INTERNATIONAL STANDARD

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)

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

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.

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.

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 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 $W_0, W_1, \ldots, W_s$ as follows:

$$W = W_0 \| W_1 \| \ldots \| W_s.$$

g) Convert W to a finite field element c = OS2FEP (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 $W_i = H(X_i)$.