Network Working Group
Internet-Draft
Intended status: Informational
Expires: May 30, 2015

B. Black
Microsoft
J. Bos
NXP Semiconductors
C. Costello
Microsoft Research
A. Langley
Google Inc
P. Longa
M. Naehrig
Microsoft Research

November 26, 2014

Rigid Parameter Generation for Elliptic Curve Cryptography
draft-black-rpgecc-00

Abstract

This memo describes algorithms for deterministically generating parameters for elliptic curves over prime fields offering high practical security in cryptographic applications, including Transport Layer Security (TLS) and X.509 certificates. The algorithms can generate domain parameters at any security level for modern (twisted) Edwards curves.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction 2
   1.1. Requirements Language 3
2. Scope and Relation to Other Specifications 3
3. Security Requirements 3
4. Notation 3
5. Parameter Generation 4
   5.1. Deterministic Curve Parameter Generation 4
      5.1.1. Twisted Edwards Curves 4
      5.1.2. Edwards Curves 5
6. Generators 6
7. Test Vectors 6
8. Acknowledgements 7
9. Security Considerations 7
10. Intellectual Property Rights 7
11. IANA Considerations 7
12. References 8
   12.1. Normative References 8
   12.2. Informative References 8
Authors' Addresses 9

1. Introduction

Since the initial standardization of elliptic curve cryptography (ECC) in [SEC1] there has been significant progress related to both efficiency and security of curves and implementations. Notable examples are algorithms protected against certain side-channel attacks, different 'special' prime shapes which allow faster modular arithmetic, and a larger set of curve models from which to choose. There is also concern in the community regarding the generation and potential weaknesses of the curves defined in [NIST].

This memo describes a deterministic algorithm for generation of elliptic curves for cryptography. The constraints in the generation process produce curves that support constant-time, exception-free scalar multiplications that are resistant to a wide range of side-channel attacks including timing and cache attacks, thereby offering

high practical security in cryptographic applications. The deterministic algorithm operates without any hidden parameters, reliance on randomness or any other processes offering opportunities for manipulation of the resulting curves. The selection between curve models is determined by choosing the curve form that supports the fastest (currently known) complete formulas for each modularity option of the underlying field prime. Specifically, the twisted Edwards curve -x^2 + y^2 = 1 + dx^2y^2 is used for primes p with p = 1 mod 4, and the Edwards curve x^2 + y^2 = 1 + dx^2y^2 is used with primes p with p = 3 mod 4.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Scope and Relation to Other Specifications

This document specifies a deterministic algorithm for generating elliptic curve domain parameters over prime fields GF(p), with p having a length of twice the desired security level in bits, in (twisted) Edwards form. Furthermore, this document identifies the security and implementation requirements for the generated domain parameters.

3. Security Requirements

For each curve at a specific security level:

1. The domain parameters SHALL be generated in a simple, deterministic manner, without any secret or random inputs. The derivation of the curve parameters is defined in Section 5.

2. The trace of Frobenius MUST NOT be in {0, 1} in order to rule out the attacks described in [Smart], [AS], and [S], as in [EBP].

3. MOV Degree: the embedding degree k MUST be greater than (r - 1) / 100, as in [EBP].

4. CM Discriminant: discriminant D MUST be greater than 2^100, as in [SC].

4. Notation

Throughout this document, the following notation is used:


```
       p: Denotes the prime number defining the base field.
   GF(p): The finite field with p elements.
       d: An element in the finite field GF(p), different from -1,0.
      Ed: The elliptic curve Ed/GF(p): x^2 + y^2 = 1 + dx^2y^2 in
          Edwards form, defined over GF(p) by the parameter d.
     tEd: The elliptic curve tEd/GF(p): -x^2 + y^2 = 1 + dx^2y^2 in
          twisted Edwards form, defined over GF(p) by the parameter d.
      rd: The largest odd divisor of the number of GF(p)-rational
          points on Ed or tEd.
      td: The trace of Frobenius of Ed or tEd such that
          #Ed(GF(p)) = p + 1 - td or #tEd(GF(p)) = p + 1 - td,
          respectively.
     rd': The largest odd divisor of the number of GF(p)-rational
          points on Ed' or tEd'.
      hd: The index (or cofactor) of the subgroup of order rd in the
          group of GF(p)-rational points on Ed or tEd.
     hd': The index (or cofactor) of the subgroup of order rd' in the
          group of GF(p)-rational points on the non-trivial quadratic
          twist of Ed or tEd.
       P: A generator point defined over GF(p) of prime order rd on Ed
          or tEd.
    X(P): The x-coordinate of the elliptic curve point P.
    Y(P): The y-coordinate of the elliptic curve point P.
```

5. Parameter Generation

This section describes the generation of the curve parameters, namely the curve parameter d, and a generator point P of the prime order subgroup of the elliptic curve.

5.1. Deterministic Curve Parameter Generation

5.1.1. Twisted Edwards Curves

For a prime p = 1 mod 4, the elliptic curve tEd in twisted Edwards form is determined by the non-square element d from GF(p), different from -1,0 with smallest absolute value such that #tEd(GF(p)) = hd * rd, #tEd'(GF(p)) = hd' * rd', {hd, hd'} = {4, 8} and both subgroup orders rd and rd' are prime. In addition, care must be taken to ensure the MOV degree and CM discriminant requirements from Section 3 are met.


```
Input: a prime p, with p = 1 mod 4
Output: the parameter d defining the curve tEd
1. Set d = 0
2. repeat
     repeat
       if (d > 0) then
         d = -d
       else
         d = -d + 1
       end if
     until d is not a square in GF(p)
     Compute rd, rd', hd, hd' where #tEd(GF(p)) = hd * rd,
     #tEd'(GF(p)) = hd' * rd', hd and hd' are powers of 2 and rd, rd'
     are odd
   until ((hd + hd' = 12) and rd is prime and rd' is prime)
3. Output d
```

GenerateCurveTEdwards

5.1.2. Edwards Curves

For a prime p = 3 mod 4, the elliptic curve Ed in Edwards form is

determined by the non-square element d from GF(p), different from -1,0 with smallest absolute value such that #Ed(GF(p)) = hd * rd, #Ed'(GF(p)) = hd' * rd', hd = hd' = 4, and both subgroup orders rd and rd' are prime. In addition, care must be taken to ensure the MOV degree and CM discriminant requirements from Section 3 are met.

```
Input: a prime p, with p = 3 mod 4
Output: the parameter d defining the curve Ed
1. Set d = 0
2. repeat
     repeat
       if (d > 0) then
         d = -d
       else
         d = -d + 1
       end if
     until d is not a square in GF(p)
     Compute rd, rd', hd, hd' where #Ed(GF(p)) = hd * rd,
     #Ed'(GF(p)) = hd' * rd', hd and hd' are powers of 2 and rd, rd'
     are odd
   until ((hd = hd' = 4) and rd is prime and rd' is prime)
3. Output d
```

GenerateCurveEdwards

6. Generators

The generator points P = (X(P),Y(P)) for all curves are selected by taking the smallest positive value x in GF(p) (when represented as an integer) such that (x, y) is on the curve and such that (X(P),Y(P)) = 8 * (x, y) has large prime order rd.

```
Input: a prime p and curve parameters d and
       a = -1 for twisted Edwards (p = 1 mod 4) or
       a = 1 for Edwards (p = 3 mod 4)
Output: a generator point P = (X(P), Y(P)) of order rd
1. Set x = 0 and found_gen = false
2. while (not found_gen) do
     x = x + 1
     while ((d * x^2 = 1 mod p)
```

```
         or ((1 - a * x^2) * (1 - d * x^2) is not a quadratic residue
         mod p)) do
       x = x + 1
     end while
     Compute an integer s, 0 < s < p, such that
            s^2 * (1 - d * x^2) = 1 - a * x^2 mod p
     Set y = min(s, p - s)

     (X(P), Y(P)) = 8 * (x, y)

     if ((X(P), Y(P)) has order rd on Ed or tEd, respectively) then
         found_gen = true
     end if
   end while
3. Output (X(P),Y(P))
```

GenerateGen

7. Test Vectors

The following figures give parameters for twisted Edwards and Edwards curves generated using the algorithms defined in previous sections. All integer values are unsigned.


```
      p = 0x7FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
            FFFFFFFFFFED
      d = 0x15E93
      r = 0x20000000000000000000000000000000016241E6093B2CE59B6B9
            8FD8849FAF35
   x(P) = 0x3B7C1D83A0EF56F1355A0B5471E42537C26115EDE4C948391714
            C0F582AA22E2
   y(P) = 0x775BE0DEC362A16E78EFFE0FF4E35DA7E17B31DC1611475CB4BE
```

```
         1DA9A3E5A819
      h = 0x4

                             p = 2^255 - 19

        p = 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
                FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEC3
        d = 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
                FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFD19F
        r = 0x3FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFE2471A1
                CB46BE1CF61E4555AAB35C87920B9DCC4E6A3897D
     x(P) = 0x61B111FB45A9266CC0B6A2129AE55DB5B30BF446E5BE4C005763FFA
                8F33163406FF292B16545941350D540E46C206BDE
     y(P) = 0x82983E67B9A6EEB08738B1A423B10DD716AD8274F1425F56830F98F
                7F645964B0072B0F946EC48DC9D8D03E1F0729392
        h = 0x4

                             p = 2^384 - 317
```

8. Acknowledgements

The authors would like to thank Tolga Acar, Karen Easterbrook and Brian LaMacchia for their contributions to the development of this draft.

9. Security Considerations

TBD

10. Intellectual Property Rights

The authors have no knowledge about any intellectual property rights that cover the usage of the domain parameters defined herein.

11. IANA Considerations

There are no IANA considerations for this document.

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

12.2. Informative References

[AS] Satoh, T. and K. Araki, "Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves", 1998.

[EBP] ECC Brainpool, "ECC Brainpool Standard Curves and Curve Generation", October 2005, <http://www.ecc-brainpool.org/download/Domain-parameters.pdf>.

[ECCP] Bos, J., Halderman, J., Heninger, N., Moore, J., Naehrig, M., and E. Wustrow, "Elliptic Curve Cryptography in Practice", December 2013, <https://eprint.iacr.org/2013/734>.

[FPPR] Faugere, J., Perret, L., Petit, C., and G. Renault, 2012, <http://dx.doi.org/10.1007/978-3-642-29011-4_4>.

[MSR] Bos, J., Costello, C., Longa, P., and M. Naehrig, "Selecting Elliptic Curves for Cryptography: An Efficiency and Security Analysis", February 2014, <http://eprint.iacr.org/2014/130.pdf>.

[NIST] National Institute of Standards, "Recommended Elliptic Curves for Federal Government Use", July 1999, <http://csrc.nist.gov/groups/ST/toolkit/documents/dss/NISTReCur.pdf>.

[RFC3279] Bassham, L., Polk, W., and R. Housley, "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3279, April 2002.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

[RFC4050] Blake-Wilson, S., Karlinger, G., Kobayashi, T., and Y. Wang, "Using the Elliptic Curve Signature Algorithm (ECDSA) for XML Digital Signatures", RFC 4050, April 2005.

[RFC4492] Blake-Wilson, S., Bolyard, N., Gupta, V., Hawk, C., and B. Moeller, "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)", RFC 4492, May 2006.

[RFC4754] Fu, D. and J. Solinas, "IKE and IKEv2 Authentication Using the Elliptic Curve Digital Signature Algorithm (ECDSA)", RFC 4754, January 2007.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5480] Turner, S., Brown, D., Yiu, K., Housley, R., and T. Polk, "Elliptic Curve Cryptography Subject Public Key Information", RFC 5480, March 2009.

[RFC5753] Turner, S. and D. Brown, "Use of Elliptic Curve Cryptography (ECC) Algorithms in Cryptographic Message Syntax (CMS)", RFC 5753, January 2010.

[RFC6090] McGrew, D., Igoe, K., and M. Salter, "Fundamental Elliptic Curve Cryptography Algorithms", RFC 6090, February 2011.

[S] Semaev, I., "Evaluation of discrete logarithms on some elliptic curves", 1998.

[SC] Bernstein, D. and T. Lange, "SafeCurves: choosing safe curves for elliptic-curve cryptography", June 2014, <http://safecurves.cr.yp.to/>.

[SEC1] Certicom Research, "SEC 1: Elliptic Curve Cryptography", September 2000, <http://www.secg.org/collateral/sec1_final.pdf>.

[Smart] Smart, N., "The discrete logarithm problem on elliptic curves of trace one", 1999.

Authors' Addresses

Benjamin Black
Microsoft
One Microsoft Way
Redmond, WA 98115
US

Email: benblack@microsoft.com

Joppe W. Bos
NXP Semiconductors
Interleuvenlaan 80
3001 Leuven
Belgium

Email: joppe.bos@nxp.com

Craig Costello
Microsoft Research
One Microsoft Way
Redmond, WA 98115
US

Email: craigco@microsoft.com

Adam Langley
Google Inc

Email: agl@google.com

Patrick Longa
Microsoft Research
One Microsoft Way
Redmond, WA 98115
US

Email: plonga@microsoft.com

Michael Naehrig
Microsoft Research
One Microsoft Way
Redmond, WA 98115
US

Email: mnaehrig@microsoft.com