Computing information on domain parameters from public keys selected uniformly at random

Martin Ekerå*

September 10, 2015

Abstract

The security of many cryptographic schemes and protocols rests on the conjectured computational intractability of the discrete logarithm problem in some group $\langle g \rangle$ of prime order. Such schemes and protocols require domain parameters that specify $\langle g \rangle$ and a specific generator g. In this paper we consider the problem of computing information on the domain parameters from public keys selected uniformly at random from $\langle g \rangle$.

We show that it is not possible to compute any information on the generator g regardless of the number of public keys observed.

In the case of elliptic curves $E(\mathbb{F}_p)$ or $E(\mathbb{F}_{2^n})$ on short Weierstrass form, or $E(\mathbb{K})$ on Edwards form, twisted Edwards form or Montgomery form, where $\mathbb{K}$ is a non-binary field, we show how to compute the domain parameters excluding the generator from four keys on affine form.

Hence, if the domain parameters excluding the generator are to be kept private, points may not be transmitted on affine form. It is an open question whether point compression is a sufficient requirement.

Regardless of whether points are transmitted on affine or compressed form, it is in general possible to create a distinguisher for the domain parameters, excluding the generator, both in the case of the elliptic curve groups previously mentioned, and in the case of subgroups of $\mathbb{F}_p^*$.

We propose that a good method for preventing all of the above attacks may be to use blinding schemes, and suggest new applications for existing blinding schemes originally designed for steganographic applications.

1 Introduction

The security of many cryptographic schemes and protocols rests on the conjectured computational intractability of the discrete logarithm problem in some group $\langle g \rangle$ of prime order. Such schemes and protocols require domain parameters that specify the group $\langle g \rangle$ and a specific generator g. In this paper, we consider the problem of computing information on the domain parameters from a set of public keys selected uniformly at random from $\langle g \rangle$.

One instance of this problem, that may be of particular interest, and that serves as motivation for this work, is the Diffie-Hellman key exchange where a passive adversary observes exchanges of public keys selected uniformly at random from $\langle g \rangle$ and where the domain parameters are not transmitted.

*Swedish NCSA, Swedish Armed Forces, SE-107 85 Stockholm, Sweden.

Although the domain parameters used for a Diffie-Hellman key exchange are ordinarily considered to be public, this is an interesting problem to study since the generic approach to breaking the Diffie-Hellman key exchange in this setting is to compute the discrete logarithm of one of the public keys to the base of the generator. It is not possible to perform this computation without knowledge of the domain parameters.

If the domain parameters are not transmitted on explicit form in the key exchange, but are rather kept private, they could hence potentially serve the role of a long term secret. The purpose of this paper is in part to explore to what extent it is possible to have the domain parameters serve such a role.

1.1 The post-quantum era

With the advent of quantum computers, it will become computationally feasible to compute discrete logarithms in the elliptic curve groups and subgroups of $\mathbb{F}_p^*$ that are in general use today, assuming that the domain parameters are known.

If the domain parameters are kept private, and may not be computed, the security of existing schemes that rely on the computational intractability of the discrete logarithm problem will be weakened and properties such a perfect forward secrecy in schemes such as Diffie-Hellman will be irrevocably lost.

However, the security of such schemes will not necessarily be completely compromised, since knowledge of the domain parameters is required to compute discrete logarithms using Shor's algorithm.

If the domain parameters are kept private, and may not be computed using either classical or quantum computers, the domain parameters could still potentially serve the role of a long term secret in the post-quantum era.

1.2 Selecting the domain parameters

A good guiding principle when selecting domain parameters is to use different domain parameters in different applications that need not be interoperable, so as not to put all eggs in one basket if some domain parameters would turn out to be weak. Adherence to this principle furthermore reduces susceptibility to massive pre-computation attacks, of the kind described in [1].

Another good guiding principle when selecting domain parameters is to select the domain parameters in a randomized but easily verifiable manner, see for example [3, 5]. Adherence to this principle makes it non-trivial to guess the parameters, allows trust in the parameters to be built, and helps to avoid small unknown classes of weak parameters.

Unfortunately, these guiding principles are seldom adhered to in practise. Instead, a very small set of standardized domain parameters are used in the vast majority of applications. This may potentially be a security concern.

We believe that this is in part caused by a need for interoperability, but also in part by existing standards and regulations that require that specific domain parameters be used. In some cases it is furthermore non-trivial to create cryptographically strong domain parameters.

1.3 Protecting the domain parameters

In many applications, the domain parameters will have to be public in order to enable interoperability. However, there are applications where this is not necessary. In such applications, it makes sense not to volunteer the domain parameters to potential adversaries by transmitting them in the clear, but to rather attempt to keep the parameters private.

However, this is only the case provided that the domain parameters have been selected in accordance with the guiding principles in the previous subsection, so that they cannot be trivially guessed, and provided that the adversary is unable to compute information on the domain parameters from public keys that have to be transmitted in the clear.

2 Domain parameters

The domain parameters specify the group $\langle g \rangle$ of large prime order q in which the discrete logarithm problem is conjectured to be computationally intractable, and a specific generator g. Although there are many groups that could conceivably be used in cryptographic applications, the most commonly used groups are subgroups of $\mathbb{F}_p^*$ and elliptic curve groups.

In the case of elliptic curve groups, curves on various forms are in use. Common choices include curves on short Weierstrass form, on Montgomery form, on Edwards form and twisted Edwards form, defined over prime fields and various extension fields of small characteristic.

Below, we introduce notation and very briefly describe what information makes up the domain parameters for the aforementioned groups. In the case of elliptic curve groups, we denote the generator by G, since it is common to use capital letters to denote elements of such groups.

- In the case of subgroups of $\mathbb{F}_p^*$, the domain parameters consist of the prime p that defines the multiplicative group $\mathbb{F}_p^*$, and of a generator $g \in \mathbb{F}_p^*$.

  It must be the case that $p = 2rq + 1$ for some $r \geq 1$ for g to have order q.

- In the case of elliptic curve groups $E(\mathbb{F}_p)$ on short Weierstrass form

  $$y^2 = x^3 + ax + b,$$

  the domain parameters consist of a prime $p > 3$ that defines the field $\mathbb{F}_p$, of the two Weierstrass coefficients $a, b \in \mathbb{F}_p$, and of a generator $G \in E$.

- In the case of elliptic curve groups $E(\mathbb{F}_{2^n})$ on the Weierstrass form

  $$y^2 + xy = x^3 + ax^2 + b,$$

  the domain parameters consist of an irreducible polynomial of degree n with coefficient in $\mathbb{F}_2$ that defines the field $\mathbb{F}_{2^n}$, of the two Weierstrass coefficients $a, b \in \mathbb{F}_{2^n}$, and of a generator $G \in E$.

- In the case of elliptic curve groups $E(\mathbb{K})$ on Edwards form

  $$x^2 + y^2 = c^2(1 + dx^2y^2),$$

  the domain parameters consist of parameters that define a non-binary field $\mathbb{K}$, of the two coefficients $c, d \in \mathbb{K}$, and of a generator $G \in E$.

- In the case of elliptic curve groups $E(\mathbb{K})$ on twisted Edwards form

$$ax^2 + y^2 = 1 + dx^2y^2,$$

the domain parameters consist of parameters that define a non-binary field $\mathbb{K}$, of the two coefficients $a, d \in \mathbb{K}$, and of a generator $G \in E$.

- In the case of elliptic curve groups $E(\mathbb{K})$ on Montgomery form

$$by^2 = x^3 + ax^2 + x,$$

the domain parameters consist of parameters that define a non-binary field $\mathbb{K}$, of the two coefficients $a, b \in \mathbb{K}$, and of a generator $G \in E$.

In this paper, we consider the problem of computing information on the domain parameters in the general case, and in the specific cases described above.

3 Public keys

The public keys that we observe are on different forms depending on from which group the public keys are selected.

- In the case of subgroups of $\mathbb{F}_p^*$ a public key $z = g^e$ is an element selected uniformly at random from $\langle g \rangle$.

- In the case of subgroups of elliptic curves a public key $Z = (x, y) = [e]\,G$ is a point selected uniformly at random from $\langle G \rangle$.

  The public key may be transmitted on affine form (x, y), or on compressed form $(x, \mathrm{sgn}(y))$ provided that the curve form does admit standard point compression, see appendix A.

  To decompress a compressed point $y^2 = f(x)$ is first computed, where $f(x)$ is obtained by re-writing the curve equation, again see appendix A. Then y is computed by taking the square root. The function $\mathrm{sgn}(y)$ outputs a single sign bit that serves to identify which of the two roots should be returned. A common choice for elliptic curves $E(\mathbb{F}_p)$ of prime characteristic $p > 3$ is to let $\mathrm{sgn}(y) = y \bmod 2$.

4 Distinguishing the group

If an adversary suspects that the public keys may have been selected from a specific group, she or he may test whether this is indeed the case, by using the methods described below.

- In the case of subgroups of $\mathbb{F}_p^*$, the adversary may for example test that all public keys are in $\mathbb{F}_p^*$ and that the order of all public keys is q.

- In the case of elliptic curve groups $E(\mathbb{K})$, where the public keys are on affine form, the adversary may for example test if the coordinates of all public keys are in $\mathbb{K}$, and if the coordinates of all public keys fulfill the curve equation.

  In the case of $\langle G \rangle$ being a strict subgroup of $E(\mathbb{K})$ the adversary may optionally proceed to test that the order of all public keys is q.

- In the case of elliptic curve groups $E(\mathbb{K})$, where the public keys are on compressed form, the adversary first tests that the x coordinate is in $\mathbb{K}$ and that the point can be decompressed to affine form, that is the adversary forms the number she or he expects to equal y^2 and verifies that it is indeed a square in $\mathbb{K}$.

  In the case of $\langle G \rangle$ being a strict subgroup of $E(\mathbb{K})$ the adversary may optionally proceed to test that the order of all public keys is q.

The above tests should in general allow the adversary to distinguish between groups. It may conceivably be the case that two groups could be crafted such that the above tests does not allow an efficient distinguisher to be created.

However, if we assume that the groups have not been deliberately selected to be hard to distinguish, the probability of the groups being difficult to distinguish may be conjectured to be negligible.

The fact that groups may in general be distinguished implies that even if the domain parameters are kept private in an application, an adversary seeking to determine the parameters can test the public keys against a database of previously observed domain parameters in the hope of finding a match.

Consequently, it makes little sense to attempt to keep the domain parameters private in an application, if the same domain parameters are used in other applications where they are not kept private, or if the parameters are known to have been otherwise published or disclosed.

5 Distinguishing or computing the generator

The tests described above provide no information on the generator.

Of course, if the adversary has access to a database of domain parameters previously observed that includes generators, she or he may guess that the generator in the database was used to select the public keys.

If the public keys have been selected uniformly at random, however, the adversary can not determine whether this guess is correct, nor can she or he hope to compute the generator from the information in the public keys.

To see why this is the case, let $g, \tilde{g} \neq g$ both be generators of the same group $\langle g \rangle = \langle \tilde{g} \rangle$ of prime order q. In multiplicative notation this implies that $\tilde{g} = g^k$ for some k. Furthermore, let $e_1, e_2, \ldots$ be private keys selected independently and uniformly at random from the interval $0 \leq e_i < q$.

Then the two sets $\{g^{e_1}, g^{e_2}, \ldots\}$ and $\{\tilde{g}^{e_1}, \tilde{g}^{e_2}, \ldots\} = \{g^{e_1 k}, g^{e_2 k}, \ldots\}$ both contain public keys selected independently and uniformly at random from $\langle g \rangle$.

If the adversary is allowed to observe one of these sets of public keys, the adversary cannot distinguish between the case that it was selected using g or $\tilde{g}$ as generator. This is true even if the adversary is able to compute discrete logarithms, since the exponents will be selected uniformly at random from the above stated interval regardless of which generator is used as base.

Since the group $\langle g \rangle$ has prime order q it has $q - 1$ distinct generators. If the generator is selected uniformly at random, then by definition all of these generators $q - 1$ are equally likely to have been used.

The method for computing the private key given a public key is to compute the discrete logarithm of the public key to the base of the generator. This requires knowledge of the generator. If g is used as base when $\tilde{g}$ was actually

used to select the public keys, the resulting private keys will all be concealed by the same unknown $k = \log_g \tilde{g}$.

The above implies that the generator could potentially serve the role of a long term secret, provided that it is selected uniformly at random and is kept private. This is attractive since it is trivial to select a new generator given some existing domain parameters. It may be considerably more difficult to generate new domain parameters from scratch. Furthermore existing implementations of elliptic curve cryptography that support domain parameters being loaded should easily be able to accommodate a change of generator.

6 Computing the domain parameters

In this section, we consider the problem of computing the domain parameters, excluding the generator, from a set of observed public keys.

6.1 Elliptic curves and public keys on affine form

Let us start by considering the case of elliptic curve groups $E(\mathbb{F}_p)$ on short Weierstrass form. We seek to compute a, b and p, the order r of E and the order q of G given a set of public keys on affine form.

The trick to accomplishing this is to first compute the prime p by forming one or more expressions that depend only on the coordinates of the public keys, and that are congruent to zero modulo p but that are non-zero over $\mathbb{Z}$. We demonstrate one way of doing this below.

If (x_1, y_1) and (x_2, y_2) are public keys, it must hold that

$$y_1^2 - x_1^3 - ax_1 - b \equiv 0, \qquad y_2^2 - x_2^3 - ax_2 - b \equiv 0,$$

so the b coefficient may be eliminated by subtracting the expressions

$$(y_1^2 - y_2^2) - (x_1^3 - x_2^3) - a(x_1 - x_2) \equiv 0.$$

If (x_3, y_3) and (x_4, y_4) are two more public keys, it must hold that

$$(y_3^2 - y_4^2) - (x_3^3 - x_4^3) - a(x_3 - x_4) \equiv 0,$$

so the a coefficient may be eliminated by forming the expression

$$\begin{aligned}(x_3 - x_4)\left[(y_1^2 - y_2^2) - (x_1^3 - x_2^3) - a(x_1 - x_2)\right] - \\ (x_1 - x_2)\left[(y_3^2 - y_4^2) - (x_3^3 - x_4^3) - a(x_3 - x_4)\right] = \\ (x_3 - x_4)\left[(y_1^2 - y_2^2) - (x_1^3 - x_2^3)\right] - (x_1 - x_2)\left[(y_3^2 - y_4^2) - (x_3^3 - x_4^3)\right] \equiv 0.\end{aligned}$$

This expression depends only on the coordinates of the public keys, is congruent to zero modulo p, and is non-zero over $\mathbb{Z}$ with high probability.

Hence, we may compute p by evaluating the expression and factoring the resulting number. From the factored number, we obtain one or more candidates for p. In the vast majority of cases, we only obtain a single candidate that is of the correct bit length and greater than all of the x-coordinates observed.

If factoring the number proves difficult, another approach is to compute several numbers that are congruent to zero modulo p by permuting the indices in the above expression, or by involving additional public keys.

By computing the greatest common divisor of these numbers, and then eliminating any small prime factors in the result using trial division, we very efficiently compute p. In our practical experiments, knowledge of the affine coordinates of four points proved sufficient to compute p.

Once p has been computed, it is a trivial matter to compute a and b as

$$\begin{aligned} a &\equiv (x_1 - x_2)^{-1} \left[(y_1^2 - y_2^2) - (x_1^3 - x_2^3) \right], \\ b &\equiv y_1^2 - x_1^3 - a x_1. \end{aligned}$$

The order r of the elliptic curve may then be computed using standard point counting algorithms. Assuming that $r = hq$, where h is some small cofactor, this allows us to compute q and to identify the subgroup $\langle G \rangle$. However, as previously stated, we cannot compute the specific generator G.

It is trivial to see that the above approach may be generalized to elliptic curves $E(\mathbb{F}_{2^n})$ on short Weierstrass form, and to elliptic curves $E(\mathbb{K})$ on Edwards form, twisted Edwards form and Montgomery form, see appendix A, as well as to elliptic curves on general Weierstrass form over arbitrary finite fields.

To summarize, we observe that in the elliptic curve setting, the domain parameters, excluding the generator, may be computed very efficiently from only a very small set of public keys on affine form selected uniformly at random.

If the domain parameters are to be kept private, it is necessary therefore to at least employ point compression, and furthermore to not use parameters that the adversary may previously have observed.

6.2 Elliptic curves and public keys on compressed form

Let us now consider the case of public keys on standard compressed form.

6.2.1 The compressed y coordinate

When point compression is employed, the compressed y coordinate provides no information since for all x coordinates on the curve, except for up to three points with zero y coordinate, there is exactly one point with compressed y coordinate equal to one, and exactly one point with compressed y coordinate equal to zero.

Assuming the public keys were selected uniformly at random, the compressed y coordinate may hence for all intents and purposes be modelled as a single bit selected uniformly at random and independently of the x coordinate. We shall therefore ignore the compressed y coordinate henceforth.

6.2.2 The x coordinate

Let $E(\mathbb{F}_p)$ be an elliptic curve on short Weierstrass form over a field of prime characteristic $p > 3$ and let $G \in E$ be a generator of prime order q.

In analogy, let $\tilde{E}(\mathbb{F}_{\tilde{p}})$ be an elliptic curve on short Weierstrass form over a field of prime characteristic $\tilde{p} > 3$ such that $\tilde{p} \sim p$, and let $\tilde{G} \in \tilde{E}$ be a generator of prime order $\tilde{q}$ such that $\tilde{q} \sim q$, where $\sim$ denotes equal length in bits.

Given an x coordinate on E it is then easy to show that the probability that $x < \tilde{p}$ is greater than or equal to $1/2$. Let $\tilde{a}, \tilde{b}$ be the coefficients in the short Weierstrass equation of $\tilde{E}$. Then, the probability of $u = x_i^3 + \tilde{a} x_i + \tilde{b}$ being a square in $\mathbb{F}_{\tilde{p}}$ should be about $1/2$. Again, it is conceivable that two groups

such that u is a square with considerably higher probability than $1/2$ could be deliberately crafted. However, if we assume that the curves are selected at random then the above statements should hold.

We may use the above probabilities not only to build distinguishers but also to set a naïve upper bound on how many x coordinates the adversary should need to observe for an elliptic curve $E(\mathbb{F}_p)$ on short Weierstrass form to be uniquely determined. If p is of length m bits, then less than $3m$ randomly selected public keys should be sufficient to determine the curve.

However, we have yet to find an efficient algorithm that allows the prime p and the two Weierstrass coefficients to be computed. To the best of our knowledge, it is still an open question whether such an efficient algorithm exists, or whether this is a hard problem.

6.2.3 Simplifications and generalizations

Possible ways of simplifying this problem include assuming that portions of the domain parameters are known. For instance, it is plausible to assume that $a \equiv -3$ since this is a very common choice. Also, one may perhaps assume that p is known since there are those who advocate selecting non-random primes with certain properties that allow fast arithmetic.

It is easy to see that the above discussion may be generalized to elliptic curves $E(\mathbb{K})$ on Edwards form, twisted Edwards form and Montgomery form.

6.3 Subgroups of $\mathbb{F}_p^*$

Finally, let us briefly consider public keys selected from subgroups of $\mathbb{F}_p^*$.

If we assume that p can somehow first be computed from the public keys observed, then the subgroup $\langle g \rangle$ may be identified by testing if $z^q \equiv 1$ for different q such that $q \mid p-1$. Typically this requires factoring $(p-1)/2 = rq$.

It is common to set $r = 1 \Rightarrow q = (p-1)/2$. Another common choice is to set q sufficiently large for naïve cycle-finding algorithms to run slower in the subgroup of order q than GNFS-based algorithms do in $\mathbb{F}_p^*$. In the latter case, computing q by factoring rq may be difficult if r has large prime factors.

This having been said, it is by no means evident how one would efficiently compute p from the public keys observed. If this may be accomplished, it may well be that some information on the subgroup of order q would have to be used or derived. To the best of our knowledge, it is still an open question whether an efficient algorithm for computing p exists, or whether this is a hard problem.

7 Blinding schemes

As was described in section 4, if we are allowed to observe public keys selected uniformly at random from some unknown group, we may in general distinguish whether these public keys all belong to some known group.

We propose that a good method for preventing the creation of such distinguishers is to use blinding schemes. Furthermore, we propose that by using blinding schemes we may prevent adversaries from computing information on the domain parameters in cases where efficient methods for performing such computations from non-blinded public keys do exist.

In particular, we propose to consider blinding schemes with the following property: Let S be a set of groups and let $\mathbb{G}$ and $\tilde{\mathbb{G}} \neq \mathbb{G}$ be any two groups selected from S by the adversary. Let the adversary observe a set of blinded public keys selected at random from either $\mathbb{G}$ or $\tilde{\mathbb{G}}$. Then, the adversary should be unable to determine whether the keys were selected from $\mathbb{G}$ or $\tilde{\mathbb{G}}$.

7.1 Existing blinding schemes

In [6] Fouque, Joux, and Tibouchi introduced an efficient injective map from bit strings to a subset of the points on an elliptic curve. This map was subsequently used as a basis for the Elligator blinding scheme [4] originally introduced by Bernstein, Krasnova and Lange and subsequently revised by Bernstein, Hamburg, Krasnova and Lange to support larger classes of elliptic curves.

After the introduction of the Elligator blinding scheme a few related works have been published, for example [2] by Aranha, Fouque, Qian, Tibouchi and Zapalowicz to extend support for blinding to elliptic curves over binary fields, and [7] by Tibouchi to support larger classes of elliptic curves, including curves of prime order, and to complete prime order subgroups of such curves.

7.2 A new application for existing blinding schemes

The motivation for the Elligator blinding scheme, and the recent related works, has thus far been to conceal the use of elliptic curve cryptography in various steganographic schemes. In this paper, we make the observation that these blinding schemes have an additional application.

More specifically, since these schemes may be used as is to prevent an adversary from creating distinguishers for elliptic curve groups, they may be used to prevent an adversary from computing information on the elliptic curve domain parameters in the settings where the domain parameters are to be kept private.

Using the above notation, assume that the blinding scheme describes efficient injective maps $\varphi : D \to \mathbb{G}$ and $\tilde{\varphi} : D \to \tilde{\mathbb{G}}$. Then elements may be selected uniformly at random from the image of φ or $\tilde{\varphi}$ and mapped to D prior to transmission. This will result in the adversary observing an element selected uniformly at random from D regardless of whether the public key was selected from $\mathbb{G}$ or $\tilde{\mathbb{G}}$. Hence, the adversary cannot distinguish between these two groups.

The above referenced blinding schemes describe maps from D to a large set S of distinct groups. This implies that the adversary will be unable to distinguish between any two groups in this set, which in turn implies that it will be impossible for the adversary to compute information on the group.

7.2.1 Practical considerations and limitations

It should be noted that previously in this paper we have assumed that the public keys are selected uniformly at random from $\mathbb{G}$. Above we instead assume that they are selected uniformly at random from the image of φ. These two sets are not necessarily identical.

The use of a blinding scheme may hence in some cases introduce a bias in the public keys selected from $\mathbb{G}$. Care must be exercised so as to ensure that any such bias does not result in any adverse consequences for the security of the cryptographic scheme in which blinding is used.

In order to ensure that all maps to the groups in S have the same domain D it may furthermore be necessary to impose additional restrictions on the size of the domain D, resulting in the introduction of an additional bias.

8 Encryption as an alternative to blinding

An obvious alternative to using a blinding scheme is to encrypt the public keys using a symmetric cipher under some pre-shared symmetric key. Since it is possible to distribute the symmetric key alongside the domain parameters, which are also private, the fact that a key needs to be distributed need not entail any increased complexity from a key distribution perspective.

Encrypting the public keys gives protection against distinguishing attacks and attacks where the adversary attempts to compute the domain parameters for as long as the private key is kept secret and for as long as the symmetric cipher itself cannot be broken by the adversary. However, if the private key is compromised the adversary can mount distinguishing attacks.

If a blinding scheme is used instead of point encryption, it is impossible to mount distinguishing attacks. Even if all information, including the domain parameters, are available to the adversary she or he cannot prove or disprove the hypothesis that these domain parameters are used. This is a clear advantage of using a blinding scheme compared to standard symmetric encryption.

9 Open problems

Below we list some open problems that may be of interest for further study by the academic community.

- Compute p in the case where the public keys belong to a subgroup of $\mathbb{F}_p^*$, or demonstrate that this is a hard problem.
- Develop blinding schemes for public keys belonging to a subgroup of $\mathbb{F}_p^*$.
- Compute p, a and b in the case where the public keys are on compressed form and belong to an elliptic curve group $E(\mathbb{F}_p)$ on short Weierstrass form, or demonstrate that this is a hard problem. Possible simplifications include assuming a subset of $\{p, a, b\}$ to be known.
- In the more general case, determine $\mathbb{K}$ and the coefficients on the curve equation in the case where the public keys are on compressed form and belong to an elliptic curve group $E(\mathbb{K})$ on short Weierstrass form, Edwards form, twisted Edwards form or Montgomery form.

  There are various ways of simplifying this problem, in analogy with the simplifications proposed in the previous bullet.

Above it is assumed that a set of public keys selected uniformly at random are observed by a passive adversary. The settings where the adversary is active, or where the public keys have a bias, or where the adversary can impose requirements on the public keys selected, are not considered.

In some special cases where the public keys are not selected uniformly at random, we are aware that information on the domain parameters may be computed. However, these special cases are not very likely to occur in practise.

Final remarks

In the above discussion, we have considered the setting where the adversary is only allowed to observe a set of public keys.

In order to implement a cryptologic scheme or protocol it is often necessary to transmit additional messages that bear some relation to the public keys and hence to the domain parameters. It may be the case that an adversary who is allowed to observe these messages in addition to the public keys will be able to compute additional information on the domain parameters. However, such attacks are out of the scope of this paper.

References

[1] D. Adrian, K. Bhargavan, Z. Durumeric, P. Gaudry, M. Green, J. A. Halderman, N. Heninger, D. Springall, E. Thomé, L. Valenta, B. van der Sloot, E. Wustrow, S. Zanella-Béguelin and P. Zimmerman, "*Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practise*". Accessible via `https://weakdh.org/imperfect-forward-secrecy.pdf`.

[2] D. F. Aranha, P.-A. Fouque, C. Qian, M. Tibouchi and J.-C. Zapalowicz, "*Binary Elligator Squared*". IACR ePrint 2014/486.

[3] The Brainpool Group, "*ECC Brainpool Standard – Curves and Curve Generation*", 2005. Accessible via `http://www.ecc-brainpool.org`.

[4] D. J. Bernstein, M. Hamburg, A. Krasnova and T. Lange, "*Elligator: Elliptic-curve points indistinguishable from uniform random strings*", in ACM SIGSAC CCS 2013, p.p. 967-980. IACR ePrint 2013/325.

[5] J.-P. Flori, J. Plût, J.-R. Reinhard and M. Ekerå, "*Diversity and Transparency for ECC*", in the NIST Workshop on Elliptic Curve Cryptography Standards, 2015. IACR ePrint 2015/659.

[6] P.-A. Fouque, A. Joux and M. Tibouchi, "*Injective Encodings to Elliptic Curves*", in ACISP (2013), C. Boyd and L. Simpson, Eds., vol. 7959 of LNCS, Springer, pp. 203–218.

[7] M. Tibouchi, "*Elligator Squared: Uniform Points on Elliptic Curves of Prime Order as Uniform Random Strings*". IACR ePrint 2014/043.

A Equations

In this appendix we provide congruence equations for elliptic curves on all forms previously mentioned as a matter of convenience. Furthermore, we provide the equations used for standard point compression on curves of these forms.

Below it is assumed that $(x_1, y_1), \ldots, (x_4, y_4)$ are four public keys selected uniformly at random from $\langle G \rangle$ or from E.

- Elliptic curves $E(\mathbb{F}_p)$ on short Weierstrass form $y^2 = x^3 + ax + b$:

$$(x_3 - x_4)\left[(y_1^2 - y_2^2) - (x_1^3 - x_2^3)\right] - (x_1 - x_2)\left[(y_3^2 - y_4^2) - (x_3^3 - x_4^3)\right] \equiv 0$$

  It is assumed that $p > 3$ is a prime. This equation also hold for elliptic curves $E(\mathbb{F}_{p^n})$ but such curves are not common in practical applications.

- Elliptic curves $E(\mathbb{F}_{2^n})$ on short Weierstrass form $y^2 + xy = x^3 + ax^2 + b$:

$$\begin{aligned}(x_1^2 - x_2^2)\left[(y_3^2 - y_4^2) + (x_3 y_3 - x_4 y_4) - (x_3^3 - x_4^3)\right] - \\ (x_3^2 - x_4^2)\left[(y_1^2 - y_2^2) + (x_1 y_1 - x_2 y_2) - (x_1^3 - x_2^3)\right] \equiv 0\end{aligned}$$

  Curves on this form do not admit standard point compression. Alternative point compression techniques exist.

- Elliptic curves $E(\mathbb{K})$ on Montgomery form $by^2 = x^3 + ax^2 + x$:

$$\begin{aligned}(x_3^2 y_4^2 - x_4^2 y_3^2)\left[(x_1^3 y_2^2 - x_2^3 y_1^2) + (x_1 y_2^2 - x_2 y_1^2)\right] - \\ (x_1^2 y_2^2 - x_2^2 y_1^2)\left[(x_3^3 y_4^2 - x_4^3 y_3^2) + (x_3 y_4^2 - x_4 y_3^2)\right] \equiv 0\end{aligned}$$

  For point compression rewrite the equation as $y^2 = (x^3 + ax^2 + x)/b$.

- Elliptic curves $E(\mathbb{K})$ on Edwards form $x^2 + y^2 = c^2(1 + dx^2y^2)$:

$$\begin{aligned}(x_3^2 y_3^2 - x_4^2 y_4^2)\left[(x_1^2 - x_2^2) + (y_1^2 - y_2^2)\right] - \\ (x_1^2 y_1^2 - x_2^2 y_2^2)\left[(x_3^2 - x_4^2) + (y_3^2 - y_4^2)\right] \equiv 0\end{aligned}$$

  For point compression rewrite the equation as $y^2 = (c^2 - x^2)/(1 - dc^2x^2)$.

- Elliptic curves $E(\mathbb{K})$ on twisted Edwards form $ax^2 + y^2 = 1 + dx^2y^2$:

$$\begin{aligned}x_3^2 x_4^2 (y_3^2 - y_4^2)\left[(x_2^2 y_1^2 - x_1^2 y_2^2) - (x_2^2 - x_1^2)\right] - \\ x_1^2 x_2^2 (y_1^2 - y_2^2)\left[(x_4^2 y_3^2 - x_3^2 y_4^2) - (x_4^2 - x_3^2)\right] \equiv 0\end{aligned}$$

  For point compression rewrite the equation as $y^2 = (1 - ax^2)/(1 - dx^2)$.

It is possible to develop similar equations for other curves, including for example the generalized Weierstrass form. However, such forms are not common in practical applications so we do not list these equations.