

Information technology — Security techniques — Cryptographic technique based on elliptic curves

Part 5: Elliptic curve generation

ICS 35.040

NO COPYING WITHOUT BSI PERMISSION EXCEPT AS PERMITTED BY COPYRIGHT LAW

Copyright British Standards Institution
Provided by IHS under license with BSI - Uncontrolled Copy
No reproduction or networking permitted without license from IHS

Not for Resale

National foreword

This British Standard is the UK implementation of ISO/IEC 15946-5:2009.

The UK participation in its preparation was entrusted to Technical Committee IST/33, IT - Security techniques.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 28 February 2010.

© BSI 2010

ISBN 978 0 580 57707 9

Amendments/corrigenda issued since publication

Date	Comments

INTERNATIONAL
STANDARD

BS ISO/IEC 15946-5:2009
ISO/IEC
15946-5

First edition
2009-12-15

**Information technology — Security
techniques — Cryptographic techniques
based on elliptic curves —**

Part 5:
Elliptic curve generation

*Technologies de l'information — Techniques de sécurité — Techniques
cryptographiques fondées sur les courbes elliptiques —*

Partie 5: Génération de courbes elliptiques

Reference number
ISO/IEC 15946-5:2009(E)



PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2009

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword	iv
Introduction.....	v
1 Scope.....	1
2 Normative reference(s)	1
3 Terms and definitions	1
4 Notation and conversion functions.....	2
4.1 Notation	2
4.2 Conversion functions.....	3
5 Framework for elliptic curve generation.....	3
5.1 Types of trusted elliptic curve	3
5.2 Overview of elliptic curve generation.....	4
6 Verifiably Pseudo-Random Elliptic curve generation.....	4
6.1 Constructing Verifiably Pseudo-Random Elliptic Curves (prime case).....	4
6.1.1 Construction algorithm.....	4
6.1.2 Test for Near Primality	5
6.1.3 Finding a Point of Large Prime Order	6
6.1.4 Verification of Elliptic Curve Pseudo-Randomness	6
6.2 Constructing Verifiably Pseudo-Random Elliptic Curves (binary case).....	7
6.2.1 Construction algorithm.....	7
6.2.2 Verification of Elliptic Curve Pseudo-Randomness	8
7 Constructing Elliptic Curves by Complex Multiplication	9
7.1 General Construction (prime case)	9
7.2 MNT curve (Miyaji-Nakabayashi-Takano curve).....	10
7.3 BN curve (Barreto-Naehrig curve)	11
7.4 F curve (Freeman curve).....	12
7.5 CP curve (Cocks-Pinch curve).....	13
8 Constructing Elliptic Curves by Lifting.....	14
Annex A (informative) Background information on elliptic curves	16
Annex B (informative) Background Information on elliptic curve cryptosystems.....	18
Annex C (informative) Numerical examples.....	21
Annex D (informative) Summary of properties of Elliptic Curves generated by a Complex Multiplication method	29
Bibliography.....	30

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 15946-5 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT Security techniques*.

ISO/IEC 15946 consists of the following parts, under the general title *Information technology — Security techniques — Cryptographic techniques based on elliptic curves*:

- *Part 1: General*
- *Part 5: Elliptic curve generation*

Introduction

Some of the most interesting alternatives to the RSA and $F(p)$ based systems are cryptosystems based on elliptic curves defined over finite fields. The concept of an elliptic curve based public-key cryptosystem is rather simple:

- Every elliptic curve over a finite field is endowed with an addition operation “+”, under which it forms a finite abelian group.
- The group law on elliptic curves extends in a natural way to a “discrete exponentiation” on the point group of the elliptic curve.
- Based on the discrete exponentiation on an elliptic curve, one can easily derive elliptic curve analogues of the well-known public-key schemes of Diffie-Hellman and ElGamal type.

The security of such a public-key system depends on the difficulty of determining discrete logarithms in the group of points of an elliptic curve. This problem is — with current knowledge — much harder than the factorization of integers or the computation of discrete logarithms in a finite field. Indeed, since Miller and Koblitz in 1985 independently suggested the use of elliptic curves for public-key cryptographic systems, the elliptic curve discrete logarithm problem has only been shown to be solvable in certain specific, and easily recognizable, cases. There has been no substantial progress in finding an efficient method for solving the elliptic curve discrete logarithm problem on arbitrary elliptic curves. Thus, it is possible for elliptic curve based public-key systems to use much shorter parameters than the RSA system or the classical discrete logarithm based systems that make use of the multiplicative group of a finite field. This yields significantly shorter digital signatures and system parameters.

This part of ISO/IEC 15946 describes elliptic curve generation techniques useful for implementing the elliptic curve based mechanisms defined in ISO/IEC 9796-3, ISO/IEC 11770-3, ISO/IEC 14888-3, and ISO/IEC 18033-2.

It is the purpose of this part of ISO/IEC 15946 to meet the increasing interest in elliptic curve based public-key technology by describing elliptic curve generation methods to support key-exchange, key-transport and digital signatures based on an elliptic curve.

© ISO/IEC 2009 – All rights reserved

Information technology — Security techniques — Cryptographic techniques based on elliptic curves —

Part 5: Elliptic curve generation

1 Scope

ISO/IEC 15946 specifies public-key cryptographic techniques based on elliptic curves.

This part of ISO/IEC 15946 defines elliptic curve generation techniques useful for implementing the elliptic curve based mechanisms defined in ISO/IEC 9796-3, ISO/IEC 11770-3, ISO/IEC 14888-3 and ISO/IEC 18033-2.

The scope of this part of ISO/IEC 15946 is restricted to cryptographic techniques based on elliptic curves defined over finite fields of prime power order (including the special cases of prime order and characteristic two). The representation of elements of the underlying finite field (i.e. which basis is used) is outside the scope of this part of ISO/IEC 15946.

ISO/IEC 15946 does not specify the implementation of the techniques it defines. Interoperability of products complying with ISO/IEC 15946 will not be guaranteed.

2 Normative reference(s)

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 15946-1, *Information technology — Security techniques — Cryptographic techniques based on elliptic curves — Part 1: General*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1

definition field of an elliptic curve

field that includes all the coefficients of the equation describing an elliptic curve

3.2

elliptic curve

cubic curve without a singular point

NOTE 1 A definition of a cubic curve is given in [29].

NOTE 2 The set of points of E under a certain addition law forms an abelian group. In this part of ISO/IEC 15946, we only deal with finite fields F as the definition field. When we describe the definition field F of an elliptic curve E explicitly, we denote the curve as E/F .

NOTE 3 A detailed definition of an elliptic curve is given in Clause 4.

[ISO/IEC 15946-1:2008]

3.3

finite field

field containing a finite number of elements

NOTE 1 A definition of field is given in [29].

NOTE 2 For any positive integer m and a prime p , there exists a finite field containing exactly p^m elements. This field is unique up to isomorphism and is denoted by $F(p^m)$, where p is called the characteristic of $F(p^m)$.

[ISO/IEC 15946-1:2008]

3.4

hash-function

function which maps strings of bits to fixed-length strings of bits, satisfying the following two properties:

- for a given output, it is computationally infeasible to find an input which maps to this output;
- for a given input, it is computationally infeasible to find a second input which maps to the same output.

[ISO/IEC 10118-1]

NOTE 1 Computational feasibility depends on the specific security requirements and environment.

NOTE 2 For the purposes of this document, the recommended hash-functions are those defined in ISO/IEC 10118-2 and ISO/IEC 10118-3.

3.5

nearly prime number

positive integer $n = m \cdot r$, where m is a large prime number and r is a small smooth integer

NOTE The meaning of the terms large and small prime numbers is dependent on the application, and is based on bounds determined by the designer.

3.6

order of an elliptic curve $E(F)$

number of points on an elliptic curve E defined over a finite field F

3.7

smooth integer

integer r whose prime factors are all small (i.e. less than some defined bound)

4 Notation and conversion functions

4.1 Notation

In this part of ISO/IEC 15946, the following notation is used to describe public-key systems based on elliptic curve technology.

B An embedding degree, the smallest B such that $\#E(F(q)) \mid q^B - 1$.

E An elliptic curve, given by an equation of the form $Y^2 = X^3 + aX + b$ over the field $F(p^m)$ for $p > 3$, by an equation of the form $Y^2 + XY = X^3 + aX^2 + b$ over the field $F(2^m)$, or by an equation of the form $Y^2 = X^3 + aX^2 + b$ over the field $F(3^m)$, together with an extra point O_E referred to as the point at infinity. The elliptic curve is denoted by $E/F(p^m)$, $E/F(2^m)$, or $E/F(3^m)$, respectively.

NOTE 1 In applications not based on a pairing, $E/F(p)$ or $E/F(2^m)$ is preferable from an efficiency point of view. In applications that use a pairing, $E/F(p)$ or $E/F(3^m)$ is preferable from an efficiency point of view.

#E(F(q)) The order (or cardinality) of $E(F(q))$.

n A prime divisor of $\#E(F(q))$.

N The number of points on an elliptic curve E over $F(q)$, $\#E(F(q))$.

r The cofactor, that is $\#E(F(q)) = rn$.

4.2 Conversion functions

For the purposes of this part of ISO/IEC 15946, the following conversion functions, defined in ISO/IEC 15946-1:2008, are used.

BS2IP The bit string to integer conversion primitive

BS2OSP The bit string to octet string conversion primitive

EC2OSP_E The elliptic curve point to octet string conversion primitive

FE2IP_F The finite field element to integer conversion primitive

FE2OSP_F The finite field element to octet string conversion primitive

I2BSP The integer to bit string conversion primitive

I2OSP The integer to octet string conversion primitive

I2ECP The integer to elliptic curve conversion primitive

OS2BSP The octet string to bit string conversion primitive

OS2FEP_F The octet string to finite field element conversion primitive

OS2ECP_E The octet string to elliptic curve point conversion primitive

OS2IP The octet string to integer conversion primitive

5 Framework for elliptic curve generation

5.1 Types of trusted elliptic curve

There are a number of ways in which a user can obtain trust in the provenance of an elliptic curve, including the following.

- The curve may be obtained from an impartial trusted source (e.g. an international or national standard).
- The curve may be generated and/or verified by a trusted third party.

- The curve may be generated and/or verified by the user.

5.2 Overview of elliptic curve generation

There are three main ways to generate elliptic curves.

- Generate an elliptic curve by applying the order counting algorithms to a (pseudo-)randomly chosen elliptic curve. Such a technique is specified in Clause 6.
- Generate an elliptic curve by applying the complex multiplication method. Such a technique is specified in Clause 7.
- Generate an elliptic curve by lifting an elliptic curve over a small finite field to that over a reasonably large field. Such a technique is specified in Clause 8.

6 Verifiably Pseudo-Random Elliptic curve generation

6.1 Constructing Verifiably Pseudo-Random Elliptic Curves (prime case)

6.1.1 Construction algorithm

The following algorithm produces a set of elliptic curve parameters over a field $F(p)$ selected (pseudo-)randomly from the curves of appropriate order, along with sufficient information for others to verify that the curve was indeed chosen pseudo-randomly.

NOTE 1 The algorithm is consistent with [16].

It is assumed that the following quantities have been chosen:

- lower bound n_{\min} for the order of the base point.
- a cryptographic hash function H with output length L_{Hash} bits.
- the bit length L of inputs to H , satisfying $L \geq L_{\text{Hash}}$.

The following notation is adopted below:

- $v = \lceil \log_2 p \rceil$,
- $s = \lfloor (v - 1)/L_{\text{Hash}} \rfloor$,
- $w = v - sL_{\text{Hash}} - 1$.

Input: a prime number p ; lower bound n_{\min} for n ; a trial division bound l_{\max} .

Output: a bit string X ; EC parameters a , b , n , and G .

- Choose an arbitrary bit string X of bit length L .
- Compute $h = H(X)$.
- Let W_0 be the bit string obtained by taking the w rightmost bits of h .
- Convert X to an integer $Z = \text{BS2IP}(X)$.

- e) For i from 1 to s do:
 - 1) Convert the integer $(Z + i) \bmod 2^L$ to a length- L bit string X_i using I2BSP.
 - 2) Compute $W_i = H(X_i)$.
- f) Let W be the bit string obtained by the concatenation of W_0, W_1, \dots, W_s as follows:

$$W = W_0 \parallel W_1 \parallel \dots \parallel W_s.$$

- g) Convert W to a finite field element $c = \text{OS2FEP}(\text{BS2OSP}(W))$.
 - h) If $c = 0_F$ or $4c + 27 = 0_F$, then go to Step a).
 - i) Choose finite field elements $a, b \in F(p)$ such that $b \neq 0_F$ and $cb^2 - a^3 = 0_F$.
- NOTE 2 The simplest choice is $a = c$ and $b = c$. However, an implementer may want to choose differently for performance reasons.
- j) Compute the order $\#E(F(p))$ of the elliptic curve E over $F(p)$ given by $y^2 = x^3 + ax + b$.
 - k) Test whether $\#E(F(p))$ is a nearly prime number using the algorithm specified in 6.1.2. If so, the output of the algorithm specified in 6.1.2 consists of the integers r, n . If not, then go to Step a).
 - l) Check $E(F(p))$ satisfies the MOV-condition specified in B.2.3, that is the smallest integer B such that n divides $q^B - 1$ ensures the desirable security level. If not, then go to Step a).
 - m) Test whether $\#E(F(p)) \neq p$ in order to be secure against the attack specified in B.2.2. If not, then go to Step a).
 - n) Test whether the prime divisor n satisfies the condition described in B.2.4 for cryptosystems based on ECDLP, ECDHP, or BDHP with auxiliary inputs as in B.1.5. If not, then go to Step a).
 - o) Generate a point G on E of order n using the algorithm specified in 6.1.3.
 - p) Output X, a, b, n, G .

NOTE 3 The necessity of near primality is described in B.2.2.

NOTE 4 Methods to compute the order $\#E(F(p))$ are described in [5], [26] and [29].

6.1.2 Test for Near Primality

Given a lower bound n_{\min} and a trial division bound l_{\max} , the following procedures test $N = \#E(F(p))$ for near primality.

Input: positive integers N, l_{\max} , and n_{\min} .

Output: if N is nearly prime, output a prime n with $n_{\min} \leq n$ and a smooth integer r such that $N = rn$. If N is not nearly prime, output the message "not nearly prime".

- a) Set $n = N, r = 1$.
- b) For l from 2 to l_{\max} do
 - 1) If l is composite then go to Step 3).
 - 2) While (l divides n)

- Set $n = n/l$ and $r = rl$.
- If $n < n_{\min}$ then output “not nearly prime” and stop.

3) Next l .

- c) Test n for primality.
- d) If n is prime then output r and n and stop.
- e) Output “not nearly prime”.

NOTE Methods to test for primality are described in [3] and [4].

6.1.3 Finding a Point of Large Prime Order

If the order $\#E(F(q))$ of an elliptic curve E is nearly prime, the following algorithm efficiently produces a random point in $E(F(q))$ whose order is the large prime factor n of $\#E(F(q)) = rn$.

Input: an elliptic curve E over the field $F(q)$, a prime n , and a positive integer r not divisible by n .

Output: if $\#E(F(q)) = rn$, a point G on E of order n ; if not, the message “wrong order.”

- a) Generate a random point P (not O_E) on E .
- b) Set $G = rP$.
- c) If $G = O_E$ then go to Step a).
- d) Set $Q = nG$.
- e) If $Q \neq O_E$ then output “wrong order” and stop.
- f) Output G .

6.1.4 Verification of Elliptic Curve Pseudo-Randomness

The following algorithm determines whether an elliptic curve over $F(p)$ was generated using the method of 6.1.1. The quantities L_{Hash} , L , v , s , and w , and the hash function H , are as in 6.1.1.

Input: a bit string X of length L , EC parameters $q = p$, a , b , n , and $G = (x_G, y_G)$, and a positive integer n_{\min} .

Output: “True” or “False”.

- a) Compute $h = H(X)$.
- b) Let W_0 be the bit string obtained by taking the w rightmost bits of h .
- c) Convert X to an integer $Z = \text{BS2IP}(X)$.
- d) For i from 1 to s do:
 - 1) Compute $Z = Z + i \bmod 2^L$.
 - 2) Convert $Z \bmod (2^L)$ to a bit string $X_i = \text{I2BSP}(Z)$.
 - 3) Compute $W_i = H(X_i)$.

e) Let W be the bit string obtained by the concatenation of W_0, W_1, \dots, W_s as follows:

$$W = W_0 \parallel W_1 \parallel \dots \parallel W_s.$$

f) Convert W to a finite field element $c = \text{OS2FEP}(\text{BS2OSP}(W))$.

g) Verify the following conditions.

- $n \geq n_{\min}$
- n is a prime.
- $c \neq 0_F$
- $4c + 27 \neq 0_F$.
- $b \neq 0_F$
- $cb^2 - a^3 = 0_F$.
- $G \neq O_E$
- $y_G^2 = x_G^3 + ax_G + b$.
- $nG = O_E$

h) If all the conditions in Step g) hold, then output “True”; otherwise output “False”.

6.2 Constructing Verifiably Pseudo-Random Elliptic Curves (binary case)

6.2.1 Construction algorithm

The following algorithm produces a set of elliptic curve parameters for a pseudo-random curve over a field $F(2^m)$, along with sufficient information for others to verify that the curve was indeed chosen pseudo-randomly.

NOTE 1 The algorithm is consistent with [16].

It is assumed that the following quantities have been chosen:

- a field $F(2^m)$
- a lower bound n_{\min} for the order of the base point
- a cryptographic hash function H with output length L_{Hash} bits
- the bit length L of inputs to H , satisfying $L \geq L_{\text{Hash}}$.

The following notation is adopted below:

- $s = \lfloor (m - 1) / L_{\text{Hash}} \rfloor$,
- $w = m - sL_{\text{Hash}}$.

Input: a field $F(2^m)$; a lower bound n_{\min} for n ; a trial division bound l_{\max} .

Output: a bit string X ; EC parameters a, b, n , and G .

- a) Choose an arbitrary bit string X of bit length L .
- b) Compute $h = H(X)$.
- c) Let W_0 be the bit string obtained by taking the w rightmost bits of h .
- d) Convert the length- L bit string X to an integer z using BS2IP.
- e) For i from 1 to s do:
 - Convert the integer $(z + i) \bmod (2^L)$ to a length- L bit string X_i using I2BSP.
 - Compute $W_i = H(X_i)$.
- f) Let W be the bit string obtained by the concatenation of W_0, W_1, \dots, W_s as follows:
$$W = W_0 \parallel W_1 \parallel \dots \parallel W_s.$$
- g) Convert the length- m bit string W to a field element b using BS2OSP and OS2FEP.
- h) If $b = 0_F$, then go to Step a).
- i) Let a be an arbitrary element in $F(2^m)$. (The simplest choice is $a = 0_F$. However, one may want to choose differently for performance reasons.)
- j) Compute the order $\#E(F(2^m))$ of the elliptic curve E over $F(2^m)$ given by $y^2 + xy = x^3 + ax^2 + b$.
- k) Test whether $\#E(F(2^m))$ is a nearly prime number using the algorithm specified in 6.1.2. If so, the output of the algorithm specified in 6.1.2 consists of the integers r, n . If not, then go to Step a).
- l) Check $E(F(2^m))$ satisfies the MOV-condition specified in B.2.3. If not, then go to Step a).
- m) Test whether the prime divisor n satisfies the condition described in B.2.4 for cryptosystems based on ECDLP, ECDHP, or BDHP with auxiliary inputs as in B.1.5. If not, then go to Step a).
- n) Generate a point G on E of order n using the algorithm specified in 6.1.3.
- o) Output X, a, b, n, G .

NOTE 2 The necessity of near primality is described in B.2.2.

NOTE 3 Methods of computing the order $\#E(F(2^m))$ are described in [5], [26] and [29].

6.2.2 Verification of Elliptic Curve Pseudo-Randomness

The following algorithm verifies the validity of a set of elliptic curve parameters. In addition, it determines whether an elliptic curve over $F(2^m)$ was generated using the method of 6.2.1.

The quantities L_{Hash}, L, s , and w , and the hash function H , are as in 6.2.1.

Input: a bit string X of length L , EC parameters $q = 2^m, a, b, n$, and $G = (x_G, y_G)$, and a positive integer n_{min} .

Output: "True" or "False".

- a) Compute $h = H(X)$.
- b) Let W_0 be the bit string obtained by taking the w rightmost bits of h .
- c) Convert the bit string X to an integer z via BS2IP.

- d) For i from 1 to s do:
- 1) Convert the integer $(z + i) \bmod (2^L)$ to a length- L bit string X_i via I2BSP.
 - 2) Compute $W_i = H(X_i)$.
- e) Let W be the bit string obtained by the concatenation of W_0, W_1, \dots, W_s as follows:

$$W = W_0 \parallel W_1 \parallel \dots \parallel W_s$$

- f) Convert the length- m bit string W to the field element b' via BS2OSP and OS2FEP.
- g) Verify the following conditions.

- $n \geq n_{\min}$
- n is a prime.
- $b \neq 0_F$
- $b = b'$
- $G \neq O_E$
- $y_G^2 + x_G y_G = x_G^3 + ax_G^2 + b$
- $nG = O_E$

- h) If all the conditions in Step g) hold, then output "True"; otherwise output "False".

7 Constructing Elliptic Curves by Complex Multiplication

7.1 General Construction (prime case)

The following algorithm produces an elliptic curve E over $F(p)$ with the given number of rational points N .

NOTE 1 The algorithm is based on [11], which is applied to primality proving [4].

Input: The definition field $F(p)$ and the number of points $N = rn$, where n is the largest prime divisor of N and r is a cofactor.

Output: curve parameters of elliptic curve E with $\#E(F(p)) = N$ and base point G

- a) Test whether the prime divisor n satisfies the condition described in B.2.4 for cryptosystems based on ECDLP, ECDHP, or BDHP with auxiliary inputs as in B.1.5. If not, then execute a new input.
- b) Set $t = p + 1 - N$.
- c) Choose a pair of integers (D, V) such that $4p - t^2 = DV^2$.
- d) Construct the Hilbert class polynomial $P_D(X)$.
- e) Find a solution j_0 in $F(p)$ of $P_D(X) = 0$ modulo p .
- f) Choose $c \in F(p)^*$ and construct an elliptic curve over $F(p)$ with the j -invariant j_0 .
 - $E_{D,j_0,c}: y^2 = x^3 + (3c^2j_0 / (1728 - j_0))x + 2c^3j_0 / (1728 - j_0)$ (if $j_0 \neq 0_F, 1728$).

- $E_{D,j_0,c}: y^2 = x^3 + c$ (if $j_0 = 0_F$).
 - $E_{D,j_0,c}: y^2 = x^3 + cx$ (if $j_0 = 1728$).
- g) Construct a random point G on $E_{D,j_0,c}(F(p))$ such that $G \neq O_E$ and $r \cdot G \neq O_E$.
- h) Set $G = r \cdot G$.
- i) If $n \cdot G = O_E$, output curve parameters of $E_{D,j_0,c}$ and the base point G . If $n \cdot G \neq O_E$, go to Step f) to choose another c .

NOTE 2 Any pair of integers (D,V) such that $4p - t^2 = DV^2$ can be used in Step c).

NOTE 3 The definition of the Diophantine equation used in Step c) is given in A.5.

NOTE 4 The definition of the Hilbert class polynomial $P_D(X)$ is given in A.2.

7.2 MNT curve (Miyaji-Nakabayashi-Takano curve)

The following algorithm produces an elliptic curve E over $F(p)$ with the embedding degree $B = 6$, which is useful for cryptosystems based on a bilinear pairing. The pairing and the embedding degree are described in A.3 and B.2.2, respectively.

NOTE 1 Detailed information is given in [20].

Input: lower and upper bound (odd integer) p_{\min} and p_{\max} for the definition field (in bits) and upper bound D_{\max} for size of D .

Output: prime p , curve parameters of elliptic curve $E/F(p)$, the order $n = \#E(F(p))$, and basepoint G .

- a) Choose a small positive integer $D < D_{\max}$ such that $D \equiv 3 \pmod{8}$ and go to Step c).
- b) If such D does not exist, then stop and output "fail".
- c) Find a pair of integers (T,U) with the smallest $U > 0$ that satisfies $T^2 - 3DU^2 = \pm 1$ using the continued fraction algorithm.
- d) Find a pair of integers (x, y) that satisfies $x^2 - 3Dy^2 = -8$ and

$$0 \leq x < 2U\sqrt{2D}, 2\sqrt{2/D} \leq y < 2T\sqrt{2/D},$$

using the algorithm of Lagrange. If not, go to Step a).

- e) $i = 0$.
- f) Find a pair of primes (p,n) as follows:
 - 1) Compute integers x_i and y_i such that $x_i + y_i\sqrt{3D} = (x + y\sqrt{3D})(T + U\sqrt{3D})^i$.
 - 2) If $x_i \equiv 1 \pmod{6}$, then $s = (x_i - 1)/6$ and $p = 4s^2 - 1$;
 - else if $x_i \equiv -1 \pmod{6}$, then $s = (x_i + 1)/6$ and $p = 4s^2 - 1$;
 - else, $i = i + 1$ and go to Step 1).
 - 3) If $p < p_{\min}$, then $i = i + 1$ and go to Step 1).
 - 4) If $p > p_{\max}$, then go to Step a).

- 5) If p is prime, then $n_1 = 4s^2 + 2s + 1$ and $n_2 = 4s^2 - 2s + 1$;
 – else, $i = i + 1$ and go to Step 1).
- 6) If n_1 or n_2 prime, then $n = n_1$ or $n = n_2$, respectively and go to Step g);
 – else, $i = i + 1$ and go to Step 1).
- g) Test whether the prime divisor n satisfies the condition described in B.2.4 for cryptosystems based on ECDLP, ECDHP, or BDHP with auxiliary inputs as in B.1.5. If not, then go to Step a).
- h) Construct the Hilbert class polynomial $P_D(X)$.
- i) Find a solution j_0 in $F(p)$ of $P_D(X) = 0$ modulo p .
- j) Choose $c \in F(p)^*$ and construct an elliptic curve over $F(p)$ with the j -invariant j_0 .
 – $E_{D,j_0,c}: y^2 = x^3 + (3c^2j_0 / (1728 - j_0))x + 2c^3j_0 / (1728 - j_0)$ (if $j_0 \neq 0_F, 1728$).
 – $E_{D,j_0,c}: y^2 = x^3 + c$ (if $j_0 = 0_F$).
 – $E_{D,j_0,c}: y^2 = x^3 + cx$ (if $j_0 = 1728$).
- k) Construct a random point G on $E_{D,j_0,c}(F(p))$, not equal to the point at infinity O_E .
- l) If $n \cdot G = O_E$, output p , E , n , and G .
- m) Else, go to Step h) to choose another $c \in F(p)^*$.

NOTE 2 The definition of the Hilbert class polynomial $P_D(X)$ is given in A.2.

NOTE 3 The continued fraction algorithm in Step c) is given in A.3 and [23].

NOTE 4 The algorithm of Lagrange in Step d) is given in A.4, [18] and [21].

NOTE 5 A technique for speeding up a protocol based on a bilinear pairing is described in [6].

7.3 BN curve (Barreto-Naehrig curve)

The following algorithm produces an elliptic curve E over $F(p)$ with the embedding degree $B = 12$, which is useful for cryptosystems based on bilinear pairings. The embedding degree is described in B.2.2.

NOTE 1 Detailed information is given in [6].

Input: the approximate desired size m of the curve order (in bits) and upper bound (odd integer) p_{\max} for the definition field

Output: prime p , curve parameters of elliptic curve $E/F(p)$, the order $n = \#E(F(p))$, and basepoint G .

- a) Let $P(u) = 36u^4 + 36u^3 + 24u^2 + 6u + 1$.
- m. b) Compute the smallest $u \approx 2^{m/4}$ such that $\lceil \log_2 P(-u) \rceil =$
- c) While $p \leq p_{\max}$
- 1) $t = 6u^2 + 1$.
 - 2) $p = P(-u)$ and $n = p + 1 - t$.

- 3) If p and n are prime then go to Step e).
 - 4) $p = P(u)$ and $n = p + 1 - t$.
 - 5) If p and n are prime, then go to Step e).
 - 6) $u = u + 1$ and go to Step 1).
- d) Stop and output "fail".
 - e) Test whether the prime divisor n satisfies the condition described in B.2.4 for cryptosystems based on ECDLP, ECDHP, or BDHP with auxiliary inputs as in B.1.5. If not, then go to Step a).
 - f) $b = 0$.
 - g) If $b + 1$ is not represented by $b + 1 = y_0^2$ modulo p for an integer y_0 , then $b = b + 1$ and go to Step g).
 - h) Set an elliptic curve $E: y^2 = x^3 + b$.
 - i) Compute a square root $y_0 = \sqrt{(b + 1)}$ modulo p .
 - j) Set the basepoint $G = (1, y_0) \in E$.
 - k) If $n \cdot G \neq O_E$, then set $b = b + 1$ and go to Step g).
 - l) Output p , E , n , and G .

NOTE 2 A technique for speeding up a protocol based on a bilinear pairing is described in [6].

7.4 F curve (Freeman curve)

The following algorithm produces an elliptic curve E over $F(p)$ with embedding degree $B = 10$, which is useful for cryptosystems based on a bilinear pairing. The embedding degree is described in B.2.2.

NOTE 1 Detailed information is given in [13].

Input: lower and upper bound p_{\min} and p_{\max} for the size of the definition field (in bits) and upper bound D_{\max} for size of D .

Output: prime p , curve parameters of elliptic curve $E/F(p)$, the order $n = \#E(F(p))$, and basepoint G .

- a) Choose a small positive integer $D < D_{\max}$ such that $D \equiv 43$ or $67 \pmod{120}$ and $15D$ is square-free and go to Step c).
- b) If such D does not exist, then stop and output "fail".
- c) Find a pair of integers (T, U) with the smallest $U > 0$ that satisfies $T^2 - 15DU^2 = \pm 1$ using the continued fraction algorithm.
- d) Let $d = T - U\sqrt{15D}$.
- e) Let $g = d^2$ if $T^2 - 15DU^2 = -1$, and let $g = d$ otherwise.
- f) Find a pair of integers (x, y) that satisfies $x^2 - 15Dy^2 = -20$ and $0 \leq x < 10U\sqrt{3D}$, $2\sqrt{1/(3D)} \leq y < 2T\sqrt{1/(3D)}$ using the algorithm of Lagrange.
- g) For the solution (x, y) in Step f).

- 1) If $x = \pm 5 \pmod{15}$, then:
 - Let $s = (-5 \pm x)/15$.
 - Let $p = 25s^4 + 25s^3 + 25s^2 + 10s + 3$.
 - Let $n = 25s^4 + 25s^3 + 15s^2 + 5s + 1$.
- 2) Else, go to Step f) to the next (x, y) .
- 3) If $p > p_{\max}$, go to Step f) to use the next (x, y) .
- 4) Else if $p < p_{\min}$, then go to Step 6).
- 5) If p and n are primes, go to Step h).
- 6) Find a pair of integers (x', y') such that $x' + y'\sqrt{15D} = (x + y\sqrt{15D}) \cdot g$.
- 7) Let $x = x'$ and $y = y'$ and return to Step 1).
- h) Test whether the prime divisor n satisfies the condition described in B.2.4 for cryptosystems based on ECDLP, ECDHP, or BDHP with auxiliary inputs as in B.1.5. If not, then go to Step a).
- i) Construct the Hilbert class polynomial $P_D(X)$.
- j) Find a solution j_0 in $F(p)$ of $P_D(X) = 0$ modulo p .
- k) Choose $c \in F(p)^*$ and construct an elliptic curve E over $F(p)$ with j -invariant j_0 :
 - $E_{D,j_0,c} : y^2 = x^3 + (3c^2j_0 / (1728 - j_0))x + 2c^3j_0 / (1728 - j_0)$ (if $j_0 \neq 0_F, 1728$).
 - $E_{D,j_0,c} : y^2 = x^3 + c$ (if $j_0 = 0_F$).
 - $E_{D,j_0,c} : y^2 = x^3 + cx$ (if $j_0 = 1728$).
- l) Construct a random point G on $E_{D,j_0,c}(F(p))$, not equal to the point at infinity O_E .
- m) If $n \cdot G = O_E$, output p, E, n , and G .
- n) Else, go to Step k) to choose another $c \in F(p)^*$.

NOTE 2 The definition of the Hilbert class polynomial $P_D(X)$ in Step i) is given in A.2.

NOTE 3 The continued fraction algorithm in Step c) is given in A.3 and [23].

NOTE 4 The algorithm of Lagrange in Step f) is given in A.4, [18] and [21].

NOTE 5 A technique to speed up a protocol based on a bilinear pairing is described in [6].

7.5 CP curve (Cocks-Pinch curve)

The following algorithm produces an elliptic curve E over $F(p)$ with arbitrary embedding degree B , which is useful for cryptosystems based on a bilinear pairing. The embedding degree is described in B.2.2.

NOTE 1 Detailed information is given in [7].

Input: a positive integer B and a set R of prime numbers n ($n-1$ is divisible by B).

Output: prime p , curve parameters of elliptic curve $E/F(p)$, the order $n \cdot r = \#E(F(p))$, and basepoint G .

- a) Choose a small square-free positive integer D and n in R such that $-D$ is a square modulo n .
 - 1) Find a B -th root of unity z in $F(n) \setminus \{0, 1\}$.
 - 2) $t' = z + 1$
 - 3) $y' = (t'-2) / \sqrt{-D} \pmod{n}$
 - 4) Let t be an integer such that t is equal to t' modulo n , and let y be an integer such that y is equal to y' modulo n .

NOTE 2 $t = t'$ and $y = y'$ are permitted.

- 5) $p = (t^2 + Dy^2) / 4$
- 6) If p is not prime, then go to Step a).
- b) Test whether the prime divisor n satisfies the condition described in B.2.4 for cryptosystems based on ECDLP, ECDHP, or BDHP with auxiliary inputs as in B.1.5. If not, then go to Step a).
- c) Construct the Hilbert class polynomial $P_D(X)$.
- d) Find a solution j_0 in $F(p)$ of $P_D(X) = 0$ modulo p .
- e) Choose $c \in F(p)^*$ and construct an elliptic curve over $F(p)$ with the j -invariant j_0 .
 - $E_{D,j_0,c}: y^2 = x^3 + (3c^2j_0 / (1728 - j_0))x + 2c^3j_0 / (1728 - j_0)$ (if $j_0 \neq 0_F, 1728$).
 - $E_{D,j_0,c}: y^2 = x^3 + c$ (if $j_0 = 0_F$).
 - $E_{D,j_0,c}: y^2 = x^3 + cx$ (if $j_0 = 1728$).
- f) Set a cofactor $r = (p + 1 - t) / n$.
- g) Construct a random point G on $E_{D,j_0,c}(F(p))$ such that $G \neq O_E$ and $r \cdot G \neq O_E$.
- h) Set $G = r \cdot G$.
- i) If $n \cdot G = O_E$, output n , G , and the elliptic curve E .
- j) Else, go to Step e) to choose another $c \in F(p)^*$.

NOTE 3 The definition of the Hilbert class polynomial $P_D(X)$ in Step c) is given in A.2.

NOTE 4 A technique for speeding up a protocol based on a bilinear pairing is described in [6].

8 Constructing Elliptic Curves by Lifting

The following algorithm produces an elliptic curve E over $F(p^m)$ by lifting an elliptic curve E over $F(p)$.

NOTE The algorithm is based on [29].

Input: small finite field $F(p)$, elliptic curve E over $F(p)$, lower and upper bound N_{\min} and N_{\max} for the order of elliptic curve (in bits).

Output: extension degree m , order $N_m = \#E(F(p^m))$, basepoint G , and order n of G .

- a) Count the order of $N = \#E(F(p))$, which is easily executed since $F(p)$ is small.
- b) Set $t = p + 1 - N$ and compute algebraic integers α and β that satisfy $t = \alpha + \beta$ and $p = \alpha \beta$.
- c) Set $m = 1$.
- d) Find a triple of (m, N_m, n) as follows:
 - 1) Compute $N_m = p^m + 1 - (\alpha^m + \beta^m)$ and $q = p^m$, which is an integer.
 - 2) If $N_m < N_{\min}$, then $m = m + 1$ and go to Step 1).
 - 3) If $N_m > N_{\max}$, then stop and output "fail".
 - 4) Test whether N_m is a nearly prime number using the algorithm specified in 6.1.2. If so, the output of 6.1.2 consists of the integers r and n . If not, then $m = m + 1$ and go to Step 1).
 - 5) Check whether $E(F(q))$ satisfies the MOV-condition specified in B.2.3, that is the smallest integer B such that n divides $q^B - 1$ ensures the desirable security level. If not, then $m = m + 1$ and go to Step 1).
- e) Generate a point G on $E(F(q))$ of order n using the algorithm specified in 6.1.3.
- f) Output an extension degree m , the order $N_m = \#E(F(q))$, a basepoint G and the order n .

Annex A (informative)

Background information on elliptic curves

A.1 j-invariant

Let $F(q)$ be a finite field with $q = p^m$, where prime $p > 3$. Let E be an elliptic curve over $F(q)$ given by the short Weierstrass equation,

$$Y^2 = X^3 + aX + b \text{ with } a, b \in F(q),$$

where the inequality $4a^3 + 27b^2 \neq 0_F$ holds in $F(q)$. Then the j-invariant is defined as

$$j = 1728 \cdot (4a^3) / (4a^3 + 27b^2).$$

Let $F(2^m)$, for some $m \geq 1$, be a finite field. Let E be an elliptic curve over $F(2^m)$ given by the equation

$$Y^2 + XY = X^3 + aX + b \text{ with } a, b \in F(2^m)$$

where $b \neq 0_F$. Then the j-invariant is defined as

$$j = 1/b.$$

Let $F(3^m)$, for some $m \geq 1$, be a finite field. Let E be an elliptic curve over $F(3^m)$ given by the equation

$$Y^2 = X^3 + aX^2 + b \text{ with } a, b \in F(3^m)$$

such that $a, b \neq 0_F$. Then the j-invariant is defined as

$$j = -a^3/b.$$

A.2 Hilbert class polynomial

The construction of elliptic curves by complex multiplication uses the theory of imaginary quadratic fields $Q(\sqrt{-D})$. In the case of the imaginary quadratic field $Q(\sqrt{-D})$, the Hilbert class field K is the extension field of $Q(\sqrt{-D})$, which is the unramified abelian extension of $Q(\sqrt{-D})$. The Hilbert class polynomial $P_D(X)$ is defined by the minimum polynomial of K over $Q(\sqrt{-D})$. In the construction of elliptic curves by complex multiplication, the fact that the j-invariants of elliptic curves $E/F(p)$ are given as a solution of a Hilbert class polynomial $P_D(X)$ modulo p is used.

NOTE 1 These facts are described in [7] and [10].

NOTE 2 Online databases of Hilbert class polynomials are available in [17].

A.3 Cryptographic pairing

A cryptographic pairing e_n satisfies the conditions of non-degeneracy, bilinearity, and computability. A pairing e_n is defined over $\langle G_1 \rangle \times \langle G_2 \rangle$ as follows,

$$e_n : \langle G_1 \rangle \times \langle G_2 \rangle \rightarrow \mu_n,$$

where $\langle G_1 \rangle$ and $\langle G_2 \rangle$ are the cyclic groups of order n and μ_n is the cyclic group of the n -th roots of unity. A pairing e_n is realized by restricting the domain of the Weil or Tate pairings.

A.4 Pell Equation

The Pell equation is of the form

$$T^2 - dU^2 = \pm 1,$$

where d is a fixed integer. In the construction of elliptic curves by complex multiplication, the Pell equation with a positive integer d that is not a perfect square is used. Then all positive integer solutions of (T,U) are given by using the least positive solution (T_0, U_0) with the smallest $U_0 > 0$ as follows:

$$T + U\sqrt{d} = (T_0 + U_0\sqrt{d})^k$$

for $k = 1, 2, \dots$

NOTE These facts are described in [24].

A.5 The Diophantine equation $x^2 - dy^2 = n$

In the construction of elliptic curves by complex multiplication, the Diophantine equation $x^2 - dy^2 = n$ is used. Here n is an integer and d is a positive integer that is not a perfect square. The number of integer solutions of this equation is zero or infinite. All positive solutions (x, y) are given by using the least positive solution (T_0, U_0) with the smallest $U_0 > 0$ of the related Pell equation of $T^2 - dU^2 = 1$.

NOTE Details are described in [24].

Annex B (informative)

Background information on elliptic curve cryptosystems

B.1 Definition of cryptographic problems

B.1.1 The elliptic curve discrete logarithm problem (ECDLP)

For an elliptic curve $E/F(q)$, the base point $G \in E(F(q))$ with order n , and a point $P \in E(F(q))$, the elliptic curve discrete logarithm problem (with respect to the base point G) is to find the integer $x \in [0, n-1]$ such that $P = xG$ if such an x exists.

The security of elliptic curve cryptosystems is based on the believed hardness of the elliptic curve discrete logarithm problem.

B.1.2 The computational elliptic curve Diffie-Hellman problem (ECDHP)

For an elliptic curve $E/F(q)$, the base point $G \in E(F(q))$ with order n , and points $aG, bG \in E(F(q))$, the computational elliptic curve Diffie-Hellman problem is to compute abG .

The security of some elliptic curve cryptosystems is based on the believed hardness of the computational elliptic curve Diffie-Hellman problem.

B.1.3 The decisional elliptic curve Diffie-Hellman problem (ECDDHP)

For an elliptic curve $E/F(q)$, the base point $G \in E(F(q))$ with order n , and points $aG, bG, Y \in E(F(q))$, the decisional elliptic curve Diffie-Hellman problem is to decide whether $Y = abG$ or not.

The security of some elliptic curve cryptosystems is based on the believed hardness of the decisional elliptic curve Diffie-Hellman problem.

B.1.4 The bilinear Diffie-Hellman problem (BDHP)

The bilinear Diffie-Hellman problems are described in two ways according to the corresponding cryptographic bilinear maps.

- For two groups $\langle G_1 \rangle$ and $\langle G_2 \rangle$ with order n , a cryptographic bilinear map $e_n : \langle G_1 \rangle \times \langle G_2 \rangle \rightarrow \mu_n$, $aG_1, bG_1 \in \langle G_1 \rangle$, and $aG_2, cG_2 \in \langle G_2 \rangle$, the bilinear Diffie-Hellman problem is to compute $e_n(G_1, G_2)^{abc}$.
- For a group $\langle G_1 \rangle$ with order n , a cryptographic bilinear map $e_n : \langle G_1 \rangle \times \langle G_1 \rangle \rightarrow \mu_n$, and $aG_1, bG_1, cG_1 \in \langle G_1 \rangle$, the bilinear Diffie-Hellman problem is to compute $e_n(G_1, G_1)^{abc}$.

The security of some elliptic curve cryptosystems is based on the believed hardness of the elliptic curve bilinear Diffie-Hellman problem.

B.1.5 The elliptic curve discrete logarithm problem with auxiliary inputs (ECDLP with auxiliary inputs)

The security of some cryptosystems is based on the elliptic curve discrete logarithm problem with auxiliary inputs. Some of them are as follows (the notation follows from the original definitions of the problems in B.1.1, B.1.2, and B.1.4).

- ECDLP with additional inputs x^2G, x^3G, \dots, x^kG
- ECDHP with additional inputs a^2G, a^3G, \dots, a^kG
- BDHP with additional inputs $a^2G_1, a^3G_1, \dots, a^kG_1$

B.2 Algorithms to determine discrete logarithms on elliptic curves

B.2.1 Security of ECDLP

The security of ECDLP depends on the selection of elliptic curves $E/F(q)$ and the size n of the order of the base point G . The size of n should be 160 bits or more to achieve the desired level of security.

This section gives the overviews of algorithms to solve ECDLP. The elliptic curve $E/F(q)$ shall be chosen to meet the defined security objectives against the following algorithms to solve ECDLP. The size of n shall be set to meet the defined security objectives against the baby-step-giant-step algorithm and various variants of the Pollard ρ algorithm.

B.2.2 Overview of algorithms

The following techniques are available to determine discrete logarithms on an elliptic curve:

- The Pohlig-Silver-Hellman algorithm. This is a “divide-and-conquer” method which reduces the discrete logarithm problem for an elliptic curve E defined over $F(q)$ to the discrete logarithm in the cyclic subgroups of prime order dividing $\#E(F(q))$.
- The baby-step-giant-step algorithm and various variants of the Pollard- ρ algorithm.

NOTE 1 The various variants of the Pollard- ρ algorithm are described in [29].

- The algorithm of Frey-Rück and the Menezes-Okamoto-Vanstone algorithm which both transform the discrete logarithm problem in a cyclic subgroup of E with prime order n to the smallest extension field $F(q^B)$ of $F(q)$ such that n divides $(q^B - 1)$, where B is called the embedding degree. The Frey-Rück algorithm runs under weaker conditions than the algorithm published by Menezes-Okamoto-Vanstone.
- The algorithm of Araki-Satoh, Smart and Semaev which solves the discrete logarithm problem for an elliptic curve E defined over $F(p^m)$ in the case $\#E(F(p^m)) = p^m$.

Unlike the situation of the discrete logarithm in the multiplicative group of some finite field there is no known “index-calculus” available in the case of elliptic curves. As for attacks using covering for special type of covers, e.g. the Weil descent attack, the GHS attack, etc. See Chapter 22 of [5].

NOTE 2 The Pohlig-Silver-Hellman and baby-step-giant-step algorithms work generally on all kinds of elliptic curves while the Frey-Rück, the Menezes-Okamoto-Vanstone, Araki-Satoh, Smart, and Semaev algorithms work only on curves with special properties.

B.2.3 The MOV-condition

Let n be as defined in the set of elliptic curve domain parameters, where n is a prime divisor of $\#E(F(q))$ and q is a power of a prime p . A value B is given as the smallest integer such that n divides $p^B - 1$. As mentioned above, the Frey-Rück and Menezes-Okamoto-Vanstone algorithms reduce the discrete logarithm problem in an elliptic curve over $F(q)$ to the discrete logarithm in the finite field $F(p^B)$ for some $B \geq 1$. By using the attack, the difficulty of the discrete logarithm problem in an elliptic curve $E/F(q)$ is related to the discrete logarithm problem in a finite field $F(p^B)$. The *subfield-adjusted MOV-condition* describes the degree B that ensures the security level of the discrete logarithm problem in an elliptic curve by the discrete logarithm problem in finite field. For some applications based on the Weil and Tate pairing, a reasonably small value of B such as 6 or more is preferable.

NOTE Information on the degree B is described in [15].

B.2.4 The condition of prime divisor n

For some cryptosystems based on ECDLP, ECDHP, or BDHP with auxiliary inputs as in B.1.5, the prime divisor n should satisfy the following conditions: there is no divisor d of n - 1 such that $(\log n)^2 < d < n^{1/2}$ and there is no divisor e of n + 1 such that $(\log n)^2 < e < n^{1/2}$. The divisors d and e are possibly composite.

NOTE Detailed information on d and e is given in [8].

Annex C (informative)

Numerical examples

C.1 Numerical examples of Verifiably Pseudo-Random Elliptic Curves

C.1.1 Introduction

Refer to [12] for this Clause. The parameters are chosen from a seed using SHA-1.

C.1.2 Elliptic curve over a prime field (192 bits)

p	ffffffff ffffffff ffffffff ffffffff e ffffffff ffffffff	$2^{192}-2^{64}-1$
Equation of E		$y^2 = x^3 + ax + b$
a	ffffffff ffffffff ffffffff ffffffff e ffffffff ffffffff c	
b	64210519 e59c80e7 0fa7e9ab 72243049 feb8deec c146b9b1	
(seed) X	3045ae6f c8422f64 ed579528 d38120ea e12196d5	
(compressed) G	03 188da80e b03090f6 7cbf20eb 43a18800 f4ff0afd 82ff1012	
(uncompressed) G	04 188da80e b03090f6 7cbf20eb 43a18800 f4ff0afd 82ff1012 07192b95 ffc8da78 631011ed 6b24cdd5 73f977a1 1e794811	
n	ffffffff ffffffff ffffffff 99def836 146bc9b1 b4d22831	
(cofactor) r		1

C.1.3 Elliptic curve over a prime field (224 bits)

p	ffffffff ffffffff ffffffff 00000000 00000000 00000001	ffffffff $2^{224}-2^{96}+1$
a	ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff e	ffffffff
b	0c04b3ab f5413256 5044b0b7 d7bfd8ba 270b3943 2355ffb4	b4050a85
(seed) X	db713447 99d5c7fc dc45cb59f a3b9ab8f 6a948bc5	

(compressed) G 02 b70e0cbd
6bb4bf7f 321390b9 4a03c1d3 56c21122 343280d6 115c1d21

(uncompressed) G 04 b70e0cbd 6bb4bf7f
321390b9 4a03c1d3 56c21122 343280d6 115c1d21 bd376388
b5f723fb 4c22dfe6 cd4375a0 5a074764 44d58199 85007e34

N ffffffff
ffffffff ffffffff ffff16a2 e0b8f03e 13dd2945 5c5c2a3d

(cofactor) r 1

C.1.4 Elliptic curve over a prime field (256 bits)

P ffffffff 00000001
00000000 00000000 00000000 ffffffff ffffffff ffffffff
 $2^{224}(2^{32}-1) + 2^{192} + 2^{96} - 1$

A ffffffff 00000001
00000000 00000000 00000000 ffffffff ffffffff ffffffff

B 5ac635d8 aa3a93e7
b3ebbd55 769886bc 651d06b0 cc53b0f6 3bce3c3e 27d2604b

(seed) X c49d3608 86e70493 6a6678e1 139d26b7 819f7e90

(compressed) G 03 6b17d1f2 e12c4247
f8bce6e5 63a440f2 77037d81 2deb33a0 f4a13945 d898c296

(uncompressed) G 04 6b17d1f2 e12c4247 f8bce6e5 63a440f2
77037d81 2deb33a0 f4a13945 d898c296 4fe342e2 fe1a7f9b 8ee7eb4a
7c0f9e16 2bce3357 6b315ece cbb64068 37bf51f5

N ffffffff 00000000
ffffffff ffffffff bce6faad a7179e84 f3b9cac2 fc632551

(cofactor) r 1

C.1.5 Elliptic curve over a prime field (384 bits)

P ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
ffffffff ffffffff ffffffff 00000000 00000000 ffffffff
 $2^{384} - 2^{128} - 2^{96} + 2^{32} - 1$

A ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
ffffffff ffffffff ffffffff 00000000 00000000 ffffffff

B b3312fa7 e23ee7e4 988e056b e3f82d19 181d9c6e fe814112
0314088f 5013875a c656398d 8a2ed19d 2a85c8ed d3ec2aef

(seed) X a335926a a319a27a 1d00896a 6773a482 7acdac73

C.2.2 Elliptic curve over a prime field (160 bits)

p	8c72d321 e48aa141 9b22f914 cb43c112 b76d7ae5	
a	8c72d321 e48aa141 9b22f914 cb43c112 b76d7ae2	
b	299ce219 b7b01348 fc2b5007 b6ab1ee1 005676f7	
(compressed) G	00000000 00000000 00000000 00000000 00000002	03
(uncompressed) G	00000000 00000000 00000000 00000000 00000002 0be8f0d3 623edada ce4c2fac a541679b 002f1d07	04
n	8c72d321 e48aa141 9b23b6b2 e4a85a07 3822640f	
(cofactor) r		1

C.2.3 Elliptic curve over a prime field (256 bits)

p	b1d5054e 2f7b68aa ee7ef918 74dd140c 6919af9b 719ed905 f6529c2a 424a6332	
a	b1d5054e 2f7b68aa ee7ef918 74dd140c 6919af9b 719ed902 f6529c2a 424a6332	
b	ae3dd5d1 f97c497c 1d5452d1 b074a6c0 6a25d4e5 819ccd1c 6e974d68 ef44f266	
(compressed) G	00000000 00000000 00000000 00000000 00000000 00000003 02 00000000 00000000	
(uncompressed) G	00000000 00000000 00000000 00000003 693d7af8 c4a29f8d e56e477f f569661c 4dcd2227 aac17b09 e4b4b0b7 03b978ce 04 00000000 00000000 00000000 00000000	
n	b1d5054e 2f7b68ab e99c585a 8419ae9f b45c620e 5ef666c3 f6529c2a 424a6332	
(cofactor) r		1

C.3 Numerical examples of BN curve

C.3.1 Summary

All of the following examples are chosen so that p is the largest prime satisfying $p = 3 \pmod{4}$ and $p = 4 \pmod{9}$ for the largest parameter u with minimum Hamming weight, allowing the extension field $F(p^2)$ to be represented as $F(p)[i]/(i^2 + 1)$ and the extension field $F(p^{2m})$ to be represented as $F(p^2)[z]/(z^m - v)$ for $m = 2, 3, 6$ and $v = 1 + i$. Computation of square (or cube) roots needed for point and/or pairing compression is also simplified in both $F(p)$ and $F(p^2)$. Furthermore, the curve equation has the form $E: y^2 = x^3 + 3$ with the obvious basepoint $G = (1, 2)$, and the sextic twist $E'/F(p^2)$ of the form $E': y'^2 = x'^3 + 3v$ contains a subgroup of order n and cofactor $h = 2p - n$, with basepoint $G' = hG_0'$ where G_0' is a point with x -coordinate $x_0' = 1$. Finally, the isomorphism $\psi: E'/F(p^2) \rightarrow E/F(p^{12})$ takes the form $\psi(x', y') = (x' v^{-1} z^4, y' v^{-1} z^3)$, with $z^6 = v$. These properties

effectively facilitate the implementation of the (plain or compressed) Tate or Weil pairing $e: E \times E' \rightarrow F(p^{2m})$, with optimal pairings especially benefiting from the sparse form of u . Detailed information on these examples is given in [6].

C.3.2 Elliptic curve over a prime field (160 bits)

p	ffffffda 48afd02c ccf4fe55 0dc1ddf3 f4046e43	
a		0
b		3
(compressed) G	02 00000000 00000000 00000000 00000000 00000001	
(uncompressed) G	04 00000000 00000000 00000000 00000000 00000000 00000001 00000000 00000000 00000000 00000000 00000002	
n	ffffffda 48afd02c ccf3fe55 0dd4bad5 95810cdd	
(cofactor) r		1

C.3.3 Elliptic curve over a prime field (192 bits)

p	ffffff5 26bac3d5 23661124 f38543e9 1f0186c1 f247719b	
a		0
b		3
(compressed) G	00000000 00000000 00000000 00000000 00000000 00000001	02
(uncompressed) G	00000000 00000000 00000000 00000000 00000000 00000001 00000000 00000000 00000000 00000000 00000000 00000002	04
n	ffffff5 26bac3d5 23661123 f38543ee 8ba2eb5d 35910e65	
(cofactor) r		1

C.3.4 Elliptic curve over a prime field (224 bits)

p	ffffff5 26bac3d5 23661124 f38543e9 1f0186c1 f247719b ffffff	
a	fff10728 8ec29e60 2c4520db 42180823 bb907d12 87127833	0
b		3
(compressed) G	02 00000000 00000000 00000000 00000000 00000000 00000000 00000001	

(uncompressed) G 04 00000000 00000000
 00000000 00000000 00000000 00000000 00000001 00000000
 00000000 00000000 00000000 00000000 00000000 00000002

n ffffff
 fff10728 8ec29e60 2c4420db 4218082b 36c2accf f76c58ed

(cofactor) r 1

C.3.5 Elliptic curve over a prime field (256 bits)

p ffffff ffcf0cd
 46e5f25e ee71a49f 0cdc65fb 12980a82 d3292ddb aed33013

a 0

b 3

(compressed) G 02 00000000 00000000
 00000000 00000000 00000000 00000000 00000000 00000001

(uncompressed) G 04 00000000 00000000 00000000 00000000
 00000000 00000000 00000000 00000001 00000000 00000000
 00000000 00000000 00000000 00000000 00000000 00000002

n ffffff ffcf0cd
 46e5f25e ee71a49e 0cdc65fb 1299921a f62d536c d10b500d

(cofactor) r 1

C.3.6 Elliptic curve over a prime field (384 bits)

p ffffff ffffffff fff2a968 23d5920d 2a127e3f 6fbca024
 c8fbe295 31892c79 534f9d30 63282615 50a7cabd 7cccd10b

a 0

b 3

(compressed) G 02
 00000000 00000000 00000000 00000000 00000000 00000000
 00000000 00000000 00000000 00000000 00000000 00000001

(uncompressed) G 04
 00000000 00000000 00000000 00000000 00000000 00000000
 00000000 00000000 00000000 00000000 00000000 00000001
 00000000 00000000 00000000 00000000 00000000 00000000
 00000000 00000000 00000000 00000000 00000000 00000002

n ffffff ffffffff fff2a968 23d5920d 2a127e3f 6fbca023
 c8fbe295 31892c79 5356487d 8ac63e4f 4db17384 341a5775

(cofactor) r 1

C.3.7 Elliptic curve over a prime field (512 bits)

p	ffffff fffffff fffffff fff9ec7f 01c60ba1 d8cb5307 c0bbe3c1 11b0ef45 5146cf1e acbe98b8 e48c65de ab236fel 916a55ce 5f4c6467 b4eb2809 22adef33	
a		0
b		3
(compressed) G	02 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000001	
(uncompressed) G	04 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000001 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000002	
n	ffffff fffffff fffffff fff9ec7f 01c60ba1 d8cb5307 c0bbe3c1 11b0ef44 5146cf1e acbe98b8 e48c65de ab2679a3 4a10313e 04f9a2b4 06a64a5f 519a09ed	
(cofactor) r		1

C.4 Numerical examples of Freeman curve

C.4.1 Introduction

Detailed information on the examples in this Clause is given in [13].

C.4.2 Elliptic curve over a prime field (234 bits)

p	2a3 81f6c6d1 423bd477 5aa52e8b 38ffe9f2 36dfdd4a 4b6f5b03 8b3218db
a	2a3 81f6c6d1 423bd477 5aa52e8b 38ffe9f2 36dfdd4a 4b6f5b03 8b3218d8
b	248 268b780f a06cef9c 31295050 153fd4c6 f94dbbe6 21d5d68d a6487f2b
n	2a3 81f6c6d1 423bd477 5aa52e8b 38cbeecb 69176e35 bb5f3716 e4fe375b
D	492c7f03
j ₀	a9 73a6d60b 2d03c9c1 f0b998dd 65cc6903 ba365672 b7ad36ca 6e08d4bf
x	2232f3d6 535caaf9

y 1084 2631ac0b
(cofactor) r 1

C.4.3 Elliptic curve over a prime field (252 bits)

p e4989d4 fd7e87ff
6ff300ef d7e4393d 2c7ed585 2b0bf1c7 7b422e6f e4911f8b

a e4989d4 fd7e87ff
6ff300ef d7e4393d 2c7ed585 2b0bf1c7 7b422e6f e4911f88

b aefa431 ec51a8a3
7e999461 fe75b15b f85dd6e1 19ab1142 1b798e51 c565610d

n e4989d4 fd7e87ff
6ff300ef d7e4393c b38a4f5e ceb8cc58 c00200ff c97e408d

D 3df4c893

j₀ 3ffc8aa e4acb124
39f5498d 1a8e5324 9a94a137 e75637b5 c5ededb1 d1c76c5b

x 3 42afc7c6 f6b26de7

y 1b615 02ae3383
(cofactor) r 1

Annex D (informative)

Summary of properties of Elliptic Curves generated by the Complex Multiplication method

In this annex, the properties of the four generating methods by complex multiplication for MNT curve, BN curve, F curve, and CP curve, are summarized, where we use the following notation. Table 1 gives a summary of each curve.

- $E(F(p)), \#E(F(p)) = rn$ (r: cofactor, n: prime divisor)
- B: embedding degree
- $4p - t^2 = Dy^2$ (t: trace, D: square-free integer)

Table 1 — Summary of elliptic curves generated by the Complex Multiplication method

	B	D	$\log_2 p / \log_2 n$	Characteristic
MNT curve	3, 4, 6	Arbitrary	1	All prime-order elliptic curves with $B = 3, 4, 6$ can be constructed.
BN curve	12	3	1	An elliptic curve with $D = 3$ and $B = 12$ (not all) is constructed.
F curve	10	Arbitrary	1	An elliptic curve with $B = 10$ (not all) is constructed.
CP curve	Arbitrary	Arbitrary	>2	$\log_2 p > 2 \log_2 n$

Bibliography

- [1] ISO/IEC 9796 (all parts), *Information technology — Security techniques — Digital signature schemes giving message recovery*
- [2] ISO/IEC 10118 (all parts), *Information technology — Security techniques — Hash-functions*
- [3] ISO/IEC 18032:2005, *Information technology — Security techniques — Prime number generation*
- [4] A.O.L. Atkin and F. Morain, “*Elliptic curves and primality proving*”, *Math. of Computation*, 61 (1993), 29-68
- [5] H. Cohen, G. Frey, R. Avanzi, C. Doche, T. Lange, K. Nguyen, and F. Vercauteren, “*Handbook of Elliptic and Hyperelliptic Curve Cryptography*”, Chapman & Hall/CRC, 2006
- [6] P.S.L.M. Barreto and M. Naehrig. “*Pairing-friendly elliptic curves of prime order*”, In *Selected Areas in Cryptography-SAC’2005*, LNCS 3897 (2006), Springer-Verlag, 319-331
- [7] I. Blake, G. Seroussi and N. Smart, “*Advances in elliptic curve cryptography*”, London Mathematical society Lecture Note Series 317
- [8] J. Cheon, “*Security Analysis of the Strong Diffie-Hellman Problem*”, In *Eurocrypt 2006*, LNCS 4004 (2006), Springer-Verlag, 1-11
- [9] H. Cohen, “*A course in computational algebraic number theory*”, Graduate Texts in Math. 138, Springer-Verlag, 1993, Third corrected printing, 1996
- [10] D.A. Cox, “*Primes of the form $x^2 + ny^2$* ”, A Wiley-Interscience Publication, 1989
- [11] M. Deuring, “*Die Typen der Multiplikatorenringe elliptischer Funktionenkörper*”, *Abh. Math. Sem. Hamburg*, 14(1941), 197-272
- [12] FIPS 186-2, Digital Signature Standard. *Federal Information Processing Standards Publication 186-2, 2000*. Available from: <http://csrc.nist.gov/>
- [13] D. Freeman, “*Constructing pairing-friendly elliptic curves with embedding degree 10*”, In *ANTS-VII*, Springer-Verlag, LNCS 4076, Berlin, 2006, 452-465
- [14] G. Frey and H.G. Rück, “*A remark concerning m -divisibility and the discrete logarithm in the divisor class group of curves*”, *Mathematics of Computation*, 62 (1994), 865-874
- [15] L. Hitt, “*On the minimal embedding field*”, In *Proceedings of the International Conference on Pairing-Based Cryptography 2007 (Pairing 2007)*, LNCS 4575 (2007), Springer-Verlag, 294–301
- [16] IEEE P1363-2000, *Standard Specifications for Public-Key Cryptography*
- [17] D.R. Kohel, “*Algorithms for Algebra and Geometry Experimentation*”
<http://echidna.maths.usyd.edu.au/~kohel/dbs/>
- [18] K. Matthews, “*The Diophantine equation $x^2 \pm Dy^2 = N$, $D > 1$ In integers*”, *Expositions Mathematician* 18 (2000), 323-331
- [19] A. Menezes, T. Okamoto and S. Vanstone, “*Reducing elliptic curve logarithms to logarithms in a finite field*”, *Proceedings of the 22nd Annual ACM Symposium on the Theory of Computing* (1991), 80-89

- [20] A. Miyaji, M. Nakabayashi and S. Takano, "New explicit conditions of Elliptic Curve Traces under FR-reduction", IEICE Trans., Fundamentals. vol. E84-A, No.5 (2001), 1234-1243
- [21] R. Mollin, Fundamental Number Theory with Applications, CRC Press, Boca Raton, 1998
- [22] D. Page, N.P. Smart and F. Vercauteren, "A comparison of MNT curves and supersingular curves", AAEECC, 17 (2006), Springer-Verlag, 379-392
- [23] J. Robertson, "Solving the generalized Pell equation", Unpublished manuscript (2004), available at <http://www.jpr2718.org/pell.pdf>
- [24] K.H. Rosen, "Elementary number theory and its applications", Addison Welsley Longman, 1999
- [25] T. Satoh and K. Araki, "Fermat quotients and the polynomial time discrete logalgorithm for anomalous elliptic curves", Commentarii Math. Univ. St. Pauli., vol.47 (1998), 81-92
- [26] R. Schoof, "Elliptic Curves Over Finite Fields and the Computation of Square Roots mod p ", Mathematics of Computation, 44 (1985), 483-494
- [27] I.A. Semaev, "Evaluation of discrete logarithms in a group of p -torsion points of an elliptic curve in characteristic p ", Mathematics of Computation, 67 (1998), 353-356
- [28] N.P. Smart, "The discrete logarithm problem on elliptic curves of trace one", J. Cryptology, 12 (1999), 193-196
- [29] L.C. Washington, "Elliptic Curves-Number Theory and Cryptography", Chapman & Hall/CRC, 2nd edition, 2008
- [30] ISO/IEC 11770-3, *Information technology — Security techniques — Key management — Part 3: Mechanisms using asymmetric techniques*
- [31] ISO/IEC 14888-3, *Information technology — Security techniques — Digital signatures with appendix — Part 3: Discrete logarithm based mechanisms*
- [32] ISO/IEC 18033-2, *Information technology — Security techniques — Encryption algorithms — Part 2: Asymmetric ciphers*

ICS 35.040

Price based on 31 pages

BSI - British Standards Institution

BSI is the independent national body responsible for preparing British Standards. It presents the UK view on standards in Europe and at the international level. It is incorporated by Royal Charter.

Revisions

British Standards are updated by amendment or revision. Users of British Standards should make sure that they possess the latest amendments or editions.

It is the constant aim of BSI to improve the quality of our products and services. We would be grateful if anyone finding an inaccuracy or ambiguity while using this British Standard would inform the Secretary of the technical committee responsible, the identity of which can be found on the inside front cover. Tel: +44 (0)20 8996 9000. Fax: +44 (0)20 8996 7400.

BSI offers members an individual updating service called PLUS which ensures that subscribers automatically receive the latest editions of standards.

Buying standards

Orders for all BSI, international and foreign standards publications should be addressed to Customer Services. Tel: +44 (0)20 8996 9001. Fax: +44 (0)20 8996 7001 Email: orders@bsigroup.com You may also buy directly using a debit/credit card from the BSI Shop on the Website <http://www.bsigroup.com/shop>

In response to orders for international standards, it is BSI policy to supply the BSI implementation of those that have been published as British Standards, unless otherwise requested.

Information on standards

BSI provides a wide range of information on national, European and international standards through its Library and its Technical Help to Exporters Service. Various BSI electronic information services are also available which give details on all its products and services. Contact Information Centre. Tel: +44 (0)20 8996 7111 Fax: +44 (0)20 8996 7048 Email: info@bsigroup.com

Subscribing members of BSI are kept up to date with standards developments and receive substantial discounts on the purchase price of standards. For details of these and other benefits contact Membership Administration. Tel: +44 (0)20 8996 7002 Fax: +44 (0)20 8996 7001 Email: membership@bsigroup.com

Information regarding online access to British Standards via British Standards Online can be found at <http://www.bsigroup.com/BSOL>

Further information about BSI is available on the BSI website at <http://www.bsigroup.com>.

Copyright

Copyright subsists in all BSI publications. BSI also holds the copyright, in the UK, of the publications of the international standardization bodies. Except as permitted under the Copyright, Designs and Patents Act 1988 no extract may be reproduced, stored in a retrieval system or transmitted in any form or by any means – electronic, photocopying, recording or otherwise – without prior written permission from BSI.

This does not preclude the free use, in the course of implementing the standard, of necessary details such as symbols, and size, type or grade designations. If these details are to be used for any other purpose than implementation then the prior written permission of BSI must be obtained.

Details and advice can be obtained from the Copyright and Licensing Manager. Tel: +44 (0)20 8996 7070 Email: copyright@bsigroup.com

BSI Group
Headquarters 389
Chiswick High Road,
London, W4 4AL, UK
Tel +44 (0)20 8996 9001
Fax +44 (0)20 8996 7001
[www.bsigroup.com/
standards](http://www.bsigroup.com/standards)

