CFRG Y. Sakemi
Internet-Draft Lepidum
Intended status: Experimental T. Kobayashi
Expires: August 27, 2020 T. Saito
NTT
February 24, 2020

Pairing-Friendly Curves
draft-irtf-cfrg-pairing-friendly-curves-01

Abstract

This memo introduces pairing-friendly curves used for constructing pairing-based cryptography. It describes recommended parameters for each security level and recent implementations of pairing-friendly curves.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

1.1. Pairing-Based Cryptography

Elliptic curve cryptography is one of the important areas in recent cryptography. The cryptographic algorithms based on elliptic curve cryptography, such as ECDSA (Elliptic Curve Digital Signature Algorithm), are widely used in many applications.

Pairing-based cryptography, a variant of elliptic curve cryptography, has attracted the attention for its flexible and applicable functionality. Pairing is a special map defined over elliptic curves. Thanks to the characteristics of pairing, it can be applied to construct several cryptographic algorithms and protocols such as identity-based encryption (IBE), attribute-based encryption (ABE), authenticated key exchange (AKE), short signatures and so on. Several applications of pairing-based cryptography are now in practical use.

As the importance of pairing grows, elliptic curves where pairing is efficiently computable are studied and the special curves called pairing-friendly curves are proposed.

1.2. Applications of Pairing-Based Cryptography

Several applications using pairing-based cryptography are standardized and implemented. We show example applications available in the real world.

IETF publishes RFCs for pairing-based cryptography such as Identity-Based Cryptography [RFC5091], Sakai-Kasahara Key Encryption (SAKKE) [RFC6508], and Identity-Based Authenticated Key Exchange (IBAKE) [RFC6539]. SAKKE is applied to Multimedia Internet KEYing (MIKEY) [RFC6509] and used in 3GPP [SAKKE].

Pairing-based key agreement protocols are standardized in ISO/IEC [ISOIEC11770-3]. In [ISOIEC11770-3], a key agreement scheme by Joux [Joux00], identity-based key agreement schemes by Smart-Chen-Cheng [CCS07] and by Fujioka-Suzuki-Ustaoglu [FSU10] are specified.

MIRACL implements M-Pin, a multi-factor authentication protocol [M-Pin]. M-Pin protocol includes a kind of zero-knowledge proof, where pairing is used for its construction.

Trusted Computing Group (TCG) specifies ECDAA (Elliptic Curve Direct Anonymous Attestation) in the specification of Trusted Platform Module (TPM) [TPM]. ECDAA is a protocol for proving the attestation held by a TPM to a verifier without revealing the attestation held by that TPM. Pairing is used for constructing ECDAA. FIDO Alliance [FIDO] and W3C [W3C] also published ECDAA algorithm similar to TCG.

Intel introduces Intel Enhanced Privacy ID (EPID) which enables remote attestation of a hardware device while preserving the privacy of the device as a functionality of Intel Software Guard Extensions (SGX) [EPID]. They extend TPM ECDAA to realize such functionality. A pairing-based EPID has been proposed [BL10] and distributed along with Intel SGX applications.

Zcash implements their own zero-knowledge proof algorithm named zk-SNARKs (Zero-Knowledge Succinct Non-Interactive Argument of Knowledge) [Zcash]. zk-SNARKs is used for protecting privacy of transactions of Zcash. They use pairing for constructing zk-SNARKS.

Cloudflare introduces Geo Key Manager [Cloudflare] to restrict distribution of customers' private keys to the subset of their data centers. To achieve this functionality, attribute-based encryption is used and pairing takes a role as a building block.

Recently, Boneh-Lynn-Shacham (BLS) signature schemes are being standardized [I-D.boneh-bls-signature] and utilized in several blockchain projects such as Ethereum [Ethereum], Algorand [Algorand], Chia Network [Chia] and DFINITY [DFINITY]. The aggregation functionality of BLS signatures is effective for their applications of decentralization and scalability.

1.3. Goal

The goal of this memo is to consider the security of pairing-friendly curves used in pairing-based cryptography and introduce secure parameters of pairing-friendly curves. Specifically, we explain the recent attack against pairing-friendly curves and how much the security of the curves is reduced. We show how to evaluate the security of pairing-friendly curves and give the parameters for 100 bits of security, which is no longer secure, 128, 192 and 256 bits of security.

1.4. Requirements Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Preliminaries

2.1. Elliptic Curve

Let p > 3 be a prime and q = p^n for a natural number n. Let F_q be a finite field. The curve defined by the following equation E is called an elliptic curve.

    E : y^2 = x^3 + A * x + B,

where x and y are in F_q, and A and B in F_q satisfy the discriminant inequality 4 * A^3 + 27 * B^2 != 0 mod q. This is called Weierstrass normal form of an elliptic curve.

Solutions (x, y) for an elliptic curve E, as well as the point at infinity, O_E, are called F_q-rational points. If P and Q are two points on the curve E, we can define R = P + Q as the opposite point of the intersection between the curve E and the line that passes through P and Q.
We can define P + O_E = P = O_E + P as well. Similarly, we can define 2P = P + P and a scalar multiplication S = [a]P for a positive integer a can be defined as an (a-1)-time addition of P.

The additive group, denoted by E(F_q), is constructed by the set of F_q-rational points and the addition law described above. We can define the cyclic additive group with a prime order r by taking a base point BP in E(F_q) as a generator. This group is used for the elliptic curve cryptography.

We define terminology used in this memo as follows.

O_E: the point at infinity over an elliptic curve E.

E(F_q): a group constructed by F_q-rational points of E.

#E(F_q): the number of F_q-rational points of E.

h: a cofactor such that h = #E(F_q) / r.

2.2. Pairing

Pairing is a kind of the bilinear map defined over two elliptic curves E and E'. Examples include Weil pairing, Tate pairing, optimal Ate pairing [Ver09] and so on. Especially, optimal Ate pairing is considered to be efficient to compute and mainly used for practical implementation.

Let E be an elliptic curve defined over a prime field F_p and E' be an elliptic curve defined over an extension field of F_p. Let k be a minimum integer such that r is a divisor of p^k - 1, which is called an embedding degree. Let G_1 be a cyclic subgroup on the elliptic curve E with order r, and G_2 be a cyclic subgroup on the elliptic curve E' with order r. Let G_T be an order r subgroup of a multiplicative group (F_p^k)^*.

Pairing is defined as a bilinear map e: (G_1, G_2) -> G_T satisfying the following properties:

1. Bilinearity: for any S in G_1, T in G_2, and integers a and b, e([a]S, [b]T) = e(S, T)^{a * b}.

2. Non-degeneracy: for any T in G_2, e(S, T) = 1 if and only if S = O_E. Similarly, for any S in G_1, e(S, T) = 1 if and only if T = O_E.

3. Computability: for any S in G_1 and T in G_2, the bilinear map is efficiently computable.

2.3. Barreto-Naehrig Curve

A BN curve [BN05] is one of the instantiations of pairing-friendly curves proposed in 2005. A pairing over BN curves constructs optimal Ate pairings.

A BN curve is defined by elliptic curves E and E' parameterized by a well chosen integer t. E is defined over F_p, where p is a prime more than or equal to 5, and E(F_p) has a subgroup of prime order r. The characteristic p and the order r are parameterized by

```
p = 36 * t^4 + 36 * t^3 + 24 * t^2 + 6 * t + 1
r = 36 * t^4 + 36 * t^3 + 18 * t^2 + 6 * t + 1
```

for an integer t.

The elliptic curve E has an equation of the form E: y^2 = x^3 + b, where b is an element of multiplicative group of order p.

BN curves always have order 6 twists. If m is an element which is neither a square nor a cube in an extension field F_p^2, the twisted curve E' of E is defined over an extension field F_p^2 by the equation E': y^2 = x^3 + b' with b' = b / m or b' = b * m. BN curves are called D-type if b' = b / m, and M-type if b' = b * m. The embedded degree k is 12.

A pairing e is defined by taking G_1 as a subgroup of E(F_p) of order r, G_2 as a subgroup of E'(F_p^2), and G_T as a subgroup of a multiplicative group (F_p^12)^* of order r.

2.4. Barreto-Lynn-Scott Curve

A BLS curve [BLS02] is another instantiations of pairings proposed in 2002. Similar to BN curves, a pairing over BLS curves constructs optimal Ate pairings.

A BLS curve is elliptic curves E and E' parameterized by a well chosen integer t. E is defined over a finite field F_p by an

equation of the form E: y^2 = x^3 + b, and its twisted curve, E': y^2 = x^3 + b', is defined in the same way as BN curves. In contrast to BN curves, E(F_p) does not have a prime order. Instead, its order is divisible by a large parameterized prime r and denoted by h * r with cofactor h. The pairing will be defined on the r-torsions points. In the same way as BN curves, BLS curves can be categorized into D-type and M-type.

BLS curves vary according to different embedding degrees. In this memo, we deal with BLS12 and BLS48 families with embedding degrees 12 and 48 with respect to r, respectively.

In BLS curves, parameterized p and r are given by the following equations:

```
BLS12:
    p = (t - 1)^2 * (t^4 - t^2 + 1) / 3 + t
    r = t^4 - t^2 + 1
BLS48:
    p = (t - 1)^2 * (t^16 - t^8 + 1) / 3 + t
    r = t^16 - t^8 + 1
```

for a well chosen integer t.

A pairing e is defined by taking G_1 as a subgroup of E(F_p) of order r, G_2 as an order r subgroup of E'(F_p^2) for BLS12 and of E'(F_p^8) for BLS48, and G_T as an order r subgroup of a multiplicative group (F_p^12)^* for BLS12 and of a multiplicative group (F_p^48)^* for BLS48.

2.5. Representation Convention for an Extension Field

Pairing-friendly curves use a tower of some extension fields. In order to encode an element of an extension field, we adopt the representation convention shown in Appendix J.4 of [I-D.draft-lwig-curve-representations] .

Let F_p be a finite field of characteristic p and F_p^d be an extension field of F_p of degree d and an indeterminate i.

For an element s in F_p^d such that s = s_0 + s_1 * i + ... + s_{d - 1} * i^{d - 1} for s_0, s_1, ... , s_{d - 1} in a basefield F_p, s is represented as octet string by oct(s) = s_0 || s_1 || ... || s_{d - 1}.

Let F_p^d' be an extension field of F_p^d of degree d' / d and an indeterminate j.

For an element s' in F_p^d' such that s' = s'_0 + s'_1 * j + ... + s'_{d' / d - 1} * j^{d' / d - 1} for s'_0, s'_1, ..., s'_{d' / d - 1} in a basefield F_p^d, s' is represented as integer by oct(s') = oct(s'_0) || oct(s'_1) || ... || oct(s'_{d' / d - 1}), where oct(s'_0), ... , oct(s'_{d' / d - 1}) are octet strings encoded by above convention.

In general, one can define encoding between integer and an element of any finite field tower by inductively applying the above convention.

The parameters and test vectors of extension fields described in this memo are encoded by this convention and represented in octet stream.

3. Security of Pairing-Friendly Curves

3.1. Evaluating the Security of Pairing-Friendly Curves

The security of pairing-friendly curves is evaluated by the hardness of the following discrete logarithm problems.

- The elliptic curve discrete logarithm problem (ECDLP) in G_1 and G_2

- The finite field discrete logarithm problem (FFDLP) in G_T

There are other hard problems over pairing-friendly curves used for proving the security of pairing-based cryptography. Such problems include computational bilinear Diffie-Hellman (CBDH) problem and bilinear Diffie-Hellman (BDH) Problem, decision bilinear Diffie-Hellman (DBDH) problem, gap DBDH problem, etc [ECRYPT]. Almost all of these variants are reduced to the hardness of discrete logarithm problems described above and believed to be easier than the discrete logarithm problems.

There would be the case where the attacker solves these reduced problems to break pairing-based cryptography. Since such attacks have not been discovered yet, we discuss the hardness of the discrete logarithm problems in this memo.

The security level of pairing-friendly curves is estimated by the computational cost of the most efficient algorithm to solve the above discrete logarithm problems. The well-known algorithms for solving the discrete logarithm problems include Pollard's rho algorithm [Pollard78], Index Calculus [HR83] and so on. In order to make index calculus algorithms more efficient, number field sieve (NFS) algorithms are utilized.

3.2. Impact of the Recent Attack

In 2016, Kim and Barbulescu proposed a new variant of the NFS algorithms, the extended tower number field sieve (exTNFS), which drastically reduces the complexity of solving FFDLP [KB16]. Due to exTNFS, the security level of pairing-friendly curves asymptotically dropped down. For instance, Barbulescu and Duquesne estimated that the security of the BN curves which had been believed to provide 128 bits of security (BN256, for example) dropped down to approximately 100 bits [BD18].

Some papers showed the minimum bit length of the parameters of pairing-friendly curves for each security level when applying exTNFS as an attacking method for FFDLP. For 128 bits of security, Menezes, Sarkar and Singh estimated the minimum bit length of p of BN curves after exTNFS as 383 bits, and that of BLS12 curves as 384 bits [MSS17]. For 256 bits of security, Kiyomura et al. estimated the minimum bit length of p^k of BLS48 curves as 27,410 bits, which implied 572 bits of p [KIK17].

4. Security Evaluation of Pairing-Friendly Curves

We give security evaluation for pairing-friendly curves based on the evaluating method presented in Section 3. We also introduce secure parameters of pairing-friendly curves for each security level. The parameters introduced here are chosen with the consideration of security, efficiency and global acceptance.

For security, we introduce the parameters with 100 bits, 128 bits, 192 bits and 256 bits of security. We note that 100 bits of security is no longer secure and recommend 128 bits, 192 bits and 256 bits of security for secure applications. We follow TLS 1.3 [RFC8446] which specifies the cipher suites with 128 bits and 256 bits of security as mandatory-to-implement for the choice of the security level.

Implementers of the applications have to choose the parameters with appropriate security level according to the security requirements of the applications. For efficiency, we refer to the benchmark by mcl [mcl] for 128 bits of security, and by Kiyomura et al. [KIK17] for 256 bits of security, and then choose sufficiently efficient parameters. For global acceptance, we give the implementations of pairing-friendly curves in Section 5.

4.1. For 100 Bits of Security

Before exTNFS, BN curves with 256-bit size of underlying finite field (so-called BN256) were considered to achieve 128 bits of security.

After exTNFS, however, the security level of BN curves with 256-bit size of underlying finite field fell into 100 bits.

Implementers who will newly develop the applications of pairing-based cryptography SHOULD NOT use pairing-friendly curves with 100 bits of security (i.e. BN256).

There exists applications which already implemented pairing-based cryptography with 100-bit secure pairing-friendly curves. In such a case, implementers MAY use 100 bits of security only if they need to keep interoperability with the existing applications.

4.2. For 128 Bits of Security

4.2.1. BN Curves

A BN curve with 128 bits of security is shown in [BD18], which we call BN462. BN462 is defined by a parameter

    t = 2^114 + 2^101 - 2^14 - 1

for the definition in Section 2.3.

For the finite field F_p, the towers of extension field F_p^2, F_p^6 and F_p^12 are defined by indeterminates u, v, w as follows:

    F_p^2 = F_p[u] / (u^2 + 1)
    F_p^6 = F_p^2[v] / (v^3 - u - 2)
    F_p^12 = F_p^6[w] / (w^2 - v).

Defined by t, the elliptic curve E and its twisted curve E' are represented by E: y^2 = x^3 + 5 and E': y^2 = x^3 - u + 2, respectively. The size of p becomes 462-bit length. A pairing e is defined by taking G_1 as a cyclic group of order r generated by a base point BP = (x, y) in F_p, G_2 as a cyclic group of order r generated by a based point BP' = (x', y') in F_p^2, and G_T as a subgroup of a multiplicative group (F_p^12)^* of order r. BN462 is D-type.

We give the following parameters for BN462.

- G_1 defined over E: y^2 = x^3 + b

  o p : a characteristic

  o r : an order

  o BP = (x, y) : a base point

   o h : a cofactor

   o b : a coefficient of E

- G_2 defined over E': y^2 = x^3 + b'

   o r' : an order

   o BP' = (x', y') : a base point (encoded with [I-D.draft-lwig-curve-representations])

      * x' = x'_0 + x'_1 * u (x'_0, x'_1 in F_p)

      * y' = y'_0 + y'_1 * u (y'_0, y'_1 in F_p)

   o h' : a cofactor

   o b' : a coefficient of E'

```
p: 0x240480360120023ffffffffff6ff0cf6b7d9bfca0000000000d812908f41c802
   0ffffffffff6ff66fc6ff687f640000000002401b00840138013

r: 0x240480360120023ffffffffff6ff0cf6b7d9bfca0000000000d812908ee1c201
   f7fffffffff6ff66fc7bf717f7c0000000002401b007e010800d

x: 0x21a6d67ef250191fadba34a0a30160b9ac9264b6f95f63b3edbec3cf4b2e689d
   b1bbb4e69a416a0b1e79239c0372e5cd70113c98d91f36b6980d

y: 0x0118ea0460f7f7abb82b33676a7432a490eeda842cccfa7d788c659650426e6a
   f77df11b8ae40eb80f475432c66600622ecaa8a5734d36fb03de

h: 1

b: 5

r':  0x240480360120023ffffffffff6ff0cf6b7d9bfca0000000000d812908ee1c2
   01f7fffffffff6ff66fc7bf717f7c0000000002401b007e010800d

x'_0:  0x0257ccc85b58dda0dfb38e3a8cbdc5482e0337e7c1cd96ed61c913820408
   208f9ad2699bad92e0032ae1f0aa6a8b48807695468e3d934ae1e4df

x'_1:  0x1d2e4343e8599102af8edca849566ba3c98e2a354730cbed9176884058b1
   8134dd86bae555b783718f50af8b59bf7e850e9b73108ba6aa8cd283

y'_0:  0x0a0650439da22c1979517427a20809eca035634706e23c3fa7a6bb42fe81
   0f1399a1f41c9ddae32e03695a140e7b11d7c3376e5b68df0db7154e
```

y'_1: 0x073ef0cbd438cbe0172c8ae37306324d44d5e6b0c69ac57b393f1ab370fd
725cc647692444a04ef87387aa68d53743493b9eba14cc552ca2a93a

h': 0x240480360120023ffffffffff6ff0cf6b7d9bfca0000000000d812908fa1ce
0227fffffffff6ff66fc63f5f7f4c0000000002401b008a0168019

b': -u + 2

4.2.2. BLS Curves

A BLS12 curve with 128 bits of security shown in [BLS12-381], BLS12-381, is defined by a parameter

t = -2^63 - 2^62 - 2^60 - 2^57 - 2^48 - 2^16

and the size of p becomes 381-bit length.

For the finite field F_p, the towers of extension field F_p^2, F_p^6 and F_p^12 are defined by indeterminates u, v, w as follows:

F_p^2 = F_p[u] / (u^2 + 1)
F_p^6 = F_p^2[v] / (v^3 - u - 1)
F_p^12 = F_p^6[w] / (w^2 - v).

Defined by t, the elliptic curve E and its twisted curve E' are represented by E: y^2 = x^3 + 4 and E': y^2 = x^3 + 4(u + 1).

A pairing e is defined by taking G_1 as a cyclic group of order r generated by a base point BP = (x, y) in F_p, G_2 as a cyclic group of order r generated by a based point BP' = (x', y') in F_p^2, and G_T as a subgroup of a multiplicative group (F_p^12)^* of order r. BLS12-381 is M-type.

We have to note that, according to [MSS17], the bit length of p for BLS12 to achieve 128 bits of security is calculated as 384 bits and more, which BLS12-381 does not satisfy. They state that BLS12-381 achieves 127-bit security level evaluated by the computational cost of Pollard's rho, whereas NCC group estimated that the security level of BLS12-381 is between 117 and 120 bits at most [NCCG]. Therefore, we regard BN462 as a "conservative" parameter, and BLS12-381 as an "optimistic" parameter.

We give the following parameters for BLS12-381.

- G_1 defined over E: y^2 = x^3 + b

  o p : a characteristic

o r : an order

o BP = (x, y) : a base point

o h : a cofactor

o b : a coefficient of E

- G_2 defined over E': y^2 = x^3 + b'

o r' : an order

o BP' = (x', y') : a base point (encoded with [I-D.draft-lwig-curve-representations])

* x' = x'_0 + x'_1 * u (x'_0, x'_1 in F_p)

* y' = y'_0 + y'_1 * u (y'_0, y'_1 in F_p)

o h' : a cofactor

o b' : a coefficient of E'

```
p: 0x1a0111ea397fe69a4b1ba7b6434bacd764774b84f38512bf6730d2a0f6b0f624
   1eabfffeb153ffffb9feffffffffaaab

r: 0x73eda753299d7d483339d80809a1d80553bda402fffe5bfeffffffff00000001

x: 0x17f1d3a73197d7942695638c4fa9ac0fc3688c4f9774b905a14e3a3f171bac58
   6c55e83ff97a1aeffb3af00adb22c6bb

y: 0x08b3f481e3aaa0f1a09e30ed741d8ae4fcf5e095d5d00af600db18cb2c04b3ed
   d03cc744a2888ae40caa232946c5e7e1

h: 0x396c8c005555e1568c00aaab0000aaab

b: 4

r':  0x1a0111ea397fe69a4b1ba7b6434bacd764774b84f38512bf6730d2a0f6b0f6
   241eabfffeb153ffffb9feffffffffaaab

x'_0:  0x024aa2b2f08f0a91260805272dc51051c6e47ad4fa403b02b4510b647ae3
   d1770bac0326a805bbefd48056c8c121bdb8

x'_1:  0x13e02b6052719f607dacd3a088274f65596bd0d09920b61ab5da61bbdc7f
   5049334cf11213945d57e5ac7d055d042b7e
```

y'_0: 0x0ce5d527727d6e118cc9cdc6da2e351aadfd9baa8cbdd3a76d429a695160d12c923ac9cc3baca289e193548608b82801

y'_1: 0x0606c4a02ea734cc32acd2b02bc28b99cb3e287e85a763af267492ab572e99ab3f370d275cec1da1aaa9075ff05f79be

h': 0x5d543a95414e7f1091d50792876a202cd91de4547085abaa68a205b2e5a7ddfa628f1cb4d9e82ef21537e293a6691ae1616ec6e786f0c70cf1c38e31c7238e5

b': 4 * (u + 1)

4.3. For 192 Bits of Security

(TBD)

4.4. For 256 Bits of Security

As shown in Section 3.2, it is unrealistic to achieve 256 bits of security by BN curves since the minimum size of p becomes too large to implement. Hence, we consider BLS48 for 256 bits of security.

A BLS48 curve with 256 bits of security is shown in [KIK17], which we call BLS48-581. It is defined by a parameter

t = -1 + 2^7 - 2^10 - 2^30 - 2^32.

For the finite field F_p, the towers of extension field F_p^2, F_p^4, F_p^8, F_p^24 and F_p^48 are defined by indeterminates u, v, w, z, s as follows:

```
F_p^2 = F_p[u] / (u^2 + 1)
F_p^4 = F_p^2[v] / (v^2 + u + 1)
F_p^8 = F_p^4[w] / (w^2 + v)
F_p^24 = F_p^8[z] / (z^3 + w)
F_p^48 = F_p^24[s] / (s^2 + z).
```

The elliptic curve E and its twisted curve E' are represented by E: y^2 = x^3 + 1 and E': y^2 = x^3 - 1 / w. A pairing e is defined by taking G_1 as a cyclic group of order r generated by a base point BP = (x, y) in F_p, G_2 as a cyclic group of order r generated by a based point BP' = (x', y') in F_p^8, and G_T as a subgroup of a multiplicative group (F_p^48)^* of order r. The size of p becomes 581-bit length. BLS48-581 is D-type.

We then give the parameters for BLS48-581 as follows.

- G_1 defined over E: y^2 = x^3 + b

o p : a characteristic

o r : a prime which divides an order of G_1

o BP = (x, y) : a base point

o h : a cofactor

o b : a coefficient of E

- G_2 defined over E': y^2 = x^3 + b'

o r' : an order

o BP' = (x', y') : a base point (encoded with [I-D.draft-lwig-curve-representations])

* x' = x'_0 + x'_1 * u + x'_2 * v + x'_3 * u * v + x'_4 * w + x'_5 * u * w + x'_6 * v * w + x'_7 * u * v * w (x'_0, ..., x'_7 in F_p)

* y' = y'_0 + y'_1 * u + y'_2 * v + y'_3 * u * v + y'_4 * w + y'_5 * u * w + y'_6 * v * w + y'_7 * u * v * w (y'_0, ..., y'_7 in F_p)

o h' : a cofactor

o b' : a coefficient of E'

p: 0x1280f73ff3476f313824e31d47012a0056e84f8d122131bb3be6c0f1f3975444
a48ae43af6e082acd9cd30394f4736daf68367a5513170ee0a578fdf721a4a48ac
3edc154e6565912b

r: 0x2386f8a925e2885e233a9ccc1615c0d6c635387a3f0b3cbe003fad6bc972c2e6
e741969d34c4c92016a85c7cd0562303c4ccbe599467c24da118a5fe6fcd671c01

x: 0x02af59b7ac340f2baf2b73df1e93f860de3f257e0e86868cf61abdbaedffb9f7
544550546a9df6f9645847665d859236ebdbc57db368b11786cb74da5d3a1e6d8c
3bce8732315af640

y: 0x0cefda44f6531f91f86b3a2d1fb398a488a553c9efeb8a52e991279dd41b720e
f7bb7beffb98aee53e80f678584c3ef22f487f77c2876d1b2e35f37aef7b926b57
6dbb5de3e2587a70

x'_0: 0x05d615d9a7871e4a38237fa45a2775debabbefc70344dbccb7de64db3a2e
f156c46ff79baad1a8c42281a63ca0612f400503004d80491f510317b797663221
54dec34fd0b4ace8bfab

```
x'_1:  0x07c4973ece2258512069b0e86abc07e8b22bb6d980e1623e9526f6da1230
   7f4e1c3943a00abfedf16214a76affa62504f0c3c7630d979630ffd75556a01afa
   143f1669b36676b47c57

x'_2:  0x01fccc70198f1334e1b2ea1853ad83bc73a8a6ca9ae237ca7a6d6957ccba
   b5ab6860161c1dbd19242ffae766f0d2a6d55f028cbdfbb879d5fea8ef4cded6b3
   f0b46488156ca55a3e6a

x'_3:  0x0be2218c25ceb6185c78d8012954d4bfe8f5985ac62f3e5821b7b92a393f
   8be0cc218a95f63e1c776e6ec143b1b279b9468c31c5257c200ca52310b8cb4e80
   bc3f09a7033cbb7feafe

x'_4:  0x038b91c600b35913a3c598e4caa9dd63007c675d0b1642b5675ff0e7c580
   5386699981f9e48199d5ac10b2ef492ae589274fad55fc1889aa80c65b5f746c9d
   4cbb739c3a1c53f8cce5

x'_5:  0x0c96c7797eb0738603f1311e4ecda088f7b8f35dcef0977a3d1a58677bb0
   37418181df63835d28997eb57b40b9c0b15dd7595a9f177612f097fc7960910fce
   3370f2004d914a3c093a

x'_6:  0x0b9b7951c6061ee3f0197a498908aee660dea41b39d13852b6db908ba2c0
   b7a449cef11f293b13ced0fd0caa5efcf3432aad1cbe4324c22d63334b5b0e205c
   3354e41607e60750e057

x'_7:  0x0827d5c22fb2bdec5282624c4f4aaa2b1e5d7a9defaf47b5211cf7417197
   28a7f9f8cfca93f29cff364a7190b7e2b0d4585479bd6aebf9fc44e56af2fc9e97
   c3f84e19da00fbc6ae34

y'_0:  0x00eb53356c375b5dfa497216452f3024b918b4238059a577e6f3b39ebfc4
   35faab0906235afa27748d90f7336d8ae5163c1599abf77eea6d659045012ab12c
   0ff323edd3fe4d2d7971

y'_1:  0x0284dc75979e0ff144da6531815fcadc2b75a422ba325e6fba01d7296473
   2fcbf3afb096b243b1f192c5c3d1892ab24e1dd212fa097d760e2e588b423525ff
   c7b111471db936cd5665

y'_2:  0x0b36a201dd008523e421efb70367669ef2c2fc5030216d5b119d3a480d37
   0514475f7d5c99d0e90411515536ca3295e5e2f0c1d35d51a652269cbc7c46fc3b
   8fde68332a526a2a8474

y'_3:  0x0aec25a4621edc0688223fbbd478762b1c2cded3360dcee23dd8b0e710e1
   22d2742c89b224333fa40dced2817742770ba10d67bda503ee5e578fb3d8b8a1e5
   337316213da92841589d

y'_4:  0x0d209d5a223a9c46916503fa5a88325a2554dc541b43dd93b5a959805f11
   29857ed85c77fa238cdce8a1e2ca4e512b64f59f430135945d137b08857fdddfcf
   7a43f47831f982e50137
```

```
y'_5:  0x07d0d03745736b7a513d339d5ad537b90421ad66eb16722b589d82e2055a
   b7504fa83420e8c270841f6824f47c180d139e3aafc198caa72b679da59ed8226c
   f3a594eedc58cf90bee4

y'_6:  0x0896767811be65ea25c2d05dfdd17af8a006f364fc0841b064155f14e4c8
   19a6df98f425ae3a2864f22c1fab8c74b2618b5bb40fa639f53dccc9e884017d9a
   a62b3d41faeafeb23986

y'_7:  0x035e2524ff89029d393a5c07e84f981b5e068f1406be8e50c87549b6ef8e
   ca9a9533a3f8e69c31e97e1ad0333ec719205417300d8c4ab33f748e5ac66e8406
   9c55d667ffcb732718b6

h: 0x85555841aaaec4ac

b: 1

r':  0x2386f8a925e2885e233a9ccc1615c0d6c635387a3f0b3cbe003fad6bc972c2
   e6e741969d34c4c92016a85c7cd0562303c4ccbe599467c24da118a5fe6fcd671c
   01

h':  0x170e915cb0a6b7406b8d94042317f811d6bc3fc6e211ada42e58ccfcb3ac07
   6a7e4499d700a0c23dc4b0c078f92def8c87b7fe63e1eea270db353a4ef4d38b59
   98ad8f0d042ea24c8f02be1c0c83992fe5d7725227bb27123a949e0876c0a8ce0a
   67326db0e955dcb791b867f31d6bfa62fbdd5f44a00504df04e186fae033f1eb43
   c1b1a08b6e086eff03c8fee9ebdd1e191a8a4b0466c90b389987de5637d5dd13da
   b33196bd2e5afa6cd19cf0fc3fc7db7ece1f3fac742626b1b02fcee04043b2ea96
   492f6afa51739597c54bb78aa6b0b99319fef9d09f768831018ee6564c68d054c6
   2f2e0b4549426fec24ab26957a669dba2a2b6945ce40c9aec6afdeda16c79e1554
   6cd7771fa544d5364236690ea06832679562a68731420ae52d0d35a90b8d10b688
   e31b6aee45f45b7a5083c71732105852decc888f64839a4de33b99521f0984a418
   d20fc7b0609530e454f0696fa2a8075ac01cc8ae3869e8d0fe1f3788ffac4c01aa
   2720e431da333c83d9663bfb1fb7a1a7b90528482c6be7892299030bb51a51dc7e
   91e9156874416bf4c26f1ea7ec578058563960ef92bbbb8632d3a1b695f954af10
   e9a78e40acffc13b06540aae9da5287fc4429485d44e6289d8c0d6a3eb2ece3501
   2452751839fb48bc14b515478e2ff412d930ac20307561f3a5c998e6bcbfebd97e
   ffc6433033a2361bfcdc4fc74ad379a16c6dea49c209b1

b':  -1 / w
```

5. Implementations of Pairing-Friendly Curves

We show the pairing-friendly curves selected by existing standards, cryptographic libraries and applications.

ISO/IEC 15946-5 [ISOIEC15946-5] shows examples of BN curves with the size of 160, 192, 224, 256, 384 and 512 bits of p. There is no action so far after the proposal of exTNFS.

TCG adopts an BN curve of 256 bits specified in ISO/IEC 15946-5 (TPM_ECC_BN_P256) and that of 638 bits specified by their own (TPM_ECC_BN_P638). FIDO Alliance [FIDO] and W3C [W3C] adopt the same BN curves as TCG, a 512-bit BN curve shown in ISO/IEC 15946-5 and another 256-bit BN curve.

Cryptographic libraries which implement pairings include PBC [PBC], mcl [mcl], RELIC [RELIC], TEPLA [TEPLA], AMCL [AMCL], Intel IPP [Intel-IPP] and a library by Kyushu University [BLS48].

Cloudflare published a new cryptographic library CIRCL (Cloudflare Interoperable, Reusable Cryptographic Library) in 2019 [CIRCL]. The plan for the implementation of secure pairing-friendly curves is stated in their roadmap.

MIRACL implements BN curves and BLS12 curves [MIRACL].

Zcash implements a BN curve (named BN128) in their library libsnark [libsnark]. After exTNFS, they propose a new parameter of BLS12 as BLS12-381 [BLS12-381] and publish its experimental implementation [zkcrypto].

Ethereum 2.0 adopts BLS12-381 (BLS12_381), BN curves with 254 bits of p (CurveFp254BNb) and 382 bits of p (CurveFp382_1 and CurveFp382_2) [go-bls]. Their implementation calls mcl [mcl] for pairing computation. Chia Network publishs their implementation [Chia] by integrating the RELIC toolkit [RELIC].

Table 1 shows the adoption of pairing-friendly curves in existing standards, cryptographic libraries and applications. In this table, the curves marked as (*) indicate that the security level is evaluated less than the one labeld in the table.

Name	100 bit	128 bit	192 bit	256 bit
ISO/IEC 15946-5	BN256	BN384		
TCG	BN256			
FIDO/W3C	BN256			
PBC	BN			
mcl	BN254 / BN_SNARK1	BN381_1 (*) / BN462 / BLS12-381		
RELIC	BN254 / BN256	BLS12-381 / BLS12-455		
TEPLA	BN254			
AMCL	BN254 / BN256	BLS12-381 (*) / BLS12-383 (*) / BLS12-461		BLS48
Intel IPP	BN256			
Kyushu Univ.				BLS48
MIRACL	BN254	BLS12		
Zcash	BN128 (CurveSNARK)	BLS12-381		
Ethereum	BN254	BN382 (*) / BLS12-381 (*)		
Chia Network		BLS12-381 (*)		

Table 1: Adoption of Pairing-Friendly Curves

6. Security Considerations

This memo entirely describes the security of pairing-friendly curves, and introduces secure parameters of pairing-friendly curves. We give these parameters in terms of security, efficiency and global acceptance. The parameters for 100, 128, 192 and 256 bits of security are introduced since the security level will different in the requirements of the pairing-based applications. Implementers can select these parameters according to their security requirements.

7. IANA Considerations

This document has no actions for IANA.

8. Acknowledgements

The authors would like to thank Akihiro Kato and Shoko Yonezawa for their significant contribution to the early version of this memo. The authors would also like to acknowledge Sakae Chikara, Hoeteck Wee, Sergey Gorbunov and Michael Scott for their valuable comments.

9. References

9.1. Normative References

[BD18] Barbulescu, R. and S. Duquesne, "Updating Key Size Estimations for Pairings", Journal of Cryptology, DOI 10.1007/s00145-018-9280-5, January 2018.

[BLS02] Barreto, P., Lynn, B., and M. Scott, "Constructing Elliptic Curves with Prescribed Embedding Degrees", Security in Communication Networks pp. 257-267, DOI 10.1007/3-540-36413-7_19, 2003.

[BN05] Barreto, P. and M. Naehrig, "Pairing-Friendly Elliptic Curves of Prime Order", Selected Areas in Cryptography pp. 319-331, DOI 10.1007/11693383_22, 2006.

[KB16] Kim, T. and R. Barbulescu, "Extended Tower Number Field Sieve: A New Complexity for the Medium Prime Case", Advances in Cryptology - CRYPTO 2016 pp. 543-571, DOI 10.1007/978-3-662-53018-4_20, 2016.

[KIK17] Kiyomura, Y., Inoue, A., Kawahara, Y., Yasuda, M., Takagi, T., and T. Kobayashi, "Secure and Efficient Pairing at 256-Bit Security Level", Applied Cryptography and Network Security pp. 59-79, DOI 10.1007/978-3-319-61204-1_4, 2017.

[MSS17] Menezes, A., Sarkar, P., and S. Singh, "Challenges with Assessing the Impact of NFS Advances on the Security of Pairing-Based Cryptography", Lecture Notes in Computer Science pp. 83-108, DOI 10.1007/978-3-319-61273-7_5, 2017.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[Ver09] Vercauteren, F., "Optimal Pairings", IEEE Transactions on Information Theory Vol. 56, pp. 455-461, DOI 10.1109/tit.2009.2034881, January 2010.

9.2. Informative References

[Algorand] Gorbunov, S., "Efficient and Secure Digital Signatures for Proof-of-Stake Blockchains", <https://medium.com/algorand/digital-signatures-for-blockchains-5820e15fbe95>.

[AMCL] The Apache Software Foundation, "The Apache Milagro Cryptographic Library (AMCL)", 2016, <https://github.com/apache/incubator-milagro-crypto>.

[BL10] Brickell, E. and J. Li, "Enhanced Privacy ID from Bilinear Pairing for Hardware Authentication and Attestation", 2010 IEEE Second International Conference on Social Computing, DOI 10.1109/socialcom.2010.118, August 2010.

[BLS12-381] Bowe, S., "BLS12-381: New zk-SNARK Elliptic Curve Construction", <https://electriccoin.co/blog/new-snark-curve/>.

[BLS48] Kyushu University, "bls48 - C++ library for Optimal Ate Pairing on BLS48", 2017, <https://github.com/mk-math-kyushu/bls48>.

[CCS07] Chen, L., Cheng, Z., and N. Smart, "Identity-based key agreement protocols from pairings", International Journal of Information Security Vol. 6, pp. 213-241, DOI 10.1007/s10207-006-0011-9, January 2007.

[Chia] Chia Network, "BLS signatures in C++, using the relic toolkit", <https://github.com/Chia-Network/bls-signatures>.

[CIRCL] Cloudflare, "CIRCL: Cloudflare Interoperable, Reusable Cryptographic Library", 2019, <https://github.com/cloudflare/circl>.

[Cloudflare]
Sullivan, N., "Geo Key Manager: How It Works", <https://blog.cloudflare.com/geo-key-manager-how-it-works/>.

[DFINITY] Williams, D., "DFINITY Technology Overview Series Consensus System Rev. 1", n.d., <https://dfinity.org/pdf-viewer/library/dfinity-consensus.pdf>.

[ECRYPT] ECRYPT, "Final Report on Main Computational Assumptions in Cryptography".

[EPID] Intel Corporation, "Intel (R) SGX: Intel (R) EPID Provisioning and Attestation Services", <https://software.intel.com/en-us/download/intel-sgx-intel-epid-provisioning-and-attestation-services>.

[Ethereum]
Jordan, R., "Ethereum 2.0 Development Update #17 - Prysmatic Labs", <https://medium.com/prysmatic-labs/ethereum-2-0-development-update-17-prysmatic-labs-ed5bcf82ec00>.

[FIDO] Lindemann, R., "FIDO ECDAA Algorithm - FIDO Alliance Review Draft 02", <https://fidoalliance.org/specs/fido-v2.0-rd-20180702/fido-ecdaa-algorithm-v2.0-rd-20180702.html>.

[FSU10] Fujioka, A., Suzuki, K., and B. Ustaoglu, "Ephemeral Key Leakage Resilient and Efficient ID-AKEs That Can Share Identities, Private and Master Keys", Lecture Notes in Computer Science pp. 187-205, DOI 10.1007/978-3-642-17455-1_12, 2010.

[go-bls] Prysmatic Labs, "go-bls - Go wrapper for a BLS12-381 Signature Aggregation implementation in C++", 2018, <https://godoc.org/github.com/prysmaticlabs/go-bls>.

[HR83] Hellman, M. and J. Reyneri, "Fast Computation of Discrete Logarithms in GF (q)", Advances in Cryptology pp. 3-13, DOI 10.1007/978-1-4757-0602-4_1, 1983.

[I-D.boneh-bls-signature]
Boneh, D., Gorbunov, S., Wee, H., and Z. Zhang, "BLS Signature Scheme", draft-boneh-bls-signature-00 (work in progress), February 2019.

[I-D.draft-lwig-curve-representations]
Struik, R., "Alternative Elliptic Curve Representations", draft-ietf-lwig-curve-representations-08 (work in progress), July 2019.

[Intel-IPP]
Intel Corporation, "Developer Reference for Intel Integrated Performance Primitives Cryptography 2019", 2018, <https://software.intel.com/en-us/ipp-crypto-reference-arithmetic-of-the-group-of-elliptic-curve-points>.

[ISOIEC11770-3]
ISO/IEC, "ISO/IEC 11770-3:2015", ISO/IEC Information technology -- Security techniques -- Key management -- Part 3: Mechanisms using asymmetric techniques, 2015.

[ISOIEC15946-5]
ISO/IEC, "ISO/IEC 15946-5:2017", ISO/IEC Information technology -- Security techniques -- Cryptographic techniques based on elliptic curves -- Part 5: Elliptic curve generation, 2017.

[Joux00] Joux, A., "A One Round Protocol for Tripartite Diffie-Hellman", Lecture Notes in Computer Science pp. 385-393, DOI 10.1007/10722028_23, 2000.

[libsnark]
SCIPR Lab, "libsnark: a C++ library for zkSNARK proofs", 2012, <https://github.com/zcash/libsnark>.

[M-Pin] Scott, M., "M-Pin: A Multi-Factor Zero Knowledge Authentication Protocol", July 2019, <https://www.miracl.com/miracl-labs/m-pin-a-multi-factor-zero-knowledge-authentication-protocol>.

[mcl] Mitsunari, S., "mcl - A portable and fast pairing-based cryptography library", 2016, <https://github.com/herumi/mcl>.

[MIRACL] MIRACL Ltd., "MIRACL Cryptographic SDK", 2018, <https://github.com/miracl/MIRACL>.

[NCCG] NCC Group, "Zcash Overwinter Consensus and Sapling Cryptography Review", <https://www.nccgroup.trust/us/our-research/zcash-overwinter-consensus-and-sapling-cryptography-review/>.

[PBC] Lynn, B., "PBC Library - The Pairing-Based Cryptography Library", 2006, <https://crypto.stanford.edu/pbc/>.

[Pollard78] Pollard, J., "Monte Carlo methods for index computation $({\rm mod}\ p)$", Mathematics of Computation Vol. 32, pp. 918-918, DOI 10.1090/s0025-5718-1978-0491431-9, September 1978.

[RELIC] Gouvea, C., "RELIC is an Efficient LIbrary for Cryptography", 2013, <https://github.com/relic-toolkit/relic>.

[RFC5091] Boyen, X. and L. Martin, "Identity-Based Cryptography Standard (IBCS) #1: Supersingular Curve Implementations of the BF and BB1 Cryptosystems", RFC 5091, DOI 10.17487/RFC5091, December 2007, <https://www.rfc-editor.org/info/rfc5091>.

[RFC6508] Groves, M., "Sakai-Kasahara Key Encryption (SAKKE)", RFC 6508, DOI 10.17487/RFC6508, February 2012, <https://www.rfc-editor.org/info/rfc6508>.

[RFC6509] Groves, M., "MIKEY-SAKKE: Sakai-Kasahara Key Encryption in Multimedia Internet KEYing (MIKEY)", RFC 6509, DOI 10.17487/RFC6509, February 2012, <https://www.rfc-editor.org/info/rfc6509>.

[RFC6539] Cakulev, V., Sundaram, G., and I. Broustis, "IBAKE: Identity-Based Authenticated Key Exchange", RFC 6539, DOI 10.17487/RFC6539, March 2012, <https://www.rfc-editor.org/info/rfc6539>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[SAKKE] 3GPP, "Security of the mission critical service (Release 15)", 3GPP TS 33.180 15.3.0, 2018.

[TEPLA] University of Tsukuba, "TEPLA: University of Tsukuba Elliptic Curve and Pairing Library", 2013, <http://www.cipher.risk.tsukuba.ac.jp/tepla/index_e.html>.

[TPM] Trusted Computing Group (TCG), "Trusted Platform Module Library Specification, Family \"2.0\", Level 00, Revision 01.38", <https://trustedcomputinggroup.org/resource/tpm-library-specification/>.

[W3C] Lundberg, E., "Web Authentication: An API for accessing Public Key Credentials Level 1 - W3C Recommendation", <https://www.w3.org/TR/webauthn/>.

[Zcash] Lindemann, R., "What are zk-SNARKs?", <https://z.cash/technology/zksnarks.html>.

[zkcrypto] zkcrypto, "zkcrypto - Pairing-friendly elliptic curve library", 2017, <https://github.com/zkcrypto/pairing>.

Appendix A. Computing Optimal Ate Pairing

Before presenting the computation of optimal Ate pairing e(P, Q) satisfying the properties shown in Section 2.2, we give subfunctions used for pairing computation.

The following algorithm Line_Function shows the computation of the line function. It takes A = (A[1], A[2]), B = (B[1], B[2]) in G_2 and P = ((P[1], P[2])) in G_1 as input and outputs an element of G_T.

```
if (A = B) then
    l := (3 * A[1]^2) / (2 * A[2]);
else if (A = -B) then
    return P[1] - A[1];
else
    l := (B[2] - A[2]) / (B[1] - A[1]);
end if;
return (l * (P[1] -A[1]) + A[2] -P[2]);
```

When implementing the line function, implementers should consider the isomorphism of E and its twisted curve E' so that one can reduce the computational cost of operations in G_2. We note that the function Line_function does not consider such isomorphism.

Computation of optimal Ate pairing for BN curves uses Frobenius map. Let a Frobenius map pi for a point Q = (x, y) over E' be pi(p, Q) = (x^p, y^p).

A.1. Optimal Ate Pairings over Barreto-Naehrig Curves

Let c = 6 * t + 2 for a parameter t and c_0, c_1, ... , c_L in {-1,0,1} such that the sum of c_i * 2^i (i = 0, 1, ..., L) equals to c.

The following algorithm shows the computation of optimal Ate pairing over Barreto-Naehrig curves. It takes P in G_1, Q in G_2, an integer c, c_0, ...,c_L in {-1,0,1} such that the sum of c_i * 2^i (i = 0, 1, ..., L) equals to c, and an order r as input, and outputs e(P, Q).

```
f := 1; T := Q;
if (c_L = -1)
    T := -T;
end if
for i = L-1 to 0
    f := f^2 * Line_function(T, T, P); T := 2 * T;
    if (c_i = 1 | c_i = -1)
        f := f * Line_function(T, c_i * Q); T := T + c_i * Q;
    end if
end for
Q_1 := pi(p, Q); Q_2 := pi(p, Q_1);
f := f * Line_function(T, Q_1, P); T := T + Q_1;
f := f * Line_function(T, -Q_2, P);
f := f^{(p^k - 1) / r}
return f;
```

A.2. Optimal Ate Pairings over Barreto-Lynn-Scott Curves

Let c = t for a parameter t and c_0, c_1, ... , c_L in {-1,0,1} such that the sum of c_i * 2^i (i = 0, 1, ..., L) equals to c. The following algorithm shows the computation of optimal Ate pairing over Barreto-Lynn-Scott curves. It takes P in G_1, Q in G_2, a parameter c, c_0, c_1, ..., c_L in {-1,0,1} such that the sum of c_i * 2^i (i = 0, 1, ..., L), and an order r as input, and outputs e(P, Q).

```
f := 1; T := Q;
if (c_L = -1)
    T := -T;
end if
for i = L-1 to 0
    f := f^2 * Line_function(T, T, P); T := 2 * T;
    if (c_i = 1 | c_i = -1)
        f := f * Line_function(T, c_i * Q, P); T := T + c_i * Q;
    end if
end for
f := f^{(p^k - 1) / r};
return f;
```

Appendix B. Test Vectors of Optimal Ate Pairing

We provide test vectors for Optimal Ate Pairing e(P, Q) given in Appendix A for the curves BN462, BLS12-381 and BLS48-581 given in Section 4. Here, the inputs P = (x, y) and Q = (x', y') are the corresponding base points BP and BP' given in Section 4.

For BN462 and BLS12-381, Q = (x', y') is given by

```
    x' = x'_0 + x'_1 * u and
    y' = y'_0 + y'_1 * u,
```

where u is a indeterminate and x'_0, x'_1, y'_0, y'_1 are elements of F_p.

For BLS48-581, Q = (x', y') is given by

```
  x' = x'_0 + x'_1 * u + x'_2 * v + x'_3 * u * v
      + x'_4 * w + x'_5 * u * w + x'_6 * v * w + x'_7 * u * v * w and
  y' = y'_0 + y'_1 * u + y'_2 * v + y'_3 * u * v
      + y'_4 * w + y'_5 * u * w + y'_6 * v * w + y'_7 * u * v * w,
```

where u, v and w are indeterminates and x'_0, ..., x'_7 and y'_0, ..., y'_7 are elements of F_p. The representation of Q = (x', y') given below is followed by [I-D.draft-lwig-curve-representations].

BN462:

```
Input x value:  0x21a6d67ef250191fadba34a0a30160b9ac9264b6f95f63b3edb
   ec3cf4b2e689db1bbb4e69a416a0b1e79239c0372e5cd70113c98d91f36b6980d

Input y value:  0x0118ea0460f7f7abb82b33676a7432a490eeda842cccfa7d788
   c659650426e6af77df11b8ae40eb80f475432c66600622ecaa8a5734d36fb03de

Input x'_0 value:  0x0257ccc85b58dda0dfb38e3a8cbdc5482e0337e7c1cd96ed
   61c913820408208f9ad2699bad92e0032ae1f0aa6a8b48807695468e3d934ae1e4
   df

Input x'_1 value:  0x1d2e4343e8599102af8edca849566ba3c98e2a354730cbed
   9176884058b18134dd86bae555b783718f50af8b59bf7e850e9b73108ba6aa8cd2
   83

Input y'_0 value:  0x0a0650439da22c1979517427a20809eca035634706e23c3f
   a7a6bb42fe810f1399a1f41c9ddae32e03695a140e7b11d7c3376e5b68df0db715
   4e
```

Input y'_1 value: 0x073ef0cbd438cbe0172c8ae37306324d44d5e6b0c69ac57b393f1ab370fd725cc647692444a04ef87387aa68d53743493b9eba14cc552ca2a93a

e_0: 0x0cf7f0f2e01610804272f4a7a24014ac085543d787c8f8bf07059f93f87ba7e2a4ac77835d4ff10e78669be39cd23cc3a659c093dbe3b9647e8c

e_1: 0x00ef2c737515694ee5b85051e39970f24e27ca278847c7cfa709b0df408b830b3763b1b001f1194445b62d6c093fb6f77e43e369edefb1200389

e_2: 0x04d685b29fd2b8faedacd36873f24a06158742bb2328740f93827934592d6f1723e0772bb9ccd3025f88dc457fc4f77dfef76104ff43cd430bf7

e_3: 0x090067ef2892de0c48ee49cbe4ff1f835286c700c8d191574cb424019de11142b3c722cc5083a71912411c4a1f61c00d1e8f14f545348eb7462c

e_4: 0x1437603b60dce235a090c43f5147d9c03bd63081c8bb1ffa7d8a2c31d673230860bb3dfe4ca85581f7459204ef755f63cba1fbd6a4436f10ba0e

e_5: 0x13191b1110d13650bf8e76b356fe776eb9d7a03fe33f82e3fe5732071f305d201843238cc96fd0e892bc61701e1844faa8e33446f87c6e29e75f

e_6: 0x07b1ce375c0191c786bb184cc9c08a6ae5a569dd7586f75d6d2de2b2f075787ee5082d44ca4b8009b3285ecae5fa521e23be76e6a08f17fa5cc8

e_7: 0x05b64add5e49574b124a02d85f508c8d2d37993ae4c370a9cda89a100cdb5e1d441b57768dbc68429ffae243c0c57fe5ab0a3ee4c6f2d9d34714

e_8: 0x0fd9a3271854a2b4542b42c55916e1faf7a8b87a7d10907179ac7073f6a1de044906ffaf4760d11c8f92df3e50251e39ce92c700a12e77d0adf3

e_9: 0x17fa0c7fa60c9a6d4d8bb9897991efd087899edc776f33743db921a689720c82257ee3c788e8160c112f18e841a3dd9a79a6f8782f771d542ee5

e_10: 0x0c901397a62bb185a8f9cf336e28cfb0f354e2313f99c538cdceedf8b8aa22c23b896201170fc915690f79f6ba75581f1b76055cd89b7182041c

e_11: 0x20f27fde93cee94ca4bf9ded1b1378c1b0d80439eeb1d0c8daef30db0037104a5e32a2ccc94fa1860a95e39a93ba51187b45f4c2c50c16482322

BLS12-381:

Input x value: 0x17f1d3a73197d7942695638c4fa9ac0fc3688c4f9774b905a14e3a3f171bac586c55e83ff97a1aeffb3af00adb22c6bb

Input y value: 0x08b3f481e3aaa0f1a09e30ed741d8ae4fcf5e095d5d00af600db18cb2c04b3edd03cc744a2888ae40caa232946c5e7e1

```
Input x'_0 value:  0x024aa2b2f08f0a91260805272dc51051c6e47ad4fa403b02
   b4510b647ae3d1770bac0326a805bbefd48056c8c121bdb8

Input x'_1 value:  0x13e02b6052719f607dacd3a088274f65596bd0d09920b61a
   b5da61bbdc7f5049334cf11213945d57e5ac7d055d042b7e

Input y'_0 value:  0x0ce5d527727d6e118cc9cdc6da2e351aadfd9baa8cbdd3a7
   6d429a695160d12c923ac9cc3baca289e193548608b82801

Input y'_1 value:  0x0606c4a02ea734cc32acd2b02bc28b99cb3e287e85a763af
   267492ab572e99ab3f370d275cec1da1aaa9075ff05f79be

e_0:  0x11619b45f61edfe3b47a15fac19442526ff489dcda25e59121d9931438907
   dfd448299a87dde3a649bdba96e84d54558

e_1:  0x153ce14a76a53e205ba8f275ef1137c56a566f638b52d34ba3bf3bf22f277
   d70f76316218c0dfd583a394b8448d2be7f

e_2:  0x095668fb4a02fe930ed44767834c915b283b1c6ca98c047bd4c272e9ac3f3
   ba6ff0b05a93e59c71fba77bce995f04692

e_3:  0x16deedaa683124fe7260085184d88f7d036b86f53bb5b7f1fc5e248814782
   065413e7d958d17960109ea006b2afdeb5f

e_4:  0x09c92cf02f3cd3d2f9d34bc44eee0dd50314ed44ca5d30ce6a9ec0539be7a
   86b121edc61839ccc908c4bdde256cd6048

e_5:  0x111061f398efc2a97ff825b04d21089e24fd8b93a47e41e60eae7e9b2a38d
   54fa4dedced0811c34ce528781ab9e929c7

e_6:  0x01ecfcf31c86257ab00b4709c33f1c9c4e007659dd5ffc4a735192167ce19
   7058cfb4c94225e7f1b6c26ad9ba68f63bc

e_7:  0x08890726743a1f94a8193a166800b7787744a8ad8e2f9365db76863e894b7
   a11d83f90d873567e9d645ccf725b32d26f

e_8:  0x0e61c752414ca5dfd258e9606bac08daec29b3e2c57062669556954fb227d
   3f1260eedf25446a086b0844bcd43646c10

e_9:  0x0fe63f185f56dd29150fc498bbeea78969e7e783043620db33f75a05a0a2c
   e5c442beaff9da195ff15164c00ab66bdde

e_10:  0x10900338a92ed0b47af211636f7cfdec717b7ee43900eee9b5fc24f0000c
   5874d4801372db478987691c566a8c474978

e_11:  0x1454814f3085f0e6602247671bc408bbce2007201536818c901dbd4d2095
   dd86c1ec8b888e59611f60a301af7776be3d
```

BLS48-581:

```
Input x value:  0x02af59b7ac340f2baf2b73df1e93f860de3f257e0e86868cf61
   abdbaedffb9f7544550546a9df6f9645847665d859236ebdbc57db368b11786cb7
   4da5d3a1e6d8c3bce8732315af640

Input y value:  0x0cefda44f6531f91f86b3a2d1fb398a488a553c9efeb8a52e99
   1279dd41b720ef7bb7beffb98aee53e80f678584c3ef22f487f77c2876d1b2e35f
   37aef7b926b576dbb5de3e2587a70

x'_0:  0x05d615d9a7871e4a38237fa45a2775debabbefc70344dbccb7de64db3a2e
   f156c46ff79baad1a8c42281a63ca0612f400503004d80491f510317b797663221
   54dec34fd0b4ace8bfab

x'_1:  0x07c4973ece2258512069b0e86abc07e8b22bb6d980e1623e9526f6da1230
   7f4e1c3943a00abfedf16214a76affa62504f0c3c7630d979630ffd75556a01afa
   143f1669b36676b47c57

x'_2:  0x01fccc70198f1334e1b2ea1853ad83bc73a8a6ca9ae237ca7a6d6957ccba
   b5ab6860161c1dbd19242ffae766f0d2a6d55f028cbdfbb879d5fea8ef4cded6b3
   f0b46488156ca55a3e6a

x'_3:  0x0be2218c25ceb6185c78d8012954d4bfe8f5985ac62f3e5821b7b92a393f
   8be0cc218a95f63e1c776e6ec143b1b279b9468c31c5257c200ca52310b8cb4e80
   bc3f09a7033cbb7feafe

x'_4:  0x038b91c600b35913a3c598e4caa9dd63007c675d0b1642b5675ff0e7c580
   5386699981f9e48199d5ac10b2ef492ae589274fad55fc1889aa80c65b5f746c9d
   4cbb739c3a1c53f8cce5

x'_5:  0x0c96c7797eb0738603f1311e4ecda088f7b8f35dcef0977a3d1a58677bb0
   37418181df63835d28997eb57b40b9c0b15dd7595a9f177612f097fc7960910fce
   3370f2004d914a3c093a

x'_6:  0x0b9b7951c6061ee3f0197a498908aee660dea41b39d13852b6db908ba2c0
   b7a449cef11f293b13ced0fd0caa5efcf3432aad1cbe4324c22d63334b5b0e205c
   3354e41607e60750e057

x'_7:  0x0827d5c22fb2bdec5282624c4f4aaa2b1e5d7a9defaf47b5211cf7417197
   28a7f9f8cfca93f29cff364a7190b7e2b0d4585479bd6aebf9fc44e56af2fc9e97
   c3f84e19da00fbc6ae34

y'_0:  0x00eb53356c375b5dfa497216452f3024b918b4238059a577e6f3b39ebfc4
   35faab0906235afa27748d90f7336d8ae5163c1599abf77eea6d659045012ab12c
   0ff323edd3fe4d2d7971
```

```
y'_1:  0x0284dc75979e0ff144da6531815fcadc2b75a422ba325e6fba01d7296473
   2fcbf3afb096b243b1f192c5c3d1892ab24e1dd212fa097d760e2e588b423525ff
   c7b111471db936cd5665

y'_2:  0x0b36a201dd008523e421efb70367669ef2c2fc5030216d5b119d3a480d37
   0514475f7d5c99d0e90411515536ca3295e5e2f0c1d35d51a652269cbc7c46fc3b
   8fde68332a526a2a8474

y'_3:  0x0aec25a4621edc0688223fbbd478762b1c2cded3360dcee23dd8b0e710e1
   22d2742c89b224333fa40dced2817742770ba10d67bda503ee5e578fb3d8b8a1e5
   337316213da92841589d

y'_4:  0x0d209d5a223a9c46916503fa5a88325a2554dc541b43dd93b5a959805f11
   29857ed85c77fa238cdce8a1e2ca4e512b64f59f430135945d137b08857fdddfcf
   7a43f47831f982e50137

y'_5:  0x07d0d03745736b7a513d339d5ad537b90421ad66eb16722b589d82e2055a
   b7504fa83420e8c270841f6824f47c180d139e3aafc198caa72b679da59ed8226c
   f3a594eedc58cf90bee4

y'_6:  0x0896767811be65ea25c2d05dfdd17af8a006f364fc0841b064155f14e4c8
   19a6df98f425ae3a2864f22c1fab8c74b2618b5bb40fa639f53dccc9e884017d9a
   a62b3d41faeafeb23986

y'_7:  0x035e2524ff89029d393a5c07e84f981b5e068f1406be8e50c87549b6ef8e
   ca9a9533a3f8e69c31e97e1ad0333ec719205417300d8c4ab33f748e5ac66e8406
   9c55d667ffcb732718b6

e_0:  0x0e26c3fcb8ef67417814098de5111ffcccc1d003d15b367bad07cef2291a9
   3d31db03e3f03376f3beae2bd877bcfc22a25dc51016eda1ab56ee3033bc4b4fec
   5962f02dffb3af5e38e

e_1:  0x069061b8047279aa5c2d25cdf676ddf34eddbc8ec2ec0f03614886fa828e1
   fc066b26d35744c0c38271843aa4fb617b57fa9eb4bd256d17367914159fc18b10
   a1085cb626e5bedb145

e_2:  0x02b9bece645fbf9d8f97025a1545359f6fe3ffab3cd57094f862f7fb9ca01
   c88705c26675bcc723878e943da6b56ce25d063381fcd2a292e0e7501fe5727441
   84fb4ab4ca071a04281

e_3:  0x0080d267bf036c1e61d7fc73905e8c630b97aa05ef3266c82e7a111072c0d
   2056baa8137fba111c9650dfb18cb1f43363041e202e3192fced29d2b0501c8825
   43fb370a56bfdc2435b

e_4:  0x03c6b4c12f338f9401e6a493a405b33e64389338db8c5e592a8dd79eac772
   0dd83dd6b0c189eeda20809160cd57cdf3e2edc82db15f553c1f6c953ea27114cb
   6bd8a38e273f407dae0
```

e_5: 0x016e46224f28bfd8833f76ac29ee6e406a9da1bde55f5e82b3bd977897a91
04f18b9ee41ea9af7d4183d895102950a12ce9975669db07924e1b432d9680f5ce
7e5c67ed68f381eba45

e_6: 0x008ddce7a4a1b94be5df3ceea56bef0077dcdde86d579938a50933a47296d
337b7629934128e2457e24142b0eeaa978fd8e70986d7dd51fccbbeb8a1933434f
ec4f5bc538de2646e90

e_7: 0x060ef6eae55728e40bd4628265218b24b38cdd434968c14bfefb87f0dcbfc
76cc473ae2dc0cac6e69dfdf90951175178dc75b9cc08320fcde187aa58ea047a2
ee00b1968650eec2791

e_8: 0x0c3943636876fd4f9393414099a746f84b2633dfb7c36ba6512a0b48e66dc
b2e409f1b9e150e36b0b4311165810a3c721525f0d43a021f090e6a27577b42c7a
57bed3327edb98ba8f8

e_9: 0x02d31eb8be0d923cac2a8eb6a07556c8951d849ec53c2848ee78c5eed4026
2eb21822527a8555b071f1cd080e049e5e7ebfe2541d5b42c1e414341694d6f16d
287e4a8d28359c2d2f9

e_10: 0x07f19673c5580d6a10d09a032397c5d425c3a99ff1dd0abe5bec40a0d47a
6b8daabb22edb6b06dd8691950b8f23faefcdd80c45aa3817a840018965941f424
7f9f97233a84f58b262e

e_11: 0x0d3fe01f0c114915c3bdf8089377780076c1685302279fd9ab12d07477aa
c03b69291652e9f179baa0a99c38aa8851c1d25ffdb4ded2c8fe8b30338c144286
07d6d822610d41f51372

e_12: 0x0662eefd5fab9509aed968866b68cff3bc5d48ecc8ac6867c212a2d82cee
5a689a3c9c67f1d611adac7268dc8b06471c0598f7016ca3d1c01649dda4b43531
cffc4eb41e691e27f2eb

e_13: 0x0aad8f4a8cfdca8de0985070304fe4f4d32f99b01d4ea50d9f7cd2abdc0a
eea99311a36ec6ed18208642cef9e09b96795b27c42a5a744a7b01a617a91d9fb7
623d636640d61a6596ec

e_14: 0x0ffcf21d641fd9c6a641a749d80cab1bcad4b34ee97567d905ed9d5cfb74
e9aef19674e2eb6ce3dfb706aa814d4a228db4fcd707e571259435393a27cac68b
59a1b690ae8cde7a94c3

e_15: 0x0cbe92a53151790cece4a86f91e9b31644a86fc4c954e5fa04e707beb69f
c60a858fed8ebd53e4cfd51546d5c0732331071c358d721ee601bfd3847e0e9041
01c62822dd2e4c7f8e5c

e_16: 0x0202db83b1ff33016679b6cfc8931deea6df1485c894dcd113bacf564411
519a42026b5fda4e16262674dcb3f089cd7d552f8089a1fec93e3db6bca43788cd
b06fc41baaa5c5098667

```
e_17:  0x070a617ed131b857f5b74b625c4ef70cc567f619defb5f2ab67534a1a8aa
   72975fc4248ac8551ce02b68801703971a2cf1cb934c9c354cadd5cfc4575cde8d
   bde6122bd54826a9b3e9

e_18:  0x070e1ebce457c141417f88423127b7a7321424f64119d5089d883cb95328
   3ee4e1f2e01ffa7b903fe7a94af4bb1acb02ca6a36678e41506879069cee11c9dc
   f6a080b6a4a7c7f21dc9

e_19:  0x058a06be5a36c6148d8a1287ee7f0e725453fa1bb05cf77239f235b41712
   7e370cfa4f88e61a23ea16df3c45d29c203d04d09782b39e9b4037c0c4ac8e8653
   e7c533ad752a640b233e

e_20:  0x0dfdfaaeb9349cf18d21b92ad68f8a7ecc509c35fcd4b8abeb93be7a204a
   c871f2195180206a2c340fccb69dbc30b9410ed0b122308a8fc75141f673ae5ec8
   2b6a45fc2d664409c6b6

e_21:  0x0d06c8adfdd81275da2a0ce375b8df9199f3d359e8cf50064a3dc10a5924
   17124a3b705b05a7ffe78e20f935a08868ecf3fc5aba0ace7ce4497bb59085ca27
   7c16b3d53dd7dae5c857

e_22:  0x0708effd28c4ae21b6969cb9bdd0c27f8a3e341798b6f6d4baf27be259b4
   a47688b50cb68a69a917a4a1faf56cec93f69ac416512c32e9d5e69bd8836b6c2b
   a9c6889d507ad571dbc4

e_23:  0x09da7c7aa48ce571f8ece74b98431b14ae6fb4a53ae979cd6b2e82320e8d
   25a0ece1ca1563aa5aa6926e7d608358af8399534f6b00788e95e37ef1b549f43a
   58ad250a71f0b2fdb2bf

e_24:  0x0a7150a14471994833d89f41daeaa999dfc24a9968d4e33d88ed9e9f07aa
   2432c53e486ba6e3b6e4f4b8d9c989010a375935c06e4b8d6c31239fad6a61e264
   7b84a0e3f76e57005ff7

e_25:  0x084696f31ff27889d4dccdc4967964a5387a5ae071ad391c5723c9034f16
   c2557915ada07ec68f18672b5b2107f785c15ddf9697046dc633b5a23cc0e442d2
   8ef6eea9915d0638d4d8

e_26:  0x0398e76e3d2202f999ac0f73e0099fe4e0fe2de9d223e78fc65c56e209cd
   f48f0d1ad8f6093e924ce5f0c93437c11212b7841de26f9067065b1898f48006bc
   c6f2ab8fa8e0b93f4ba4

e_27:  0x06d683f556022368e7a633dc6fe319fd1d4fc0e07acff7c4d4177e83a911
   e73313e0ed980cd9197bd17ac45942a65d90e6cb9209ede7f36c10e009c9d337ee
   97c4068db40e34d0e361

e_28:  0x0d764075344b70818f91b13ee445fd8c1587d1c0664002180bbac9a396ad
   4a8dc1e695b0c4267df4a09081c1e5c256c53fd49a73ffc817e65217a44fc0b20e
   f5ee92b28d4bc3e38576
```

```
e_29:  0x0aa6a32fdc4423b1c6d43e5104159bcd8e03a676d055d4496f7b1bc87611
         64a2908a3ff0e4c4d1f4362015c14824927011e2909531b8d87ee0acd676e7221a
         1ca1c21a33e2cf87dc51

e_30:  0x1147719959ac8eeab3fc913539784f1f947df47066b6c0c1beafecdb5fa7
         84c3be9de5ab282a678a2a0cbef8714141a6c8aaa76500819a896b46af20509953
         495e2a85eff58348b38d

e_31:  0x11a377bcebd3c12702bb34044f06f8870ca712fb5caa6d30c48ace96898f
         cbcddbcf31f331c9e524684c02c90db7f30b9fc470d6e651a7e8b1f684383f3705
         d7a47a1b4fe463d623c8

e_32:  0x0b8b4511f451ba2cc58dc28e56d5e1d0a8f557ecb242f4d994a627e07cf3
         fa44e6d83cb907deacf303d2f761810b5d943b46c4383e1435ec23fec196a70e33
         946173c78be3c75dfc83

e_33:  0x090962d632ee2a57ce4208052ce47a9f76ea0fdad724b7256bb07f3944e9
         639a981d3431087241e30ae9bf5e2ea32af323ce7ed195d383b749cb25bc09f678
         d385a49a0c09f6d9efca

e_34:  0x0931c7befc80acd185491c68af886fa8ee39c21ed3ebd743b9168ae3b298
         df485bfdc75b94f0b21aecd8dca941dfc6d1566cc70dc648e6ccc73e4cbf2a1ac8
         3c8294d447c66e74784d

e_35:  0x020ac007bf6c76ec827d53647058aca48896916269c6a2016b8c06f01309
         01c8975779f1672e581e2dfdbcf504e96ecf6801d0d39aad35cf79fbe7fe193c6c
         882c15bce593223f0c7c

e_36:  0x0c0aed0d890c3b0b673bf4981398dcbf0d15d36af6347a39599f3a225841
         84828f78f91bbbbd08124a97672963ec313ff142c456ec1a2fc3909fd4429fd699
         d827d48777d3b0e0e699

e_37:  0x0ef7799241a1ba6baaa8740d5667a1ace50fb8e63accc3bc30dc07b11d78
         dc545b68910c027489a0d842d1ba3ac406197881361a18b9fe337ff22d730fa44a
         fabb9f801f759086c8e4

e_38:  0x016663c940d062f4057257c8f4fb9b35e82541717a34582dd7d55b41ebad
         f40d486ed74570043b2a3c4de29859fdeae9b6b456cb33bb401ecf38f968564669
         2300517e9b035d6665fc

e_39:  0x1184a79510edf25e3bd2dc793a5082fa0fed0d559fa14a5ce9ffca4c61f1
         7196e1ffbb84326272e0d079368e9a735be1d05ec80c20dc6198b50a22a765defd
         c151d437335f1309aced

e_40:  0x120e47a747d942a593d202707c936dafa6fed489967dd94e48f317fd3c88
         1b1041e3b6bbf9e8031d44e39c1ab5ae41e487eac9acd90e869129c38a8e6c97cf
         55d6666d22299951f91a
```

e_41: 0x026b6e374108ecb2fe8d557087f40ab7bac8c5af0644a655271765d57ad7
1742aa331326d871610a8c4c30ccf5d8adbeec23cdff20d9502a5005fce2593caf
0682c82e4873b89d6d71

e_42: 0x041be63a2fa643e5a66faeb099a3440105c18dca58d51f74b3bf281da4e6
89b13f365273a2ed397e7b1c26bdd4daade710c30350318b0ae9a9b16882c29fe3
1ca3b884c92916d6d07a

e_43: 0x124018a12f0f0af881e6765e9e81071acc56ebcddadcd107750bd8697440
cc16f190a3595633bb8900e6829823866c5769f03a306f979a3e039e620d6d2f57
6793d36d840b168eeedd

e_44: 0x0d422de4a83449c535b4b9ece586754c941548f15d50ada6740865be9c0b
066788b6078727c7dee299acc15cbdcc7d51cdc5b17757c07d9a9146b01d2fdc7b
8c562002da0f9084bde5

e_45: 0x1119f6c5468bce2ec2b450858dc073fea4fb05b6e83dd20c55c9cf694cbc
c57fc0effb1d33b9b5587852d0961c40ff114b7493361e4cfdff16e85fbce66786
9b6f7e9eb804bcec46db

e_46: 0x061eaa8e9b0085364a61ea4f69c3516b6bf9f79f8c79d053e646ea637215
cf6590203b275290872e3d7b258102dd0c0a4a310af3958165f2078ff9dc3ac9e9
95ce5413268d80974784

e_47: 0x0add8d58e9ec0c9393eb8c4bc0b08174a6b421e15040ef558da58d241e5f
906ad6ca2aa5de361421708a6b8ff6736efbac6b4688bf752259b4650595aa395c
40d00f4417f180779985

Authors' Addresses

Yumi Sakemi
Lepidum

Email: yumi.sakemi@lepidum.co.jp

Tetsutaro Kobayashi
NTT

Email: tetsutaro.kobayashi.dr@hco.ntt.co.jp

Tsunekazu Saito
NTT

Email: tsunekazu.saito.hg@hco.ntt.co.jp