GM/T 0003.4

SM2 Public Key Cryptographic Algorithms Based on Elliptic Curves

Part 4: Public Key Encryption Algorithm

Cryptography Standardization
Technical Committee of China

Issued on: 2012-03-21 Translated on: 2024-10-30

Contents

Foreword

GM/T 0003 “SM2 public key cryptographic algorithms based on elliptic curves” consists of 5 parts:

- Part 1: General
- Part 2: Digital signature algorithm
- Part 3: Key exchange protocol
- Part 4: Public key encryption algorithm
- Part 5: Parameter definition

This section is the fourth part of GM/T 0003.

1 Scope

This part of GM/T 0003 specifies the public encryption algorithm of SM2 public key cryptographic algorithms based on elliptic curves, and gives examples of encryption and decryption of messages and the corresponding processes.

This part is applicable to encryption and decryption of messages in commercial cryptographic applications. A sender of messages can encrypt messages using the public key of a receiver, while the receiver can make decryption with his corresponding private key to get the messages. Besides, this part also provides standardization guidance and standardization references to products and technology for manufacturers of security products, improving the credibility and maneuverability of security products.

2 Normative references

The following documents are necessary for the application of this document. For the references with noted dates, only the version on the specific date applies to this part. For the references without dates, the newest version (including all the modified lists) applies to this part.

GM/T 0003.1, SM2 Public key cryptographic algorithms based on elliptic curves — Part 1: General

3 Terms and definitions

The following terms and definitions apply to this part.

3.1 secret key

A kind of key, which is shared between the sender and the receiver while unknown to the third party.

3.2 message

Any bit string of finite length.

3.3 key derivation function

The function that generates one or more shared secret keys on input shared secrets and other parameters known to the two parties.

4 Symbols

The following symbols are applicable to this part.

A, B: two users who use the public key cryptography system.

a, b: elements in F_q which define an elliptic curve E over F_q.

d_A: the private key of user A.

d_B: the private key of user B.

$E(F_q)$: the set of rational points on elliptic curves E over F_q (including the infinity point O).

F_q: the finite field with q elements.

G: a base point of an elliptic curve with prime order.

$Hash()$: a cryptographic hash function.

$H_v()$: a cryptographic hash function with v bits message digest.

$KDF()$: key derivation function.

M: the message to be encrypted.

M': the message obtained by decryption.

n: the order of a base point G where n is a prime factor of $\#E(F_q)$.

O: a special point on an elliptic curve called the infinity point or zero point. it is the identity of the additive group of an elliptic curve.

P_B: public key of user B.

q: the number of elements of finite field F_q.

$x||y$: the concatenation of x and y, where x and y are bit strings or byte strings.

$[k]P$: a point which is k times of point P on elliptic curves, i.e., $[k]P = \underbrace{P + P + \cdots + P}_{Add\ k\ times}$, where k is a positive integer.

$[x, y]$: the set of integers which are greater than or equal to x and less than or equal to y

$\lceil x \rceil$: ceiling function which maps x to the smallest integer greater than or equal to x. For example, $\lceil 7 \rceil = 7$, $\lceil 8.3 \rceil = 9$.

$\lfloor x \rfloor$: floor function which maps x to the largest integer less than or equal to x. For example, $\lfloor 7 \rfloor = 7$, $\lfloor 8.3 \rfloor = 8$.

$\#E(F_q)$: the number of points on $E(F_q)$, called the order of elliptic curves $E(F_q)$.

5 Algorithm parameters and auxiliary functions

5.1 General

Public key encryption algorithm stipulates that a sender generates a ciphertext by encrypting a message with a receiver's public key, while the receiver can get the original message by decrypting the received ciphertext with his own private key.

5.2 System parameters of elliptic curves

The system parameters of elliptic curves include the size q of finite field F_q (when $q = 2^m$, also identifiers of representation of elements and reduced polynomial are involved), two elements $a, b \in F_q$ which define a equation of the elliptic curve $E(F_q)$, a

base point $G = (x_G, y_G)$ $(G \neq O)$ over $E(F_q)$ where x_G and y_G are two elements of F_q, the order n of G, and other optional parameters (e.g. the cofactor h of n etc.).

The system parameters of elliptic curves and their validation should be in line with the regulations in Clause 5 of GM/T 0003.1.

5.3 User key pair

The key pair of user B consists of the private key d_B and the public key $P_B = [d_B]G$.

The generation algorithm of user key pairs and the validation algorithm of public keys should be consistent with the specification in Clause 6 of GM/T 0003.1.

5.4 Auxiliary functions

5.4.1 Overview

Three kinds of auxiliary functions are involved in the public key encryption algorithm based on elliptic curves specified in this part: cryptographic hash function, key derivation function and pseudo-random number generator. The strength of these three types of auxiliary functions has direct impact on the security of the encryption algorithm.

5.4.2 Cryptographic hash function

This part should adopt the secure cryptographic hash function, approved by the State Cryptography Administration, such as the GM/T 0004 SM3 Cryptographic Hash Algorithm.

5.4.3 Key derivation function

The functionality of key derivation function is to derive key data from a shared secret bit string. In the process of key agreement, on input of the shared secret bit string obtained by key exchange protocol, the key derivation function outputs a required session key or a key data required by further encryption.

Key derivation function needs to invoke the cryptographic hash function.

Let $H_v()$ be a cryptographic hash function which outputs a hash value of length v bits.

Key derivation function $KDF(Z, klen)$ is defined as follows:

Input: a bit string Z, an integer $klen$ which represents the bit length of the resulting secret key data and is required to be smaller than $(2^{32} - 1)v$.

Output: the secret key bit string K of length $klen$.

a) Initialize a 32-bit counter $ct = 0x00000001$;

b) For i from 1 to $\left\lceil \frac{klen}{v} \right\rceil$:

b.1) compute $Ha_i = H_v(Z \| ct)$;

b.2) $ct++$;

c) If $\frac{klen}{v}$ is an integer, let $Ha!_{\left\lceil \frac{klen}{v} \right\rceil} = Ha_{\left\lceil \frac{klen}{v} \right\rceil}$; Otherwise let $Ha!_{\left\lceil \frac{klen}{v} \right\rceil}$ be the leftmost $(klen - (v \times \left\lfloor \frac{klen}{v} \right\rfloor))$ bits of $Ha_{\left\lceil \frac{klen}{v} \right\rceil}$.

d) Let $K = Ha_1 \| \cdots \| Ha_{\left\lceil \frac{klen}{v} \right\rceil - 1} \| Ha!_{\left\lceil \frac{klen}{v} \right\rceil}$.

5.4.4 Random number generator

This part should adopt random number generators, approved by the State Cryptography Administration, such as GM/T 0105 software-based random number generators.

6 Encryption algorithm and its process

6.1 Encryption algorithm

Suppose the message to be sent is the bit string M and $klen$ represents the bit-length of M.

To encrypt the plaintext M, user A should perform the following procedures:

A1: generate a random number $k \in [1, n-1]$ with the random number generator.

A2: compute point $C_1 = [k]G = (x_1, y_1)$ of the elliptic curve, and convert the data type of C_1 to bit string as specified in Clauses 4.2.8 and 4.2.4 of GM/T 0003.1.

A3: compute point $S = [h]P_B$ of the elliptic curve; if S is the infinity point, report error and the protocol fails.

A4: compute point $[k]P_B = (x_2, y_2)$ of the elliptic curve, convert the data type of x_2, y_2 to bit string as specified in Clauses 4.2.5 and 4.2.4 of GM/T 0003.1.

A5: compute $t = KDF(x_2||y_2, klen)$. If t is an all-zero bit string, go to A1.

A6: compute $C_2 = M \oplus t$.

A7: compute $C_3 = Hash(x_2||M||y_2)$.

A8: output the ciphertext $C = C_1||C_3||C_2$.

Note: Examples of the process of encryption are described in Annex A.

6.2 Process of encryption algorithm

The process of the encryption algorithm is depicted in Figure 1.

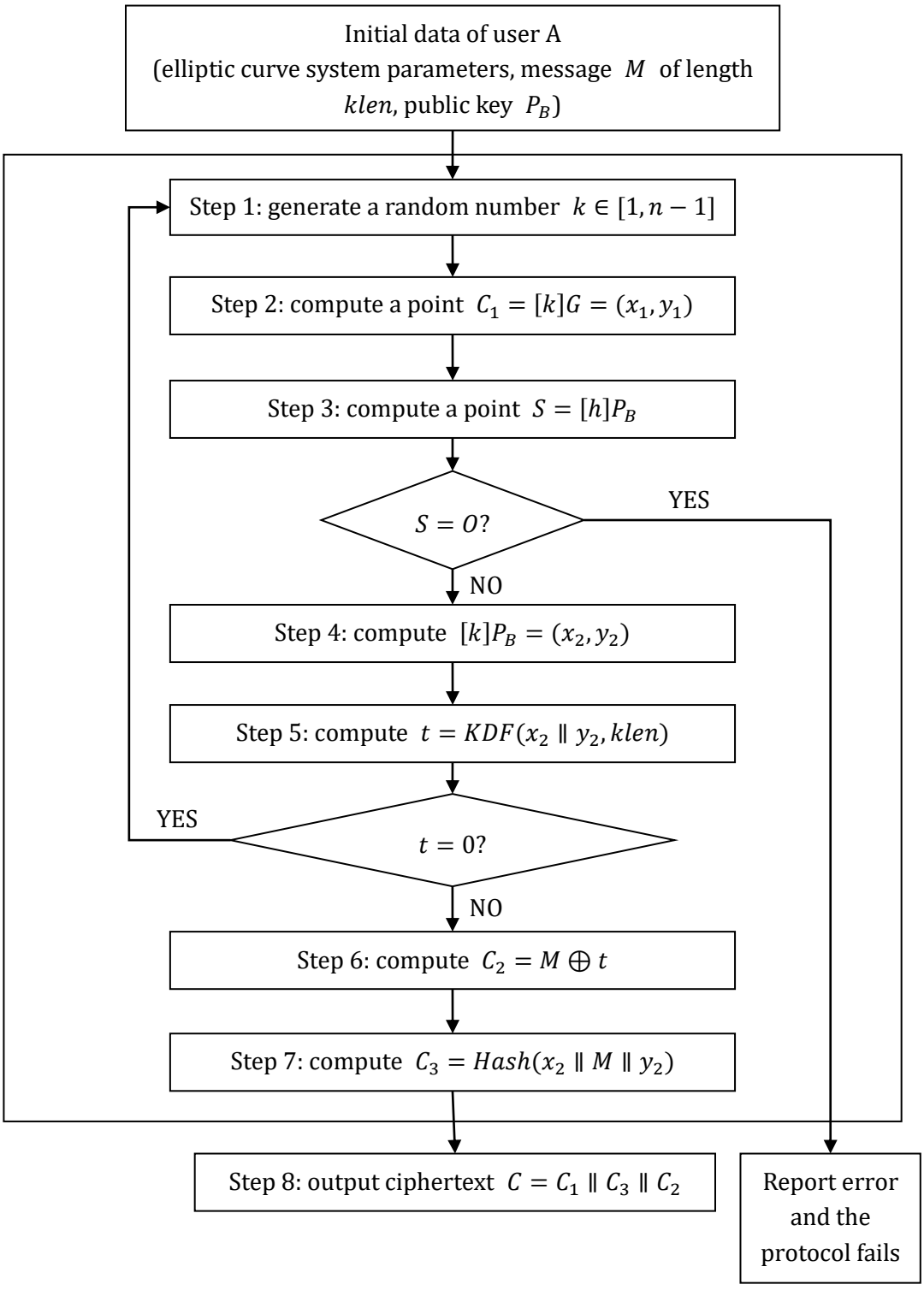

Figure 1: Encryption algorithm process

7 Decryption algorithm and its process

7.1 Decryption algorithm

Suppose $klen$ is the length of C_2 in the ciphertext.

To decrypt the ciphertext $C = C_1 \| C_3 \| C_2$, user B should perform the following procedures:

B1: get C_1 from C, and convert the data type of C_1 to the point of the elliptic curve as specified in Clauses 4.2.3 and 4.2.9 of GM/T 0003.1. Then, verify whether C_1 on the elliptic curve. If not, output "ERROR", and the protocol fails.

B2: compute point $S = [h]C_1$ of the elliptic curve. If S is the infinity point, then output "ERROR" and the protocol fails.

B3: compute $[d_B]C_1 = (x_2, y_2)$ and convert the data type of x_2, y_2 to bit string as specified in Clauses 4.2.5 and 4.2.4 of GM/T 0003.1.

B4: compute $t = KDF(x_2 \| y_2, klen)$. If t is an all-zero bit string, then output "ERROR", and the protocol fails.

B5: get C_2 from C, and compute $M' = C_2 \oplus t$.

B6: compute $u = Hash(x_2 \| M' \| y_2)$. Get C_3 from C. If $u \neq C_3$, output "ERROR", and the protocol fails.

B7: output the plaintext M'.

Note: Examples of the process of decryption are described in Annex A.

7.2 Process of decryption algorithm

The process of decryption algorithm is depicted in Figure 2.

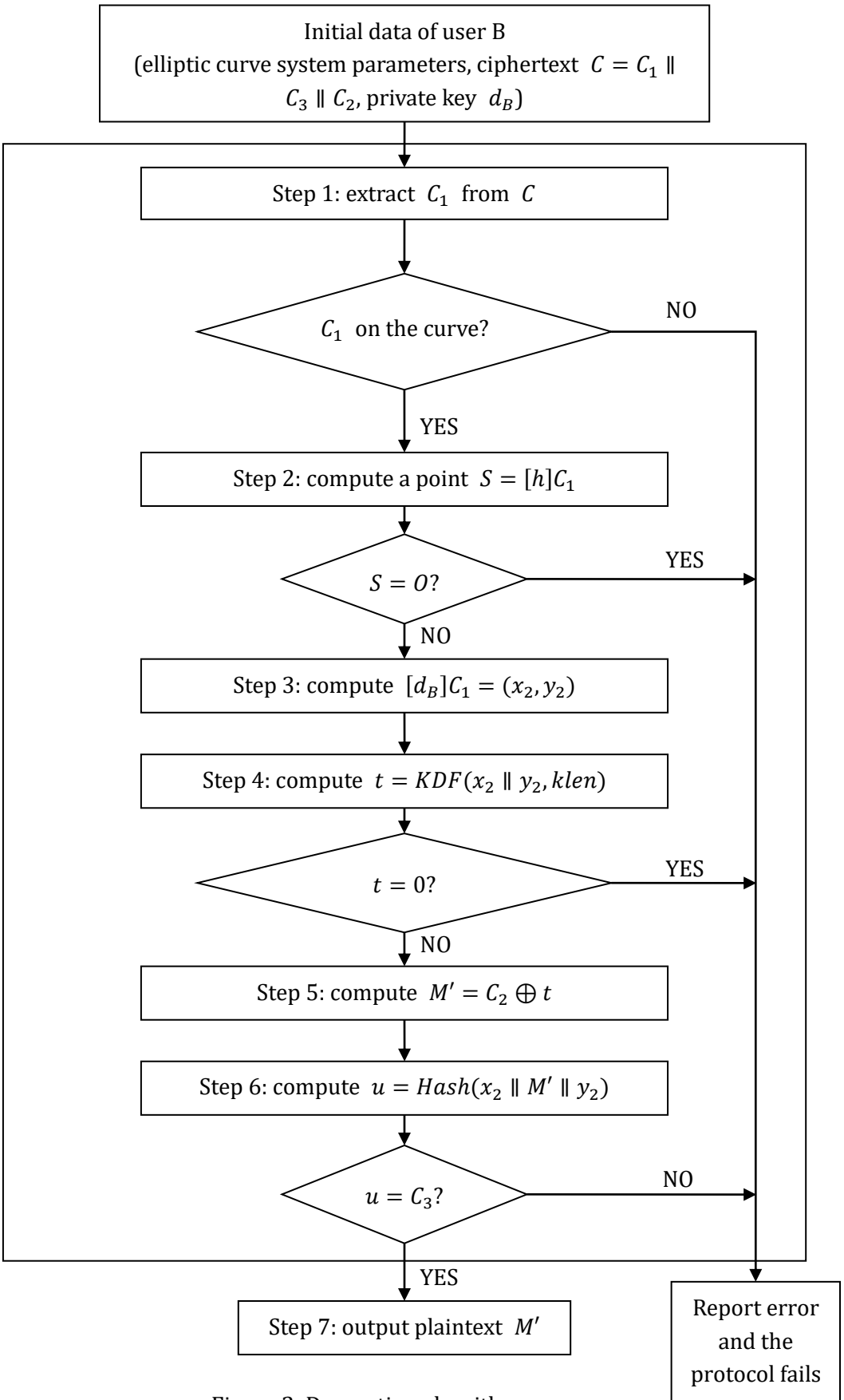

Figure 2: Decryption algorithm process

Annex A
(informative)
Examples of message encryption and decryption

A.1 General requirements

This annex adopts the cryptographic hash function specified in GM/T 0004 SM3 Cryptographic Hash Algorithm, whose input is a bit string of length less than 2^{64}, and output is a hash value of length 256 bits, denoted $H_{256}()$.

In this annex, for all values represented in hexadecimal form, the left is the most significant side and the right is the least significant side.

In this annex, plaintexts are denoted as ASCII encoding.

A.2 Message encryption and decryption on elliptic curves over F_p

The elliptic curve equation is: $y^2 = x^3 + ax + b$

Example 1: $F_p - 192$

prime p: BDB6F4FE 3E8B1D9E 0DA8C0D4 6F4C318C EFE4AFE3 B6B8551F

coefficient a: BB8E5E8F BC115E13 9FE6A814 FE48AAA6 F0ADA1AA 5DF91985

coefficient b: 1854BEBD C31B21B7 AEFC80AB 0ECD10D5 B1B3308E 6DBF11C1

base point $G = (x_G, y_G)$, whose order is n

coordinate x_G: 4AD5F704 8DE709AD 51236DE6 5E4D4B48 2C836DC6 E4106640

coordinate y_G: 02BB3A02 D4AAADAC AE24817A 4CA3A1B0 14B52704 32DB27D2

order n: BDB6F4FE 3E8B1D9E 0DA8C0D4 0FC96219 5DFAE76F 56564677

message M to be encrypted: encryption standard

hexadecimal form of message M: 656E63 72797074 696F6E20 7374616E 64617264

private key d_B: 58892B80 7074F53F BF67288A 1DFAA1AC 313455FE 60355AFD

public key $P_B = (x_B, y_B)$:

coordinate x_B: 79F0A954 7AC6D100 531508B3 0D30A565 36BCFC81 49F4AF4A

coordinate y_B: AE38F2D8 890838DF 9C19935A 65A8BCC8 994BC792 4672F912

Related values in steps of the encryption algorithm:

generate random number k: 384F3035 3073AEEC E7A16543 30A96204 D37982A3 E15B2CB5

compute point $C_1 = [k]G = (x_1, y_1)$ of the elliptic curve:

coordinate x_1: 23FC680B 124294DF DF34DBE7 6E0C38D8 83DE4D41 FA0D4CF5

coordinate y_1: 70CF14F2 0DAF0C4D 777F738D 16B16824 D31EEFB9 DE31EE1F

choose the uncompressed form of C_1, convert the point to byte string of form $\mathrm{PC} \| x_1 \| y_1$ where PC is a single byte and $\mathrm{PC} = 04$, and denoted still by C_1.

compute point $[k]P_B = (x_2, y_2)$ of the elliptic curve:

coordinate x_2: 57E7B636 23FAE5F0 8CDA468E 872A20AF A03DED41 BF140377

coordinate y_2: 0E040DC8 3AF31A67 991F2B01 EBF9EFD8 881F0A04 93000603

bit length of message M: $klen = 152$

compute $t = KDF(x_2 \| y_2, klen)$: 046B04 A9ADF53B 389B9E2A AFB47D90 F4D08978

compute $C_2 = M \oplus t$: 610567 DBD4854F 51F4F00A DCC01CFE 90B1FB1C

compute $C_3 = Hash(x_2 \| M \| y_2)$:

$x_2 \| M \| y_2$: 57E7B636 23FAE5F0 8CDA468E 872A20AF A03DED41 BF140377 656E6372 79707469 6F6E2073 74616E64 6172640E 040DC83A F31A6799 1F2B01EB F9EFD888 1F0A0493 000603

C_3: 6AFB3BCE BD76F82B 252CE5EB 25B57996 86902B8C F2FD8753 6E55EF76 03B09E7C

Output the ciphertext $C = C_1 \| C_3 \| C_2$:

04 23FC680B 124294DF DF34DBE7 6E0C38D8 83DE4D41 FA0D4CF5 70CF14F2 0DAF0C4D 777F738D 16B16824
D31EEFB9 DE31EE1F 6AFB3BCE BD76F82B 252CE5EB 25B57996 86902B8C F2FD8753 6E55EF76 03B09E7C
610567DB D4854F51 F4F00ADC C01CFE90 B1FB1C

Related values in steps of the decryption algorithm:

compute point $[d_B]C_2 = (x_2, y_2)$:

coordinate x_2: 57E7B636 23FAE5F0 8CDA468E 872A20AF A03DED41 BF140377

coordinate y_2: 0E040DC8 3AF31A67 991F2B01 EBF9EFD8 881F0A04 93000603

compute $t = KDF(x_2 \| y_2, klen)$: 046B04 A9ADF53B 389B9E2A AFB47D90 F4D08978

compute $M' = C_2 \oplus t$: 656E63 72797074 696F6E20 7374616E 64617264

compute $u = Hash(x_2 \| M' \| y_2)$:

6AFB3BCE BD76F82B 252CE5EB 25B57996 86902B8C F2FD8753 6E55EF76 03B09E7C

plaintext M': 656E63 72797074 696F6E20 7374616E 64617264, i.e. encryption standard

Example 2: $F_p - 256$

prime p: 8542D69E 4C044F18 E8B92435 BF6FF7DE 45728391 5C45517D 722EDB8B 08F1DFC3

coefficient a: 787968B4 FA32C3FD 2417842E 73BBFEFF 2F3C848B 6831D7E0 EC65228B 3937E498

coefficient b: 63E4C6D3 B23B0C84 9CF84241 484BFE48 F61D59A5 B16BA06E 6E12D1DA 27C5249A

base point $G = (x_G, y_G)$, whose order is n

coordinate x_G: 421DEBD6 1B62EAB6 746434EB C3CC315E 32220B3B ADD50BDC 4C4E6C14 7FEDD43D

coordinate y_G: 0680512B CBB42C07 D47349D2 153B70C4 E5D7FDFC BFA36EA1 A85841B9 E46E09A2

order n: 8542D69E 4C044F18 E8B92435 BF6FF7DD 29772063 0485628D 5AE74EE7 C32E79B7

message M to be encrypted: encryption standard

hexadecimal form of message M: 656E63 72797074 696F6E20 7374616E 64617264

private key d_B: 1649AB77 A00637BD 5E2EFE28 3FBF3535 34AA7F7C B89463F2 08DDBC29 20BB0DA0

public key $P_B = (x_B, y_B)$:

coordinate x_B: 435B39CC A8F3B508 C1488AFC 67BE491A 0F7BA07E 581A0E48 49A5CF70 628A7E0A

coordinate y_B: 75DDBA78 F15FEECB 4C7895E2 C1CDF5FE 01DEBB2C DBADF453 99CCF77B BA076A42

Related values in steps of the encryption algorithm:

generate random number k:

4C62EEFD 6ECFC2B9 5B92FD6C 3D957514 8AFA1742 5546D490 18E5388D 49DD7B4F

compute point $C_1 = [k]G = (x_1, y_1)$ of the elliptic curve:

coordinate x_1: 245C26FB 68B1DDDD B12C4B6B F9F2B6D5 FE60A383 B0D18D1C 4144ABF1 7F6252E7

coordinate y_1: 76CB9264 C2A7E88E 52B19903 FDC47378 F605E368 11F5C074 23A24B84 400F01B8

choose the uncompressed form of C_1, convert the point to byte string of form $\mathrm{PC} \| x_1 \| y_1$ where PC is a single byte and $\mathrm{PC} = 04$, and denoted still by C_1.

compute point $[k]P_B = (x_2, y_2)$ of the elliptic curve:

coordinate x_2: 64D20D27 D0632957 F8028C1E 024F6B02 EDF23102 A566C932 AE8BD613 A8E865FE

coordinate y_2: 58D225EC A784AE30 0A81A2D4 8281A828 E1CEDF11 C4219099 84026537 5077BF78

bit length of message M: $klen = 152$

compute $t = KDF(x_2 \| y_2, klen)$: 006E30 DAE231B0 71DFAD8A A379E902 64491603

compute $C_2 = M \oplus t$: 650053 A89B41C4 18B0C3AA D00D886C 00286467

compute $C_3 = Hash(x_2 \| M \| y_2)$:

$x_2 \| M \| y_2$: 64D20D27 D0632957 F8028C1E 024F6B02 EDF23102 A566C932 AE8BD613 A8E865FE 656E6372 79707469 6F6E2073 74616E64 61726458 D225ECA7 84AE300A 81A2D482 81A828E1 CEDF11C4 21909984 02653750 77BF78

C_3: 9C3D7360 C30156FA B7C80A02 76712DA9 D8094A63 4B766D3A 285E0748 0653426D

output the ciphertext $C = C_1 \| C_3 \| C_2$:

04 245C26FB 68B1DDDD B12C4B6B F9F2B6D5 FE60A383 B0D18D1C 4144ABF1 7F6252E7 76CB9264 C2A7E88E
52B19903 FDC47378 F605E368 11F5C074 23A24B84 400F01B8 9C3D7360 C30156FA B7C80A02 76712DA9
D8094A63 4B766D3A 285E0748 0653426D 650053A8 9B41C418 B0C3AAD0 0D886C00 286467

Related values in steps of the decryption algorithm:

compute point $[d_B]C_1 = (x_2, y_2)$:

coordinate x_2: 64D20D27 D0632957 F8028C1E 024F6B02 EDF23102 A566C932 AE8BD613 A8E865FE

coordinate y_2: 58D225EC A784AE30 0A81A2D4 8281A828 E1CEDF11 C4219099 84026537 5077BF78

compute $t = KDF(x_2 \| y_2, klen)$: 006E30 DAE231B0 71DFAD8A A379E902 64491603

compute $M' = C_2 \oplus t$: 656E63 72797074 696F6E20 7374616E 64617264

compute $u = Hash(x_2 \| M' \| y_2)$:

9C3D7360 C30156FA B7C80A02 76712DA9 D8094A63 4B766D3A 285E0748 0653426D

plaintext M': 656E63 72797074 696F6E20 7374616E 64617264, i.e. encryption standard

A.3 Message encryption and decryption on elliptic curves over F_{2^m}

The elliptic curve equation is: $y^2 + xy = x^3 + ax^2 + b$

Example 3: $F_{2^m} - 193$

generator polynomial of base field: $x^{193} + x^{15} + 1$

coefficient a: 0

coefficient b: 00 2FE22037 B624DBEB C4C618E1 3FD998B1 A18E1EE0 D05C46FB

base point $G = (x_G, y_G)$ whose order is n

coordinate x_G: D78D47E8 5C936440 71BC1C21 2CF994E4 D21293AA D8060A84

coordinate y_G: 615B9E98 A31B7B2F DDEEECB7 6B5D8755 86293725 F9D2FC0C

order n: 80000000 00000000 00000000 43E9885C 46BF45D8 C5EBF3A1

message M to be encrypted: encryption standard

hexadecimal form of message M: 656E63 72797074 696F6E20 7374616E 64617264

private key d_B: 6C205C15 89087376 C2FE5FEE E153D4AC 875D643E B8CAF6C5

public key $P_B = (x_B, y_B)$:

coordinate x_B: 00 E788F191 C5591636 FA992CE6 7CDC8D3B 16E4F4D4 6AF267B8

coordinate y_B: 00 BD6E7E5E 4113D790 20ED5A10 287C14B7 A6767C4D 814ADBFD

Related values in steps of the encryption algorithm:

generate random number k: 6E51C537 3D5B4705 DC9B94FA 9BCF30A7 37ED8D69 1E76D9F0

compute point $C_1 = [k]G = (x_1, y_1)$ of the elliptic curve:

coordinate x_1: 00 95A8B866 7ACF097F 65CE96EB FE53422F CF15876D 16446B8A

coordinate y_1: 01 7A1EC7C9 BAB0DE07 0522311E 75CD31C3 C4D74150 E84E0A95

choose the uncompressed form of C_1, convert the point to byte string of form $\mathrm{PC} \| x_1 \| y_1$ where PC is a single byte and $\mathrm{PC} = 04$, and denoted still by C_1.

compute point $[k]P_B = (x_2, y_2)$ of the elliptic curve:

coordinate x_2: 01 C6271B31 F6BE396A 4166C061 6CF4A8AC DA5BEF4D CBF2DD42

coordinate y_2: 01 47AF35DF A1BFE2F1 61521BCF 59BAB835 64868D92 95881735

bit length of message M: $klen = 152$

compute $t = KDF(x_2 \| y_2, klen)$: BC5F0D 50F2B2BC F2DC3027 0BAA5249 3B8A67A4

compute $C_2 = M \oplus t$: D9316E 228BC2C8 9BB35E07 78DE3327 5FEB15C0

compute $C_3 = Hash(x_2 \| M \| y_2)$:

$x_2 \| M \| y_2$: 01C6271B 31F6BE39 6A4166C0 616CF4A8 ACDA5BEF 4DCBF2DD 42656E63 72797074 696F6E20 7374616E

64617264 0147AF35 DFA1BFE2 F161521B CF59BAB8 3564868D 92958817 35

C_3: F0A41F6F 48AC723C ECFC4B76 7299A5E2 5C064167 9FBD2D4D 20E9FFD5 B9F0DAB8

output the ciphertext $C = C_1 \| C_3 \| C_2$:

04 0095A8B8 667ACF09 7F65CE96 EBFE53422FCF 15876D16 446B 8A017A1E C7C9BAB0 DE070522 311E75CD

31C3C4D7 4150E84E 0A95F0A4 1F6F48AC 723CECFC 4B767299 A5E25C06 41679FBD 2D4D20E9 FFD5B9F0

DAB8D931 6E228BC2 C89BB35E 0778DE33 275FEB15 C0

Related values in steps of the decryption algorithm:

compute point $[d_B]C_1 = (x_2, y_2)$:

coordinate x_2: 01 C6271B31 F6BE396A 4166C061 6CF4A8AC DA5BEF4D CBF2DD42

coordinate y_2: 01 47AF35DF A1BFE2F1 61521BCF 59BAB835 64868D92 95881735

compute $t = KDF(x_2 \| y_2, klen)$: BC5F0D 50F2B2BC F2DC3027 0BAA5249 3B8A67A4

compute $M' = C_2 \oplus t$: 656E63 72797074 696F6E20 7374616E 64617264

compute $u = Hash(x_2 \| M' \| y_2)$:

F0A41F6F 48AC723C ECFC4B76 7299A5E2 5C064167 9FBD2D4D 20E9FFD5 B9F0DAB8

plaintext M': 656E63 72797074 696F6E20 7374616E 64617264, i.e. encryption standard

Example 4: $F_{2^m} - 257$

generator polynomial of base field: $x^{257} + x^{12} + 1$

coefficient a: 0

coefficient b: 00 E78BCD09 746C2023 78A7E72B 12BCE002 66B9627E CB0B5A25 367AD1AD 4CC6242B

base point $G = (x_G, y_G)$, whose order is n

coordinate x_G: 00 CDB9CA7F 1E6B0441 F658343F 4B10297C 0EF9B649 1082400A 62E7A748 5735FADD

coordinate y_G: 01 3DE74DA6 5951C4D7 6DC89220 D5F7777A 611B1C38 BAE260B1 75951DC8 060C2B3E

order n: 7FFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF BC972CF7 E6B6F900 945B3C6A 0CF6161D

message M to be encrypted: encryption standard

hexadecimal form of message M: 656E63 72797074 696F6E20 7374616E 64617264

private key d_B: 56A270D1 7377AA9A 367CFA82 E46FA526 7713A9B9 1101D077 7B07FCE0 18C757EB

public key $P_B = (x_B, y_B)$:

coordinate x_B: 00 A67941E6 DE8A6180 5F7BCFF0 985BB3BE D986F1C2 97E4D888 0D82B821 C624EE57

coordinate y_B: 01 93ED5A67 07B59087 81B86084 1085F52E EFA7FE32 9A5C8118 43533A87 4D027271

Related values in steps of the encryption algorithm:

generate random number k:

6D3B4971 53E3E925 24E5C122 682DBDC8 705062E2 0B917A5F 8FCDB8EE 4C66663D

compute point $C_1 = [k]G = (x_1, y_1)$ of the elliptic curve:

coordinate x_1: 01 9D236DDB 305009AD 52C51BB9 32709BD5 34D476FB B7B0DF95 42A8A4D8 90A3F2E1

coordinate y_1: 00 B23B938D C0A94D1D F8F42CF4 5D2D6601 BF638C3D 7DE75A29 F02AFB7E 45E91771

choose the uncompressed form of C_1, convert the point to byte string of form $\mathrm{PC} \| x_1 \| y_1$ where PC is a single byte and $\mathrm{PC} = 04$, and denoted still by C_1.

compute point $[k]P_B = (x_2, y_2)$ of the elliptic curve:

coordinate x_2: 00 83E628CF 701EE314 1E8873FE 55936ADF 24963F5D C9C64805 66C80F8A 1D8CC51B

coordinate y_2: 01 524C647F 0C0412DE FD468BDA 3AE0E5A8 0FCC8F5C 990FEE11 60292923 2DCD9F36

bit length of message M: $klen = 152$

compute $t = KDF(x_2 \| y_2, klen)$: 983BCF 106AB2DC C92F8AEA C6C60BF2 98BB0117

compute $C_2 = M \oplus t$: FD55AC 6213C2A8 A040E4CA B5B26A9C FCDA7373

compute $C_3 = Hash(x_2 \| M \| y_2)$:

$x_2 \| M \| y_2$: 0083E628 CF701EE3 141E8873 FE55936A DF24963F 5DC9C648 0566C80F 8A1D8CC5 1B656E63 72797074 696F6E20 7374616E 64617264 01524C64 7F0C0412 DEFD468B DA3AE0E5 A80FCC8F 5C990FEE 11602929 232DCD9F 36

C_3: 73A48625 D3758FA3 7B3EAB80 E9CFCABA 665E3199 EA15A1FA 8189D96F 579125E4

output the ciphertext $C = C_1 \| C_3 \| C_2$:

04 019D236D DB305009 AD52C51B B932709B D534D476 FBB7B0DF 9542A8A4 D890A3F2 E100B23B 938DC0A9
4D1DF8F4 2CF45D2D 6601BF63 8C3D7DE7 5A29F02A FB7E45E9 177173A4 8625D375 8FA37B3E AB80E9CF
CABA665E 3199EA15 A1FA8189 D96F5791 25E4FD55 AC6213C2 A8A040E4 CAB5B26A 9CFCDA73 73

Related values in steps of the decryption algorithm:

compute point $[d_B]C_1 = (x_2, y_2)$:

coordinate x_2: 00 83E628CF 701EE314 1E8873FE 55936ADF 24963F5D C9C64805 66C80F8A 1D8CC51B

coordinate y_2: 01 524C647F 0C0412DE FD468BDA 3AE0E5A8 0FCC8F5C 990FEE11 60292923 2DCD9F36

compute $t = KDF(x_2 \| y_2, klen)$: 983BCF 106AB2DC C92F8AEA C6C60BF2 98BB0117

compute $M' = C_2 \oplus t$: 656E63 72797074 696F6E20 7374616E 64617264

compute $u = Hash(x_2 \| M' \| y_2)$:

73A48625 D3758FA3 7B3EAB80 E9CFCABA 665E3199 EA15A1FA 8189D96F 579125E4

plaintext M': 656E63 72797074 696F6E20 7374616E 64617264, i.e. encryption standard