obtain criteria which guarantees that implementing such an attack is more complex then implementing Pollard ρ -method when it is met. Suppose $p - 1 = p_1 \cdot \dots \cdot p_s$, where all p_i are prime and not necessarily distinct. For some fixed $B \in \mathbb{N}$ let us set

$$k = \log_2 \left(\prod_{j=1}^{n'} p_{i_j} \right) + 1,$$

where $p_{i_j} \in \{p_1, \dots, p_s\}$, $p_{i_j} < B$ and $n = \lfloor \log_2(p) \rfloor$, $m = \lfloor n/k \rfloor$. We will denote by $d_{\max}$ the value $\max\{4, p_{i_1}, \dots, p_{i_n}\}$.

Estimate. *If for every $B < n/4$ the following inequality holds:*

$$k + m (\log_2 m - 1) + 2 d_{\max} \log_2 m n' > \frac{n}{2},$$

then the proposed method is computationally harder then the Pollard ρ -method.

The proof of this Estimate and the proof of the fact that this condition is satisfied for all the curves may be found in Appendix A.

Conclusion

In the papers mentioned above it was shown that there are no generic methods solving DLP more efficiently than the Pollard's ρ -method. These papers also state that the best way to solve DDH, CDH or DSLP is to compute a value of corresponding discrete logarithm. So there are no known efficient generic algorithms that solve these problems.

We have also shown that none of known specific conditions (including the one proposed by Petit, Kesters and Messeng) that may lead to speed-ups in solving DLP holds for the examined curves.

The curve generation procedure was also presented here. This method employs a hash function in order to create values b and δ for the short Weierstrass and twisted Edwards form respectively. One can see that it hardens intentional embedding of the properties that could lead to vulnerabilities.

Thus we can conclude that the use of Russian standardized curves does not violate the security of the national standards and other high-level primitives.

Acknowledgements

The authors would like to thank Denis M. Dygin for his great help in calculations.

References

- [1] *Information technology. Cryptographic data security. Signature and verification processes of [electronic] digital signature, GOST R 34.10-2001*: Gosudarstvennyi Standart of Russian Federation, Government Committee of Russia for Standards, 2001 (in Russian).
- [2] Popov V., Kurepkin I., Leontiev S., "Additional cryptographic algorithms for use with GOST 28147-89, GOST R 34.10-94, GOST R 34.10-2001, and GOST R 34.11-94 algorithms", CRYPTO-PRO, 2006 (in Russian), <https://tools.ietf.org/html/rfc4357>.
- [3] *Information technology. Cryptographic data security. Signature and verification processes of [electronic] digital signature, GOST R 34.10-2012*: Federal Agency on Technical Regulating and Metrology, 2012 (in Russian).
- [4] *The use of cryptographic algorithms accompanying the usage of standards GOST R 34.10-2012 and GOST R 34.11-2012. Standardization recommendations*: Technical Committee 26. Federal Agency on Technical Regulating and Metrology, 2014 (in Russian).
- [5] Smyshlyaev S. (Ed.), Alekseev E., Oshkin I., Popov V., Leontiev S., Podobaev V., Belyavsky D., *Guidelines on the cryptographic algorithms to accompany the usage of standards GOST R 34.10-2012 and GOST R 34.11-2012*: RFC 7836, 2016 (in Russian), <https://tools.ietf.org/html/rfc7836.html>.
- [6] Alekseev E.K., Oshkin I.B., Popov V.O., Smyshlyaev S.V., Sonina L.A., "On the perspectives of the usage of twisted Edwards curves with the GOST R 34.11-2012 digital signature and the corresponding key agreement algorithm", *Information Security Problems. Computer Systems*, **3** (2014), 60–66 (in Russian).

- [7] *Information technology. Cryptographic data security. Parameters of elliptic curves for cryptographic algorithms and protocols*: Federal Agency on Technical Regulating and Metrology, 2016 (in Russian).
- [8] *Information technology. Cryptographic Data Security. Hashing function, GOST R 34.11-2012*: Federal Agency on Technical Regulating and Metrology, 2012 (in Russian).
- [9] SEC 2, “Standards for Efficient Cryptography Group: Recommended Elliptic Curve Domain Parameters. Version 1.0” (2000), www.secg.org/SEC2-Ver-1.0.pdf.
- [10] Bernstein D. J., Chou T., Chuengsatiansup C., Huelsing A., Lambooj E., Lange T., Niederhagen R., van Vredendaal C., “How to manipulate curve standards: a white paper for the black hat”, <http://bada55.cr.yp.to/pubs.html>.
- [11] Varnovskii N. P., “Provable security of digital signatures in the tamper-proof device model”, *Diskretnaya Matematika*, **20**:3 (2008), 147–159 (in Russian); *Discrete Math. Appl.*, **18**:4 (2008), 427–437 (in English).
- [12] Alekseev E. K., Oshkin I. B., Popov V. O., Smyshlyaev S. V., “On the cryptographic properties of algorithms accompanying the applications of standards GOST R 34.11-2012 and GOST R 34.10-2012”, *Mathematical Aspects of Cryptography*, **7**:1 (2016), 5–38 (in Russian).
- [13] Shoup V., “Lower bounds for discrete logarithms and related problems”. In: “EUROCRYPT 97”, *Lect. Notes Comput. Sci.*, **1233**, 1997, 256–266.
- [14] Nechaev V. I., “Complexity of a determinate algorithm for the discrete logarithm”, *Matem. Zametki*, **55**:2 (1994), 91–101 (in Russian); *Mathematical Notes*, **55**:2 (1994), 165–172 (in English).
- [15] Pollard J. M., “Monte Carlo methods for index computation (mod p)”, *Math. Comput.*, **32** (1978), 918–924.
- [16] *Final Report on Main Computational Assumptions in Cryptography*, ed. Vercauteren F.: ECRYPT II, ICT-2007-216676, <http://www.ecrypt.eu.org/ecrypt2/documents/D.MAYA.6.pdf>.
- [17] SafeCurves: choosing safe curves for elliptic-curve cryptography <https://safecurves.cr.yp.to/index.html>.
- [18] Semaev I. A., “Evaluation of discrete logarithms in a group of p -torsion points of an elliptic curve in characteristic p ”, *Math. Comput.*, **67** (1998), 353–356.
- [19] Menezes A., Vanstone S., “Reducing elliptic curve logarithms to logarithms in a finite field”, *IEEE Trans. Inform. Theory*, **IT-39**:5 (1993), 1639–1646.
- [20] Petit C., Kisters M., Messeng A., “Algebraic approaches for the elliptic curve discrete logarithm problem over prime fields”. In: “PKC 2016, Part II”, *Lect. Notes Comput. Sci.*, **9615**, 2016, 3–18.
- [21] Koblitz N., *A Course in Number Theory and Cryptography*, Berlin–Heidelberg etc.: Springer-Verlag, 1987, viii+208 pp.; Коблиц Н., *Курс теории чисел и криптографии*, М.: ТБП, 2001, 254 с.
- [22] Semaev I. A., *Summation polynomials and the discrete logarithm problem on elliptic curves*, Cryptology ePrint Archive: Report 2004/031.
- [23] Semaev I. A., *New algorithm for the discrete logarithm problem on elliptic curves*, Cryptology ePrint Archive: Report 2015/310.
- [24] Faugere J.-C., “A new efficient algorithm for computing Groebner bases (F_4)”, *J. Pure Appl. Algebra*, **139**:1–3 (1999), 61–88.

- [25] Crandall R., Pomerance C., *Prime Numbers: A Computational Perspective*, N.-Y. etc.: Springer, 2001, xv+545 pp.
- [26] Collins G., “The calculation of multivariate polynomial resultants”, *J. Assoc. Comput. Mach.*, **18**:4 (1971), 515–532.
- [27] Diem C., “On the discrete logarithm problem in elliptic curves. II”, *Algebra & Number Theory*, **7**:6 (2013), 1281–1323.
- [28] Huang Y.-J., Petit C., Shinohara N., Takagi T., *On generalized first fall degree assumptions*, Cryptology ePrint Archive: Report 2015/358.

Appendices

A Resistance to Petit–Kosters–Messeng attack

Here we give a brief description of the algorithm proposed by Petit, Kusters and Messeng in [20] and prove Estimate. Here we denote by $\overline{K}$ the algebraic closure of a field K and by the expression “ $a \in_R B$ ” we mean “ a is taken uniformly random from the set B ”. We also denote by $\text{Res}_X(f(X), g(X))$ a resultant of polynomials $f(X)$ and $g(X)$ by variable X .

A.1 Algorithm description

The algorithm proposed by Petit, Kusters and Messeng in [20] uses so called factor bases (see [21]). Suppose Ω is some factor base which consists of elliptic curve points. For some fixed $m \in \mathbb{N}$ (we will provide comments on this value later) we need to construct $N + 1$ equations:

$$[\alpha_i]P \oplus [\beta_i]Q = \bigoplus_{j=1}^m W_j, \quad W_j \in \Omega, \quad \alpha_i, \beta_i \in_R \{1, \dots, q-1\}, \quad i \in \{1, \dots, N+1\}.$$

We obtain a $(N+1) \times N$ matrix of coefficients. Then using Gaussian elimination we can obtain equation $[\alpha]P \oplus [\beta]Q = O$ and deduce the value of the discrete logarithm as $-\alpha\beta^{-1} \pmod{q}$.

These equations may be constructed by means of Semaev polynomials defined in [22].

Definition A.1. The i -th Semaev summation polynomial $S_i(x_1, \dots, x_i) \in \mathbb{F}_p[x_1, \dots, x_i]$, $i \geq 2$, is a polynomial which turns to 0 on the input $x_1, \dots, x_i \in \overline{\mathbb{F}}_p$ if and only if there exist $y_1, \dots, y_i \in \overline{\mathbb{F}}_p$: $(x_1, y_1) \in E(\overline{\mathbb{F}}_p), \dots, (x_i, y_i) \in E(\overline{\mathbb{F}}_p)$ such that $\bigoplus_{j=1}^i (x_j, y_j) = O$.

Theorem A.1 (I. A. Semaev, [22]).

1. The following equalities hold: for any $n \geq 4$, $1 \leq k \leq n-3$

$$\begin{aligned} S_2(x_1, x_2) &= x_1 - x_2, \\ S_3(x_1, x_2, x_3) &= (x_1 - x_2)^2 x_3^2 - 2((x_1 + x_2)(x_1 x_2 + a) + 2b)x_3 + \\ &\quad + ((x_1 x_2 - a)^2 - 4b(x_1 + x_2)), \\ S_n(x_1, \dots, x_n) &= \text{Res}_X(S_{n-k}(x_1, \dots, x_{n-k-1}, X), S_{k+2}(x_{n-k}, \dots, x_n, X)), \end{aligned}$$

2. All polynomials are symmetric and have degree 2^{n-2} in each variable for $n \geq 3$.

It is easy to see that the total degree of i -th Semaev polynomial does not exceed $i \cdot 2^{i-2}$ when $i > 2$. Indeed, the monomial $x_1^{2^{i-2}} \cdot \dots \cdot x_i^{2^{i-2}}$ has the highest total degree in the i -th Semaev polynomial.

Suppose an elliptic curve point R has coordinates (R_x, R_y) . The problem of decomposing of R into a sum of m points of $E(\overline{\mathbb{F}}_p)$ may be expressed as

$$S_{m+1}(x_{1,1}, \dots, x_{m,1}, R_x) = 0.$$

Since we have to decompose R not just into m random points but into the points from the factor base Ω we have to add constraints on the variables $x_{1,1}, \dots, x_{m,1}$. The authors of [20] write down these constraints as a system of additional polynomial equations which are satisfied if and only if a point specified by its coordinates belongs to Ω .

The authors of [20] propose to define factor base Ω as a set of points having abscissas belonging to some subgroup of a multiplicative group of a finite field the curve is defined on. Suppose the following equality holds:

$$p - 1 = r \prod_{i=1}^{n'} p_i,$$

where p_i are primes (not necessarily pairwise distinct). Let us select a subgroup of F_p^* with order equal to $\prod_{i=1}^{n'} p_i$ as a subgroup of possible abscissas of points of factor base. Then the polynomial system of constraints may be written as follows:

$$\begin{cases} x_{i,j+1} - x_{i,j}^{p_j} = 0, & i = 1, \dots, m, j = 1, \dots, n' - 1, \\ x_{i,n'}^{p_{n'}} - 1 = 0, & i = 1, \dots, m. \end{cases} \quad (\text{A.1})$$

Here variables $x_{i,j}$, $j \geq 2$, are introduced to decrease the degrees of the constraint equations added to the system.

Finally we obtain a system of equations over $\mathbb{F}_p$ which consists of $mn' + 1$ equations with mn' variables:

$$\begin{cases} S_{m+1}(x_{1,1}, \dots, x_{m,1}, R_x) = 0, \\ x_{i,j+1} - x_{i,j}^{p_j} = 0, & i = 1, \dots, m, j = 1, \dots, n' - 1, \\ x_{i,n'}^{p_{n'}} - 1 = 0, & i = 1, \dots, m. \end{cases} \quad (\text{A.2})$$

The maximal degree of an equation in the system equals

$$\max(B, (m+1) \cdot 2^{m-1}), \quad \text{where} \quad B = \max_{1 \leq i \leq n'} p_i + 1.$$

Here we may also try to construct a system with a smaller maximal degree using the technique proposed by Semaev in [23]. We change one

$$S_{m+1}(x_{1,1}, \dots, x_{m,1}, R_x) = 0$$

equation into a set of equations employing S_3 polynomial. The system (A.2) may be rewritten as

$$\begin{cases} S_3(x_{1,1}, x_{2,1}, u_1) = 0, \\ S_3(u_i, x_{i+2,1}, u_{i+1}) = 0, \quad i \in 1, \dots, m-3, \\ S_3(u_{m-2}, x_{m,1}, R_X) = 0, \\ x_{i,j+1} - x_{i,j}^{p_j} = 0, \quad i = 1, \dots, m, \quad j = 1, \dots, n' - 1, \\ x_{i,n'}^{p_{n'}} - 1 = 0, \quad i = 1, \dots, m. \end{cases} \quad (\text{A.3})$$

The new system (A.3) consists of $mn' + m - 1$ equations over $\mathbb{F}_p$ with $mn' + m - 2$ variables. The equations have now maximal degree equal to $\max(B, 4)$.

Systems (A.2) and (A.3) may be solved by means of Grobner bases. In order to construct such a basis one can use common algorithms like F_4 (see [24]).

A.2 Complexity estimations

Complexity of the method described may be estimated for each of three parts the algorithm consists of. First part is the initialization step. Second part is the decomposition generation step. The third part is the linear algebra step. Now we are going to estimate complexities of all steps.

The initialization step is performed once for each curve. One has to factor n -bit length value $p - 1$, this may be done in $T_{\text{factor}} = O(2^{c_1 n^{1/3} \log_2^{2/3} n})$ operations using number field sieve method [25]. The other problem is to generate $(m+1)$ -th Semaev polynomial. It may be done in $T_{\text{Sem}} = O(2^{m^2})$ operations (according to the complexity of computing the resultant of two polynomials obtained in [26]). If one uses (A.3), then only third Semaev polynomial is used. This polynomial is implicitly defined in [22], therefore in this case $T_{\text{Sem}} = 0$.

Now we are going to estimate the complexity of the third step. After collecting all the decompositions we obtain a sparse matrix with size $(N + 1) \times N$, which has only m non-null n -bit elements in each row. Complexity of finding a nontrivial equation connecting P and Q may be estimated as $T_{GE} = O(mnN^{\omega'})$, where $\omega' \leq 2$.

The main problem is to estimate the complexity of the second step. One has to find $N + 1$ decompositions of random elliptic curve points into m points of the factor base Ω . We will denote by T_{PDP} (PDP here means point decomposition problem) the mean complexity of constructing such decomposition, thus the complexity of the second step equals $(N + 1) \cdot T_{PDP}$. One obtains a decomposition by solving (A.2) or (A.3). We will denote the probability that one instance of the system has solutions as P_{PDP} . We will also denote the complexity of solving one instance of the system (A.2) or (A.3) as T_{trial} . Therefore $T_{PDP} = P_{PDP}^{-1} T_{\text{trial}}$. Following [23] the complexity of solving one instance of a system may be estimated as $T_{\text{trial}} = O(V^{\omega D_{\text{reg}}})$, where V is a number of variables, ω is a linear algebra constant and D_{reg} is the «regularity degree».

Following [27] we can estimate P_{PDP} as

$$\frac{N^m}{m! |E(\mathbb{F}_p)|}.$$

Suppose $p - 1 = r \prod_{i=1}^{n'} p_i$, $p_i < B = 2^l$. We will put $\prod_{i=1}^{n'} p_i \approx 2^k$ and $p \approx 2^n$. The size N of the factor base Ω may be estimated as $O(2^k)$. This may be justified as follows. For a random $x \in G \subset \mathbb{F}_p^*$, where G is a subgroup of $\mathbb{F}_p^*$ of order $\prod_{i=1}^{n'} p_i$, the probability that $x^3 + ax + b$ is a quadratic residue modulo p is about $1/2$, while every solution gives us two points. Thus there exist about $O(2^{mk})$ sums of m points of the factor base Ω . Since the order of the summands does not change the sum we have to divide the number of sums by $m!$ in order to estimate the number of different sums. We use quite inaccurate estimate of the number of points using Hasse theorem as $O(2^n)$. Then we get the value $P_{PDP} = O(2^{mk-n}/m!)$.

In order to estimate the complexity of solving one instance of (A.2) and (A.3) one has to estimate the «regularity degree» of the corresponding polynomial ideal.

For some polynomial ring ideal $I \subseteq \mathbb{F}_p[x_1, \dots, x_t]$ with basis $f_1(x_1, \dots, x_t), \dots, f_s(x_1, \dots, x_t)$ the following inequality holds (see [28]):

$$D_{\text{reg}} \geq \max_{1 \leq i \leq s} \deg(f_i(x_1, \dots, x_t)).$$

Thus the (A.2) system may be solved in $T_{\text{trial}} = O(2^{\log_2(mn')\omega D_{\text{reg}}})$ operations, where $D_{\text{reg}} \geq \max(B, (m + 1) \cdot 2^{m-1})$, while the (A.3) system may be solved in $T_{\text{trial}} = O(2^{\log_2(mn' + m - 2)\omega D_{\text{reg}}})$, where $D_{\text{reg}} \geq \max(B, 4)$.

The total complexity of retrieving discrete logarithm T_{ECDLP} is:

$$T_{\text{ECDLP}} = T_{\text{factor}} + T_{\text{Sem}} + N \cdot P_{\text{PDP}}^{-1} T_{\text{trial}} + T_{\text{GE}},$$

what gives for the (A.2) system

$$T_{\text{ECDLP}} = O(2^{c_1 n^{1/3} \log_2^{2/3} n}) + O(2^{m^2}) + \\ + O(2^{k+m \log_2 m - m + n - m k + \log_2(m n') \omega D_{\text{reg}}}) + O(2^{w' k + \log_2 m + \log_2 n}),$$

and for the (A.3) system

$$T_{\text{ECDLP}} = O(2^{c_1 n^{1/3} \log_2^{2/3} n}) + O(2^{k+m \log_2 m - m + n - m k + \log_2(m n' + m - 2) \omega D_{\text{reg}}}) + \\ + O(2^{w' k + \log_2 m + \log_2 n}).$$

In order to avoid parameter m in the exponent, the authors of [20] suggested to assume $m = \lceil n/k \rceil$. Then for the both systems we may rewrite the complexity estimations as

$$T_{\text{ECDLP}} = O(2^{c_1 n^{1/3} \log_2^{2/3} n}) + O(2^{m^2}) + \\ + O(2^{k+m(\log_2 m - 1) + \log_2(m n') \omega D_{\text{reg}}}) + O(2^{w' k + \log_2 m + \log_2 n}),$$

$$T_{\text{ECDLP}} = O(2^{c_1 n^{1/3} \log_2^{2/3} n}) + \\ + O(2^{k+m(\log_2 m - 1) + \log_2(m n' + m - 2) \omega D_{\text{reg}}}) + O(2^{w' k + \log_2 m + \log_2 n}).$$

Now we are going to analyse the results in order to find out the values of m, k, n', B that make the complexity of the described method higher then the Pollard ρ -method, which has the complexity $T_{\text{Pollard}} = O(2^{n/2})$. We will denote by d_{max} the maximal degree of the polynomials emerging in the system. The following relations should hold:

$$\begin{cases} mk = n, & (\text{A.4a}) \\ m^2 < n/2, & (\text{A.4b}) \\ 2B < n/2, & (\text{A.4c}) \\ k + m(\log_2 m - 1) + 2d_{\text{max}} \log_2 m n' < n/2. & (\text{A.4d}) \end{cases}$$

The last three inequalities are the sufficient conditions for the search of the required amount of decompositions to take less time then running the Pollard ρ -method. If one of them is violated, then Pollard ρ -method remains more efficient than the described method. These relations correspond to the system (A.2). The corresponding relations for (A.3) differ only in the absence of the (A.4b) inequality.

Thus, we have proved Estimate.

A.3 256-bit curves

$$1. \quad p = 2^{256} - 617$$

$$p - 1 = 2 \cdot 7 \cdot 43 \cdot 9109 \cdot 87640387787 \cdot 16876409960174552741 \cdot \\ \cdot 14276683752608433211265709130033043243453$$

$$2. \quad p = 2^{255} + 3225$$

$$p - 1 = 2^3 \cdot 11 \cdot 33797 \cdot 633062117 \cdot 43400749232432159 \cdot \\ \cdot 39607009966486015397 \cdot 17888439653017795004024467$$

$$3. \quad p = 703900853520833051995477180190184378410795166300451804712 \\ 84346843705633502619$$

$$p - 1 = 2 \cdot 17 \cdot 37 \cdot 113 \cdot 244997 \cdot 7044765983457327077589232961 \cdot \\ \cdot 286896380833551689651093852714669043701$$

A.4 512-bit curves

$$1. \quad p = 2^{512} - 569$$

$$p - 1 = 2 \cdot 23 \cdot 41 \cdot 353 \cdot 105095387 \cdot 45130584520747958722981 \cdot \\ \cdot 582271299047893027187874292913927407 \cdot \\ \cdot 2440563294432588452310063876982204011061 \cdot \\ \cdot 2987936166061269764733822017919288608395313$$

$$2. \quad p = 2^{511} + 111$$

$$p - 1 = 2 \cdot 7 \cdot 17 \cdot 9433 \cdot \\ \cdot 29860769339941482698353859190482380663180854047591713 \\ 55727648743353559431103560755510084456357598398009587 \\ 265648506906928241789019762235105800049577$$

A.5 The effectiveness of using Petit–Kosters–Messeng method for breaking Russian standardized elliptic curves

Keeping in mind the fact that $2 < \omega \leq 3$ we have that for 256-bit curves B should be less than 64. Now we are going to estimate m and k for each curve using (A.4a) and (A.4c).

Curve	Divisors	k	m	n'	$d_{\max}$ ((A.3) case)
1	2	1	256	1	4
1	$2 \cdot 7$	4	53	2	7
1	$2 \cdot 43$	6	39	2	43
1	$2 \cdot 7 \cdot 43$	10	25	3	43
1	7	2	91	1	7
1	$7 \cdot 43$	9	27	2	43
1	43	5	47	1	43
2	2	1	256	1	4
2	$2 \cdot 2$	2	128	2	4
2	$2 \cdot 2 \cdot 2$	3	85	3	4
2	$2 \cdot 2 \cdot 2 \cdot 11$	6	39	4	11
2	$2 \cdot 2 \cdot 11$	5	46	3	11
2	$2 \cdot 11$	4	57	2	11
2	11	3	74	1	11
3	2	1	256	1	4
3	$2 \cdot 17$	5	50	2	17
3	$2 \cdot 37$	6	41	2	37
3	$2 \cdot 17 \cdot 37$	11	22	3	37
3	17	4	62	1	17
3	$17 \cdot 37$	9	27	2	37
3	37	5	49	1	37

It is easy to see that (A.4d) does not hold for any of the parameters specified, so the method that uses (A.2) or (A.3) is slower than the Pollard ρ -method.

Now we are going to study 512-bit curves. Here $B < 128$. Now we have the following limitations.

Curve	Divisors	k	m	n'	$d_{\max}$ ((A.3) case)
1	2	1	512	1	4
1	$2 \cdot 23$	5	92	2	23
1	$2 \cdot 41$	6	80	2	41
1	$2 \cdot 23 \cdot 41$	10	47	3	41
1	23	4	113	1	23
1	$23 \cdot 41$	9	51	2	41
1	41	5	95	1	41
2	2	1	512	1	4
2	$2 \cdot 7$	4	106	2	7
2	$2 \cdot 17$	5	100	2	17
2	$2 \cdot 7 \cdot 17$	7	64	3	17
2	7	2	182	1	7
2	$7 \cdot 17$	6	74	2	17
2	17	4	125	1	17

Here we see that (A.4d) does not hold for any of the parameter sets again, hence the Petit–Kosters–Messeng method does not provide any improvement of DLP solving in comparison with the Pollard ρ -method.

A.6 On some probably vulnerable curve

The authors of [20] mentioned NIST P-224 elliptic curve as probably vulnerable. This curve is defined over $\mathbb{F}_p$, where

$$p = 2^{224} - 2^{96} + 1.$$

Value $p - 1$ is factored as

$$2^{96} \cdot 3 \cdot 5 \cdot 17 \cdot 257 \cdot 641 \cdot 65537 \cdot 274177 \cdot 6700417 \cdot 67280421310721.$$

One can easily check having fixed $B = 56$ that inequality (A.4d) holds for some parameter sets. This fact, however, does not imply that method provided in [20] is computationally easier for this curve than the Pollard ρ -method.

B Parameters of the examined curves

Here we denote by $x(P)$ and $y(P)$ coordinates of the elliptic curve point P in short Weierstrass form. By $u(P)$ and $v(P)$ we denote coordinates of the elliptic curve point P in twisted Edwards form.

- id-GostR3410-2001-CryptoPro-A-ParamSet (256-bit Weierstrass curve)

$$p = 2^{256} - 617$$

$$a = p - 3 = -3 \bmod p$$

$$b = 166$$

$$m = 115792089237316195423570985008687907853073762908499243225$$

$$378155805079068850323$$

$$q = m$$

$$x(P) = 1$$

$$y(P) = 6403388114292720268364988145043347398593176026888494128$$

$$8852745803908878638612$$
- id-GostR3410-2001-CryptoPro-B-ParamSet (256-bit Weierstrass curve)

$$p = 2^{255} + 3225$$

$$a = p - 3 = -3 \bmod p$$

$$b = 2809101935305809009699697900030956075912436855801486595765$$

$$5842872397301267595$$

$$m = 578960446186580977117854925043439539271021331602558268200$$

$$68844496087732066703$$

$$q = m$$

$$x(P) = 1$$

$$y(P) = 2879266581485461129699234745838028413502863677822911300$$

$$5756334730996303888124$$
- id-GostR3410-2001-CryptoPro-C-ParamSet (256-bit Weierstrass curve)

$$p = 7039008535208330519954771801901843784107951663004518047128$$

$$4346843705633502619$$

$$a = p - 3 = -3 \bmod p$$

$$b = 32858$$

$$m = 703900853520833051995477180190184378409208826471640810353$$

$$22601458352298396601$$

$$q = m$$

$$x(P) = 0$$

$$y(P) = 2981889391773124073347127324031476992724055081238369568$$

$$9146495261604565990247$$

- id-tc26-gost-3410-12-512-paramSetA (512-bit Weierstrass curve)

$$p = 2^{512} - 569$$

$$a = p - 3 = -3 \bmod p$$

$$b = 1219058002426623015618942475834009407551484406473623125220 \\ 8772337825397464478540423418981074322718899427039088997221 \\ 609947354520590448683948135300824418144$$

$$m = 134078079299425970995740249982058461274793658205923933777 \\ 235614437217640300734492323182905858176364980496286125565 \\ 96899500625279906416653993875474742293109$$

$$q = m$$

$$x(P) = 3$$

$$y(P) = 6128567132159368375550676650534153371826708807906353132 \\ 2960495468664645454726071191345292217033369215164051073 \\ 69028606191097747738367571924466694236795556$$

- id-tc26-gost-3410-12-512-paramSetB (512-bit Weierstrass curve)

$$p = 2^{511} + 111$$

$$a = p - 3 = -3 \bmod p$$

$$b = 5472517130514047254760433071281657274171034389553769779747 \\ 9416031257965496939070366962372739527026378575800712932542 \\ 40945079496484373854264998452887027990$$

$$m = 670390396497129854978701249910292306373968291029619668886 \\ 178072186088201503692258541985374819038361506291094774340 \\ 556751014839882071710028285687776119229$$

$$q = m$$

$$x(P) = 2$$

$$y(P) = 1391087797795557258711735874750463328666729297647553860 \\ 7943404349820727624912779633246684899931850893657030334 \\ 94204180568181905548968011075910357787492797$$

- id-tc26-gost-3410-2012-256-paramSetA (256-bit twisted Edwards curve)

$$p = 2^{256} - 617$$

$$a = 87789765485885808793369751294406841171614589925193456909855962166505018127157$$

$$b = 18713751737015403763890503457318596560459867796169830279162511461744901002515$$

$$\varepsilon = 1$$

$$\delta = 2724414110474605931834268501164757645998726878473076809432604223414351675387$$

$$m = 115792089237316195423570985008687907853354241192369013770048613635142121435548$$

$$q = m/4 = 28948022309329048855892746252171976963338560298092253442512153408785530358887$$

$$x(P) = 65987350182584560790308640619586834712105545126269759365406768962453298326056$$

$$y(P) = 22855189202984962870421402504110399293152235382908105741749987405721320435292$$

$$u(P) = 13$$

$$v(P) = 43779144989398987843428779166090436406934195821915183574454224403186176950503$$

- id-tc26-gost-3410-2012-512-paramSetC (512-bit twisted Edwards curve)

$$p = 2^{512} - 569$$

$$a = 1155220774172662408138485443175427045341999095815853654745363047275328427985602901303342173019597772912484970560977054897563749457966985165428182284278739$$

$$b = 9467654314974239364849779893497935997616546680893642377235981868741051215651032446828994750528267630604306101610711521055955290148577159125187794668181473$$

$$\varepsilon = 1$$

$$\delta = 8291368582540391759956325599449696250171737838651241289585997940557058703682611983276933821315724273262456979490783602284025399599007870613061010570769744$$

$$\begin{aligned}
m &= 134078079299425970995740249982058461274793658205923933777 \\
&\quad 235614437217640300734484634732003373968850976753928234033 \\
&\quad 66582058868465127637383742173859717091252 \\
q = m/4 &= 3351951982485649274893506249551461531869841455148098 \\
&\quad 3444308903609304410075183621158683000843492212744188 \\
&\quad 48205850841645514717116281909345935543464929272813 \\
x(P) &= 1188304634094941753595925361103163743848612198935774824 \\
&\quad 7963585015455167053565085942161130870937622596747831459 \\
&\quad 979590245849590330315393322885186213222089032 \\
y(P) &= 1287388791229141876216321917489924902778890935496427956 \\
&\quad 1044704584180798942832869356886395871011373467652642378 \\
&\quad 30933785897290140286858111689735138773336704015 \\
u(P) &= 18 \\
v(P) &= 3697901750350036466195501370680965130892925445528794106 \\
&\quad 5157006855305279130383310151063822343988427973147742640 \\
&\quad 61702328469726236276369898526828850803907133
\end{aligned}$$

C Curve generation

Here we present the seeds used to generate twisted Edwards curves. By H_{512} we denote 512-bit variant of the Streebog hash function ([8]).

- id-tc26-gost-3410-2012-256-paramSetA:

seed:

```

97 B9 57 56 E0 59 D5 75 E6 05 99 7A FE A2 46 D7
95 C6 84 FB 87 74 86 9A 9B 18 53 8F 13 F3 8A C2
14 42 8D 87 1D 12 CA 10 9C 9A 4F E6 C4 DD F3 AF
26 F6 38 6D 2D 51 4C CC B6 D1 AB 82 83 3C 30 D0
79 8E 9B 30 37 FC 86 A3 B6 15 60 2E A1 56 E1 2D
30 CD 63 53 E6 7D F6 42 82 D4 52 A2 03 3C BB 03

```

$H_{512}(\text{seed})$:

```

AF 83 BE 4E C7 BA C6 DD ED 83 3C 44 F1 D0 CF 6A
18 E7 B1 9D EA 5F 4E 69 BA E3 C2 CC B2 F9 9B 07
06 05 F6 B7 C1 83 FA 81 57 8B C3 9C FA D5 18 13
2B 9D F6 28 97 00 9A F7 E5 22 C3 2D 6D C7 BF FB

```

Coefficient δ :

$$\delta = 0x605f6b7c183fa81578bc39cfad518132b9df62897009af7e522c32d6dc7bffb$$

- id-tc26-gost-3410-2012-512-paramSetC:

seed:

```
1F BB 79 69 B9 1B 3E A0 81 17 FB 10 74 BF BF 55
49 DD 66 07 63 F6 A5 AF 09 57 77 5B 66 4C B1 13
CF CB 91 C4 A7 7D 27 98 06 BC F2 4A 56 77 F2 5E
AF FE C6 67 76 70 2E E2 C7 AA 84 16 07 50 DA 1D
D1 50 AE D2 8C 30 26 AC 7E D6 D1 9B 97 AC 2C B5
82 7C 00 03 18 47 13 53 5B FA 65 24 B3 E4 60 83
```

$H_{512}(\text{seed})$:

```
9E 4F 5D 8C 01 7D 8D 9F 13 A5 CF 3C DF 5B FE 4D
AB 40 2D 54 19 8E 31 EB DE 28 A0 62 10 50 43 9C
A6 B3 9E 0A 51 5C 06 B3 04 E2 CE 43 E7 9E 36 9E
91 A0 CF C2 BC 2A 22 B4 CA 30 2D BB 33 EE 75 50
```

Coefficient δ :

$$\delta = 0x9e4f5d8c017d8d9f13a5cf3cdf5bfe4dab402d54198e31ebde28a0621050439ca6b39e0a515c06b304e2ce43e79e369e91a0cfc2bc2a22b4ca302dbb33ee7550$$

D Embedding degrees

- id-GostR3410-2001-CryptoPro-A-ParamSet (256-bit Weierstrass curve)

$$q - 1 = 2 \cdot 3 \cdot 7 \cdot 17 \cdot 37 \cdot 127 \cdot 121493 \cdot 5592900119 \cdot$$

$$\cdot 50791017540450015071456350284045037169936855765408748581$$

$$\text{ord } p = (q - 1)/2$$

- id-GostR3410-2001-CryptoPro-B-ParamSet (256-bit Weierstrass curve)

$$q - 1 = 2 \cdot 47336631894758162101 \cdot$$

$$\cdot 611535319489737880361765400867656717933201264526640746251$$

$$\text{ord } p = (q - 1)/2$$

- id-GostR3410-2001-CryptoPro-C-ParamSet (256-bit Weierstrass curve)

$$q-1 = 2 \cdot 2 \cdot 2 \cdot 3 \cdot 3 \cdot 5 \cdot 5 \cdot 47 \cdot \\ \cdot 207130852417 \cdot 15398703602419036183 \cdot \\ \cdot 260862815097120313262827914162129492639311 \\ \text{ord } p = q-1$$

- id-tc26-gost-3410-12-512-paramSetA (512-bit Weierstrass curve)

$$q-1 = 2 \cdot 2 \cdot 19 \cdot 41 \cdot 257 \cdot 619681 \cdot \\ \cdot 56230387 \cdot 3067250436697090551527 \cdot \\ \cdot 1566530004958535144270949664750568664934770494453380 \\ 8966395462210877730129873810997022305060734539130584 \\ 1500928811 \\ \text{ord } p = q-1$$

- id-tc26-gost-3410-12-512-paramSetB (512-bit Weierstrass curve)

$$q-1 = 2 \cdot 2 \cdot 3 \cdot 2389 \cdot \\ \cdot 2338462384879063258611348018383885539186438855272846 \\ 6195276198973981031167283809771940329803929062421734 \\ 7249467817970123836626162296536217484891787921 \\ \text{ord } p = (q-1)/6$$

- id-tc26-gost-3410-2012-256-paramSetA (256-bit twisted Edwards curve)

$$q-1 = 2 \cdot 1597 \cdot \\ \cdot 9063250566477473029396601832239191284702116561707029 \\ 881813448155537110319 \\ \text{ord } p = q-1$$

- id-tc26-gost-3410-2012-512-paramSetC (512-bit twisted Edwards curve)

$$q-1 = 2 \cdot 2 \cdot 2819 \cdot 7673 \cdot 1683089 \cdot \\ \cdot 8490713317 \cdot 11424320126732366161793 \cdot \\ \cdot 13932905458310476305278334845959 \cdot \\ \cdot 1703155140675906364123009782252316886537706280400523 \\ 7213018459123037212618699 \\ \text{ord } p = q-1$$

E Complex-multiplication discriminants

- id-GostR3410-2001-CryptoPro-A-ParamSet (256-bit Weierstrass curve)
 $|D| = 42466537897363729579564088851177574392504272854603698668$
 8706749501354362291267
 $|D| > 2^{100}$
- id-GostR3410-2001-CryptoPro-B-ParamSet (256-bit Weierstrass curve)
 $|D| = 619$
 $|D| < 2^{100}$
- id-GostR3410-2001-CryptoPro-C-ParamSet (256-bit Weierstrass curve)
 $|D| = 25639560088361227189272331572651984444562993039773368308$
 5773745599364563982115
 $|D| > 2^{100}$
- id-tc26-gost-3410-12-512-paramSetA (512-bit Weierstrass curve)
 $|D| = 44077247648723775457953059908923017372609510022789812783$
 $42883277523972388369674882662761310976156558737585696712$
 $6319214595429328501251794033479398868138547$
 $|D| > 2^{100}$
- id-tc26-gost-3410-12-512-paramSetB (512-bit Weierstrass curve)
 $|D| = 18342779240273929075899703402487093574031114847104527093$
 $89308702891906954803415545067768171221828265052217247434$
 $50519754049331850431306583272275935941515$
 $|D| > 2^{100}$
- id-tc26-gost-3410-2012-256-paramSetA (256-bit twisted Edwards curve)
 $|D| = 11401729866323070445829319093245706896112552656877950460$
 1000849844981660882323
 $|D| > 2^{100}$
- id-tc26-gost-3410-2012-512-paramSetC (512-bit twisted Edwards curve)
 $|D| = 10981588948336641788044089745090748140849244929279553143$
 $44904547220088092760679498341489100555689058926856569099$
 $4331013363862909856927021016359633109168483$
 $|D| > 2^{100}$

F Twist security

Here we present auxiliary data needed to analyse twist security of Russian standardized curves. For each curve we provide the number of points of the non-trivial quadratic twist (m'), the order of largest prime subgroup (q') and order of p in $Z_{q'}^*$. We do not estimate complex multiplication discriminants of the twists as they are equal to the discriminants of the original curves.

- id-GostR3410-2001-CryptoPro-A-ParamSet (256-bit Weierstrass curve)

$$m' = 11579208923731619542357098500868790785346620642278188485 \\ 3537012210747190428317$$

$$q' = m' / (67 \cdot 1197515797 \cdot 112960388171533961) =$$

$$= 12776051685464693103815859587282398207663184917003$$

$$q' < 2^{200}$$

$$q' \neq p$$

$$q' - 1 = 2 \cdot 3 \cdot 157 \cdot 179 \cdot 1553 \cdot$$

$$\cdot 48788927410949333231569954455332516953313$$

$$\text{ord } p = (q' - 1) / 2$$

- id-GostR3410-2001-CryptoPro-B-ParamSet (256-bit Weierstrass curve)

$$m' = 57896044618658097711785492504343953926167851505384737219 \\ 388739511825397579685$$

$$q' = m' / (5 \cdot 113093) =$$

$$= 102386610344863250089369797431041627556379000478163524213 \\ 503469731681709$$

$$q' > 2^{200}$$

$$q' \neq p$$

$$q' - 1 = 2^2 \cdot 3^3 \cdot 6113 \cdot$$

$$\cdot 155083292959241764802045727428251915402480143225674 \\ 979572228386577$$

$$\text{ord } p = (q' - 1) / 3$$

- id-GostR3410-2001-CryptoPro-C-ParamSet (256-bit Weierstrass curve)

$$m' = 70390085352083305199547718019018437841238150612926279907$$

$$246092229058968608639$$

$$q' = m' / (11 \cdot 433 \cdot 1951561 \cdot 9245309 \cdot 104138341) =$$

$$= 7865325388019627074611138979082913271426129070825717$$

$$q' < 2^{200}$$

$$q' \neq p$$

$$q - 1 = 2^2 \cdot 3469 \cdot 259421 \cdot 531572647637 \cdot 426526067872013 \cdot$$

$$\cdot 9636938087179541$$

$$\text{ord } p = (q' - 1) / 2$$
- id-tc26-gost-3410-12-512-paramSetA (512-bit Weierstrass curve)

$$m' = 13407807929942597099574024998205846127479365820592393377$$

$$72356144372176403007364472128545801051617035733043510381$$

$$6375202206882485717476485898991823269873947$$

$$q' = m' / (23 \cdot 61 \cdot 4447 \cdot 142799 \cdot 205285055011140558581260164490369) =$$

$$= 733078495620179237339561502178730969216876708146598508325$$

$$97789030697445789538267547684057083080453035490394257$$

$$q' > 2^{200}$$

$$q' \neq p$$

$$q' - 1 = 2^4 \cdot 3 \cdot 328302760325728843 \cdot$$

$$\cdot 465194646660935860197301403277751550961407981648077$$

$$1074075608475236947578881524343127730729$$

$$\text{ord } p = (q' - 1) / 16$$
- id-tc26-gost-3410-12-512-paramSetB (512-bit Weierstrass curve)

$$m' = 67039039649712985497870124991029230637396829102961966888$$

$$61780721860882015036624391382020549976519812627120910443$$

$$080483343605483991229469663576771229965091$$

$$q' = m' / (107 \cdot 457 \cdot 11279 \cdot 288867653 \cdot 8115660434350138997 \cdot$$

$$7939453682920433906546174291390263043 \cdot$$

$$48185178322054020433660035202680240077) =$$

$$= 13552840881400009935629935782603930805279121$$

$$q' < 2^{200}$$

$$q' \neq p$$

$$q' - 1 = 2^4 \cdot 5 \cdot 347 \cdot 1021 \cdot 22063110703 \cdot 21672968087599576942739749$$

$$\text{ord } p = (q' - 1) / 10$$

- id-tc26-gost-3410-2012-256-paramSetA (256-bit twisted Edwards curve)

$$m' = 11579208923731619542357098500868790785318572813891211430 \\ 8866554380684137843092$$

$$q' = m'/4$$

$$q' > 2^{200}$$

$$q' \neq p$$

$$q' - 1 = 2^2 \cdot 3^2 \cdot 97223288653 \cdot \\ \cdot 827077279482546575190449379565233863604927641138408 \\ 1918269885909$$

$$\text{ord } p = (q' - 1)/1$$

- id-tc26-gost-3410-2012-512-paramSetC (512-bit twisted Edwards curve)

$$m' = 13407807929942597099574024998205846127479365820592393377 \\ 72356144372176403007364549013054825893692175770467089296 \\ 9605519648639300496255756150693438295075804$$

$$q' = m'/4$$

$$q' > 2^{200}$$

$$q' \neq p$$

$$q' - 1 = 2 \cdot 3 \cdot 5^2 \cdot 48847 \cdot 560977 \cdot 19068983 \cdot \\ \cdot 277596723050853856375912281308912051313269373218800873 \cdot \\ \cdot 1540570974546945249583856835063039560608301462825796138 \\ 72531998189556342453721633$$

$$\text{ord } p = (q' - 1)/10$$