Alternative Elliptic Curve Representations
draft-struik-lwig-curve-representations-02

Abstract

   This document specifies how to represent Montgomery curves and
   (twisted) Edwards curves as curves in short-Weierstrass form and
   illustrates how this can be used to implement elliptic curve
   computations using existing implementations that already implement,
   e.g., ECDSA and ECDH using NIST prime curves.

Requirements Language

   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 RFC
   2119 [RFC2119].

Status of This Memo

Copyright Notice

Table of Contents

1. Fostering Code Reuse with New Elliptic Curves

It is well-known that elliptic curves can be represented using
different curve models. Recently, IETF standardized elliptic curves
that are claimed to have better performance and improved robustness
against "real world" attacks than curves represented in the
traditional "short" Weierstrass model. This draft specifies an
alternative representation of points of Curve25519, a so-called
Montgomery curve, and of points of Edwards25519, a so-called twisted
Edwards curve, which are both specified in [RFC7748], as points of a
specific so-called "short" Weierstrass curve, called Wei25519. The
draft also defines how to efficiently switch between these different
representations.

Use of Wei25519 allows easy definition of signature schemes and key
agreement schemes already specified for traditional NIST prime
curves, thereby allowing easy integration with existing
specifications, such as NIST SP 800-56a [SP-800-56a], FIPS Pub 186-4
[FIPS-186-4], and ANSI X9.62-2005 [ANSI-X9.62] and fostering code
reuse on platforms that already implement some of these schemes using
elliptic curve arithmetic for curves in "short" Weierstrass form (see
Appendix B.1).

2. Specification of Wei25519

For the specification of Wei25519 and its relationship to Curve25519
and Edwards25519, see Appendix D. For further details and background
information on elliptic curves, we refer to the other appendices.

The use of Wei25519 allows reuse of existing generic code that
implements short-Weierstrass curves, such as the NIST curve P256, to
also implement the CFRG curves Curve25519 and Ed25519. The draft
also caters to reuse of existing code where some domain parameters
may have been hardcoded, thereby widening the scope of applicability;
see Appendix F.

3. Example Uses

3.1. ECDSA-SHA256-25519

RFC 8032 [RFC8032] specifies the use of EdDSA, a "full" Schnorr
signature scheme, with instantiation by Edwards25519 and Ed448, two
so-called twisted Edwards curves. These curves can also be used with
the widely implemented signature scheme ECDSA [FIPS-186-4], by
instantiating ECDSA with the curve Wei25519 and hash function SHA-
256, where "under the hood" an implementation may carry out elliptic
curve scalar multiplication routines using the corresponding
representations of a point of the curve Wei25519 in Weierstrass form

as a point of the Montgomery curve Curve25519 or of the twisted
Edwards curve Edwards25519. (The corresponding ECDSA-SHA512-448
scheme arises if one were to specify a curve in short-Weierstrass
form corresponding to Ed448 and use the hash function SHA512.) Note
that, in either case, one can implement these schemes with the same
representation conventions as used with existing NIST specifications,
including bit/byte-ordering, compression functions, and the-like.
This allows implementations of ECDSA with the hash function SHA-256
and with the NIST curve P-256 or with the curve Wei25519 specified in
this draft to use the same implementation (instantiated with,
respectively, the NIST P-256 elliptic curve domain parameters or with
the domain parameters of curve Wei25519 specified in Appendix D).

3.2. Other Uses

Any existing specification of cryptographic schemes using elliptic
curves in Weierstrass form and that allows introduction of a new
elliptic curve (here: Wei25519) is amenable to similar constructs,
thus spawning "offspring" protocols, simply by instantiating these
using the new curve in "short" Weierstrass form, thereby allowing
code and/or specifications reuse and, for implementations that so
desire, carrying out curve computations "under the hood" on
Montgomery curve and twisted Edwards curve cousins hereof (where
these exist). This would simply require definition of a new object
identifier for any such envisioned "offspring" protocol. This could
significantly simplify standardization of schemes and help keeping
the resource and maintenance cost of implementations supporting
algorithm agility [RFC7696] at bay.

4. Security Considerations

The different representations of elliptic curve points discussed in
this draft are all obtained using a publicly known transformation.
Since this transformation is an isomorphism, this transformation maps
elliptic curve points to equivalent mathematical objects.

5. IANA Considerations

There is *currently* no IANA action required for this document. New
object identifiers would be required in case one wishes to specify
one or more of the "offspring" protocols exemplified in Section 3.

6. Normative References

   [ANSI-X9.62]
              ANSI X9.62-2005, "Public Key Cryptography for the
              Financial Services Industry: The Elliptic Curve Digital
              Signature Algorithm (ECDSA)", American National Standard
              for Financial Services, Accredited Standards Committee X9,
              Inc Anapolis, MD, 2005.

   [FIPS-186-4]
              FIPS 186-4, "Digital Signature Standard (DSS), Federal
              Information Processing Standards Publication 186-4", US
              Department of Commerce/National Institute of Standards and
              Technology Gaithersburg, MD, July 2013.

   [GECC]   D. Hankerson, A.J. Menezes, S.A. Vanstone, "Guide to
              Elliptic Curve Cryptography", New York: Springer-Verlag,
              2004.

   [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>.

   [RFC5639] Lochter, M. and J. Merkle, "Elliptic Curve Cryptography
              (ECC) Brainpool Standard Curves and Curve Generation",
              RFC 5639, DOI 10.17487/RFC5639, March 2010,
              <https://www.rfc-editor.org/info/rfc5639>.

   [RFC7696] Housley, R., "Guidelines for Cryptographic Algorithm
              Agility and Selecting Mandatory-to-Implement Algorithms",
              BCP 201, RFC 7696, DOI 10.17487/RFC7696, November 2015,
              <https://www.rfc-editor.org/info/rfc7696>.

   [RFC7748] Langley, A., Hamburg, M., and S. Turner, "Elliptic Curves
              for Security", RFC 7748, DOI 10.17487/RFC7748, January
              2016, <https://www.rfc-editor.org/info/rfc7748>.

   [RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital
              Signature Algorithm (EdDSA)", RFC 8032,
              DOI 10.17487/RFC8032, January 2017,
              <https://www.rfc-editor.org/info/rfc8032>.

   [SP-800-56a]
              NIST SP 800-56a, "Recommendation for Pair-Wise Key
              Establishment Schemes Using Discrete Log Cryptography,
              Revision 2", US Department of Commerce/National Institute
              of Standards and Technology Gaithersburg, MD, June 2013.

[Appendix A](#). Some (non-Binary) Elliptic Curves

[A.1](#). Curves in short-Weierstrass Form

   Let GF(q) denote the finite field with q elements, where q is an odd
   prime power and where q is not divisible by three. Let W_{a,b} be
   the Weierstrass curve with defining equation y^2 = x^3 + a*x + b,
   where a and b are elements of GF(q) and where 4*a^3 + 27*b^2 is
   nonzero. The points of W_{a,b} are the ordered pairs (x, y) whose
   coordinates are elements of GF(q) and that satisfy the defining
   equation (the so-called affine points), together with the special
   point O (the so-called "point at infinity").This set forms a group
   under addition, via the so-called "chord-and-tangent" rule, where the
   point at infinity serves as the identity element. See [Appendix B.1](#)
   for details of the group operation.

[A.2](#). Montgomery Curves

   Let GF(q) denote the finite field with q elements, where q is an odd
   prime power. Let M_{A,B} be the Montgomery curve with defining
   equation B*v^2 = u^3 + A*u^2 + u, where A and B are elements of GF(q)
   with A unequal to (+/-)2 and with B nonzero. The points of M_{A,B}
   are the ordered pairs (u, v) whose coordinates are elements of GF(q)
   and that satisfy the defining equation (the so-called affine points),
   together with the special point O (the so-called "point at
   infinity").This set forms a group under addition, via the so-called
   "chord-and-tangent" rule, where the point at infinity serves as the
   identity element. See [Appendix B.2](#) for details of the group
   operation.

[A.3](#). Twisted Edwards Curves

   Let GF(q) denote the finite field with q elements, where q is an odd
   prime power. Let E_{a,d} be the twisted Edwards curve with defining
   equation a*x^2 + y^2 = 1+ d*x^2*y^2, where a and d are distinct
   nonzero elements of GF(q). The points of E_{a,d} are the ordered
   pairs (x, y) whose coordinates are elements of GF(q) and that satisfy
   the defining equation (the so-called affine points). It can be shown
   that this set forms a group under addition if a is a square in GF(q),
   whereas d is not, where the point (0, 1) serves as the identity
   element. (Note that the identity element satisfies the defining
   equation.) See [Appendix B.3](#) for details of the group operation. An
   Edwards curve is a twisted Edwards curve with a=1.

Appendix B. Elliptic Curve Group Operations

B.1. Group Law for Weierstrass Curves

   For each point P of the Weierstrass curve W_{a,b}, the point at
   infinity O serves as identity element, i.e., P + O = O + P = P.

   For each affine point P:=(x, y) of the Weierstrass curve W_{a,b}, the
   point -P is the point (x, -y) and one has P + (-P) = O.

   Let P1:=(x1, y1) and P2:=(x2, y2) be distinct affine points of the
   Weierstrass curve W_{a,b} and let Q:=P1 + P2, where Q is not the
   identity element. Then Q:=(x, y), where

      x + x1 + x2 = lambda^2 and y + y1 = lambda*(x1 - x), where lambda
      = (y2 - y1)/(x2 - x1).

   Let P:= (x1, y1) be an affine point of the Weierstrass curve W_{a,b}
   and let Q:=2P, where Q is not the identity element. Then Q:= (x, y),
   where

      x + 2*x1 = lambda^2 and y + y1 = lambda*(x1 - x), where
      lambda=(3*x1^2 + a)/(2*y1).

B.2. Group Law for Montgomery Curves

   For each point P of the Montgomery curve M_{A,B}, the point at
   infinity O serves as identity element, i.e., P + O = O + P = P.

   For each affine point P:=(x, y) of the Montgomery curve M_{A,B}, the
   point -P is the point (x, -y) and one has P + (-P) = O.

   Let P1:=(x1, y1) and P2:=(x2, y2) be distinct affine points of the
   Montgomery curve M_{A,B} and let Q:=P1 + P2, where Q is not the
   identity element. Then Q:=(x, y), where

      x + x1 + x2 = B*lambda^2 - A and y + y1 = lambda*(x1 - x), where
      lambda=(y2 - y1)/(x2 - x1).

   Let P:= (x1, y1) be an affine point of the Montgomery curve M_{A,B}
   and let Q:=2P, where Q is not the identity element. Then Q:= (x, y),
   where

      x + 2*x1 = B*lambda^2 - A and y + y1 = lambda*(x1 - x), where
      lambda=(3*x1^2 + 2*A*x1+1)/(2*y1).

   Alternative and more efficient group laws exist, e.g., when using the
   so-called Montgomery ladder. Details are out of scope.

[B.3](#). **Group Law for Twisted Edwards Curves**

   Note: The group laws below hold for twisted Edwards curves E_{a,d}
   where a is a square in GF(q), whereas d is not. In this case, the
   addition formulae below are defined for each pair of points, without
   exceptions. Generalizations of this group law to other twisted
   Edwards curves are out of scope.

   For each point P of the twisted Edwards curve E_{a,d}, the point
   O=(0,1) serves as identity element, i.e., P + O = O + P = P.

   For each point P:=(x, y) of the twisted Edwards curve E_{a,d}, the
   point -P is the point (-x, y) and one has P + (-P) = O.

   Let P1:=(x1, y1) and P2:=(x2, y2) be points of the twisted Edwards
   curve E_{a,d} and let Q:=P1 + P2. Then Q:=(x, y), where

   $$x = (x1*y2 + x2*y1)/(1 + d*x1*x2*y1*y2) \text{ and } y = (y1*y2 - a*x1*x2)/(1 - d*x1*x2*y1*y2).$$

   Let P:=(x1, y1) be a point of the twisted Edwards curve E_{a,d} and
   let Q:=2P. Then Q:=(x, y), where

   $$x = (2*x1*y1)/(1 + d*x1^2*y1^2) \text{ and } y = (y1^2 - a*x1^2)/(1 - d*x1^2*y1^2).$$

   Note that one can use the formulae for point addition to implement
   point doubling, taking inverses and adding the identity element as
   well (i.e., the point addition formulae are uniform and complete
   (subject to our Note above)).

[Appendix C](#). **Relationship Between Curve Models**

   The non-binary curves specified in [Appendix A](#) are expressed in
   different curve models, viz. as curves in short-Weierstrass form, as
   Montgomery curves, or as twisted Edwards curves. These curve models
   are related, as follows.

[C.1](#). **Mapping between twisted Edwards Curves and Montgomery Curves**

   One can map points of the Montgomery curve M_{A,B} to points of the
   twisted Edwards curve E_{a,d}, where a:=(A+2)/B and d:=(A-2)/B and,
   conversely, map points of the twisted Edwards curve E_{a,d} to points
   of the Montgomery curve M_{A,B}, where A:=2(a+d)/(a-d) and where
   B:=4/(a-d). For twisted Edwards curves we consider (i.e., those
   where a is a square in GF(q), whereas d is not), this defines a one-
   to-one correspondence, which - in fact - is an isomorphism between

M_{A,B} and E_{a,d}, thereby showing that, e.g., the discrete
logarithm problem in either curve model is equally hard.

For the Montgomery curves and twisted Edwards curves we consider, the
mapping from M_{A,B} to E_{a,d} is defined by mapping the point at
infinity O and the point (0, 0) of order two of M_{A,B} to,
respectively, the point (0, 1) and the point (0, -1) of order two of
E_{a,d}, while mapping each other point (u, v) of M_{A,B} to the
point (x, y):=(u/v, (u-1)/(u+1)) of E_{a,d}. The inverse mapping from
E_{a,d} to M_{A,B} is defined by mapping the point (0, 1) and the
point (0, -1) of order two of E_{a,d} to, respectively, the point at
infinity O and the point (0, 0) of order two of M_{A,B}, while each
other point (x, y) of E_{a,d} is mapped to the point (u,
v):=((1+y)/(1-y), (1+y)/((1-y)*x)) of M_{A,B}.

Implementations may take advantage of this mapping to carry out
elliptic curve group operations originally defined for a twisted
Edwards curve on the corresponding Montgomery curve, or vice-versa,
and translating the result back to the original curve, thereby
potentially allowing code reuse.

C.2. Mapping between Montgomery Curves and Weierstrass Curves

One can map points of the Montgomery curve M_{A,B} to points of the
Weierstrass curve W_{a,b}, where a:=(3-A^2)/(3*B^2) and
b:=(2*A^3-9*A)/(27*B^3). This defines a one-to-one correspondence,
which - in fact - is an isomorphism between M_{A,B} and W_{a,b},
thereby showing that, e.g., the discrete logarithm problem in either
curve model is equally hard.

The mapping from M_{A,B} to W_{a,b} is defined by mapping the point
at infinity O of M_{A,B} to the point at infinity O of W_{a,b}, while
mapping each other point (u, v) of M_{A,B} to the point (x, y):=(u/
B+A/(3*B), v/B) of W_{a,b}. Note that not all Weierstrass curves can
be injectively mapped to Montgomery curves, since the latter have a
point of order two and the former may not. In particular, if a
Weierstrass curve has prime order, such as is the case with the so-
called "NIST curves", this inverse mapping is not defined.

This mapping can be used to implement elliptic curve group operations
originally defined for a twisted Edwards curve or for a Montgomery
curve using group operations on the corresponding elliptic curve in
short-Weierstrass form and translating the result back to the
original curve, thereby potentially allowing code reuse. Note that
implementations for elliptic curves with short-Weierstrass form that
hard-code the domain parameter a to a= -3 (which value is known to
allow more efficient implementations) cannot always be used this way,

since the curve W_{a,b} may not always be expressed in terms of a
Weierstrass curve with a=-3 via a coordinate transformation.

C.3. Mapping between twisted Edwards Curves and Weierstrass Curves

One can map points of the twisted Edwards curve E_{a,d} to points of
the Weierstrass curve W_{a,b}, via function composition, where one
uses the isomorphic mapping between twisted Edwards curve and
Montgomery curves of Appendix C.1 and the one between Montgomery and
Weierstrass curves of Appendix C.2. Obviously, one can use function
composition (now using the respective inverses) to realize the
inverse of this mapping.

Appendix D. Curve25519 and Cousins

D.1. Curve Definition and Alternative Representations

The elliptic curve Curve25519 is the Montgomery curve M_{A,B} defined
over the prime field GF(p), with p:=2^{255}-19, where A:=486662 and
B:=1. This curve has order h*n, where h=8 and where n is a prime
number. For this curve, A^2-4 is not a square in GF(p), whereas A+2
is. The quadratic twist of this curve has order h1*n1, where h1=4
and where n1 is a prime number. For this curve, the base point is
the point (Gu,Gv), where Gu=9 and where Gv is an odd integer in the
interval [0, p-1].

This curve has the same group structure as (is "isomorphic" to) the
twisted Edwards curve E_{a,d} defined over GF(p), with as base point
the point (Gx,Gy), where parameters are as specified in Appendix D.3.
This curve is denoted as Edwards25519. For this curve, the parameter
a is a square in GF(p), whereas d is not, so the group laws of
Appendix B.3 apply.

The curve is also isomorphic to the elliptic curve W_{a,b} in short-
Weierstrass form defined over GF(p), with as base point the point
(Gx',Gy'), where parameters are as specified in Appendix D.3. This
curve is denoted as Wei25519.

D.2. Switching between Alternative Representations

Each affine point (u,v) of Curve25519 corresponds to the point
(x,y):=(u + A/3,y) of Wei25519, while the point at infinity of
Curve25519 corresponds to the point at infinity of Wei25519. (Here,
we used the mapping of Appendix C.2.) Under this mapping, the base
point (Gu,Gv) of Curve25519 corresponds to the base point (Gx',Gy')
of Wei25519. The inverse mapping maps the affine point (x,y) of
Wei25519 to (u,v):=(x - A/3,y) of Curve25519, while mapping the point
at infinity of Wei25519 to the point at infinity of Curve25519. Note

that this mapping involves a simple shift of the first coordinate and can be implemented via integer-only arithmetic as a shift of (p+A)/3 for the isomorphic mapping and a shift of -(p+A)/3 for its inverse, where delta=(p+A)/3 is the element of GF(p) defined by

delta   19298681539552692372618308347813179755449974442734273399095973
        34652188435537

        (=0x2aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaad2
        451)

The curve Edwards25519 is isomorphic to the curve Curve25519, where the base point (Gu,Gv) of Curve25519 corresponds to the base point (Gx,Gy) of Edwards25519 and where the point at infinity and the point (0,0) of order two of Curve25519 correspond to, respectively, the point (0, 1) and the point (0, -1) of order two of Edwards25519 and where each other point (u, v) of Curve25519 corresponds to the point (c*u/v, (u-1)/(u+1)) of Edwards25519, where c is the element of GF(p) defined by

c   sqrt(-(A+2))

    5104256939916053613020613523314632928415220225303463182268183378
    8666877215207

    (=0x70d9120b 9f5ff944 2d84f723 fc03b081 3a5e2c2e b482e57d
    3391fb55 00ba81e7)

(Here, we used the mapping of Appendix C.1.) The inverse mapping from Edwards25519 to Curve25519 is defined by mapping the point (0, 1) and the point (0, -1) of order two of Edwards25519 to, respectively, the point at infinity and the point (0,0) of order two of Curve25519 and having each other point (x, y) of Edwards25519 correspond to the point ((1 + y)/(1 - y), c*(1 + y)/((1-y)*x)).

The curve Edwards25519 is isomorphic to the Weierstrass curve Wei25519, where the base point (Gx,Gy) of Edwards25519 corresponds to the base point (Gx',Gy') of Wei25519 and where the identity element (0,1) and the point (0,-1) of order two of Edwards25519 correspond to, respectively, the point at infinity O and the point (A/3, 0) of order two of Wei25519 and where each other point (x, y) of Edwards25519 corresponds to the point (x', y'):=((1+y)/(1-y)+A/3, c*(1+y)/((1-y)*x)) of Wei25519, where c was defined before. (Here, we used the mapping of Appendix C.3.) The inverse mapping from Wei25519 to Edwards25519 is defined by mapping the point at infinity O and the point (A/3, 0) of order two of Wei25519 to, respectively, the identity element (0,1) and the point (0,-1) of order two of

Edwards25519 and having each other point (x, y) of Wei25519 correspond to the point (c*(3*x-A)/(3*y), (3*x-A-3)/(3*x-A+3)).

Note that these mappings can be easily realized in projective coordinates, using a few field multiplications only, thus allowing switching between alternative representations with negligible relative incremental cost.

D.3. Domain Parameters

The parameters of the Montgomery curve and the corresponding isomorphic curves in twisted Edwards curve and short-Weierstrass form are as indicated below. Here, the domain parameters of the Montgomery curve Curve25519 and of the twisted Edwards curve Edwards25519 are as specified in RFC 7748; the domain parameters of Wei25519 are "new".

General parameters (for all curve models):

p $2^{255}-19$

     (=0x7fffffff ffffffff ffffffff ffffffff ffffffff ffffffff
     ffffffff ffffffed)

h   8

n   7237005577332262213973186563042994240857116359379907606001950938
     285454250989

     (=2^{252} + 0x14def9de a2f79cd6 5812631a 5cf5d3ed)

h1   4

n1   1447401115466452442794637312608598848160326344765032579786049412
     5407373907997

     (=2^{253} - 0x29bdf3bd 45ef39ac b024c634 b9eba7e3)

Montgomery curve-specific parameters (for Curve25519):

A   486662

B   1

Gu   9 (=0x9)

Gv   1478161944758954479102059356840998688726460613461647528896488183
     7755586237401

```
      (=0x20ae19a1 b8a086b4 e01edd2c 7748d14c 923d4d7e 6d7c61b2
      29e9c5a2 7eced3d9)
```

   Twisted Edwards curve-specific parameters (for Edwards25519):

   a   -1 (-0x01)

   d   -121665/121666

```
      (=37095705934669439343138083508754565189542113879843219016388785
      5330859402835555)
```

```
      (=0x52036cee 2b6ffe73 8cc74079 7779e898 00700a4d 4141d8ab
      75eb4dca 135978a3)
```

   Gx   15112221349535400772501151409588531511454012693041857206046113283
        949847762202

```
      (=0x216936d3 cd6e53fe c0a4e231 fdd6dc5c 692cc760 9525a7b2
      c9562d60 8f25d51a)
```

   Gy   4/5

```
      (=46316835694926478169428394003475163141307993866256225615783033
      603165251855960)
```

```
      (=0x66666666 66666666 66666666 66666666 66666666 66666666
      66666666 66666658)
```

   Weierstrass curve-specific parameters (for Wei25519):

   a   19298681539552699237261830834781317975544997444273427339909597334
        573241639236

```
      (=0x2aaaaaaa aaaaaaaa aaaaaaaa aaaaaaaa aaaaaaaa aaaaaaaa
      aaaaaa98 4914a144)
```

   b   55751746669818908907645289078257140818241103727901012315294400837
        956729358436

```
      (=0x7b425ed0 97b425ed 097b425e d097b425 ed097b42 5ed097b4
      260b5e9c 7710c864)
```

   Gx' 19298681539552699237261830834781317975544997444273427339909597334
        652188435546

```
      (=0x2aaaaaaa aaaaaaaa aaaaaaaa aaaaaaaa aaaaaaaa aaaaaaaa
      aaaaaaaa aaad245a)
```

```
Gy' 1478161944758954479102059356840998688726460613461647528896488183
    7755586237401

    (=0x20ae19a1 b8a086b4 e01edd2c 7748d14c 923d4d7e 6d7c61b2
    29e9c5a2 7eced3d9)
```

Appendix E. Further Mappings

The non-binary curves specified in Appendix A are expressed in
different curve models, viz. as curves in short-Weierstrass form, as
Montgomery curves, or as twisted Edwards curves. Within each curve
model, further mappings exist that induce a mapping between elliptic
curves within each curve model. This can be exploited to force some
of the domain parameter to a value that allows a more efficient
implementation of the addition formulae.

E.1. Isomorphic Mapping between Weierstrass Curves

One can map points of the Weierstrass curve W_{a,b} to points of the
Weierstrass curve W_{a',b'}, where a:=a'*u^4 and b:=b'*u^6 for some
nonzero value u of the finite field GF(q). This defines a one-to-one
correspondence, which - in fact - is an isomorphism between W_{a,b}
and W_{a',b'}, thereby showing that, e.g., the discrete logarithm
problem in either curve model is equally hard.

The mapping from W_{a,b} to W_{a',b'} is defined by mapping the point
at infinity O of W_{a,b} to the point at infinity O of W_{a',b'},
while mapping each other point (x, y) of W_{a,b} to the point (x',
y'):=(x*u^2, y*u^3) of W_{a',b'}. The inverse mapping from W_{a',b'}
to W_{a,b} is defined by mapping the point at infinity O of W_{a',b'}
to the point at infinity O of W_{a,b}, while mapping each other point
(x', y') of W_{a',b'} to the point (x, y):=(x/u^2, y/u^3) of W_{a,b}.

Implementations may take advantage of this mapping to carry out
elliptic curve group operations originally defined for a Weierstrass
curve with a generic domain parameter a on a corresponding isomorphic
Weierstrass curve with domain parameter a' that has a special form,
which is known to allow for more efficient implementations of
addition laws, and translating the result back to the original curve.
In particular, it is known that such efficiency improvements exist if
a'=-3 (mod p) and one uses so-called Jacobian coordinates with a
particular projective version of the addition laws of Appendix B.1.
While not all Weierstrass curves can be put into this form, all
traditional NIST curves have domain parameter a=-3, while all
Brainpool curves [RFC5639] are isomorphic to a Weierstrass curve of
this form. For details, we refer to [GECC].

Note that implementations for elliptic curves with short-Weierstrass
form that hard-code the domain parameter a to a= -3 (which value is
known to allow more efficient implementations) cannot always be used
this way, since the curve W_{a,b} may not always be expressed in
terms of a Weierstrass curve with a'=-3 via a coordinate
transformation: this only holds if a'/a is a fourth power in GF(q).
However, even in this case, one can still express the curve W_{a,b}
in terms of a Weierstrass curve with small a' domain parameter,
thereby still allowing a more efficient implementation than with a
general a value.

E.2. Isogeneous Mapping between Weierstrass Curves

One can still map points of the Weierstrass curve W_{a,b} to points
of the Weierstrass curve W_{a',b'}, where a':=-3 (mod p), even if
a'/a is not a fourth power in GF(q). In that case, this mappping
cannot be an isomorphism (see Appendix E.1) and, thereby, does not
define a one-to-one correspondence. Instead, the mapping is a so-
called isogeny (or homomorphism). Since most elliptic curve
operations process points of prime order or use so-called "co-factor
multiplication", in practice the resulting mapping has similar
properties. In particular, one can still take advantage of this
mapping to carry out elliptic curve group operations originally
defined for a Weierstrass curve with domain parameter a unequal to -3
(mod p) on a corresponding isogenous Weierstrass curve with domain
parameter a'=-3 (mod p) and translating the result back to the
original curve. Details of this mapping are outside scope of this
document.

Appendix F. Further Cousins of Curve25519

F.1. Further Alternative Representations

The Weierstrass curve Wei25519 is isomorphic to the Weierstrass curve
Wei25519.2 defined over GF(p), with as base point the pair (G1x,G1y),
where parameters are as specified in Appendix F.3.

F.2. Further Switching

Each affine point (x,y) of Wei25519 corresponds to the point
(x,y):=(x*u^2,y*u^3) of Wei25519.2, where u is the element of GF(p)
defined by

u   4773168724887355967255521690649675419508341069991820702939107936
     6321486119

     (=0x10e26dacae93602704c7e6cff9efe595764cb5c9e04931f6fdeefc657d4e5
     27),

while the point at infinity of Wei25519 corresponds to the point at infinity of Wei25519.2. (Here, we used the mapping of Appendix E.1.) Under this mapping, the base point (Gx',Gy') of Wei25519 corresponds to the base point (G1x',G1y') of Wei25519.2. The inverse mapping maps the affine point (x,y) of Wei25519.2 to (x,y):=(x/u^2,y/u^3) of Wei25519, while mapping the point at infinity of Wei25519.2 to the point at infinity of Wei25519. Note that this mapping (and its inverse) involves a multiplication of both coordinates with fixed constants u^2 and u^3 (respectively, 1/u^2 and 1/u^3), which can be precomputed.

F.3. Further Domain Parameters

The parameters of the Weierstrass curve with a=2 that is isomorphic with Wei25519 and the parameters of the Weierstrass curve with a=-3 that is isogeneous with Wei25519 are as indicated below. Both domain parameter sets can be exploited directly to derive more efficient point addition formulae, should an implementation facilitate this.

Weierstrass curve-specific parameters (with a=2):

a   2 (=0x2)

b   45793404337388339159414415854563976158160282736335993851976016290
     777777599260

     (=0x653e25fa 4aa43eb9 cc42c61b 806bcfd1 0e67bc23 09966e90
     95a202fe 9aac731c)

G1x' 21872607226894442744132797191435288341483620396057247222462149535754145422686

     (=0x305b74fc 935f1dad d440a88e 781f0a81 09d6a68d 98c6081a
     660528e2 0746dd5e)

G1y' 13943617903486429134407723576638679615598775530747991987186632147013341290929

     (=0x1ed3cedc e78b6b19 5d1c361c e1d4ef00 5b5b102c 99083780
     bf830f7e a89021b1)

Weierstrass curve-specific parameters (with a=-3):

[NOTE: parameters indicated with TBD still to be completed, pending completion of Sage calculations.]

a   -3

```
        (=0x7fffffff ffffffff ffffffff ffffffff ffffffff ffffffff
        ffffffff ffffffea)
```

b   [TBD]

```
        (=0x[TBD])
```

G2x' [TBD]

```
        (=0x[TBD])
```

G2y' [TBD]

```
        (=0x[TBD])
```

Author's Address

    Rene Struik
    Struik Security Consultancy

    Email: rstruik.ext@gmail.com