Open Access Library Journal
2019, Volume 6, e5423
ISSN Online: 2333-9721
ISSN Print: 2333-9705

ECDSA Private Keys Study of Security

Panagiotis V. Kontogiannis, T. Varvarigou

National Technical University of Athens, Athens, Greece
Email: panagiotiskg@gmail.com

How to cite this paper: Kontogiannis, P.V. and Varvarigou, T. (2019) ECDSA Private Keys Study of Security. *Open Access Library Journal*, **6**: e5423.
https://doi.org/10.4236/oalib.1105423

Received: April 24, 2019
Accepted: June 2, 2019
Published: June 5, 2019

Abstract

Cryptocurrencies are a mean of executing online transactions. They use a variety of cryptographic techniques to secure and verify these transactions, which are functionally supported by the Blockchain platform. Blockchain is a continuously growing, distributed ledger of files that contains all transactions between users of cryptocurrencies in a verifiable and permanent manner. It consists of blocks that are connected and secured cryptographically. Cryptocurrencies use algorithms to produce pairs of public and private keys. These pairs, cryptographically merged with a message between the participants, are the building blocks of the relevant transactions. Bitcoin uses the ECDSA algorithm to produce the above-mentioned keys. The purpose of our work is to present some useful motifs for the domain parameters of base point (P) and the order (n) of the subgroup produced by it, while choosing the elliptic curve and the Galois field on which we formulate the algorithm, in order to obtain safer private keys. The results of the research are experimental due to the limited infrastructure, but explanatory for the purpose of our work. The resulting conclusions highlight the value of the proper selection of the structural parameters of these algorithms and possible alternatives to the curve, field and domain parameters that can be used.

Subject Areas

Mathematical Economics

Keywords

ECDSA, ECDLP, Private Keys, Blockchain

1. Introduction

The most important issue a cryptocurrency such as Bitcoin has to offer to its users is the ability to handle a large volume of transactions in a short period of time with security and verifiability.

Often the developer's task with respect to the key generation algorithm is the right choice of the algorithm's domain parameters that will provide a sufficient combination of the above properties. For the Bitcoin cryptocurrency, the elliptic curve secp256k1 defined by the standards for an efficient cryptographic group (SECG) is the one used by the ECDSA algorithm, as discussed further in [1]. Although safer curves have been proposed, speed, volume of work, but above all infrastructure, force developers to compromise looking for other security controls.

For the purpose of the research, we focused exclusively on the security of the algorithm, which we applied for specific experimental data. The resulting conclusions concern the selection of appropriate domain parameters on the specific safe elliptical curves from the NSA survey conducted by the United States of America, best explained by [2], on Galois field (F_p).

2. Theoretical Background

2.1. Hash Function

In Bitcoin, on the Blockchain platform, the user's digital addresses are the result of fragmentation of a public key part Q produced by the ECDSA algorithm [3]. Hash functions have four very powerful properties, thoroughly examined by [4], that contribute to the security of transactions between users. Specifically:

- **Collision resistance:** Collision resistance: Concept in which a hash function H is resistant to collisions of input values if it is infeasible to lead to a common output value from different input values. Otherwise, for x, y with $x \neq y$ we arrive at $H(x) = H(y)$.
- **Preimage resistance:** For a predetermined output value *y*, it is infeasible to find the input value *x* that has it as output, *i.e.* it is difficult to find any preselected input value *x*, so that $H(x) = y$.
- **Second preimage resistance:** It is not possible for a different input value x', with $x' \neq x$, to arrive at a valid $H(x) = H(x')$.
- **Hiding:** A hash function has the ability to be hidden if for a hidden value *i*, which is selected from a distribution with a high min-entropy, for a given value $F(i \| z)$ it is impossible to find the value *z*.

Note that all addresses consist of letters and numbers, which are the output values of the Bitcoin-based hash function, with input values the corresponding parts of the public keys *Q*. The amount, the addresses of the contracted users and the dates of the transactions, as shown in **Figure 1**, are few of the elements contained on the Blockchain platform.

2.2. Elliptic Curves

Elliptic curves are the first mathematical tool used to create the ECDSA algorithm, as discussed in [5]. Selecting a suitable curve that will simultaneously support the collateral needs of the platform is a building block.

Elliptic curve: A curve whose shape is given by the equation:

$$y^2 = x^3 + Ax + B, \text{ with } A, B \in \mathbb{Z}$$

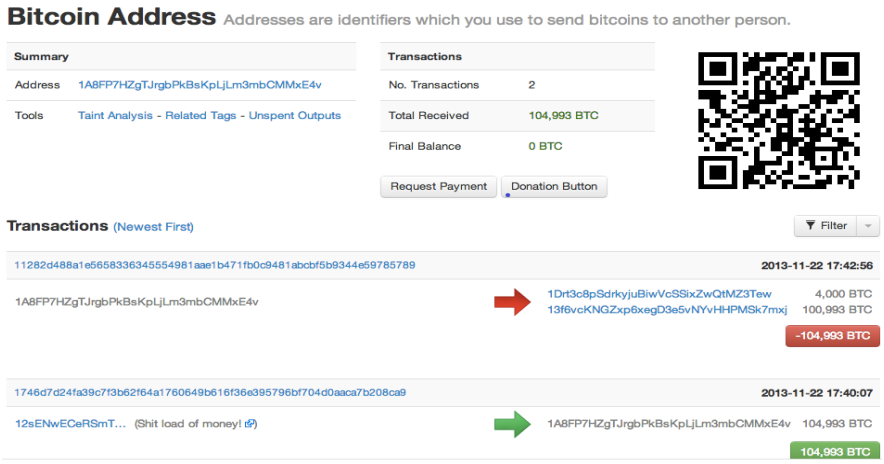


Figure 1. An example of bitcoin address and transactions history of a typical user.

where the basic condition is the discriminant $\Delta = 4 \cdot A^3 + 27 \cdot B^2 \neq 0$

Its points are given by the set:

$E = \{(x, y) : y^2 = x^3 + Ax + B\} \cup \{O\}$, where O is the point at infinity.

Algebraic properties: For two points $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ on an elliptic curve of the form $E : y^2 = x^3 + Ax + B$, the following properties, further explained by [6]:

Apply

- If $P_1 \neq P_2$ with $x_1 = x_2$, then $P_1 + P_2 = 0$.
- If $P_1 = P_2$ and $y_1 = 0$, then $P_1 + P_2 = 2P_1 = 0$.
- If $P_1 \neq P_2$ and $x_1 \neq x_2$, then $\begin{cases} \lambda = \dfrac{y_2 - y_1}{x_2 - x_1} \\ \beta = -\lambda x_1 + y_1 = \dfrac{y_1 x_2 - y_2 x_1}{x_2 - x_1} \end{cases} \Rightarrow$ Point Addition.
- If $P_1 = P_2$ with $y_1 \neq 0$, then $\begin{cases} \lambda = \dfrac{3x_1^2 + A}{2y_1} \\ \beta = -\lambda x_1 + y_1 = \dfrac{-x^3 + Ax + 2B}{2y} \end{cases} \Rightarrow$ Point Doubling.

It is true from the foregoing that in general:

$$P_1 + P_2 = \left(\lambda^2 - x_1 - x_2, -\lambda^3 + \lambda(x_1 + x_2) - \beta\right)$$

Schematic examples of the above are illustrated in **Figure 2**.

2.3. Galois Fields

The second mathematical theory that supports ECDSA are the finite fields, commonly known as Galois fields, as mentioned in [7].

- **Field:** A set of numbers defining the operations of addition, multiplication, and consequently subtraction and division, which satisfy all the essential properties of these operations.

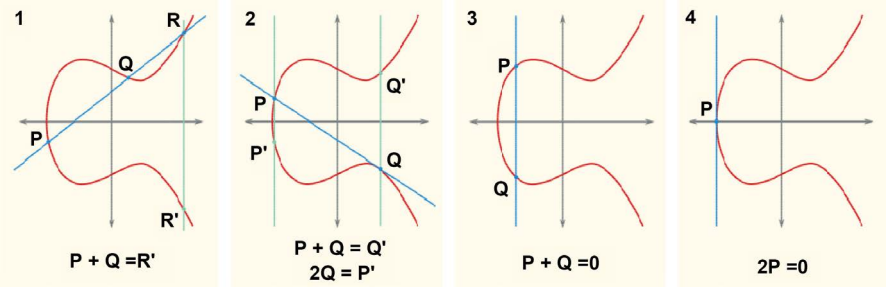


Figure 2. Examples of point addition and point doubling on $y^2 = x^3 - x - 1$ elliptic curve.

- **Galois fields** are all fields of form $F_p = \{0,1,2,\cdots,p-1\}$ with p prime number. These fields have a finite number of elements. The results of all operations are divided by modulo p and are all prime field values.

For example, F_{29} is $\{0,1,2,3,\cdots,28\}$. Examples of numerical operations are:

- Addition: 17 + 20 = 8, because 37 mod 29 = 8
- Subtraction: 17 − 20 = 26, due to −3 mode 29 = 26
- Multiplication: 17*20 = 21, because 340 mod 29 = 21
- Division: 17 − 1 = 12, due to 17*12 mod 29 = 1

Some very basic properties of Galois fields useful for conducting research are:

1) Subfield-Field Expansion: For field *F*, we call *K* a subfield of *F*, when this is a field provided with the same operations as *F*, all elements of which belong to the original *F*. By analogy, *F* is an extension of subfield *K*.

2) Galois Field Base: Algebraically a finite field F_{p^n} can be a vector space of the F_p subfield, where the vectors will be the elements of the first and gradual sizes the elements of the second (depending on the operation we perform). For $B = \{b_1, b_2, \cdots, b_n\}$ a base and $a \in F_{p^n}$, a subfield element α can be unique as $a = a_1 \cdot b_1 + a_2 \cdot b_2 + \cdots + a_n \cdot b_n$ with $(a_1, a_2, \cdots, a_n)$ elements of the F_p field.

3) Existence and Uniqueness: For every prime number p and positive integer n there is a finite field with p^n elements, *i.e.* F_{p^n}. Any other finite field with the same number of elements is isomorphic to the previous one.

4) Subfield Criterion: For F_q a finite field with $q = p^n$ elements, we have that any subfield of F_{q^m} has an order p^m, where *m* is a positive divisor of *n*. Conversely, for *m* positive divisor of *n*, there is exactly one subfield F_{q^m} of F_q with p^m elements.

Note: The Galois fields are widely used in modern cryptography. Specifically in software applications, in the development of processors due to the field arithmetic and the creation of fast desktop multipliers. These are only a few of the improvements they have been brought by Galois fields.

2.4. Weierstrass Equation, Isoforms and Hasse Theorem

The ECDSA algorithm is constructed from a specific elliptic curve (secp256k1) on a Galois field. The merge of the two previous theories is the Weierstrass equation, best explained by [8], which stated as follows:

Weierstrass equation: For an arbitrary (finite) field F we define the Weierstrass equation of the elliptic curve E on the field, *i.e. E/F*, which is of the form:

$$E: y^2 + a_1 \cdot xy + a_3 \cdot y = x^3 + a_2 \cdot x^2 + a_4 \cdot x + a_6 \text{ with } (x, y) \in F.$$

where:

For $\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_6 \in F$ we have $\Delta \neq 0$, where Δ is the discriminant of *E*.

In general, the above equation must be transformed into more friendly forms for use by any key pair generation algorithm. The following isomorphism is appropriate.

Weierstrass isomorphism (mentioned in [5]): For two distinct elliptic curves of Weierstrass form, E_1, E_2, on a finite field F with formulas:

$$E_1: y^2 + a_1 \cdot xy + a_3 \cdot y = x^3 + a_2 \cdot x^2 + a_4 \cdot x + a_6$$
$$E_2: y^2 + a_1' \cdot xy + a_3' \cdot y = x^3 + a_2' \cdot x^2 + a_4' \cdot x + a_6'$$

The curves are called isomorphic on the field if there are $u, r, s, t \in F$, with $u \neq 0$, so for the transformation:

$$(x, y) \to \left(u^2 \cdot x + r, u^3 \cdot y + u^2 \cdot s \cdot x + t\right)$$

Starting with E_1, we end up in E_2.

Basically, we use isomorphism:

$$(x, y) \to \left(\frac{x - 3a_1^2 - 12a_2}{36}, \frac{y - 3a_1 \cdot x}{216} - \frac{a_1^3 + 4a_1 \cdot a_2 - 12a_3}{24}\right)$$

Which leads us to the known form of short Weierstrass elliptic curves over Galois field F_p, p prime, with the formula:

$$y^2 = x^3 + a \cdot x + b \text{ where } 4 \cdot a^3 + 27 \cdot b^2 \pmod p \neq 0.$$

Riemann's hypothesis for elliptic curves (mentioned in [12]): For E an elliptic curve with points on a finite field F_q, with $\#E\left(F_{q^n}\right)$ the number of these points applies:

$$\left|\#E\left(F_{q^n}\right) - 1 - q^n\right| \leq 2 \cdot q^{\frac{n}{2}}, \forall n \geq 1.$$

By selecting $n = 1$ we have $\left|\#E\left(F_q\right) - 1 - q\right| \leq 2 \cdot \sqrt{q}, \forall n \geq 1$ -Hasse theorem [9]

Note: The order of the field, *i.e.* the number of points of the elliptic curve on the Galois field is to be denoted by *N*. According to Hasse theorem, an easy first estimate of the order of the curve is calculated.

Using similar isomorphisms, we result from the generalized Weierstrass equation in two other very useful elliptical curve cryptographic forms (ECC).

Specifically:

- **Montgomery equation**: concerning elliptic curves on Galois F_p fields of the form:

$$B \cdot y^2 = x^3 + A \cdot x^2 + x, \text{ where } B \cdot \left(A^2 - 4\right) \pmod p \neq 0$$

- **Edwards equation:** concerning elliptic curves over Galois F_p fields of the form:

$$x^2 + y^2 = 1 + d \cdot x^2 \cdot y^2, \text{ where } d \cdot (1-d) (\text{mod } p) \neq 0$$

Note: It is possible, through proper transformation, to determine one form of elliptic curve equation from another. Specifically:

- For a Montgomery elliptic curve through transformation, $(x, y) \to \left(B \cdot u - \frac{A}{3}, B \cdot v \right)$ we pass into a short Weierstrass form with the equation:

$$v^2 = u^3 + a \cdot u + b, \text{ where } a = \frac{3 - A^2}{3 \cdot B^2} \text{ and } b = \frac{2 \cdot A^3 - 9 \cdot A}{27 \cdot B^3}$$

- For an Edward elliptic curve through transformation $(x, y) \to (u/v, (u-1)/(u+1))$ we pass into Montgomery with the following equation:

$$B \cdot v^2 = u^3 + A \cdot u^2 + u, \text{ where } A = \frac{2 \cdot (1+d)}{1-d} \text{ and } B = \frac{4}{1-d}$$

Note: Generally, all elliptical equations on Galois $F_p, p > 3$ fields can be in the form of a short Weierstrass equation.

Rational points (explained in detail in [10]): All points (x, y), with $x, y \in F_p$ of the elliptic curves on the respective Galois F_p fields that satisfy their curve equations. If we have short Weierstrass or Montgomery equations, there is the *O*-point at infinity, while for Edward equations there is not.

2.5. Elliptic Curves Discrete Logarithmic Problem (ECDLP)

Elliptic Curve Cryptography (ECC) supports its safety in the difficulty of solving the discrete logarithmic elliptic curve problem (ECDLP). This means that the implementation of the ECDSA used to produce a key pair (*d*, *Q*) should support its functions in a robust pair of elliptic curve and Galois field, as explained in [11].

ECDLP (Short Weierstrass): We consider an elliptic curve *E* defined on a finite field F_p, with characteristic *p i.e.*:

$$E: y^2 \stackrel{?}{=} x^3 + A \cdot x + B, \text{ where } A, B \in F_p$$
$$\text{with restriction } 4 \cdot A^3 + 27 \cdot B^2 (\text{mod } p) \neq 0$$

For two points of $E(F_p)$, let *P*, *Q* we look for the integer $x, x \in \mathbb{N}$ for which:

$$Q = x \cdot P$$

Complexity of Pohlig-Hellman—ECDSA (resilience, thoroughly explained by [12]): The Pohlig-Hellman algorithm is used for calculating discrete logarithms with input a set of points of order *n* and having complexity $O(n)$.

By parameterizing $n = \prod p_i^{e_i}$ we degrade Pohlig-Hellman into a Baby-giant step algorithm, which results in the complexity of the algorithm to increase to $O\left(\sum_i e_i \left(\log n + \sqrt{p_i} \right) \right)$

Let an elliptic curve *E* be defined on a Galois field F_p, whose order is the

number $\#E(F_p) = N$. Based on the above, order N can be presented as the product of prime numbers, *i.e.* $N = p_1 p_2 \cdots p_n$ which are the orders of subgroups produced by base points P. This makes it more difficult to solve ECDLP, which implies that the ECDSA algorithm is resilient to model-based attacks.

We prefer orders $n_i \equiv p_i$ to be large prime numbers.

Domain parameters of ECDSA

The domain parameters are the composite elements on which ECDSA is designed to produce the requested keys for trading. Although the users of the platform know their values, it is common for security reasons to list the hash function outputs with input values, the values of the domain parameters. Bitcoin's ECDSA uses the elliptic curve secp256k1 with domain parameters $T = (p, a, b, G, n, h)$ where:

- $p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$, is the size of the Galois field F_p.
- The coefficients $a = 0$, $b = 7$ of the above curve.
- The generation point G = 0279BE667E F9DCBBAC55A06295CE870B07029BFCDB2DCE28D959F2815B 16F81798, from which we produce the subset of the elliptic curve points in the field.
- n = FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEBAAEDCE6AF48A03BBFD25E8C D0364141, the order of the base point G.
- The cofactor $h = 1$.

The base point is the one from which the algorithm generates the subgroup of elliptic curve points applied to the Galois field. We prefer the order of the subgroup produced for security reasons to be a large prime number. If N is the order of the curve, then we look for the largest prime number n that divides the order of the curve. **Simply selecting a base point with a non-prime subgroup n order makes the algorithm vulnerable to attacks because it is not adequately supported by the ECDLP.**

The process of finding the order n is fulfilled by the procedure of elliptic curve scalar point multiplication, *i.e.* by solving the equation:

$n \cdot P \pmod p = 0$, where P is a point of $E(F_p)$.

Cofactor $h = \dfrac{N = \#E(F_p)}{n}$, with $h \in \mathbb{N}$ is used to calculate the base point G by solving the equation:

$n \cdot (h \cdot P) = n \cdot G = 0$ with $G = h \cdot P$.

2.6. Model-Based Attacks versus ECDSA

In the present work, we will unleash three model-based methods of attack, such as Brute force, Baby-giant step and Pollard's rho. The methods will violate the ECDSA algorithm to steal the private key d of the transaction, as explained by [13]. This creates serious problems for the victim of the violation and for the Blockchain platform, whose prestige is irreversibly impaired. The purpose of the research is to demonstrate at an experimental level that any change in the domain parameters of the elliptic curve and the Galois field of the algorithm, as de-

fined by the NSA survey, is dangerous to the security of the algorithm and by extension to the users' private keys, as discussed in [14].

Brute force: Calculates the products $x_1 \cdot P, x_2 \cdot P, x_3 \cdot P, \cdots$ for random values $x_1, x_2, x_3, \cdots$ until $x \cdot P = Q$ is verified for some value. The length of time until the above verification, is $O(p)$. Clearly the most time-consuming method requiring a powerful computing system for comparable results over the next. The symbolism O is Landau's big O notation.

Baby giant step: First, we transform the equality $Q = x \cdot P$ that we verify as: $Q - a \cdot m \cdot P = b \cdot P$

Reminder: Generally, any integer $x, x \in \mathbb{Z}$ can be written as a product of three arbitrary integers $a, m, b \in \mathbb{Z}$, so that $x = a \cdot m + b$.

So, in the next step two vector lists of the starting points P, Q of $E(F_q)$ are created with the previous coefficients $(x_1, x_2, x_3, \cdots)$ *i.e.*:

$$\text{Vector 1}: x_1 \cdot P, x_2 \cdot P, x_3 \cdot P, \cdots$$
$$\text{Vector 2}: Q - x_1 \cdot P, Q - x_2 \cdot P, Q - x_2 \cdot P, \cdots$$

The process ends when a collision of the following form occurs:

$$x_i \cdot P = Q - x_j \cdot P, \text{ where } i, j = 1, 2, 3, \cdots$$

The expected execution time is $O(\sqrt{q})$ and is clearly less than the equivalent of the exhaustive method (Brute force).

(Pollard's Attack): with this method, we search for discrete pairs (a, A) and (b, B) of modulo n integers to verify equality:

$$a \cdot P + b \cdot Q = A \cdot P + B \cdot Q, \text{ with } a, b, A, B \in \mathbb{Z}$$

Specifically we calculate the value

$$x = (a - A) \cdot (B - b)^{-1} \bmod n$$

Briefly

1) For a given point $X \in \langle P \rangle$ and integers (c, d) with $X = c \cdot P + d \cdot Q$, a random iteration function $f : \langle P \rangle \to \langle P \rangle$ is defined, which calculates $X = f(X)$ and $\overline{c}, \overline{d} \in [0, n-1]$ with $\overline{X} = \overline{c} \cdot P + \overline{d} \cdot Q$.

2) Subsequently we define a random partition of $\langle P \rangle$, the set $\{S_1, S_2, \cdots, S_L\}$ in order the L sets to have an even approximate size. Typical values of L are 16 (2^4) and 32 (2^5).

3) For $X = c \cdot P + d \cdot Q$ we have $f(X) = \overline{X} = \overline{c} \cdot P + \overline{d} \cdot Q$ where $\overline{c} = c + a_j \bmod n$ and $\overline{d} = d + b_j \bmod n$.

Finally, each point $X_0 \in \langle P \rangle$ defines a sequence $\{X_i\}_{i \geq 0}$ of points, where $X_i = f(X_{i-1})$, $i \geq 1$. Because the $\langle P \rangle$ set is finite, we will definitely come to a collision. This means that there will be a small index w for which $X_w = X_{w+s}$, $s \geq 1$. In conclusion we have $X_i = X_{i-s}, \forall i \geq w + s$.

W is called "tail length", while s is the "length of the circle". A collision is expected after $\sqrt{\pi \cdot n/2}$ values, while the tail and cycle lengths are respectively $t \sim \sqrt{\pi \cdot n/8}$ and $s \sim \sqrt{\pi \cdot n/8}$. The algorithm used to find this collision is the Floyd Cycle Finding algorithm. We calculate point pairs (X_i, X_{2i}) for

$i = 1, 2, 3, \cdots$ and terminate the process when $X_i = X_{2i}$. There is a collision when for two points $X_i, X_j \Rightarrow X_i = X_j$ for $i \neq j$. The expected number of pairs to be compared is $k \in [w, w+s]$ and for random iterated function f is $1.0308 \cdot \sqrt{n}$.

Advantages of ECDSA:

1) With respect to earlier cryptographic tools such as RSA, DSA, elliptic curve algorithms offer greater security for a certain key size. This is also observed for small key sizes, which by definition are much more vulnerable than larger ones.

2) Time and memory space required to produce and distribute messages in the case of ECDSA is significantly smaller than in previous tools. In this way, the platform becomes accessible to more users and researchers.

3) From an economic point of view, systems based on elliptic curve algorithms are more cost-efficient than older algorithms in storage, cooling and energy.

Disadvantages of ECDSA:

1) Cryptographic tools are primarily free to access by all kinds of users, allowing the creation of tools for violating any kind of platform.

2) Encrypted information, authentic and digitally signed, can be difficult to access even for a legitimate user at critical decision time, especially when the platform is tampered with.

3) Cryptography by definition does not protect against the vulnerabilities and threats that result from bad design of systems, protocols and processes.

2.7. Experimental Analysis

At this point, we present all the experimental results that emerged from the research. Specifically, we have implemented the ECDSA key pair generation algorithm for two Galois F_{50101} and F_{100153} fields with appropriate base points of our choice. We then launched three model-based attacks such as Brute force, Baby-giant step and Pollard rho to steal private keys used in transactions. By counting the time required to carry out the wrenching of the private key, we have arrived at significant practical conclusions.

Before the tables and diagrams are presented, we must mention the modifications to the algorithm and the assumptions made for the better conduct of the survey. Analytically:

- We modified the GitHub mini_ecdsa algorithm, as referenced in the relevant bibliography, designed to calculate the first 10 - 12 base points of order only prime number. Specifically, by finding one of the points of the elliptic curve on the Galois field, we examined the order of the subgroup we create when that point is used as a base point. If order n is prime number then we would consider it for study, otherwise we are going to be checked at the next points. This double check to find points increases waiting time in many hours per elliptic curve, but it serves the rest of the research to a large extent.
- The Galois fields we studied are objectively very small than those used in key-generation algorithms. The reason is the limited infrastructure. It is no-

ticeable that the wait for double check increases exponentially, when the size of the Galois field increases. The results, however, are similar and expected.

- From the above points, we have chosen as base points those for which the largest possible subgroup n order has been obtained. It is known that the larger the n order is, the greater the security in minor attacks, that we are not studying in this research. In addition, it was observed that for larger n order larger private keys are emerging, which is very important for research.
- Despite the elasticity of the base point selection algorithm, we remain "loyal" to the ECDLP problem on which the ECDSA algorithm is based, *i.e.* the base point selection that gives as an order n prime number.
- In applying the algorithm, we additionally chose elliptic curves with Edward equations, which we transformed into short Weierstrass. The reason is that the mini_ecdsa algorithm, best explained in [15], has been chosen to study elliptical curves of the form:

  $y^2 = x^3 + a \cdot x^2 + b \cdot x + c$ defined on F_p, p prime number.

The above equation is an intermediate form of Short Weierstrass and Montgomery.

In particular, the equations of the above form are:

Having properly transformed the elliptic curves, which are displayed in **Table 1**, we pass on the level of the analysis of the results, as shown in **Table 2**.

Table 1. All the elliptic curves and their equations used for the research purposes in Weierstrass form.

Curve	Equation
secp256k1	$y^2 = x^3 + x + 7$
Curve25519	$y^2 = x^3 + 486662x^2 + x$
Curve383187	$y^2 = x^3 + 229969x^2 + x$
M-221	$y^2 = x^3 + 117050x^2 + x$
M-551	$y^2 = x^3 + 530438x^2 + x$
Anomalous	$y^2 = x^3 + 1.535e^{62} \cdot x + 7.444e^{60}$
BN(2,254)	$y^2 = x^3 + 2$
BrainpollP256t1	$y^2 = x^3 - 3x + 4.621e^{76}$
Curve1174	$y^2 = x^3 - 28489x + 1926947$
E-222	$y^2 = x^3 + 13333333333x$
E-382	$y^2 = x^3 - 94211737x + 352268124782$
E-521	$y^2 = x^3 + 14833242554x + 6219980097646$
Ed448-Goldilocks	$y^2 = x^3 - 318383x + 692424$
M-383	$y^2 = x^3 + 2065150x^2 + x$
NIST P-224	$y^2 = x^3 - 3x + 1.896e^{67}$

Table 2. Applying all model based methods in all above-mentioned elliptic curves on F_{50101} Galois field.

Galois Field F_{50101}							
					Method (Time in sec)		
Curve	**Base Point *P***	**Order *n***	**Private Key *d***	**Point *P***	**Brute Force**	**Baby-Giant Step**	**Pollard's rho**
secp256k1	(99, 11,436)	137	52	(35,925, 39,101)	0.008	0.001	0.002
Curve25519	(1288, 23,396)	181	156	(48,651, 17,578)	0.225	0.004	0.005
Curve383187	(88, 17,366)	1049	988	(12,662, 41,418)	0.023	0.01	0.006
M-221	(15, 14,998)	12,547	11123	(2847, 24,111)	4.239	0.068	0.032
M-551	(4, 3728)	3119	1440	(33,896, 38,549)	0.289	0.011	0.008
Anomalous	(378, 6853)	211	104	(49,646, 42,661)	0.014	0.002	0.004
BN(2,254)	(61, 4601)	1021	319	(16,955, 40,535)	0.237	0.006	0.004
BrainpollP256t1	(16, 25,549)	25,057	23719	(5992, 11700)	9.736	0.085	0.022
Curve1174	(26, 44,099)	4549	4345	(929, 5479)	1.864	0.04	0.026
E-222	(184, 13,447)	701	552	(45,080, 196)	0.096	0.006	0.005
E-382	(16, 2607)	4999	4913	(20,242, 13,648)	1.25	0.027	0.009
E-521	(280, 28,472)	991	831	(18,354, 4898)	0.158	0.008	0.005
Ed448-Goldilocks	(694, 9865)	499	99	(911, 12,900)	0.012	0.002	0.006
M-383	(76, 7159)	349	252	(46,426, 38,837)	0.036	0.004	0.004
NIST P-224	(78, 4999)	1931	1703	(10,585, 26,862)	0.356	0.012	0.004

Remarks:

1) For all curves defined on the field, the private key is stolen in a very short time. Fact expected due to the small size of the Galois field and the limited base point options.

2) Some curves like Curve1174 and Ed448-Goldilocks show much longer resistance against the Brute force method. However, all curves succumb almost immediately to Baby-giant step and Pollard's rho attacks. The reason is their search model and quick verification of the corresponding ECDLP solution.

3) The higher the n order of the subgroup of the base point P, the larger the d private key created. For the public key Q this is not observed. In all cases where a large private d key is observed, the times for all the model-based methods are comparatively much longer than for other elliptic curves for which the subgroup order n gives small-sized private keys.

4) The theory of the complexity of the Pohlig-Hellman algorithm is verified as to the difficulty of solving the ECDLP problem for large orders n of subgroups, and therefore the ECDSA algorithm's durability.

Correspondingly, for the Galois Field F_{100153} the results of the survey are displayed in Table 3.

Corresponding observations with field results F_{50101} also arise here. By comparing the tables, we also observe:

1) For elliptic curves such as secp256k1 and Curve1174 when doubling the Galois field there is a longer tolerance for the Brute force method only. The other smarter methods used to steal the private key consume almost equal times.

Table 3. Applying all model based methods in the above-mentioned curves on F50101 Galois field.

Galois Field F_{100153}							
					Method (Time in sec)		
Curve	Base Point P	Order n	Private Key d	Public Key Q	Brute Force	Baby-Giant Step	Pollard's rho
secp256k1	(149, 62,878)	1933	1360	(4678, 2942)	0.446	0.013	0.009
Curve25519	(2318, 11,352)	179	102	(69,082, 18,748)	0.022	0.003	0.003
Curve383187	(1258, 2668)	733	701	(79,043, 21,484)	0.17	0.008	0.006
M-221	(1843, 70,129)	107	100	(64,009, 10,304)	0.013	0.002	0.003
M-551	(2115, 86,231)	271	101	(44,889, 47,472)	0.021	0.003	0.005
Anomalous	(2995, 96,951)	43	40	(61381, 17502)	0.004	0.001	0.001
BN(2, 254)	(240, 321)	1231	994	(57435, 36387)	0.294	0.008	0.004
BrainpollP256t1	(2368, 5807)	83	61	(46,066, 25,833)	0.004	0.001	0.001
Curve1174	(8, 93683)	99551	96542	(15,179, 6830)	66.859	0.231	0.086
E-222	(3, 98740)	1229	781	(77,951, 68,281)	0.274	0.013	0.01
E-382	(785, 3790)	148	148	(65,000, 4657)	0.029	0.004	0.006
E-521	(858, 16,707)	893	760	(82,041, 45,996)	0.186	0.009	0.005
Ed448-Goldilocks	(6, 85,750)	50021	35057	(77,073, 78,269)	12.331	0.105	0.031
M-383	(22, 72,669)	6229	6161	(60,238, 81,175)	1.678	0.033	0.009
NIST P-224	(3079, 50,355)	229	174	(25,906, 3547)	0.024	0.003	0.002

2) On the contrary, for curves such as BrainpollP256t1 and M-221, doubling the field made them more vulnerable even to the Brute force method. For the other two methods, we observe as expected very short times.

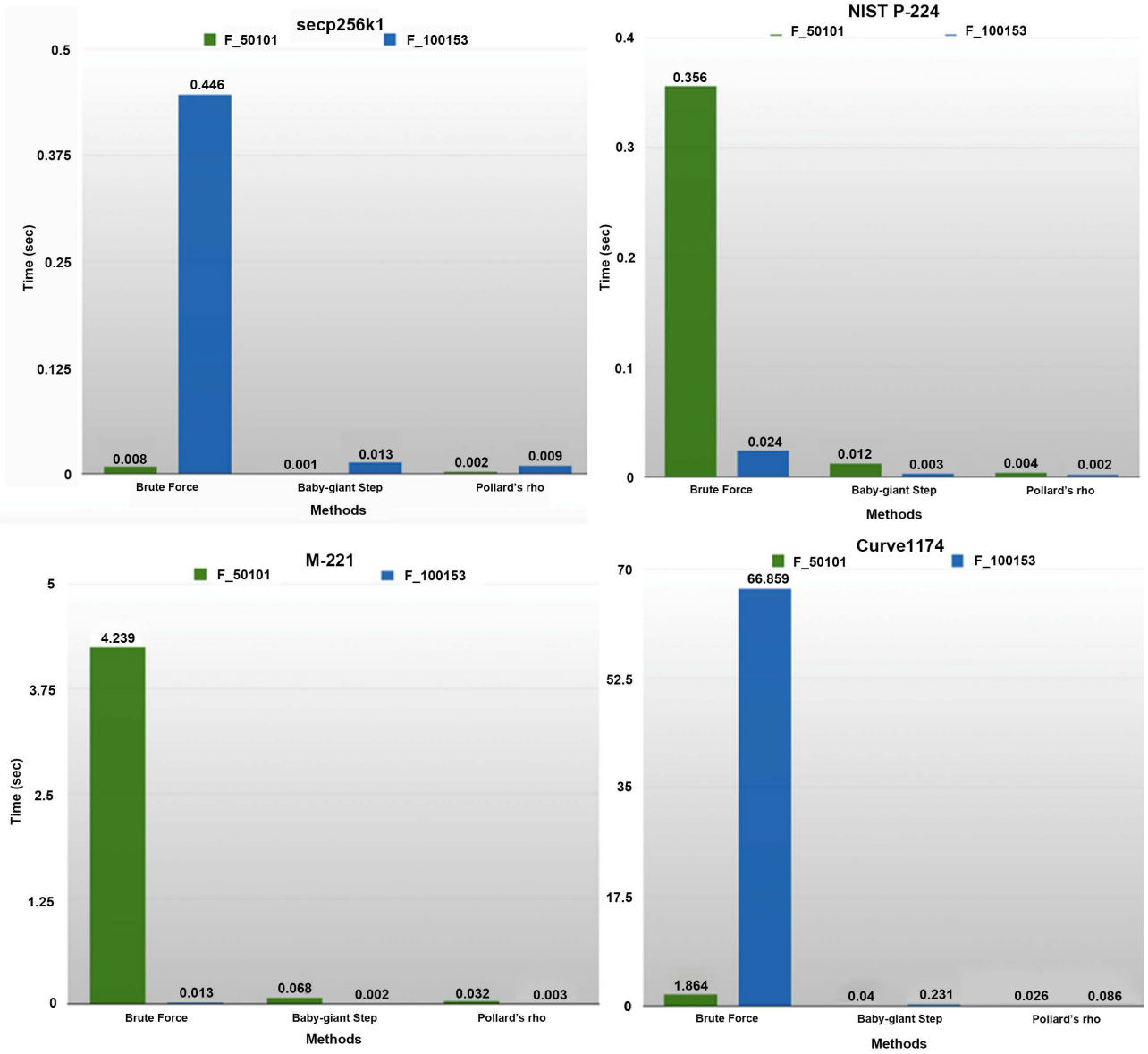
At the diagram level, the times for all the model-based methods for the elliptical curves Curve1174, M-221, NIST P-224, secp256k1 are displayed in Graph 1.

Likewise the diagram level and the time tolerance for all the model-based methods for the elliptical curves Brainpoll256t1 and Anomalous are displayed in Graph 2.

Important Note: You may wonder why search for candidate base points P takes many hours, while model-based attacks squeeze the private key d in much shorter times. It is sufficient to consider that the equations of the elliptic curves we chose are very large, which delays the finding of $E\left(F_p\right)$ points starting from a absolutely random point search in combination with the integer division defined in the Galois field. In addition, the order n check performed at each such point further delays the whole process. On the contrary, solving the $Q = d \cdot P$ equation of the ECDLP for calculating d takes place much faster, since the public keys Q and the base point P are known to all users of any platform.

Study of d private keys

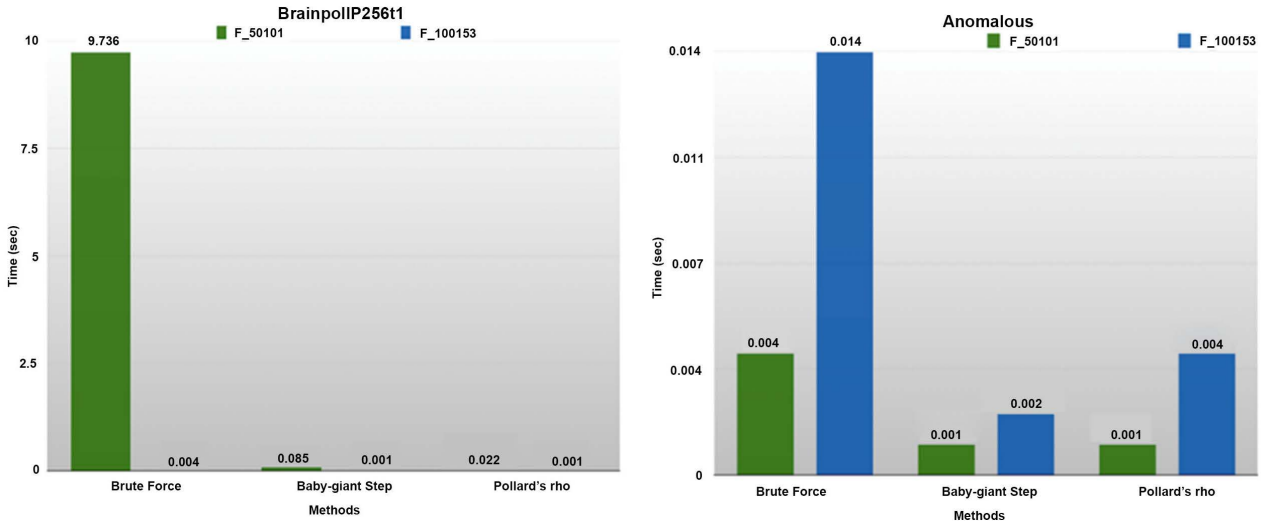
The private key d is clearly the most important building block of a transaction taking place at Bitcoin. For this reason, it is the target of potential hackers who,



Graph 1. Tolerance of secp256k1, NIST P-224, M-221 and Curve1174 against all model based attacks on both Galois fields.

use one of the above-mentioned model-based methods of attack or their variants, seek to steal it within a reasonable period of time. **Ownership of a user's account is determined by who knows the private key, *i.e.* it controls the account. For this reason, users should never disclose or make public their private keys, because their theft has irreversible consequences.** The ECDSA algorithm, which takes into account the developer-defined structural parameters, is responsible for creating the private keys and, consequently, for the security of the system.

Below we study cases of making private keys from the algorithm for some elliptic curves from those we studied in the Galois field F_{100153}. By maintaining specific domain parameters, such as the base point P and the order n of the subgroup, we have studied the impact on the production of private keys. Note that



Graph 2. Tolerance of BrainpollP256t1 and Anomalous against all model based attacks on the Galois fields.

these results are the outcome of great search in all above-mentioned elliptic curves. The following were chosen because of their representative image for the purpose of the survey, but this did not mean that there were no alternatives.

Note: Usually for the same domain parameters, the ECDSA algorithm allows code-level creation of many private keys that serve the same needs with the same efficiency. This is most common for huge Galois fields.

In the first phase, we defined the E elliptic curve M-383 according to the formula:

$$E: y^2 = x^3 + 2065150 \cdot x^2 + x \quad \text{with } x \in \mathbb{R},$$

In the finite Galois field mentioned, maintaining the base point $P(22, 72669)$ stable, for which a subgroup of points of the curve with order $n =$ 6229 is created, a value which is objectively large prime number in relation to the field. Below in **Table 4** are indicative results in tabular form:

We notice that with the creation of three different key pairs (Q, d), as the size of the private key increases, the algorithm's "resistance" to model-based attacks increases, with the exception of Pollard rho. Corresponding picture is observed in all elliptic curves of the survey.

Diagramically, the stealing times are plotted per model-based method in **Graph 3**.

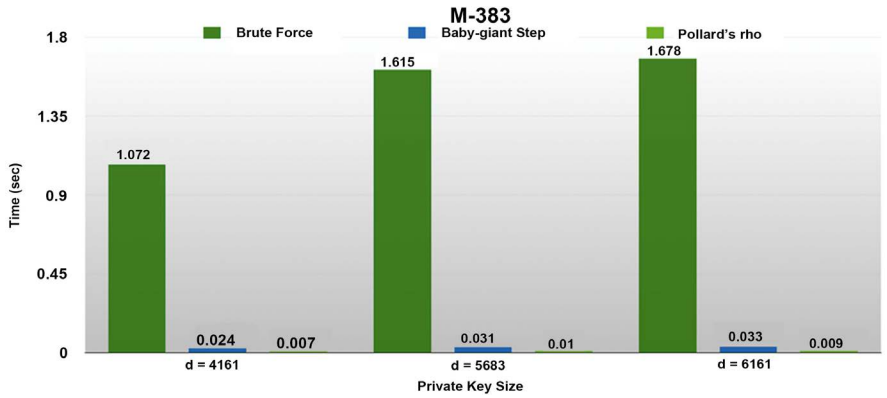
Conclusion: From an algorithm perspective, it is possible to improve the security of a user's account by generating a large-sized private key. This security involves model-based methods such as Brute force and the Baby-giant step and variations thereof. Pollard rho does not appear to be affected to some extent. Finally, the size of the public key does not affect the results at all.

Two further elliptic curves, Curve1174 and BrainpollP256t1, were then studied. Their equations are:

$$y^2 = x^3 - 28486 \cdot x + 1926947 \quad \& \quad y^2 \cong x^3 - 3 \cdot x + 4.621 \cdot 10^{76}, x \in \mathbb{R}$$

Table 4. Tolerance of M-383 curve on the F100153 Galois field for fixed Base Point P and value of order n.

Galois Field F_{100153}							
					Method (Time in sec)		
Curve	Base Point P	Order n	Private Key d	Public Key Q	Brute Force	Baby-Giant Step	Pollard's rho
M-383	(22,72669)	6229	4161	(70,789, 7108)	1.072	0.024	0.007
			5683	(63,106, 12,167)	1.615	0.031	0.01
			6161	(60,238, 81,175)	1.678	0.033	0.009



Graph 3. Test results illustrating the tolerance of M-383 curve.

Table 5 presents the collected data:

For the Curve 1174 we chose to keep stable the n order of the subgroup of points produced by different base points P.

- Initially keeping the abscissa of the base point stable, we again compared the private key d time of theft. We notice that for a larger ordinate, the generated private key increases and exhibits greater "resistance" to the Brute force and Baby-giant step model-based methods. As before, the Pollard rho method is not affected to some extent.
- In the second phase, maintaining the base point stable, we applied the algorithm several times until we arrive at two pairs (d, Q) with a larger private key. We notice as before, more resistance to Brute force and Baby-giant step. Pollard Rho method times of theft remain small.

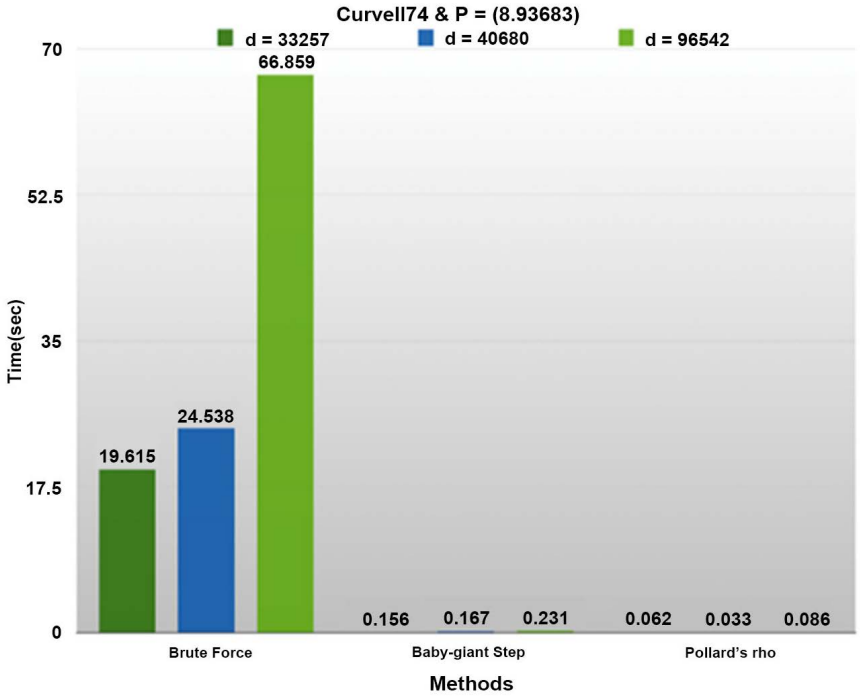
Note: While in the M-383 the Pollard Rho method times are not affected by the size of the private key d, on the contrary Curve1174 shows that the respective times are affected either negatively (d = 33,257 → d = 40,680) or positively (d = 33,257 → d = 96,542). Note that no workable conclusion emerged from the overall study of Curve1174 elliptic curve.

The results are shown diagrammatically in Graph 4 and Graph 5 respectively:

Finally, for the same reasons, we studied the BrainpollP256t1 curve. Here we observed that on the contrary with previous M-383 and Curve1174, BrainpollP256t1 presents a stable motif with unexpected increases in "resistance" across the three model-based methods. Diagrammatically came out the following, which can be seen in Graph 6:

Table 5. Tolerance of Curve1174 and BrainpollP256t1 curves against all model based methods for a number of fixed Base Points P and order n values.

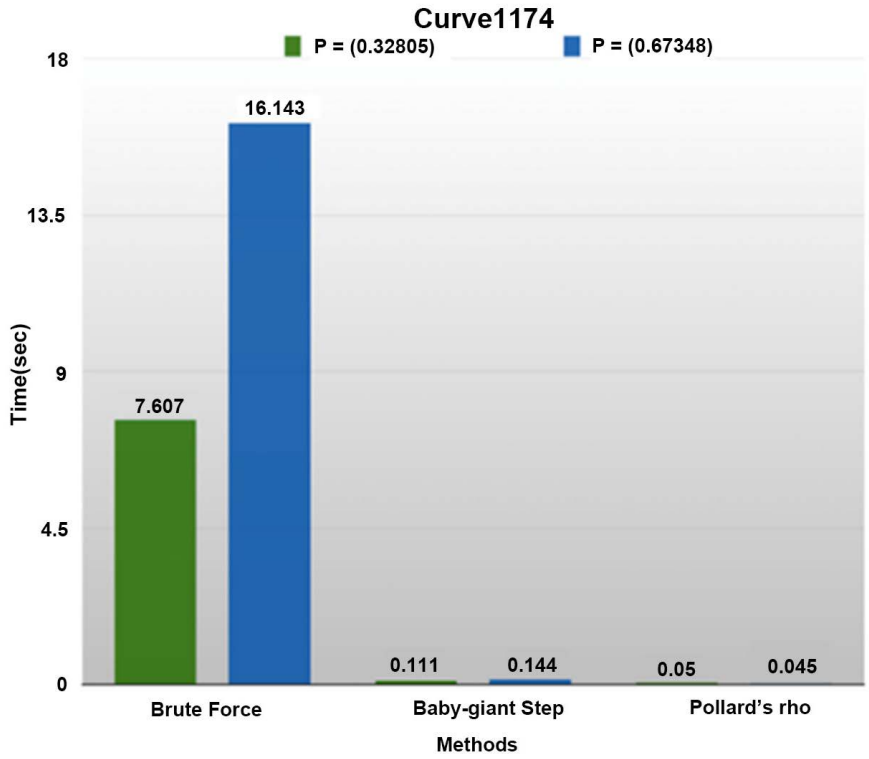
Galois Field F_{100153}							
					Method (Time in sec)		
Curve	Base Point P	Order n	Private Key d	Public Key Q	Brute Force	Baby-Giant Step	Pollard's rho
Curve1174	(0, 32,805)	99,551	14,532	(63,436, 61,932)	7.607	0.111	0.05
	(0, 67,348)		27,861	(97,688, 99,242)	16.143	0.144	0.045
	(8, 93,683)		33,257	(1310, 23,887)	19.615	0.156	0.062
	(8, 93,683)		40,680	(60,101, 4759)	24.538	0.167	0.033
	(8, 93,683)		96,542	(15,179, 6830)	66.859	0.231	0.086
BrainpollP256t1	(4, 21,591)	25,057	11,961	(23,353, 47,637)	4.699	0.053	0.015
	(4, 28,510)		13,327	(11,666, 38,300)	5.284	0.068	0.028
	(7, 24,312)		20,458	(9236, 24,170)	8.602	0.079	0.016
	(16, 25,549)		23,719	(5992, 11,700)	9.736	0.085	0.022



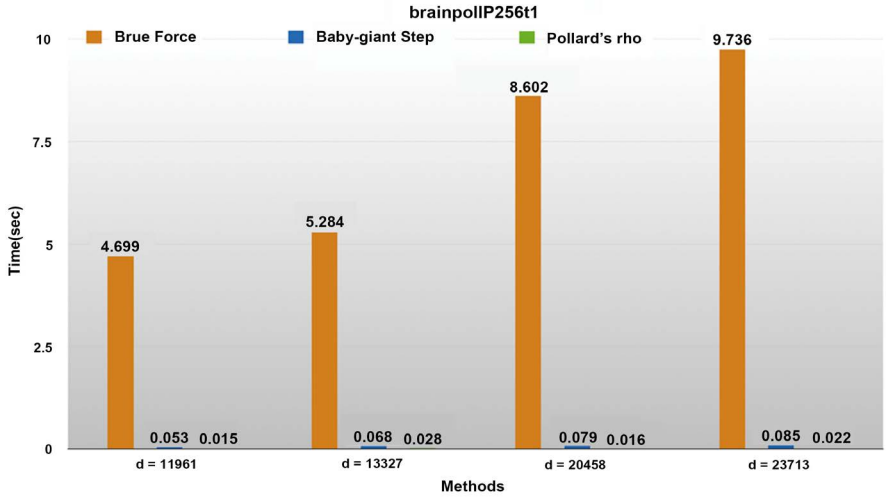
Graph 4. Tollerance of Curve1174 for different private keys d.

For BrainpollP256t1 we observe the following:

- When the ordinance of the base point increases (the first two triplets of the barplot), the private key increases, resulting in longer resistances in all model-based methods, even in Pollard rho, which has not been observed so far. Specifically from $P(4,21591)$ to $P(4,28510)$.
- In the other two column triads we get $P(7,24312)$ and $P(16,2549)$ by increasing both base point components. Again, the times of all methods until the private key d is taken up increase.



Graph 5. Tollerance of Curve1174 for different Base Point P.



Graph 6. Tolerance of BrainpollP256t1 against model based methods for a number of private keys d.

We conclude that elliptic curve selection, such as BrainpollP256t1, against another, such as M-383, may have positive effects particularly against the Pollard rho method, which is objectively the most effective of the other two studied.

General remark: The results of the survey are heavily influenced by the infrastructure. When calculating the private key times of theft executing the algorithm for the same curve and just the same domain parameters by unleashing any model-based method, many different times were observed, without of

course using another computer. The average of these times is the one completed in the data tables. The random production of the above times has not been subject of research.

Notes:

- The elliptical curve used in a cryptocurrency is chosen for a specific purpose. For example, BrainpollP256t1 for smaller base points produces larger private keys than Curve1174. However, the waiting time for calculating the base points of the former is much greater than that of the second. For a new cryptocurrency, if developers want large private keys for users, they are more likely to opt for brainpoll256t1 vs. Curve1174.
- Any mathematical model-based method of stealing will succeed over time. The point is that if times of theft approach prohibitive values with respect to natural time, then we consider them to fail. Acceptable transaction time is the one that does not allow the reversal of the transaction after verification.

3. Conclusions

On a daily base millions of transactions are executed in various cryptocurrencies around the world. The pair of keys of each user is his digital identity, which means that their security is a matter of utmost importance. Those responsible for safety are algorithms such as ECDSA. The domain parameters of the above algorithms were the subject of the research.

We chose to study in addition to the secp256k1 already used by the algorithm, all proposed elliptic curves that the NSA has declared as the most secure. Secondary criteria such as high transaction speeds were not studied.

Generally, useful patterns were observed when selecting the base point P, with respect to its coordinates, resulting in an increase in the size of the user's private keys. Accordingly, the choice of the elliptic curve to be used by the algorithm to produce the keys plays a very important role. Of course, any change to the above domain parameters should be reconsidered because it does not always lead to better safety results.

In any case, it is recommended to use the domain parameters of the safe elliptic curves as stated on https://safecurves.cr.yp.to/ if the infrastructure allows the programmer-manager of the platform to do so.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Secp256k1. https://en.bitcoin.it/wiki/Secp256k1

[2] Lange, T. and Bernstein, D.J. (2013) SafeCurves: Choosing Safe Curves for Elliptic-Curve Cryptography. https://safecurves.cr.yp.to/refs.html

[3] Entropy (Information Theory).

https://en.wikipedia.org/wiki/Entropy_(information_theory)

[4] Narayanan, A., Bonneau, J., Felten, E., Miller, A. and Goldfender, S. (2015) Bitcoin and Cryptocurrency Technologies. Princeton University, Princeton, NJ.

[5] Johnson, D., Menezes, A. and Vanstone, S. (2001) The Elliptic Curve Digital Signature Algorithm (ECDSA). Department of Combinatorics & Optimization, University of Waterloo.
http://www.cs.miami.edu/home/burt/learning/Csc609.142/ecdsa-cert.pdf

[6] Silverman, J.H. (2006) An Introduction to the Theory of Elliptic Curves. Brown University and NTRU Cryptosystems, Inc.
https://www.math.brown.edu/~jhs/Presentations/WyomingEllipticCurve.pdf

[7] Benvenuto, C.J. (2012) Galois Field in Cryptography.
https://sites.math.washington.edu/~morrow/336_12/papers/juan.pdf

[8] Edward Curve. https://en.wikipedia.org/wiki/Edwards_curve

[9] Hasse's Theorem on Elliptic Curves.
https://en.wikipedia.org/wiki/Hasse%27s_theorem_on_elliptic_curves

[10] Rational Point. https://en.wikipedia.org/wiki/Rational_point

[11] Rykwalder, E. (2014) The Math Behind Bitcoin.
https://www.coindesk.com/math-behind-bitcoin/

[12] Corbellini, A. (2015) Elliptic Curve Cryptography: A Gentle Introduction.
http://andrea.corbellini.name/2015/05/17/elliptic-curve-cryptography-a-gentle-introduction/

[13] Khalique, A., Sood, S. and Singh, K. (2010) Implementation of Elliptic Curve Digital Signature Algorithm. *Internatuonal Journal of Computer Applications*, **2**, No. 2.
http://www.ijcaonline.org/volume2/number2/pxc387876.pdf

[14] Corbellini, A. (2015) Elliptic Curve Cryptography: Breaking Security and a Comparison with RSA.
http://andrea.corbellini.name/2015/06/08/elliptic-curve-cryptography-breaking-security-and-a-comparison-with-rsa/

[15] Qubd (2017) https://github.com/qubd/mini_ecdsa/blob/master/mini_ecdsa.py