

1 **Draft NIST Special Publication 800-186**

2

3 **Recommendations for Discrete**
4 **Logarithm-Based Cryptography:**

5 *Elliptic Curve Domain Parameters*

6

7 Lily Chen

8 Dustin Moody

9 Andrew Regenscheid

10 Karen Randall

11

12

13

14

15

16

17

COMPUTER SECURITY

18

19

20

18 **Draft NIST Special Publication 800-186**

19

20 **Recommendations for Discrete**
21 **Logarithm-Based Cryptography:**
22 *Elliptic Curve Domain Parameters*

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

Lily Chen
Dustin Moody
Andrew Regenscheid
Computer Security Division
Information Technology Laboratory

Karen Randall
Randall Consulting
Dover, NH

This publication is available free of charge from:
<https://doi.org/10.6028/NIST.SP.800-186-draft>

October 2019



41

42

43

44

45

46

47

48

U.S. Department of Commerce
Wilbur L. Ross, Jr., Secretary

National Institute of Standards and Technology
Walter Copan, NIST Director and Under Secretary of Commerce for Standards and Technology

49

Authority

50 This publication has been developed by NIST in accordance with its statutory responsibilities under the
51 Federal Information Security Modernization Act (FISMA) of 2014, 44 U.S.C. § 3551 *et seq.*, Public Law
52 (P.L.) 113-283. NIST is responsible for developing information security standards and guidelines, including
53 minimum requirements for federal information systems, but such standards and guidelines shall not apply
54 to national security systems without the express approval of appropriate federal officials exercising policy
55 authority over such systems. This guideline is consistent with the requirements of the Office of Management
56 and Budget (OMB) Circular A-130.

57 Nothing in this publication should be taken to contradict the standards and guidelines made mandatory and
58 binding on federal agencies by the Secretary of Commerce under statutory authority. Nor should these
59 guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce,
60 Director of the OMB, or any other federal official. This publication may be used by nongovernmental
61 organizations on a voluntary basis and is not subject to copyright in the United States. Attribution would,
62 however, be appreciated by NIST.

63 National Institute of Standards and Technology Special Publication 800-186
64 Natl. Inst. Stand. Technol. Spec. Publ. 800-186, 78 pages (October 2019)
65 CODEN: NSPUE2

66 This publication is available free of charge from:
67 <https://doi.org/10.6028/NIST.SP.800-186-draft>

68 Certain commercial entities, equipment, or materials may be identified in this document in order to describe an
69 experimental procedure or concept adequately. Such identification is not intended to imply recommendation or
70 endorsement by NIST, nor is it intended to imply that the entities, materials, or equipment are necessarily the best
71 available for the purpose.

72 There may be references in this publication to other publications currently under development by NIST in accordance
73 with its assigned statutory responsibilities. The information in this publication, including concepts and methodologies,
74 may be used by federal agencies even before the completion of such companion publications. Thus, until each
75 publication is completed, current requirements, guidelines, and procedures, where they exist, remain operative. For
76 planning and transition purposes, federal agencies may wish to closely follow the development of these new
77 publications by NIST.

78 Organizations are encouraged to review all draft publications during public comment periods and provide feedback to
79 NIST. Many NIST cybersecurity publications, other than the ones noted above, are available at
80 <https://csrc.nist.gov/publications>.

81

82 **Public comment period: *October 31, 2019 through January 29, 2020***

83 National Institute of Standards and Technology
84 Attn: Computer Security Division, Information Technology Laboratory
85 100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930
86 Email: SP800-186-comments@nist.gov

87 All comments are subject to release under the Freedom of Information Act (FOIA).

88

Reports on Computer Systems Technology

89 The Information Technology Laboratory (ITL) at the National Institute of Standards and
90 Technology (NIST) promotes the U.S. economy and public welfare by providing technical
91 leadership for the Nation's measurement and standards infrastructure. ITL develops tests, test
92 methods, reference data, proof of concept implementations, and technical analyses to advance the
93 development and productive use of information technology. ITL's responsibilities include the
94 development of management, administrative, technical, and physical standards and guidelines for
95 the cost-effective security and privacy of other than national security-related information in federal
96 information systems. The Special Publication 800-series reports on ITL's research, guidelines, and
97 outreach efforts in information system security, and its collaborative activities with industry,
98 government, and academic organizations.

99

Abstract

100 This recommendation specifies the set of elliptic curves recommended for U.S. Government use.
101 In addition to the previously recommended Weierstrass curves defined over prime fields and
102 binary fields, this recommendation includes two newly specified Montgomery curves, which
103 claim increased performance, side-channel resistance, and simpler implementation when
104 compared to traditional curves. The recommendation also specifies alternative representations
105 for these new curves to allow more implementation flexibility. The new curves are interoperable
106 with those specified by the Crypto Forum Research Group (CFRG) of the Internet Engineering
107 Task Force (IETF).

108

Keywords

109 Computer security; discrete logarithm-based groups; elliptic curve cryptography; domain parameters.

110

Acknowledgements

111 The authors of SP 800-186 (Lily Chen, Dustin Moody, Andrew Regenscheid, and Karen
112 Randall) wish to thank Rene Struik, Rich Davis, and Scott Simon for their contributions to this
113 document. The authors also acknowledge and appreciate the support from public and private
114 sectors whose constructive and beneficial comments greatly enhanced the utility of this
115 publication, as well as their colleagues who reviewed earlier versions of this document.
116

117

Audience

118 This document is intended for implementers of cryptographic schemes that include the use of
119 elliptic curve cryptography.

120

Conformance Testing

121 Conformance testing for implementations of this Recommendation will be conducted within the
122 framework of the Cryptographic Algorithm Validation Program (CAVP) and the Cryptographic
123 Module Validation Program (CMVP). The requirements of this Recommendation are indicated
124 by the word “**shall**.” Some of these requirements may be out-of-scope for CAVP or CMVP
125 validation testing, and thus are the responsibility of entities using, implementing, installing or
126 configuring applications that incorporate this Recommendation.

127 Conformant implementations may perform procedures via an equivalent sequence of operations,
128 provided that these include all cryptographic checks included with the specifications in this
129 document. This is important because the checks are essential for the prevention of subtle attacks.
130

131

Call for Patent Claims

132 This public review includes a call for information on essential patent claims (claims whose use
133 would be required for compliance with the guidance or requirements in this Information
134 Technology Laboratory (ITL) draft publication). Such guidance and/or requirements may be
135 directly stated in this ITL Publication or by reference to another publication. This call also
136 includes disclosure, where known, of the existence of pending U.S. or foreign patent applications
137 relating to this ITL draft publication and of any relevant unexpired U.S. or foreign patents.

138 ITL may require from the patent holder, or a party authorized to make assurances on its behalf,
139 in written or electronic form, either:

140 a) assurance in the form of a general disclaimer to the effect that such party does not hold
141 and does not currently intend holding any essential patent claim(s); or

142 b) assurance that a license to such essential patent claim(s) will be made available to
143 applicants desiring to utilize the license for the purpose of complying with the guidance
144 or requirements in this ITL draft publication either:

145 i. under reasonable terms and conditions that are demonstrably free of any unfair
146 discrimination; or

147 ii. without compensation and under reasonable terms and conditions that are
148 demonstrably free of any unfair discrimination.

149 Such assurance shall indicate that the patent holder (or third party authorized to make assurances
150 on its behalf) will include in any documents transferring ownership of patents subject to the
151 assurance, provisions sufficient to ensure that the commitments in the assurance are binding on
152 the transferee, and that the transferee will similarly include appropriate provisions in the event of
153 future transfers with the goal of binding each successor-in-interest.

154 The assurance shall also indicate that it is intended to be binding on successors-in-interest
155 regardless of whether such provisions are included in the relevant transfer documents.

156 Such statements should be addressed to: SP800-186-comments@nist.gov

157 **Executive Summary**

158 This recommendation specifies the set of elliptic curves recommended for U.S. Government use.
159 It includes:

- 160 – Specification of elliptic curves previously specified in FIPS Publication 186-4, *Digital*
161 *Signature Schemes* [[FIPS 186-4](#)]. This includes both elliptic curves defined over a prime
162 field and curves defined over a binary field. Although the specifications for elliptic
163 curves over binary fields are included, these curves are now deprecated.
- 164 – Specification of new Montgomery and Edwards curves, which are detailed in *Elliptic*
165 *Curves for Security* [[RFC 7748](#)]. These curves are only to be used with the EdDSA
166 digital signature scheme in FIPS 186-5.
- 167 – A reference for the Brainpool curves, specified in [[RFC 5639](#)]. These curves are allowed
168 to be used for interoperability reasons.
- 169 – Elliptic curves in FIPS 186-4 that do not meet the current bit-security requirements put
170 forward in NIST Special Publication 800-57, Part 1, *Recommendation for Key*
171 *Management Part 1: General* [[SP 800-57](#)], are now legacy-use. They may be used to
172 process already protected information (e.g., decrypt or verify) but not to apply protection
173 to information (e.g., encrypt or sign). Also see NIST Special Publication 800-131A,
174 *Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms*
175 *and Key Lengths* [[SP 800-131A](#)].

176
177 This recommendation provides details regarding the group operations for each of the
178 specified elliptic curves and the relationship between the various curve models, allowing
179 flexibility regarding the use of curves most suitable in particular applications. It also
180 gives cryptographic criteria for the selection of suitable elliptic curves and provides more
181 details on finite field arithmetic and data representation than were available in FIPS 186-
182 4.

183

184 **Table of Contents**

185 Executive Summary v

186 1 Introduction 1

187 1.1 Background..... 1

188 1.2 Purpose and Scope 1

189 1.3 Document Organization 1

190 2 Glossary of Terms, Symbols and Abbreviations..... 3

191 2.1 Glossary..... 3

192 2.2 Symbols and Abbreviations 3

193 3 Overview of Elliptic Curves 6

194 3.1 Non-Binary Curves..... 6

195 3.1.1 Curves in Short-Weierstrass Form 6

196 3.1.2 Montgomery Curves 6

197 3.1.3 Twisted Edwards Curves..... 6

198 3.2 Binary Curves 6

199 3.2.1 Curves in Short-Weierstrass Form 6

200 4 Recommended Curves for U.S. Federal Government Use 8

201 4.1 Choices of Key Lengths, Underlying Fields, Curves, and Base Points 9

202 4.1.1 Choice of Key Lengths 9

203 4.1.2 Choice of Underlying Fields..... 9

204 4.1.3 Choice of Basis for Binary Fields..... 10

205 4.1.4 Choice of Curves 10

206 4.1.5 Choice of Base Points 11

207 4.2 Curves over Prime Fields..... 11

208 4.2.1 Weierstrass Curves 11

209 4.2.2 Montgomery Curves 18

210 4.2.3 Twisted Edwards Curves..... 20

211 4.3 Curves over Binary Fields..... 23

212 4.3.1 Koblitz Curves 24

213 4.3.2 Pseudorandom Curves..... 27

214 References 31

215 Appendix A Details of Elliptic Curve Group Operations..... 33

216 A.1 Non-Binary Curves..... 33

217 A.1.1 Group Law for Weierstrass Curves..... 33

218 A.1.2 Group Law for Montgomery Curves..... 33

219 A.1.3 Group Law for Twisted Edwards Curves 33

220 A.2 Binary Curves 34

221 A.2.1 Group Law for Weierstrass Curves..... 34

222 Appendix B Relationship between Curve Models..... 35

223 B.1 Mapping Between Twisted Edwards Curves and Montgomery Curves..... 35

224 B.2 Mapping Between Montgomery Curves and Weierstrass Curves..... 35

225 B.3 Mapping Between Twisted Edwards Curves and Weierstrass Curves..... 36

226 B.4 4-Isogenous Mapping 36

227 Appendix C Generation Details for Recommended Elliptic Curves 37

228 C.1 General Cryptographic Criteria 37

229 C.1.1 Implementation Security Criteria 37

230 C.2 Curve Generation Details..... 37

231 C.2.1 Weierstrass Curves over Prime Fields 37

232 C.2.2 Montgomery Curves 38

233 C.2.3 Twisted Edwards Curves..... 39

234 C.2.4 Weierstrass Curves over Binary Fields..... 39

235 C.3 Generation and Verification of Pseudorandom Curves 40

236 C.3.1 Generation of Pseudorandom Curves (Prime Case) 40

237 C.3.2 Verification of Curve Pseudorandomness (Prime Case) 41

238 C.3.3 Generation of Pseudorandom Curves (Binary Case) 41

239 C.3.4 Verification of Curve Pseudorandomness (Binary Case)..... 42

240 Appendix D Elliptic Curve Routines..... 44

241 D.1 Public Key Validation 44

242 D.1.1 Non-Binary Curves in Short-Weierstrass Form 44

243 D.1.2 Montgomery Curves 44

244 D.1.3 Twisted Edwards Curves..... 45

245 D.1.4 Binary Curves in Short-Weierstrass Form 46

246 D.2 Point Compression..... 47

247 D.2.1 Prime Curves in Short-Weierstrass Form 47

248 D.2.2 Binary Curves in Short-Weierstrass Form 48

249 D.3 Base Point (Generator) Selection 49

250	D.3.1	Generation of Base Points.....	49
251	D.3.2	Verifiably Random Base Points	50
252	D.3.3	Validity of Base Points.....	50
253	Appendix E	Auxiliary Functions	52
254	E.1	Computing Square Roots in Binary Fields	52
255	E.2	Solving the Equation $x^2 + x = w$ in Binary Fields	52
256	E.3	Computing Square Roots in non-Binary Fields $GF(q)$	52
257	E.4	Computing Inverses in $GF(q)$	53
258	Appendix F	Data Conversion	54
259	F.1	Conversion of a Field Element to an Integer.....	54
260	F.2	Conversion of an Integer to a Field Element.....	54
261	F.3	Conversion of an Integer to a Bit String	54
262	F.4	Conversion of a Bit String to an Integer	55
263	Appendix G	Implementation Aspects.....	56
264	G.1	Implementation of Modular Arithmetic.....	56
265	G.1.1	Curve P-224	56
266	G.1.2	Curve P-256	57
267	G.1.3	Curve P-384	57
268	G.1.4	Curve P-521	58
269	G.1.5	Curve Curve448	58
270	G.1.6	Curve Curve25519	59
271	G.2	Scalar Multiplication for Koblitz Curves.....	60
272	G.3	Polynomial and Normal Bases for Binary Fields	63
273	G.3.1	Normal Bases	63
274	G.3.2	Polynomial Basis to Normal Basis Conversion.....	65
275	G.3.3	Normal Basis to Polynomial Basis Conversion.....	66
276	Appendix H	Other Allowed Elliptic Curves.....	68
277	H.1	Brainpool Curves	68
278			
279			

280 **1 Introduction**

281 **1.1 Background**

282 Elliptic curve cryptography (ECC) has uses in applications involving digital signatures (e.g.,
283 Elliptic Curve Digital Signature Algorithm, or ECDSA) and key agreement schemes (e.g.,
284 Elliptic Curve Diffie-Hellman, or ECDH). The most widely used curves are usually expressed in
285 short-Weierstrass format. However, curves that are expressed using a different format, such as
286 Montgomery curves and twisted Edwards curves, have garnered academic interest. These curves
287 are claimed to have better performance and increased side-channel resistance.

288 A number of organizations (e.g., NIST, ANSI X9F, ISO, SEC, and IETF) have developed elliptic
289 curve standards. Other standards-setting organizations, such as the Crypto Forum Research
290 Group (CFRG) of the IETF, have discussed ECC and made recommendations for alternate
291 elliptic curves and digital signatures. In June 2015, NIST organized an ECC workshop to discuss
292 the design of curves that are secure, efficient, and easy to use while also being resilient to a wide
293 range of implementation attacks. Subsequently, NIST solicited public comments on the Digital
294 Signature Standard (FIPS 186-4), requesting specific feedback regarding the digital signature
295 schemes in FIPS 186 as well as possible new recommended elliptic curves. This publication is
296 the result of that input.

297 **1.2 Purpose and Scope**

298 This recommendation provides updated specifications of elliptic curves that are appropriate for
299 use by the U.S. Federal Government for digital signatures. It is intended for use in conjunction
300 with other NIST publications, such as NIST Special Publication SP 800-56A, *Recommendation*
301 *for Pair-Wise Key Establishment Schemes Using Discrete Logarithm-Based Cryptography* [[SP](#)
302 [800-56A](#)]; Federal Information Processing Standard FIPS 186-5, *Digital Signature Standard*
303 [[FIPS 186-5](#)]; and related specifications. The key pairs specified here are used for digital
304 signature generation and verification or key agreement only and should not be used for any other
305 purposes.

306 This recommendation is intended to provide sufficient information for a vendor to implement
307 ECC using asymmetric algorithms in FIPS 140-3 [FIPS 140-3] validated modules.

308 **1.3 Document Organization**

309 The remainder of this document includes the following sections and appendices:

- 310 • **Section 2: Glossary of Terms, Symbols, and Abbreviations**
- 311 • **Section 3: Overview of Elliptic Curves** — This section details the different curve models
312 being used with this recommendation, including notational conventions.
- 313 • **Section 4: Recommended Curves for Federal Government Use** — This section highlights
314 the domain parameters for all elliptic curves recommended for U.S. Government use.
- 315 • **References** — This section contains references for additional information and links to
316 documents referenced in the publication.

- 317 • **Appendix A: Details of Elliptic Curve Group Operations** — This appendix discusses the
318 group laws for each of the different curve models specified in this recommendation.
- 319 • **Appendix B: Relationship Between Curve Models** — This appendix details how different
320 curve models are related and how the coordinates of a point and the domain parameters of a
321 curve in one curve model relate to those in another curve model.
- 322 • **Appendix C: Generation Details for Recommended Elliptic Curves** — This appendix
323 describes the cryptographic criteria that guided the selection of suitable elliptic curves and
324 the process by which one of many such suitable elliptic curves is selected.
- 325 • **Appendix D: Elliptic Curve Routines** — This appendix details elementary routines for
326 elliptic curves, such as the verification that these curves are indeed well-formed, and point
327 compression.
- 328 • **Appendix E: Auxiliary Functions** — This appendix covers mathematical functions that are
329 used to describe elliptic curve operations and representation conversions, such as inversion,
330 and taking square roots.
- 331 • **Appendix F: Data Conversion** — This appendix documents the detailed procedure for the
332 conversion of data elements, such as integers, field elements, bit strings and octet strings,
333 and elliptic curve points.
- 334 • **Appendix G: Implementation Aspects** — This appendix discusses various implementation
335 aspects of binary curves, including conversions between different field representations; for
336 prime curves, it indicates how the special form of the underlying prime field aids in efficient
337 modular reduction.
- 338 • **Appendix H: Other Allowed Elliptic Curves** — This appendix lists other elliptic curves
339 that may be used for interoperability reasons.
- 340

341 **2 Glossary of Terms, Symbols, and Abbreviations**342 **2.1 Glossary**

Group Order	Cardinality of the group.
Identity	Unique group element 0 for which $x+0=x$ for each group element x , relative to the binary group operator $+$.
Inverse	For some group element x , the unique element y for which $x+y$ is the identity element relative to the binary group operator $+$ (y is usually denoted as $-x$).
Isogeny	Morphism from a first elliptic curve to a second elliptic curve.
Isomorphism	Morphism that is, in fact, a bijection.
Kernel	For a morphism, the set of group elements that map to the identity element.
l -isogeny	Isogeny with kernel of size l (Note: if $l=1$, an l -isogeny is an isomorphism).
Morphism	Mapping from a first group to a second group that maintains the group structure.
Point at Infinity	Identity element of a Montgomery curve or a curve in short-Weierstrass form.
Point Order	Smallest multiple of a group element that results in the group's identity element.
Quadratic Twist	Certain elliptic curve related to a specified elliptic curve.
Square	The property that some element x of a finite field $\text{GF}(q)$ can be written as $x=z^2$ for some element z in the same field $\text{GF}(q)$.

343

344 **2.2 Symbols and Abbreviations**

345 Selected acronyms and abbreviations used in this publication are defined below.

$a \bmod n$	Smallest non-negative integer r so that $a-r$ is a multiple of n .
$\lfloor a \rfloor$	The floor of a ; the largest integer that is less than or equal to a . For example, $\lfloor 5 \rfloor = 5$, $\lfloor 5.3 \rfloor = 5$, and $\lfloor -2.1 \rfloor = -3$.

$B_{a,b}$	Elliptic curve in short-Weierstrass form defined over the binary field $\text{GF}(2^m)$, with domain parameters a and b .
c	Parameter used in domain parameter generation for some curves $W_{a,b}$ in short-Weierstrass form, where $c = a^2/b^3$ (optional).
D	Domain parameters of elliptic curve.
$E_{a,d}$	Twisted Edwards curve, with domain parameters a and d .
G	Base point of order n of an elliptic curve.
$\text{GF}(q)$	Finite field of size q .
$\text{GF}(p)$	Prime field of size p , represented by the set of integers $\{0, 1, \dots, p-1\}$.
h	Co-factor of an elliptic curve.
Hf	Half-trace function (for binary fields).
$\text{len}(a)$	The length of a in bits; the integer L , where $2^{L-1} \leq a < 2^L$.
$M_{A,B}$	Montgomery curve, with domain parameters A and B .
n	Order of a prime-order subgroup of elliptic curve.
p	Prime Number.
RBG	Random Bit Generator.
<i>Seed</i>	String from which part of the domain parameters are derived (optional).
tr	Trace of an elliptic curve.
Tr	Trace function (for binary fields).
<i>Type</i>	Indication of elliptic curve type.
u, v	Coordinates on a Montgomery curve.
$W_{a,b}$	Elliptic curve in short-Weierstrass form, with domain parameters a and b .
x, y	Coordinates on a (twisted) Edwards or Weierstrass curve.
x', y'	Coordinates on an Edwards448 curve that correspond to the x, y coordinates on an E448 curve.

0x	Indication of a hexadecimal string.
∅	Identity element of an elliptic curve.
\	Indication that an integer value stretches over several lines.

346

3 Overview of Elliptic Curves

3.1 Non-Binary Curves

3.1.1 Curves in Short-Weierstrass Form

Let $\text{GF}(q)$ 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 the defining equation $y^2 = x^3 + ax + b$, where a and b are elements of $\text{GF}(q)$ with $4a^3 + 27b^2 \neq 0$. When selecting curve parameters, a *Seed* value may be used to generate the parameters a and b as described in Appendix C.2.1.1.

The points of $W_{a,b}$ are the ordered pairs (x, y) whose coordinates are elements of $\text{GF}(q)$ and that satisfy the defining equation (i.e., the affine points), together with the special point \emptyset (the “point at infinity”). This set forms a group under the operation of addition on elliptic curves via the “chord-and-tangent” rule, where the point at infinity serves as the identity element. See Appendix A.1.1 for details of the group operation.

3.1.2 Montgomery Curves

Let $\text{GF}(q)$ 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 $Bv^2 = u(u^2 + Au + 1)$, where A and B are elements of $\text{GF}(q)$ with $A \neq \pm 2$ and $B \neq 0$. The points of $M_{A,B}$ are the ordered pairs (u, v) whose coordinates are elements of $\text{GF}(q)$ and that satisfy the defining equation (i.e., the affine points), together with the special point \emptyset (the “point at infinity”). This set forms a group under the operation of addition on elliptic curves via the “chord-and-tangent” rule, where the point at infinity serves as the identity element. See Appendix A.1.2 for details of the group operation.

3.1.3 Twisted Edwards Curves

Let $\text{GF}(q)$ 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 $ax^2 + y^2 = 1 + dx^2y^2$, where a and d are elements of $\text{GF}(q)$ with $a, d \neq 0$ and $a \neq d$. The points of $E_{a,d}$ are the ordered pairs (x, y) whose coordinates are elements of $\text{GF}(q)$ and that satisfy the defining equation (i.e., the affine points). It can be shown that this set forms a group under the operation addition, where the point $(0, 1)$ serves as the identity element. If a is a square in $\text{GF}(q)$, and d is not, the addition formulae are complete, meaning that the formulae work for all inputs on the curve. See Appendix A.1.3 for details of the group operation.

An Edwards curve is a twisted Edwards curve with $a=1$. Edwards curves are to be used with the EdDSA digital signature scheme [FIPS 186-5].

3.2 Binary Curves

3.2.1 Curves in Short-Weierstrass Form

Let $\text{GF}(q)$ denote the finite field with q elements, where $q=2^m$. Let $B_{a,b}$ be the Weierstrass curve with defining equation $y^2 + xy = x^3 + ax^2 + b$, where a and b are elements of $\text{GF}(q)$ with $b \neq 0$.

383 The points of $B_{a,b}$ are the ordered pairs (x, y) whose coordinates are elements of $GF(q)$ and that
384 satisfy the defining equation (i.e., the affine points), together with the special point \mathcal{O} (the “point
385 at infinity”). This set forms a group under the operation of addition on elliptic curves via the
386 “chord-and-tangent” rule, where the point at infinity serves as the identity element. See
387 Appendix A.2.1 for details of the group operation.

388 **4 Recommended Curves for U.S. Federal Government Use**

389 This section specifies the elliptic curves recommended for U.S. Federal Government use and
390 contains choices for the private key length and underlying fields. This includes elliptic curves
391 over prime fields (Section 4.2) and elliptic curves over binary fields (Section 4.3) where each
392 curve takes one of the forms described in Section 3 (referred to as “*Type*” below).

393 Each recommended curve is uniquely defined by its domain parameters D , which indicate the
394 field $\text{GF}(q)$ over which the elliptic curve is defined and the parameters of its defining equation,
395 as well as principal parameters such as the co-factor h of the curve, the order n of its prime-order
396 subgroup, and a designated point $G=(G_x, G_y)$ on the curve of order n (i.e., the “base point”).

397 When ECDSA domain parameters are generated (i.e., the NIST-recommended curves for
398 ECDSA are not used), the value of G **should** be generated canonically (verifiably random). An
399 **approved** hash function (such as those specified in FIPS 180 or FIPS 202) **shall** be used during
400 the generation of ECDSA domain parameters. When generating these domain parameters, the
401 security strength of a hash function used **shall** meet or exceed the security strength associated
402 with the bit length of n .¹

403 Let E be an elliptic curve defined over the field $\text{GF}(q)$.

404 The cardinality $|E|$ of the curve is equal to the number of points on the curve and satisfies the
405 equation $|E|=(q+1)-tr$, where $|tr| \leq 2\sqrt{q}$. (Thus, $|E|$ and q have the same order of magnitude.)

406 The integer tr is called the trace of E over the field $\text{GF}(q)$.

407 The points on E form a commutative group under addition (for the group law for each curve
408 form, see Appendix A). Any point P on the curve is the generator of a cyclic subgroup $\langle P \rangle = \{kP$
409 $| k=0, 1, 2, \dots\}$ of E . The *order* of P in E is defined as the cardinality of $\langle P \rangle$. A curve is cyclic if
410 it is generated by some point on E . All curves of prime order are cyclic, while all curves of order
411 $|E|=h \cdot n$, where n is a large prime number and where h is small number, have a large cyclic
412 subgroup of prime order n .

413 If R is a point on the curve that is also contained in $\langle P \rangle$, there is a unique integer k in the interval
414 $[0, l-1]$ so that $R=kP$, where l is the order of P in E . This number is called the discrete logarithm
415 of R to the base P . The discrete logarithm problem is the problem of finding the discrete
416 logarithm of R to the base P for any two points P and R on the curve, if such a number exists.

417 A quadratic twist of E is a curve E' related to E , with cardinality $|E'|=(q+1)+tr$. If E is a curve in
418 one of the curve forms specified in this Recommendation, a quadratic twist of this curve can be
419 expressed using the same curve model, although (naturally) with different curve parameters.

¹ The NIST-recommended curves for ECDSA were generated prior to the formulation of this guidance and using SHA-1, which was the only approved hash function available at that time. Since SHA-1 was considered secure at the time of generation, the curves were made public, and SHA-1 will only be used to validate those curves, the NIST-recommended curves for ECDSA are still considered secure and appropriate for Federal Government use.

420 For details regarding the generation method of the elliptic curves, see Appendix C.

421 **4.1 Choices of Key Lengths, Underlying Fields, Curves, and Base Points**

422 **4.1.1 Choice of Key Lengths**

423 The principal parameters for elliptic curve cryptography are the elliptic curve E and a designated
424 point G on E called the *base point*. The base point has order n , which is a large prime. The
425 number of points on the curve is $h \cdot n$ for some integer h (the *cofactor*), which is not divisible by
426 n . For efficiency reasons, it is desirable to have the cofactor be as small as possible.

427 All of the curves given below have cofactors 1, 2, or 4. As a result, the private and public keys
428 for a curve are approximately the same length.

429 **4.1.2 Choice of Underlying Fields**

430 For each key length, two kinds of fields are provided:

- 431 • A *prime field* is the field $\text{GF}(p)$, which contains a prime number p of elements. The
432 elements of this field are the integers modulo p , and the field arithmetic is implemented
433 in terms of the arithmetic of integers modulo p .
- 434 • A *binary field* is the field $\text{GF}(2^m)$, which contains 2^m elements for some m (called the
435 *degree* of the field). The elements of this field are the bit strings of length m , and the field
436 arithmetic is implemented in terms of operations on the bits.

437 The security strengths for four ranges of the bit length of n are provided in SP 800-57, Part 1. For
438 the field $\text{GF}(p)$, the security strength is dependent on the length of the binary expansion of p . For
439 the field $\text{GF}(2^m)$, the security strength is dependent on the value of m . Table 1 provides the bit
440 lengths of the various underlying fields of the curves provided in this appendix. Column 1 lists
441 the ranges for the bit length of n . Column 2 identifies the value of p used for the curves over
442 prime fields, where $\text{len}(p)$ is the length of the binary expansion of the integer p . Column 3
443 provides the value of m for the curves over binary fields.

444 **Table 1: Bit Lengths of the Underlying Fields of the Recommended Curves**

Bit Length of n	Prime Field	Binary Field
224 – 255	$\text{len}(p) = 224$	$m = 233$
256 – 383	$\text{len}(p) = 256$	$m = 283$
384 – 511	$\text{len}(p) = 384$	$m = 409$
≥ 512	$\text{len}(p) = 521$	$m = 571$

445

446 **4.1.3 Choice of Basis for Binary Fields**

447 To describe the arithmetic of a binary field, it is first necessary to specify how a bit string is to be
 448 interpreted. This is referred to as choosing a *basis* for the field. There are two common types of
 449 bases: a *polynomial basis* and a *normal basis*.

- 450 • A polynomial basis is specified by an irreducible polynomial modulo 2, called the *field*
 451 *polynomial*. The bit string $(a_{m-1} \dots a_2 a_1 a_0)$ is used to represent the polynomial

$$452 \quad a_{m-1} t^{m-1} + \dots + a_2 t^2 + a_1 t + a_0$$

453 over GF(2). The field arithmetic is implemented as polynomial arithmetic modulo $p(t)$,
 454 where $p(t)$ is the field polynomial.

- 455 • A normal basis is specified by an element θ of a particular kind. The bit string $(a_0 a_1 a_2$
 456 $\dots a_{m-1})$ is used to represent the element

$$457 \quad a_0 \theta + a_1 \theta^2 + a_2 \theta^{2^2} + \dots + a_{m-1} \theta^{2^{m-1}}.$$

458 Normal basis field arithmetic is not easy to describe or efficient to implement in general
 459 except for a special class called *Type T low-complexity* normal bases. For a given field of
 460 degree m , the choice of T specifies the basis and the field arithmetic (see Appendix G.3).

461 There are many polynomial bases and normal bases from which to choose. The following
 462 procedures are commonly used to select a basis representation:

- 463 • *Polynomial Basis*: If an irreducible *trinomial* $t^m + t^k + 1$ exists over GF(2), then the field
 464 polynomial $p(t)$ is chosen to be the irreducible trinomial with the lowest-degree middle
 465 term t^k . If no irreducible trinomial exists, then a *pentanomial* $t^m + t^a + t^b + t^c + 1$ is
 466 selected. The particular pentanomial chosen has the following properties: the second term
 467 t^a has the lowest degree m ; the third term t^b has the lowest degree among all irreducible
 468 pentanomials of degree m and the second term t^a ; and the fourth term t^c has the lowest
 469 degree among all irreducible pentanomials of degree m , with the second term t^a , and third
 470 term t^b .
- 471 • *Normal Basis*: Choose the *Type T low-complexity* normal basis with the smallest T .

472 For each binary field, the parameters are given for the above basis representations.

473 **4.1.4 Choice of Curves**

474 Two kinds of curves are given:

- 475 • *Pseudorandom* curves are those whose coefficients are generated from the output of a
 476 seeded cryptographic hash function. If the domain parameter seed value is given along
 477 with the coefficients, it can be easily verified that the coefficients were generated by that
 478 method.
- 479 • *Special curves* are those whose coefficients and underlying field have been selected to

480 optimize the efficiency of the elliptic curve operations.

481 For each curve size range, the following curves are given:

482 → A pseudorandom curve over $\text{GF}(p)$.

483 → A pseudorandom curve over $\text{GF}(2^m)$.

484 → Special curves over $\text{GF}(p)$ called *Edwards curves* and *Montgomery curves*.

485 → A special curve over $\text{GF}(2^m)$ called a *Koblitz curve* or *anomalous binary curve*.

486 The pseudorandom curves were generated as specified in Appendix C.3.

487 **4.1.5 Choice of Base Points**

488 Since any point of order n can serve as the base point, users could, in principle, generate their
489 own base points to ensure a cryptographic separation of networks, although this does result in
490 another set of domain parameters. When generating base points, users **should** use a verifiably
491 random method and check the validity of the point generated. See Appendix D.3 for more
492 details. If a base point is generated by another entity, it is recommended that its validity be
493 verified with the procedure in Appendix D.3.3 prior to use.

494 **4.2 Curves over Prime Fields**

495 This section specifies elliptic curves over prime fields recommended for U.S. Federal
496 Government use, where each curve takes the form of a curve in short-Weierstrass form (Section
497 4.2.1), a Montgomery curve (Section 4.2.2), or a twisted Edwards curve (Section 4.2.3).

498 **4.2.1 Weierstrass Curves**

499 This specification includes pseudorandom Weierstrass curves generated over prime fields P-192,
500 P-224, P-256, P-384, and P-521 (See Sections 4.2.1.1 - 4.2.1.5) and special Weierstrass curves
501 over prime fields W-25519 (Section 4.2.1.6) and W-448 (Section 4.2.1.7). The curves W-25519
502 and W-448 may provide improved performance of the elliptic curve operations as well as
503 increased resilience against side-channel attacks while allowing for ease of integration with
504 existing implementations.

505 For each Weierstrass curve,

$$506 \quad E : y^2 \equiv x^3 + ax + b \pmod{p},$$

507 the following domain parameters $D=(p, h, n, Type, a, b, G, \{Seed, c\})$ are given:

- 508 • The prime modulus p
- 509 • The cofactor h
- 510 ○ For pseudorandom curves, the cofactor $h = 1$ so the order n is prime

- 511 ○ For special curves, the cofactor $h > 1$ so the order n is not prime
- 512 • The *Type* is “Weierstrass curve”
- 513 • The coefficient a
- 514 ○ For pseudorandom curves, $a = -3$ was made for reasons of efficiency; see IEEE
515 Std 1363-2000
- 516 • The coefficient b
- 517 ○ For pseudorandom curves, the coefficient b satisfies $b^2 c \equiv -27 \pmod{p}$
- 518 • The base point G with x coordinate G_x and y coordinate G_y
- 519 • The 160-bit input *Seed* to the SHA-1 hash algorithm in Appendix C.3 for pseudorandom
520 curves. *Seed* is not used with the special curves W-25519 (Section 4.2.1.6) and W-448
521 (Section 4.2.1.7).
- 522 • The output c of the SHA-1 hash algorithm used for pseudorandom curves. The value c is
523 not used with the special curves W-25519 (Section 4.2.1.6) and W-448 (Section 4.2.1.7).

524 The integers p and n are given in decimal form; bit strings and field elements are given in
525 hexadecimal.

526 4.2.1.1 P-192

527 The use of this curve is for legacy-use only. See [\[FIPS 186-4\]](#) for the specification.

528 4.2.1.2 P-224

529 The elliptic curve P-224 is a Weierstrass curve $W_{a,b}$ defined over the prime field $\text{GF}(p)$ that has
530 order $h \cdot n$, where $h=1$ and where n is a prime number. This curve has domain parameters $D=(p,$
531 $h, n, \textit{Type}, a, b, G, \{\textit{Seed}, c\})$, where the *Type* is “Weierstrass curve” and the other parameters
532 are defined as follows:

533

534 $p:$ $2^{224} - 2^{96} + 1$
535 $= 26959946667150639794667015087019630673557916260026308143510066298881$
536 (=0xffffffff ffffffff ffffffff ffffffff 00000000 00000000 00000001)

537 $h:$ 1

538 $n:$ 26959946667150639794667015087019625940457807714424391721682722368061
539 (=0xffffffff ffffffff ffffffff ffff16a2 e0b8f03e 13dd2945 5c5c2a3d)

540 $tr:$ 4733100108545601916421827343930821
541 (= $(p+1) - h \cdot n = 0xe95c 1f470fc1 ec22d6ba a3a3d5c5$)

542 $a:$ -3
543 $= 26959946667150639794667015087019630673557916260026308143510066298878$
544 (=0xffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff)

545 b : 18958286285566608000408668544493926415504680968679321075787234672564
546 (=0xb4050a85 0c04b3ab f5413256 5044b0b7 d7bfd8ba 270b3943 2355ffb4)
547 G_x : 19277929113566293071110308034699488026831934219452440156649784352033
548 (=0xb70e0cbd 6bb4bf7f 321390b9 4a03c1d3 56c21122 343280d6 115c1d21)
549 G_y : 19926808758034470970197974370888749184205991990603949537637343198772
550 (=0xbd376388 b5f723fb 4c22dfe6 cd4375a0 5a074764 44d58199 85007e34)
551 *Seed*: 0xbd713447 99d5c7fc dc45b59f a3b9ab8f 6a948bc5
552 c : 9585649763196999776159690989286240671136085803543320687376622326267
553 (=0x5b056c7e 11dd68f4 0469ee7f 3c7a7d74 f7d12111 6506d031 218291fb)
554

555 4.2.1.3 P-256

556 The elliptic curve P-256 is a Weierstrass curve $W_{a,b}$ defined over the prime field $GF(p)$ that has
557 order $h \cdot n$, where $h=1$ and where n is a prime number. This curve has domain parameters $D=(p,$
558 $h, n, Type, a, b, G, \{Seed, c\})$, where the *Type* is “Weierstrass curve” and the other parameters
559 are defined as follows:
560

561 p : $2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$
562 = 115792089210356248762697446949407573530\
563 086143415290314195533631308867097853951
564 (=0xffffffff 00000001 00000000 00000000 00000000 00000000 ffffffff ffffffff
565 ffffffff)
566 h : 1
567 n : 115792089210356248762697446949407573529\
568 996955224135760342422259061068512044369
569 (=0xffffffff 00000000 ffffffff ffffffff bce6faad a7179e84 f3b9cac2
570 fc632551)
571 tr : 89188191154553853111372247798585809583
572 ($= (p+1) - h \cdot n = 0x43190553 58e8617b 0c46353d 039cdaaf$)
573 a : -3
574 = 115792089210356248762697446949407573530\
575 086143415290314195533631308867097853948
576 (=0xffffffff 00000001 00000000 00000000 00000000 00000000 ffffffff ffffffff
577 ffffffff)
578 b : 41058363725152142129326129780047268409\
579 114441015993725554835256314039467401291
580 (=0x5ac635d8 aa3a93e7 b3ebbd55 769886bc 651d06b0 cc53b0f6 3bce3c3e
581 27d2604b)
582 G_x : 48439561293906451759052585252797914202\
583 762949526041747995844080717082404635286
584 (=0x6b17d1f2 e12c4247 f8bce6e5 63a440f2 77037d81 2deb33a0 f4a13945
585 d898c296)
586 G_y : 36134250956749795798585127919587881956\
587 611106672985015071877198253568414405109
588 (=0x4fe342e2 fe1a7f9b 8ee7eb4a 7c0f9e16 2bce3357 6b315ece cbb64068

589 37bf51f5)
 590 *Seed*: 0xc49d3608 86e70493 6a6678e1 139d26b7 819f7e90
 591 *c*: 57436011470200155964173534038266061871\
 592 440426244159038175955947309464595790349
 593 (=0x7efba166 2985be94 03cb055c 75d4f7e0 ce8d84a9 c5114abc af317768
 594 0104fa0d)

596 4.2.1.4 P-384

597 The elliptic curve P-384 is a Weierstrass curve $W_{a,b}$ defined over the prime field $GF(p)$ that has
 598 order $h \cdot n$, where $h=1$ and where n is a prime number. This curve has domain parameters $D=(p,$
 599 $h, n, Type, a, b, G, \{Seed, c\})$, where the *Type* is “Weierstrass curve” and the other parameters
 600 are defined as follows:

601
 602 *p*: $2^{384} - 2^{128} - 2^{96} + 2^{32} - 1$
 603 = 3940200619639447921227904010014361380507973927046544666794\
 604 8293404245721771496870329047266088258938001861606973112319
 605 (=0xffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
 606 ffffffff ffffffff ffffffff 00000000 00000000 ffffffff)
 607 *h*: 1
 608 *n*: 3940200619639447921227904010014361380507973927046544666794\
 609 6905279627659399113263569398956308152294913554433653942643
 610 (=0xffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
 611 c7634d81 f4372ddf 581a0db2 48b0a77a ecec196a ccc52973)
 612 *tr*: 1388124618062372383606759648309780106643088307173319169677
 613 (= $(p+1) - h \cdot n$ = 0x389cb27e 0bc8d21f a7e5f24c b74f5885 1313e696
 614 333ad68d)
 615 *a*: -3
 616 = 3940200619639447921227904010014361380507973927046544666794\
 617 8293404245721771496870329047266088258938001861606973112316
 618 (=0xffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
 619 ffffffff ffffffff ffffffff 00000000 00000000 ffffffff)
 620 *b*: 2758019355995970587784901184038904809305690585636156852142\
 621 8707301988689241309860865136260764883745107765439761230575
 622 (=0xb3312fa7 e23ee7e4 988e056b e3f82d19 181d9c6e fe814112
 623 0314088f 5013875a c656398d 8a2ed19d 2a85c8ed d3ec2aef)
 624 *G_x*: 2624703509579968926862315674456698189185292349110921338781\
 625 5615900925518854738050089022388053975719786650872476732087
 626 (=0xaa87ca22 be8b0537 8eb1c71e f320ad74 6e1d3b62 8ba79b98
 627 59f741e0 82542a38 5502f25d bf55296c 3a545e38 72760ab7)
 628 *G_y*: 832571096148902998554675128952010817928785304886131559470\
 629 9205902480503199884419224438643760392947333078086511627871
 630 (=0x3617de4a 96262c6f 5d9e98bf 9292dc29 f8f41dbd 289a147c
 631 e9da3113 b5f0b8c0 0a60b1ce 1d7e819d 7a431d7c 90ea0e5f)
 632 *Seed*: 0xa335926a a319a27a 1d00896a 6773a482 7acdac73

633 c : 1874980186709887347182107097135388878869033900306543902178\
 634 0101954060871745882341382251168574711376101826101037376643
 635 (=0x79d1e655 f868f02f ff48dcde e14151dd b80643c1 406d0ca1
 636 0dfe6fc5 2009540a 495e8042 ea5f744f 6e184667 cc722483)
 637

638 4.2.1.5 P-521

639 The elliptic curve P-521 is a Weierstrass curve $W_{a,b}$ defined over the prime field $GF(p)$ that has
 640 order $h \cdot n$, where $h=1$ and where n is a prime number. This curve has domain parameters $D=(p,$
 641 $h, n, Type, a, b, G, \{Seed, c\})$, where the *Type* is “Weierstrass curve” and the other parameters
 642 are defined as follows:
 643

644 p : $2^{521} - 1$
 645 = 686479766013060971498190079908139321726943530014330540939\
 646 446345918554318339765605212255964066145455497729631139148 \
 647 0858037121987999716643812574028291115057151
 648 (=0x1ff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
 649 ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
 650 ffffffff ffffffff ffffffff ffffffff)

651 h : 1
 652 n : 686479766013060971498190079908139321726943530014330540939\
 653 446345918554318339765539424505774633321719753296399637136\
 654 3321113864768612440380340372808892707005449
 655 (=0x1ff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
 656 ffffffff ffffffff 51868783 bf2f966b 7fcc0148 f709a5d0
 657 3bb5c9b8 899c47ae bb6fb71e 91386409)

658 tr : 657877501894328237357444332315020117536\
 659 923257219387276263472201219398408051703
 660 (= $(p+1) - h \cdot n = 0x5$ ae79787c 40d06994 8033feb7 08f65a2f
 661 c44a3647 7663b851 449048e1 6ec79bf7)

662 a : -3
 663 = 686479766013060971498190079908139321726943530014330540939\
 664 446345918554318339765605212255964066145455497729631139148 \
 665 0858037121987999716643812574028291115057148
 666 (=0x1ff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
 667 ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
 668 ffffffff ffffffff ffffffff ffffffff)

669 b : 1093849038073734274511112390766805569936207598951683748994\
 670 5863944959531161507350160137087375737596232485921322967063\
 671 13309438452531591012912142327488478985984
 672 (=0x051 953eb961 8e1c9a1f 929a21a0 b68540ee a2da725b 99b315f3
 673 b8b48991 8ef109e1 56193951 ec7e937b 1652c0bd 3bb1bf07
 674 3573df88 3d2c34f1 ef451fd4 6b503f00)

675 G_x : 2661740802050217063228768716723360960729859168756973147706\
 676 6713684188029449964278084915450806277719023520942412250655\
 677 58662157113545570916814161637315895999846
 678 (=0xc6 858e06b7 0404e9cd 9e3ecb66 2395b442 9c648139 053fb521

```

679         f828af60 6b4d3dba a14b5e77 efe75928 fe1dc127 a2ffa8de
680         3348b3c1 856a429b f97e7e31 c2e5bd66)
681 Gy: 37571800257700204635455072244911836035944551347697624866945\
682 67779615544477440556316691234405012945539562144444537289428\
683 522585666729196580810124344277578376784
684         (=0x118 39296a78 9a3bc004 5c8a5fb4 2c7d1bd9 98f54449 579b4468
685         17afb1d17 273e662c 97ee7299 5ef42640 c550b901 3fad0761
686         353c7086 a272c240 88be9476 9fd16650)
687 Seed: 0xd09e8800 291cb853 96cc6717 393284aa a0da64ba
688 c: 2420736670956961470587751833778383872272949280174637971106318\
689 2239560106363555573338990358663426503785752212772688861827046\
690 43828850020061383251826928984446519
691         (=0x0b4 8bfa5f42 0a349495 39d2bdfc 264eeeeb 077688e4 4fbf0ad8
692         f6d0edb3 7bd6b533 28100051 8e19f1b9 ffbe0fe9 ed8a3c22
693         00b8f875 e523868c 70c1e5bf 55bad637)
694
695

```

4.2.1.6 W-25519

696 The elliptic curve W-25519 is a Weierstrass curve $W_{a,b}$ defined over the prime field $GF(p)$, with
697 $p=2^{255}-19$, and that has order $h \cdot n$, where $h=8$ and where n is a prime number. The quadratic twist
698 of this curve has order $h_1 \cdot n_1$, where $h_1=4$ and where n_1 is a prime number. This curve has domain
699 parameters $D=(p, h, n, Type, a, b, G)$, where the *Type* is “Weierstrass curve” and the other
700 parameters are defined as follows:

```

701
702 p: 2255-19
703         (=0x7fffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
704         ffffffff)
705 h: 8
706 n: 72370055773322622139731865630429942408\
707 57116359379907606001950938285454250989
708         (=2252 + 0x14def9de a2f79cd6 5812631a 5cf5d3ed)
709 tr: -221938542218978828286815502327069187962
710         (= (p+1) - h · n = -0xa6f7cef5 17bce6b2 c09318d2 e7ae9f7a)
711 a: 19298681539552699237261830834781317975\
712 544997444273427339909597334573241639236
713         (=0x2aaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaa98
714         4914a144)
715 b: 55751746669818908907645289078257140818\
716 241103727901012315294400837956729358436
717         (=0x7b425ed0 97b425ed 097b425e d097b425 ed097b42 5ed097b4 260b5e9c
718         7710c864)
719 Gx: 19298681539552699237261830834781317975\
720 544997444273427339909597334652188435546
721         (=0x2aaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaaaaa
722         aaaaaaaaa aaad245a)
723 Gy: 43114425171068552920764898935933967039\

```

724 370386198203806730763910166200978582548
725 (=0x5f51e65e 475f794b 1fe122d3 88b72eb3 6dc2b281 92839e4d
726 d6163a5d 81312c14)
727

728 The curve W-25519 is isomorphic to the curve Curve25519 specified in Section 4.2.2.1, where
729 the base point of Curve25519 corresponds to the base point of W-25519, where the point at
730 infinity \emptyset of Curve25519 corresponds to the point at infinity \emptyset on W-25519 and where the point
731 (u, v) on Curve25519 corresponds to the point $(x, y)=(u+A/3, v)$ on $W_{a,b}$.
732

733 See Appendix B.2 for more details.
734

735 Note that Curve25519 is not isomorphic with a Weierstrass curve with domain parameter $a = -3$.
736 In particular, this means that one cannot reuse an implementation for elliptic curves with short-
737 Weierstrass form that hard-codes the domain parameter a to -3 to implement Curve25519.
738

739 **4.2.1.7 W-448**

740 The elliptic curve Curve448 is the Weierstrass curve $W_{a,b}$ defined over the prime field $GF(p)$,
741 with $p=2^{448}-2^{224}-1$, and that has order $h \cdot n$, where $h=4$ and where n is a prime number. The
742 quadratic twist of this curve has order $h_1 \cdot n_1$, where $h_1=4$ and where n_1 is a prime number. This
743 curve has domain parameters $D=(p, h, n, Type, a, b, G)$, where the *Type* is “Weierstrass curve”
744 and the other parameters are defined as follows:
745

746 $p:$ $2^{448}-2^{224}-1$
747 (=0xffffffff ffffffff ffffffff ffffffff ffffffff ffffffff fffffffe
748 ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff)
749 $h:$ 4
750 $n:$ 1817096810739017226373309519720011335884103401718295150703725497951
751 46003961539585716195755291692375963310293709091662304773755859649779
752 (=2⁴⁴⁶ - 0x8335dc16 3bb124b6 5129c96f de933d8d 723a70aa dc873d6d
753 54a7bb0d)
754 $tr:$ 28312320572429821613362531907042076847709625476988141958474579766324
755 (= $(p+1) - h \cdot n$ = 0x1 0cd77058 eec492d9 44a725bf 7a4cf635 c8e9c2ab
756 721cf5b5 529eec34)
757 $a:$ 4845591495304045936995492052586696895690942404582120401876601327870
758 74885444487181790930922465784363953392589641229091574035657199637535
759 (=0xaaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaaa9
760 ffffffff ffffffff ffffffff ffffffff ffffffff fffffffe 1a76d41f)
761 $b:$ 2691995275168914409441940029214831608717190224767844667709222959928
762 19380802492878772739401369880202196329216467349495319191685664513904
763 (=0x5ed097b4 25ed097b 425ed097 b425ed09 7b425ed0 97b425ed 097b425e
764 71c71c71 c71c71c7 1c71c71c 71c71c71 c71c71c7 1c72c87b 7cc69f70)
765 $G_x:$ 4845591495304045936995492052586696895690942404582120401876601327870\
766 7488544448718179093092246578436395339258964122909157403566534562907
767 (=0xaaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaaaaa aaaaaaaa
768 00000000 00000000 00000000 00000000 00000000 00000000 0000cb91)

769 G_y : 3552939267855681752641275020637833348089763993877142718318808984351\
 770 69088786967410002932673765864550910142774147268105838985595290606362
 771 (=0x7d235d12 95f5b1f6 6c98ab6e 58326fce cbae5d34 f55545d0 60f75dc2
 772 8df3f6ed b8027e23 46430d21 1312c4b1 50677af7 6fd7223d 457b5b1a)
 773

774 The curve W-448 is isomorphic to the curve Curve448 specified in Section 4.2.2.2, where the
 775 base point of Curve448 corresponds to the base point of W-448, where the point at infinity \emptyset of
 776 Curve448 corresponds to the point at infinity \emptyset on W-448 and where the point (u, v) on
 777 Curve448 corresponds to the point $(x, y)=(u+A/3, v)$ on $W_{a,b}$.
 778

779 See Appendix B.2 for more details.
 780

781 Note that Curve448 is not isomorphic with a Weierstrass curve with domain parameter $a = -3$. In
 782 particular, this means that one cannot reuse an implementation for curves with short-Weierstrass
 783 form that hard-codes the domain parameter a to -3 to implement Curve448.
 784

785 4.2.2 Montgomery Curves

786 Similar to W-25519 and W-448, Montgomery curves may offer improved performance with
 787 improved resistance to side-channel attacks. These curves can also provide a bridge between
 788 short-Weierstrass curves and Edwards curves.

789 4.2.2.1 Curve25519

790 The elliptic curve Curve25519 is the Montgomery curve $M_{A,B}$ defined over the prime field
 791 $\text{GF}(p)$, with $p=2^{255}-19$, and with parameters $A=486662$ and $B=1$ [[RFC 7748](#)]. This curve has
 792 order $h \cdot n$, where $h=8$ and where n is a prime number. For this curve, A^2-4 is not a square in
 793 $\text{GF}(p)$, whereas $A+2$ is. The quadratic twist of this curve has order $h_1 \cdot n_1$, where $h_1=4$ and where
 794 n_1 is a prime number. This curve has domain parameters $D=(p, h, n, \text{Type}, A, B, G)$, where the
 795 *Type* is “Montgomery curve” and where the other parameters are defined as follows:
 796

797 p : $2^{255}-19$
 798 (=0x7fffffffff ffffffffff ffffffffff ffffffffff ffffffffff ffffffffff ffffffffff
 799 ffffffffed)

800 h : 8

801 n : 72370055773322622139731865630429942408\
 802 57116359379907606001950938285454250989
 803 (=2²⁵² + 0x14def9de a2f79cd6 5812631a 5cf5d3ed)

804 tr : -221938542218978828286815502327069187962
 805 (= $(p+1) - h \cdot n = -0xa6f7cef5 17bce6b2 c09318d2 e7ae9f7a$)

806 A : 486662

807 B : 1

808 G_u : 9
 809 (=0x9)

810 G_v : 43114425171068552920764898935933967039\
 811 370386198203806730763910166200978582548

812 (=0x5f51e65e 475f794b 1fe122d3 88b72eb3 6dc2b281 92839e4d d6163a5d
813 81312c14)
814
815

816 4.2.2.2 Curve448

817 The elliptic curve Curve448 is the Montgomery curve $M_{A,B}$ defined over the prime field $GF(p)$,
818 with $p=2^{448}-2^{224}-1$, and with parameters $A=156326$ and $B=1$ [RFC 7748]. This curve has order
819 $h \cdot n$, where $h=4$ and where n is a prime number. For this curve, A^2-4 is not a square in $GF(p)$,
820 whereas $A-2$ is. The quadratic twist of this curve has order $h_1 \cdot n_1$, where $h_1 = 4$ and where n_1 is a
821 prime number. This curve has domain parameters $D=(p, h, n, Type, A, B, G)$, where the *Type* is
822 “Montgomery curve” and where the other parameters are defined as follows:

823
824 $p:$ $2^{448}-2^{224}-1$
825 (=0xffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
826 ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff)
827 $h:$ 4
828 $n:$ 1817096810739017226373309519720011335884103401718295150703725497951
829 46003961539585716195755291692375963310293709091662304773755859649779
830 (=2⁴⁴⁶ - 0x8335dc16 3bb124b6 5129c96f de933d8d 723a70aa dc873d6d
831 54a7bb0d)
832 $tr:$ 28312320572429821613362531907042076847709625476988141958474579766324
833 (= $(p+1) - h \cdot n$ = 0x1 0cd77058 eec492d9 44a725bf 7a4cf635 c8e9c2ab
834 721cf5b5 529eec34)
835 $A:$ 156326
836 $B:$ 1
837 $G_u:$ 5
838 (=0x5)
839 $G_v:$ 3552939267855681752641275020637833348089763993877142718318808984351\
840 69088786967410002932673765864550910142774147268105838985595290606362
841 (=0x7d235d12 95f5b1f6 6c98ab6e 58326fce cbae5d34 f55545d0 60f75dc2
842 8df3f6ed b8027e23 46430d21 1312c4b1 50677af7 6fd7223d 457b5b1a)
843

844 The base point of Curve448 corresponds to the base point of E448 and the point at infinity \emptyset ,
845 and the point (0,0) of order two of Curve448 correspond to, respectively, the point (0, 1) and the
846 point (0, -1) of order two on E448. Each other point (u, v) on Curve448 corresponds to the point
847 $(\alpha u/v, (u + 1)/(u - 1))$ on E448, where α is the element of $GF(p)$ defined by
848

849 $\alpha:$ 1978884672954644395383540097538580382568351525910598021481997791960\
850 87404232002515713604263127793030747855424464185691766453844835192428
851 (=0x45b2c5f7 d649eed0 77ed1ae4 5f44d541 43e34f71 4b71aa96 c945af01
852 2d182975 0734cde9 faddbda4 c066f7ed 54419ca5 2c85de1e 8aae4e6c)
853

854 See Appendix B.1 for more details.
855
856

857 **4.2.3 Twisted Edwards Curves**

858 Edwards curves offer high performance for elliptic curve calculations and protection against
859 side-channel attacks. The Edwards Curve Digital Signature Algorithm (EdDSA) is a digital
860 signature scheme based on twisted Edwards curves and is specified in FIPS 186-5.

861 **4.2.3.1 Edwards25519**

862 The elliptic curve Edwards25519 is the twisted Edwards curve $E_{a,d}$ defined over the prime field
863 $\text{GF}(p)$, with $p=2^{255}-19$, and with parameters $a=-1$ and $d=-121665/121666$ (i.e.,
864 $37095705934669439343138083508754565189542113879843219016388785533085940283555$)
865 [\[RFC 8032\]](#). This curve has order $h \cdot n$, where $h=8$ and where n is a prime number. For this curve,
866 a is a square in $\text{GF}(p)$, whereas d is not. The quadratic twist of this curve has order $h_1 \cdot n_1$, where
867 $h_1=4$ and where n_1 is a prime number. This curve has domain parameters $D=(p, h, n, \text{Type}, a, d,$
868 $G)$, where the *Type* is “twisted Edwards curve” and where the other parameters are defined as
869 follows:

```
870
871  $p:$  2255-19
872      (=0x7fffffffff ffffffffff ffffffffff ffffffffff ffffffffff ffffffffff ffffffffff
873      ffffffffed)
874  $h:$  8
875  $n:$  72370055773322622139731865630429942408\
876      57116359379907606001950938285454250989
877      (=2252 + 0x14def9de a2f79cd6 5812631a 5cf5d3ed)
878  $tr:$  -221938542218978828286815502327069187962
879      (= (p+1) - h · n = -0xa6f7cef5 17bce6b2 c09318d2 e7ae9f7a)
880  $a:$  -1
881  $d:$  -121665/121666 = 37095705934669439343138083508754565189\
882      542113879843219016388785533085940283555
883      (=0x52036cee 2b6ffe73 8cc74079 7779e898 00700a4d 4141d8ab 75eb4dca
884      135978a3)
885  $G_x:$  15112221349535400772501151409588531511\
886      454012693041857206046113283949847762202
887      (=0x216936d3 cd6e53fe c0a4e231 fdd6dc5c 692cc760 9525a7b2 c9562d60
888      8f25d51a)
889  $G_y:$  4/5 = 46316835694926478169428394003475163141\
890      307993866256225615783033603165251855960
891      (=0x66666666 66666666 66666666 66666666 66666666 66666666 66666666
892      66666658)
893
```

894 The curve Edwards25519 is isomorphic to the curve Curve25519 specified in Section 4.2.2.1,
895 where

- 896 • the base point of Curve25519 corresponds to the base point of Edwards25519;
- 897 • the point at infinity \emptyset and the point (0,0) of order two of Curve25519 correspond to,
898 respectively, the point (0, 1) and the point (0, -1) of order two on Edwards25519; and

- 899 • each other point (u, v) on Curve25519 corresponds to the point $(\alpha u/v, (u - 1)/(u + 1))$ on
900 Edwards25519, where α is the element of $\text{GF}(p)$ defined by

901
902 α : 51042569399160536130206135233146329284\
903 152202253034631822681833788666877215207
904 (=0x70d9120b 9f5fff944 2d84f723 fc03b081 3a5e2c2e b482e57d 3391fb55
905 00ba81e7).

906
907 The inverse mapping from Edwards25519 to Curve25519 is defined by

- 908 • mapping the point $(0, 1)$ and the point $(0, -1)$ of order two on Edwards25519 to,
909 respectively, the point at infinity \emptyset and the point $(0,0)$ of order two of Curve25519 and
910 • having each other point (x, y) on Edwards25519 correspond to the point $((1 + y)/(1 - y),$
911 $\alpha(1 + y)/(1 - y)x)$.

912
913 See Appendix B.1 for more details.

914 4.2.3.2 Edwards448

916 The elliptic curve Edwards448 is the Edwards curve $E_{a,d}$ defined over the prime field $\text{GF}(p)$, with
917 $p=2^{448}-2^{224}-1$, and with parameters $a=1$ and $d=-39081$ [RFC 8032]. This curve has order $h \cdot n$,
918 where $h=4$ and where n is a prime number. For this curve, a is a square in $\text{GF}(p)$, whereas d is
919 not. The quadratic twist of this curve has order $h_1 \cdot n_1$, where $h_1=4$ and where n_1 is a prime
920 number. This curve has domain parameters $D=(p, h, n, \text{Type}, a, d, G)$, where the *Type* is
921 “twisted Edwards curve” and where the other parameters are defined as follows:

922
923 p : $2^{448}-2^{224}-1$
924 (=0xffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffffe
925 ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff)
926 h : 4
927 n : 1817096810739017226373309519720011335884103401718295150703725497951
928 46003961539585716195755291692375963310293709091662304773755859649779
929 (=2⁴⁴⁶ - 0x8335dc16 3bb124b6 5129c96f de933d8d 723a70aa dc873d6d
930 54a7bb0d)
931 tr : 28312320572429821613362531907042076847709625476988141958474579766324
932 (= $(p+1) - h \cdot n$ = 0x1 0cd77058 eec492d9 44a725bf 7a4cf635 c8e9c2ab
933 721cf5b5 529eec34)
934 a : 1
935 d : -39081
936 = 7268387242956068905493238078880045343536413606873180602814901991806\
937 12328166730772686396383698676545930088884461843637361053498018326358
938 (=0xffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffffe
939 ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffff6756)
940 G_x : 2245800402959243001876043340998960362467896416325641342461254616869\
941 50415467406032909029192869357953282578032075146446173674602635247710
942 (=0x4f1970c6 6bed0ded 221d15a6 22bf36da 9e146570 470f1767 ea6de324

943 a3d3a464 12ae1af7 2ab66511 433b80e1 8b00938e 2626a82b c70cc05e)
 944 G_y : 2988192100784814926760179304439306734375440401540802420959282413723\
 945 31506189835876003536878655418784733982303233503462500531545062832660
 946 (=0x693f4671 6eb6bc24 88762037 56c9c762 4bea7373 6ca39840 87789c1e
 947 05a0c2d7 3ad3ff1c e67c39c4 fdbd132c 4ed7c8ad 9808795b f230fa14)

950 **4.2.3.3 E448**

951 The elliptic curve E448 is the Edwards curve $E_{a,d}$ defined over the prime field $GF(p)$, with
 952 $p=2^{448}-2^{224}-1$, and with parameters $a=1$ and $d=39082/39081$. This curve has order $h \cdot n$, where
 953 $h=4$ and where n is a prime number. For this curve, a is a square in $GF(p)$, whereas d is not. The
 954 quadratic twist of this curve has order $h_1 \cdot n_1$, where $h_1=4$ and where n_1 is a prime number. This
 955 curve has domain parameters $D=(p, h, n, Type, a, d, G)$, where the *Type* is “twisted Edwards
 956 curve” and where the other parameters are defined as follows:

957
 958 p : $2^{448}-2^{224}-1$
 959 (=0xffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
 960 ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff)
 961 h : 4
 962 n : 1817096810739017226373309519720011335884103401718295150703725497951
 963 46003961539585716195755291692375963310293709091662304773755859649779
 964 ($=2^{446} - 0x8335dc16 3bb124b6 5129c96f de933d8d 723a70aa dc873d6d$
 965 54a7bb0d)
 966 tr : 28312320572429821613362531907042076847709625476988141958474579766324
 967 ($=(p+1) - h \cdot n = 0x1 0cd77058 eec492d9 44a725bf 7a4cf635 c8e9c2ab$
 968 721cf5b5 529eec34)
 969 a : 1
 970 d : $39082/39081 =$
 971 6119758507445291761604232209655533175432196968710166263289689364150\
 972 87860042636474891785599283666020414768678979989378147065462815545017
 973 (=0xd78b4bdc 7f0daf19 f24f38c2 9373a2cc ad461572 42a50f37 809b1da3
 974 412a12e7 9ccc9c81 264cfe9a d0809970 58fb61c4 243cc32d baa156b9)
 975 G_x : 3453974930397295163740086041505374102666552600751832902164069702816\
 976 45695073672344430481787759340633221708391583424041788924124567700732
 977 (=0x79a70b2b 70400553 ae7c9df4 16c792c6 1128751a c9296924 0c25a07d
 978 728bdc93 e21f7787 ed697224 9de732f3 8496cd11 69871309 3e9c04fc)
 979 G_y : $3/2 =$
 980 3634193621478034452746619039440022671768206803436590301407450995903\
 981 06164083365386343198191849338272965044442230921818680526749009182718
 982 (=0x7fffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
 983 7fffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffffL)

985 The mapping from E448 to Curve448 is defined by mapping the point (0, 1) and the point (0, -1)
 986 of order two on E448 to, respectively, the point at infinity \emptyset and the point (0,0) of order two of

987 Curve448 and having each other point (x, y) on E448 correspond to the point $((y + 1)/(y - 1), \alpha(y$
988 $+ 1)/(y-1)x)$. The value of α is specified in 4.2.2.2. See Appendix B.1 for more details.

989
990 The curve Edwards448 (specified in Section 4.2.3.2) is 4-isogenous to the curve E448. See
991 Appendix B.4 for further information.

992

993 **4.3 Curves over Binary Fields**

994 This section specifies elliptic curves over binary fields where each curve takes the form of a
995 curve in short-Weierstrass form and is either a Koblitz curve (Section 4.3.1) or a pseudorandom
996 curve (Section 4.3.2). Due to their limited adoption, elliptic curves over binary fields (i.e., all the
997 curves specified in Section 4.3) are deprecated and may be removed from a subsequent revision
998 to these guidelines to facilitate interoperability and simplify elliptic curve standards and
999 implementations. New implementations should select an appropriate elliptic curve over a prime
1000 field from Section 4.2.

1001 Here, the domain parameters a and b for Koblitz curves are elements of the base field $\text{GF}(2)$, i.e.,
1002 $b=1$ and $a=0$ or $a=1$, whereas, for pseudorandom curves, $a=1$ and b is a nonzero element of
1003 $\text{GF}(2^m)$.

1004 For each field degree m , a pseudorandom curve is given, along with a Koblitz curve. The
1005 pseudorandom curve has the form

$$1006 \quad E: y^2 + xy = x^3 + x^2 + b,$$

1007 and the Koblitz curve has the form

$$1008 \quad E_a: y^2 + xy = x^3 + ax^2 + 1,$$

1009 where $a = 0$ or 1 .

1010 For each pseudorandom curve, the cofactor is $h = 2$. The cofactor of each Koblitz curve is $h = 2$
1011 if $a = 1$, and $h = 4$ if $a = 0$.

1012 The coefficients of the pseudorandom curves and the coordinates of the base points of both kinds
1013 of curves are given in terms of both the polynomial and normal basis representations discussed in
1014 Section 4.1.3.

1015 For each m , the following parameters are given:

1016 *Field Representation:*

- 1017 • The normal basis type T
- 1018 • The field polynomial (a trinomial or pentanomial)

1019 *Koblitz Curve:*

- 1020 • The coefficient a
- 1021 • The base point order n

- 1022 • The base point x coordinate G_x
 1023 • The base point y coordinate G_y

1024 *Pseudorandom curve:*

- 1025 • The base point order n

1026 *Pseudorandom curve (Polynomial Basis representation):*

- 1027 • The coefficient b
 1028 • The base point x coordinate G_x
 1029 • The base point y coordinate G_y

1030 *Pseudorandom curve (Normal Basis representation):*

- 1031 • The 160-bit input *Seed* to the SHA-1 based algorithm (i.e., the domain parameter seed)
 1032 • The coefficient b (i.e., the output of the SHA-1 based algorithm)
 1033 • The base point x coordinate G_x
 1034 • The base point y coordinate G_y

1035 Integers (such as T , m , and n) are given in decimal form; bit strings and field elements are given
 1036 in hexadecimal.

1037 **4.3.1 Koblitz Curves**

1038 **4.3.1.1 Curve K-163**

1039 The use of this curve is for legacy-use only. See FIPS 186-4 for the specification.

1040 **4.3.1.2 Curve K-233**

1041 The elliptic curve K-233 is a Weierstrass curve $B_{a,b}$ defined over the binary field $\text{GF}(2^m)$, with
 1042 $m=233$, and with parameters $a=0$ and $b=1$. This curve has order $h \cdot n$, where $h=4$ and where n is a
 1043 prime number. This curve has domain parameters $D=(m, f(z), h, n, \text{Type}, a, b, G, \{\text{Seed}, c\})$,
 1044 where the *Type* is “Weierstrass curve” and where the other parameters are defined as follows:
 1045

1046 $f(z): z^{233} + z^{74} + 1$

1047 $h: 4$

1048 $n: 345087317339528189371737793113851276057094098886225212\backslash$
 1049 6328087024741343

1050 $(=0x80\ 00000000\ 00000000\ 00000000\ 00069d5b\ b915bcd4\ 6efb1ad5\ f173abdf)$

1051 $tr: -137381546011108235394987299651366779$

1052 $(=(2^m+1) - h \cdot n = -0x1a756e\ e456f351\ bbec6b57\ c5ceaf7b)$

1053 $a: 0$

1054 $(=0x000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000)$

1055 $b: 1$

1056 $(=0x000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000001)$

1057 Polynomial basis:

1058 G_x : 0x172 32ba853a 7e731af1 29f22ff4 149563a4 19c26bf5 0a4c9d6e efad6126

1059 G_y : 0x1db 537dece8 19b7f70f 555a67c4 27a8cd9b f18aeb9b 56e0c110 56fae6a3

1060 Normal basis:

1061 G_x : 0x0fd e76d9dcd 26e643ac 26f1aa90 1aa12978 4b71fc07 22b2d056 14d650b3

1062 G_y : 0x064 3e317633 155c9e04 47ba8020 a3c43177 450ee036 d6335014 34cac978

1063 *Seed*: n/a (binary Koblitz curve)

1064

1065 4.3.1.3 Curve K-283

1066 The elliptic curve K-283 is a Weierstrass curve $B_{a,b}$ defined over the binary field $\text{GF}(2^m)$, with
 1067 $m=283$, and with parameters $a=0$ and $b=1$. This curve has order $h \cdot n$, where $h=4$ and where n is a
 1068 prime number. This curve has domain parameters $D=(m, f(z), h, n, \text{Type}, a, b, G, \{\text{Seed}, c\})$,
 1069 where the *Type* is “Weierstrass curve” and where the other parameters are defined as follows:

1070

1071 $f(z)$: $z^{283} + z^{12} + z^7 + z^5 + 1$

1072 h : 4

1073 n : 388533778445145814183892381364703781328481\

1074 1733793061324295874997529815829704422603873

1075 (=0x1fffffff ffffffff ffffffff ffffffff ffffe9ae 2ed07577

1076 265dff7f 94451e06 1e163c61)

1077 tr : 7777244870872830999287791970962823977569917

1078 ($= (2^m + 1) - h \cdot n = 0x5947 44be2a23 66880201 aeeb87e7 87a70e7d$)

1079 a : 0

1080 (=0x0000000 00000000 00000000 00000000 00000000 00000000

1081 00000000 00000000 00000000)

1082 b : 1

1083 (=0x0000000 00000000 00000000 00000000 00000000 00000000

1084 00000000 00000000 00000001)

1085 Polynomial basis:

1086 G_x : 0x503213f 78ca4488 3f1a3b81 62f188e5 53cd265f 23c1567a

1087 16876913 b0c2ac24 58492836

1088 G_y : 0x1ccda38 0f1c9e31 8d90f95d 07e5426f e87e45c0 e8184698

1089 e4596236 4e341161 77dd2259

1090 Normal basis:

1091 G_x : 0x3ab9593 f8db09fc 188f1d7c 4ac9fcc3 e57fcd3b db15024b

1092 212c7022 9de5fcd9 2eb0ea60

1093 G_y : 0x2118c47 55e7345c d8f603ef 93b98b10 6fe8854f feb9a3b3

1094 04634cc8 3a0e759f 0c2686b1

1095 *Seed*: n/a (binary Koblitz curve)

1096

1097 4.3.1.4 Curve K-409

1098 The elliptic curve K-409 is a Weierstrass curve $B_{a,b}$ defined over the binary field $\text{GF}(2^m)$, with
 1099 $m=409$, and with parameters $a=0$ and $b=1$. This curve has order $h \cdot n$, where $h=4$ and where n is a
 1100 prime number. This curve has domain parameters $D=(m, f(z), h, n, \text{Type}, a, b, G, \{\text{Seed}, c\})$,
 1101 where the *Type* is “Weierstrass curve” and where the other parameters are defined as follows:

```

1102
1103  $f(z): z^{409} + z^8 + 1$ 
1104  $h: 4$ 
1105  $n: 3305279843951242994759576540163855199142023414821406096423243\$ 
1106  $95022880711289249191050673258457777458014096366590617731358671$ 
1107  $(= 0x7ffffff ffffffff ffffffff ffffffff ffffffff ffffffff fffffffe5f$ 
1108  $83b2d4ea 20400ec4 557d5ed3 e3e7ca5b 4b5c83b8 e01e5fcf)$ 
1109  $tr: 10457288737315625927447685387048320737638796957687575791173829$ 
1110  $(=(2^m+1) - h \cdot n = 0x681 f134ac57 7effc4ee aa0a84b0 7060d692 d28df11c$ 
1111  $7f8680c5)$ 
1112  $a: 0$ 
1113  $(=0x0000000 00000000 00000000 00000000 00000000 00000000 00000000$ 
1114  $00000000 00000000 00000000 00000000 00000000 00000000)$ 
1115  $b: 1$ 
1116  $(=0x0000000 00000000 00000000 00000000 00000000 00000000 00000000$ 
1117  $00000000 00000000 00000000 00000000 00000000 00000001)$ 
1118 Polynomial basis:
1119  $G_x: 0x060f05f 658f49c1 ad3ab189 0f718421 0efd0987 e307c84c 27accfb8$ 
1120  $f9f67cc2 c460189e b5aaaa62 ee222eb1 b35540cf e9023746$ 
1121  $G_y: 0x1e36905 0b7c4e42 acba1dac bf04299c 3460782f 918ea427 e6325165$ 
1122  $e9ea10e3 da5f6c42 e9c55215 aa9ca27a 5863ec48 d8e0286b$ 
1123 Normal basis:
1124  $G_x: 0x1b559c7 cba2422e 3affe133 43e808b5 5e012d72 6ca0b7e6 a63aeafb$ 
1125  $c1e3a98e 10ca0fcf 98350c3b 7f89a975 4a8e1dc0 713cec4a$ 
1126  $G_y: 0x16d8c42 052f07e7 713e7490 eff318ba 1abd6fef 8a5433c8 94b24f5c$ 
1127  $817aeb79 852496fb ee803a47 bc8a2038 78ebf1c4 99afd7d6$ 
1128  $Seed: n/a$  (binary Koblitz curve)
1129

```

1130 4.3.1.5 Curve K-571

1131 The elliptic curve K-571 is a Weierstrass curve $B_{a,b}$ defined over the binary field $GF(2^m)$, with
 1132 $m=571$, and with parameters $a=0$ and $b=1$. This curve has order $h \cdot n$, where $h=4$ and where n is a
 1133 prime number. This curve has domain parameters $D=(m, f(z), h, n, Type, a, b, G, \{Seed, c\})$,
 1134 where the *Type* is “Weierstrass curve” and where the other parameters are defined as follows:

```

1135
1136  $f(z): z^{571} + z^{10} + z^5 + z^2 + 1$ 
1137  $h: 4$ 
1138  $n: 193226876150862917234767594546599367214946366485321749932\$ 
1139  $861762572575957114478021226813397852270671183470671280082\$ 
1140  $5351461273674974066617311929682421617092503555733685276673$ 
1141  $(=0x 2000000 00000000 00000000 00000000 00000000 00000000 00000000$ 
1142  $00000000 00000000 131850e1 f19a63e4 b391a8db 917f4138$ 
1143  $b630d84b e5d63938 1e91deb4 5cfe778f 637c1001)$ 
1144
1145  $tr: -148380926981691413899619140297051490364542\$ 
1146  $574180493936232912339534208516828973111459843$ 
1147  $(=(2^m+1) - h \cdot n = -0x4c614387 c6698f92 ce46a36e 45fd04e2 d8c3612f$ 

```

1148 9758e4e0 7a477ad1 73f9de3d 8df04003)
 1149 $a:$ 0
 1150 (=0x0000000 00000000 00000000 00000000 00000000 00000000 00000000
 1151 00000000 00000000 00000000 00000000 00000000 00000000)
 1152 00000000 00000000 00000000 00000000 00000000)
 1153 $b:$ 1
 1154 (=0x0000000 00000000 00000000 00000000 00000000 00000000 00000000
 1155 00000000 00000000 00000000 00000000 00000000 00000000)
 1156 00000000 00000000 00000000 00000000 00000001)

1157 Polynomial basis:

1158 $G_x:$ 0x26eb7a8 59923fbc 82189631 f8103fe4 ac9ca297 0012d5d4 60248048
 1159 01841ca4 43709584 93b205e6 47da304d b4ceb08c bbd1ba39
 1160 494776fb 988b4717 4dca88c7 e2945283 a01c8972
 1161 $G_y:$ 0x349dc80 7f4fbf37 4f4aeade 3bca9531 4dd58cec 9f307a54 ffc61efc
 1162 006d8a2c 9d4979c0 ac44aea7 4fbеbbbb9 f772aedc b620b01a
 1163 7ba7af1b 320430c8 591984f6 01cd4c14 3ef1c7a3

1164 Normal basis:

1165 $G_x:$ 0x04bb2db a418d0db 107adae0 03427e5d 7cc139ac b465e593 4f0bea2a
 1166 b2f3622b c29b3d5b 9aa7a1fd fd5d8be6 6057c100 8e71e484
 1167 bcd98f22 bf847642 37673674 29ef2ec5 bc3ebcf7
 1168 $G_y:$ 0x44cbb57 de20788d 2c952d7b 56cf39bd 3e89b189 84bd124e 751cefff4
 1169 369dd8da c6a59e6e 745df44d 8220ce22 aa2c852c fcbbef49
 1170 ebaa98bd 2483e331 80e04286 feaa2530 50caff60

1171 *Seed:* n/a (binary Koblitz curve)

1172

1173 4.3.2 Pseudorandom Curves

1174 4.3.2.1 Curve B-163

1175 The use of this curve is for legacy-use only. See FIPS 186-4 for the specification.

1176 4.3.2.2 Curve B-233

1177 The elliptic curve B-233 is a Weierstrass curve $B_{a,b}$ defined over the binary field $GF(2^m)$, with
 1178 $m=233$, and with parameter $a=1$. This curve has order $h \cdot n$, where $h=2$ and where n is a prime
 1179 number. This curve has domain parameters $D=(m, f(z), h, n, Type, a, b, G, \{Seed, c\})$, where the
 1180 *Type* is “Weierstrass curve” and where the other parameters are defined as follows:

1181
 1182 $f(z):$ $z^{233} + z^{74} + 1$
 1183 $h:$ 2
 1184 $n:$ 690174634679056378743475586227702555583981273734501355\
 1185 5379383634485463
 1186 (=0x100 00000000 00000000 00000000 0013e974 e72f8a69 22031d26 03cfe0d7)
 1187 $tr:$ -206777407530349254000433718821372333
 1188 (= $(2^m+1) - h \cdot n = -0x27d2e9 ce5f14d2 44063a4c 079fc1ad$)
 1189 $a:$ 1
 1190 (=0x000 00000000 00000000 00000000 00000000 00000000 00000000 00000001)

1191 Polynomial basis:

1192 b : 0x066 647ede6c 332c7f8c 0923bb58 213b333b 20e9ce42 81fe115f 7d8f90ad
 1193 G_x : 0x0fa c9dfcbac 8313bb21 39f1bb75 5fef65bc 391f8b36 f8f8eb73 71fd558b
 1194 G_y : 0x100 6a08a419 03350678 e58528be bf8a0bef f867a7ca 36716f7e 01f81052
 1195 Normal basis:
 1196 b : 0x1a0 03e0962d 4f9a8e40 7c904a95 38163adb 82521260 0c7752ad 52233279
 1197 G_x : 0x18b 863524b3 cdfefb94 f2784e0b 116faac5 4404bc91 62a363ba b84a14c5
 1198 G_y : 0x049 25df77bd 8b8ff1a5 ff519417 822bfedf 2bbd7526 44292c98 c7af6e02
 1199 $Seed$: 0x74d59ff0 7f6b413d 0ea14b34 4b20a2db 049b50c3

1200

1201 **4.3.2.3 Curve B-283**

1202 The elliptic curve B-283 is a Weierstrass curve $B_{a,b}$ defined over the binary field $GF(2^m)$, with
 1203 $m=283$, and with parameter $a=1$. This curve has order $h \cdot n$, where $h=2$ and where n is a prime
 1204 number. This curve has domain parameters $D=(m, f(z), h, n, Type, a, b, G, \{Seed, c\})$, where the
 1205 $Type$ is “Weierstrass curve” and where the other parameters are defined as follows:

1206

1207 $f(z)$: $z^{283} + z^{12} + z^7 + z^5 + 1$
 1208 h : 2
 1209 n : 7770675568902916283677847627294075626569625924376904889\
 1210 109196526770044277787378692871
 1211 (=0x3fffffff ffffffff ffffffff ffffffff ffffef90 399660fc
 1212 938a9016 5b042a7c efadb307)
 1213 tr : 2863663306391796106224371145726066910599667
 1214 ($= (2^m + 1) - h \cdot n =$ 0x 20df8cd33e06d8eadfd349f7ab0620a499f3)
 1215 a : 1
 1216 (=0x0000000 00000000 00000000 00000000 00000000 00000000
 1217 00000000 00000000 00000001)

1218 Polynomial basis:

1219 b : 0x27b680a c8b8596d a5a4af8a 19a0303f ca97fd76 45309fa2
 1220 a581485a f6263e31 3b79a2f5
 1221 G_x : 0x5f93925 8db7dd90 e1934f8c 70b0dfec 2eed25b8 557eac9c
 1222 80e2e198 f8cdbecd 0x86b12053
 1223 G_y : 0x3676854 fe24141c b98fe6d4 b20d02b4 516ff702 350eddb0
 1224 826779c8 13f0df45 be8112f4

1225 Normal basis:

1226 b : 0x157261b 894739fb 5a13503f 55f0b3f1 0c560116 66331022
 1227 01138cc1 80c0206b dafbc951
 1228 G_x : 0x749468e 464ee468 634b21f7 f61cb700 701817e6 bc36a236
 1229 4cb8906e 940948ea a463c35d
 1230 G_y : 0x62968bd 3b489ac5 c9b859da 68475c31 5bafcdc4 ccd0dc90
 1231 5b70f624 46f49c05 2f49c08c
 1232 $Seed$: 0x77e2b073 70eb0f83 2a6dd5b6 2dfc88cd 06bb84be

1233

1234 **4.3.2.4 Curve B-409**

1235 The elliptic curve B-409 is a Weierstrass curve $B_{a,b}$ defined over the binary field $GF(2^m)$, with
 1236 $m=409$, and with parameter $a=1$. This curve has order $h \cdot n$, where $h=2$ and where n is a prime

1237 number. This curve has domain parameters $D=(m, f(z), h, n, Type, a, b, G, \{Seed, c\})$, where the
1238 *Type* is “Weierstrass curve” and where the other parameters are defined as follows:

1239
1240 $f(z): z^{409} + z^{87} + 1$
1241 $h: 2$
1242 $n: 6610559687902485989519153080327710398284046829642812192846487\backslash$
1243 $98304157774827374805208143723762179110965979867288366567526771$
1244 $(=0x1000000 00000000 00000000 00000000 00000000 00000000 000001e2$
1245 $aad6a612 f33307be 5fa47c3c 9e052f83 8164cd37 d9a21173)$
1246 $tr: -6059503967182126918765909026644927652236777310526686418445029$
1247 $(=(2^m+1) - h \cdot n = -0x3c5 55ad4c25 e6660f7c bf48f879 3c0a5f07$
1248 $02c99a6f b34422e5)$
1249 $a: 1$
1250 $(=0x0000000 00000000 00000000 00000000 00000000 00000000 00000000$
1251 $00000000 00000000 00000000 00000000 00000000 00000000 00000001)$

1252 Polynomial basis:

1253 $b: 0x021a5c2 c8ee9feb 5c4b9a75 3b7b476b 7fd6422e f1f3dd67 4761fa99$
1254 $d6ac27c8 a9a197b2 72822f6c d57a55aa 4f50ae31 7b13545f$
1255 $G_x: 0x15d4860 d088ddb3 496b0c60 64756260 441cde4a f1771d4d b01ffe5b$
1256 $34e59703 dc255a86 8a118051 5603aeab 60794e54 bb7996a7$
1257 $G_y: 0x061b1cf ab6be5f3 2bbfa783 24ed106a 7636b9c5 a7bd198d 0158aa4f$
1258 $5488d08f 38514f1f df4b4f40 d2181b36 81c364ba 0273c706$

1259 Normal basis:

1260 $b: 0x124d065 1c3d3772 f7f5a1fe 6e715559 e2129bdf a04d52f7 b6ac7c53$
1261 $2cf0ed06 f610072d 88ad2fdc c50c6fde 72843670 f8b3742a$
1262 $G_x: 0x0ceacbc 9f475767 d8e69f3b 5dfab398 13685262 bcacf22b 84c7b6dd$
1263 $981899e7 318c96f0 761f77c6 02c016ce d7c548de 830d708f$
1264 $G_y: 0x199d64b a8f089c6 db0e0b61 e80bb959 34afd0ca f2e8be76 d1c5e9af$
1265 $fc7476df 49142691 ad303902 88aa09bc c59c1573 aa3c009a$
1266 $Seed: 0x4099b5a4 57f9d69f 79213d09 4c4bcd4d 4262210b$

1267

1268 4.3.2.5 Curve B-571

1269 The elliptic curve B-571 is a Weierstrass curve $B_{a,b}$ defined over the binary field $GF(2^m)$, with
1270 $m=571$, and with parameter $a=1$. This curve has order $h \cdot n$, where $h=2$ and where n is a prime
1271 number. This curve has domain parameters $D=(m, f(z), h, n, Type, a, b, G, \{Seed, c\})$, where the
1272 *Type* is “Weierstrass curve” and where the other parameters are defined as follows:

1273
1274 $f(z): z^{571} + z^{10} + z^5 + z^2 + 1$
1275 $h: 2$
1276 $n: 386453752301725834469535189093198734429892732970643499865\backslash$
1277 $723525145151914228956042453614399938941577308313388112192\backslash$
1278 $6944486246872462816813070234528288303332411393191105285703$
1279 $(=0x3fffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff$
1280 $fffffff ffffffff e661ce18 ff559873 08059b18 6823851e$
1281 $c7dd9ca1 161de93d 5174d66e 8382e9bb 2fe84e47)$
1282
1283 $tr: 9953438501360975865946981915046538223641239\backslash$

```

1284      6452349171016760770327496674607579419075443
1285      ( $= (2^m + 1) - h \cdot n =$       0x333c63ce 0154cf19 eff4c9cf 2fb8f5c2 7044c6bd
1286                                         d3c42d85 5d165322 f8fa2c89 a02f6373)
1287  a:      1
1288      (=0x0000000 00000000 00000000 00000000 00000000 00000000 00000000
1289                                         00000000 00000000 00000000 00000000 00000000
1290                                         00000000 00000000 00000000 00000000 00000001)
1291  Polynomial basis:
1292  b:      0x2f40e7e 2221f295 de297117 b7f3d62f 5c6a97ff cb8ceff1 cd6ba8ce
1293          4a9a18ad 84ffabbd 8efa5933 2be7ad67 56a66e29 4afd185a
1294          78ff12aa 520e4de7 39baca0c 7ffeff7f 2955727a
1295  Gx:    0x303001d 34b85629 6c16c0d4 0d3cd775 0a93d1d2 955fa80a a5f40fc8
1296          db7b2abd bde53950 f4c0d293 cdd711a3 5b67fb14 99ae6003
1297          8614f139 4abfa3b4 c850d927 e1e7769c 8eec2d19
1298  Gy:    0x37bf273 42da639b 6dccfffe b73d69d7 8c6c27a6 009cbbca 1980f853
1299          3921e8a6 84423e43 bab08a57 6291af8f 461bb2a8 b3531d2f
1300          0485c19b 16e2f151 6e23dd3c 1a4827af 1b8ac15b
1301  Normal basis:
1302  b:      0x3762d0d 47116006 179da356 88eeaccf 591a5cde a7500011 8d9608c5
1303          9132d434 26101a1d fb377411 5f586623 f75f0000 1ce61198
1304          3c1275fa 31f5bc9f 4be1a0f4 67f01ca8 85c74777
1305  Gx:    0x0735e03 5def5925 cc33173e b2a8ce77 67522b46 6d278b65 0a291612
1306          7dfea9d2 d361089f 0a7a0247 a184e1c7 0d417866 e0fe0feb
1307          0ff8f2f3 f9176418 f97d117e 624e2015 df1662a8
1308  Gy:    0x04a3642 0572616c df7e606f ccadaecf c3b76dab 0eb1248d d03fbdfc
1309          9cd3242c 4726be57 9855e812 de7ec5c5 00b4576a 24628048
1310          b6a72d88 0062eed0 dd34b109 6d3acbb6 b01a4a97
1311  Seed:   0x2aa058f7 3a0e33ab 486b0f61 0410c53a 7f132310

```


1312 **References**

- [FIPS 140-3] National Institute of Standards and Technology (2019) Security Requirements for Cryptographic Modules. (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 140-3. <https://doi.org/10.6028/NIST.FIPS.140-3>
- [FIPS 186-4] National Institute of Standards and Technology (2013) Digital Signature Standard (DSS). (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 186-4. <https://doi.org/10.6028/NIST.FIPS.186-4>
- [FIPS 186-5] National Institute of Standards and Technology (2019) Digital Signature Standard (DSS). (U.S. Department of Commerce, Washington, DC), Draft Federal Information Processing Standards Publication (FIPS) 186-5. <https://doi.org/10.6028/NIST.FIPS.186-5-draft>
- [IEEE 1363] IEEE (2000) *IEEE 1363-2000 – IEEE Standard Specifications for Public Key Cryptography* (IEEE Standards Association, Piscataway, NJ). Available at <https://standards.ieee.org/standard/1363-2000.html>
- [RFC 5639] Lochter M, Merkle J (2010) Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation. (Internet Engineering Task Force (IETF)), Request for Comments (RFC) 5639. <https://tools.ietf.org/html/rfc5639>
- [RFC 7748] Langley A, Hamburg M, Turner S (2016) Elliptic Curves for Security. (Internet Research Task Force (IRTF)), Request for Comments (RFC) 7748. <https://doi.org/10.17487/RFC7748>
- [RFC 8032] Josefsson S, Liusvaara I (2017) Edwards-Curve Digital Signature Algorithm (EdDSA). (Internet Research Task Force (IRTF)), Request for Comments (RFC) 8032. <https://doi.org/10.17487/RFC8032>
- [SP 800-56A] Barker EB, Chen L, Roginsky AL, Vassilev A, Davis R (2018) Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-56A, Rev. 3. <https://doi.org/10.6028/NIST.SP.800-56Ar3>
- [SP 800-57] Barker EB (2016) Recommendation for Key Management, Part 1: General. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-57 Part 1, Rev. 4. <https://doi.org/10.6028/NIST.SP.800-57pt1r4>

[SP 800-131A] Barker EB, Roginsky AL (2019) Transitioning the Use of Cryptographic Algorithms and Key Lengths. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-131A, Rev. 2. <https://doi.org/10.6028/NIST.SP.800-131Ar2>

1313

1314 **Appendix A – Details of Elliptic Curve Group Operations**1315 **A.1 Non-Binary Curves**1316 **A.1.1 Group Law for Weierstrass Curves**

1317 For each point P on the Weierstrass curve $W_{a,b}$, the point at infinity \emptyset serves as the identity
1318 element, i.e., $P + \emptyset = \emptyset + P = P$.

1319 For each point $P=(x, y)$ on the Weierstrass curve $W_{a,b}$, the point $-P$ is the point $(x, -y)$, and one
1320 has $P + (-P) = \emptyset$.

1321 Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ be points on the Weierstrass curve $W_{a,b}$, where $P_1 \neq \pm P_2$, and let
1322 $Q = P_1 + P_2$. Then $Q = (x, y)$, where

$$1323 \quad x + x_1 + x_2 = \lambda^2 \text{ and } y + y_1 = \lambda(x_1 - x), \text{ where } \lambda = (y_2 - y_1)/(x_2 - x_1).$$

1324 Let $P = (x_1, y_1)$ be a point on the Weierstrass curve $W_{a,b}$, where $P \neq -P$, and let $Q = 2P$. Then $Q =$
1325 (x, y) , where

$$1326 \quad x + 2x_1 = \lambda^2 \text{ and } y + y_1 = \lambda(x_1 - x), \text{ where } \lambda = (3x_1^2 + a)/2y_1.$$

1327 **A.1.2 Group Law for Montgomery Curves**

1328 For each point P on the Montgomery curve $M_{A,B}$, the point at infinity \emptyset serves as the identity
1329 element, i.e., $P + \emptyset = \emptyset + P = P$.

1330 For each point $P=(u, v)$ on the Montgomery curve $M_{A,B}$, the point $-P$ is the point $(u, -v)$, and
1331 one has $P + (-P) = \emptyset$.

1332 Let $P_1 = (u_1, v_1)$ and $P_2 = (u_2, v_2)$ be points on the Montgomery curve $M_{A,B}$, where $P_1 \neq \pm P_2$, and
1333 let $Q = P_1 + P_2$. Then $Q = (u, v)$, where

$$1334 \quad u + u_1 + u_2 = B\lambda^2 - A \text{ and } v + v_1 = \lambda(u_1 - u), \text{ where } \lambda = (v_2 - v_1)/(u_2 - u_1).$$

1335 Let $P=(u, v)$ be a point on the Montgomery curve $M_{A,B}$, where $P \neq -P$, and let $Q = 2P$. Then Q
1336 $= (u, v)$, where

$$1337 \quad u + 2u_1 = B\lambda^2 - A \text{ and } v + v_1 = \lambda(u_1 - u), \text{ where } \lambda = (3u_1^2 + 2Au_1 + 1)/2Bv_1.$$

1338 **A.1.3 Group Law for Twisted Edwards Curves**

1339 Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ be points on the twisted Edwards curve $E_{a,d}$ and let $Q = P_1 + P_2$.
1340 Then $Q = (x, y)$, where

$$1341 \quad (x, y) = \left(\frac{x_1y_2 + x_2y_1}{1 + dx_1x_2y_1y_2}, \frac{y_1y_2 - ax_1x_2}{1 - dx_1x_2y_1y_2} \right).$$

1342 For the twisted Edwards curves specified in this recommendation, the domain parameter a is
 1343 always a square in $\text{GF}(q)$, whereas d is not. In this case, the addition formula above is defined for
 1344 each pair of points. In particular, for each point $P = (x_1, y_1)$ on the twisted Edwards curve $E_{a,d}$,
 1345 point doubling yields the point $Q = 2P$, where $Q = (x, y)$ and

$$1346 \quad (x, y) = \left(\frac{2x_1y_1}{1 + dx_1^2y_1^2}, \frac{y_1^2 - ax_1^2}{1 - dx_1^2y_1^2} \right).$$

1347 Note that $(0, 1)$ is the identity element, since for each point $P = (x, y)$ on the twisted Edwards
 1348 curve $E_{a,d}$, one has $P + (0, 1) = (x, y) + (0, 1) = (x, y) = P$.

1349 For each point $P = (x, y)$ on the twisted Edwards curve $E_{A,B}$, the inverse point $-P$ is the point $(-x,$
 1350 $y)$ and one has $P + (-P) = \emptyset$. The point $(0, -1)$ has order 2.

1351 **A.2 Binary Curves**

1352 **A.2.1 Group Law for Weierstrass Curves**

1353 For each point P on the Weierstrass curve $B_{a,b}$, the point at infinity \emptyset serves as the identity
 1354 element, i.e., $P + \emptyset = \emptyset + P = P$.

1355 For each point $P = (x, y)$ on the Weierstrass curve $B_{a,b}$, the point $-P$ is the point $(x, x + y)$ and one
 1356 has $P + (-P) = \emptyset$.

1357 Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ be points on the Weierstrass curve $B_{a,b}$, where $P_1 \neq \pm P_2$, and let
 1358 $Q = P_1 + P_2$. Then $Q = (x, y)$, where

$$1359 \quad x + x_1 + x_2 = \lambda^2 + \lambda + a \text{ and } (x + y) + y_1 = \lambda(x_1 + x), \text{ where } \lambda = (y_2 + y_1)/(x_2 + x_1).$$

1360 Let $P = (x_1, y_1)$ be a point on the Weierstrass curve $B_{a,b}$, where $P \neq -P$, and let $Q = 2P$. Then $Q =$
 1361 (x, y) , where

$$1362 \quad x = \lambda^2 + \lambda + a = x_1^2 + b/x_1^2 \text{ and } (x + y) + y_1 = \lambda(x_1 + x), \text{ where } \lambda = x_1 + y_1/x_1.$$

1363

1364 **Appendix B – Relationship Between Curve Models**

1365 The non-binary curves specified in this recommendation are expressed in different curve models
1366 defined over the same field $\text{GF}(q)$ —namely as curves in short-Weierstrass form, as Montgomery
1367 curves, or as twisted Edwards curves. These curve models are related, as follows.

1368 **B.1 Mapping Between Twisted Edwards Curves and Montgomery Curves**

1369 One can map points on the Montgomery curve $M_{A,B}$ to points on the twisted Edwards curve $E_{a,d}$,
1370 where $a=(A+2)/B$ and $d=(A-2)/B$ and, conversely, map points on the twisted Edwards curve $E_{a,d}$
1371 to points on the Montgomery curve $M_{A,B}$, where $A=2(a+d)/(a-d)$ and where $B=4/(a-d)$. For the
1372 curves in this specification, this defines a one-to-one correspondence, which is an isomorphism
1373 between $M_{A,B}$ and $E_{a,d}$, thereby showing that the discrete logarithm problem in either curve
1374 model is equally hard.

1375 For the Montgomery curves and twisted Edwards curves in this specification, the mapping from
1376 $M_{A,B}$ to $E_{a,d}$ is defined by mapping the point at infinity \emptyset and the point $(0, 0)$ of order two on
1377 $M_{A,B}$ to, respectively, the point $(0, 1)$ and the point $(0, -1)$ of order two on $E_{a,d}$, while mapping
1378 every other point (u, v) on $M_{A,B}$ to the point $(x, y)=(u/v, (u-1)/(u+1))$ on $E_{a,d}$. The inverse
1379 mapping from $E_{a,d}$ to $M_{A,B}$ is defined by mapping the point $(0, 1)$ and the point $(0, -1)$ of order
1380 two on $E_{a,d}$ to, respectively, the point at infinity \emptyset and the point $(0, 0)$ of order two on $M_{A,B}$,
1381 while every other point (x, y) on $E_{a,d}$ is mapped to the point $(u, v)=((1+y)/(1-y), (1+y)/(1-y)x)$ on
1382 $M_{A,B}$.

1383 Implementations may take advantage of this mapping to carry out elliptic curve group operations
1384 originally defined for a twisted Edwards curve on the corresponding Montgomery curve, or vice-
1385 versa, and translating the result back to the original curve to potentially allow code reuse.

1386 **B.2 Mapping Between Montgomery Curves and Weierstrass Curves**

1387 One can map points on the Montgomery curve $M_{A,B}$ to points on the Weierstrass curve $W_{a,b}$,
1388 where $a=(3-A^2)/3B^2$ and $b=(2A^3-9A)/27B^3$. For the curves in this specification, this defines a
1389 one-to-one correspondence, which is an isomorphism between $M_{A,B}$ and $W_{a,b}$, thereby showing
1390 that the discrete logarithm problem in either curve model is equally hard.

1391 For the Montgomery curves in this specification, the mapping from $M_{A,B}$ to $W_{a,b}$ is defined by
1392 mapping the point at infinity \emptyset on $M_{A,B}$ to the point at infinity \emptyset on $W_{a,b}$, while mapping every
1393 other point (u, v) on $M_{A,B}$ to the point $(x, y)=(u/B+A/3B, v/B)$ on $W_{a,b}$.

1394 Note that not all Weierstrass curves can be mapped to Montgomery curves since the latter have a
1395 point of order two and the former may not. In particular, if a Weierstrass curve has prime
1396 order—as in the case with the curves P-224, P-256, P-385, and P-521 specified in this
1397 recommendation—this mapping is not defined.

1398 This mapping can be used to implement elliptic curve group operations originally defined for a
1399 twisted Edwards curve or for a Montgomery curve using group operations on the corresponding

1400 elliptic curve in short-Weierstrass form and translating the result back to the original curve to
1401 potentially allow code reuse.

1402 Note that implementations for elliptic curves with short-Weierstrass form that hard-code the
1403 domain parameter a to $a=-3$ cannot always be used this way since the curve $W_{a,b}$ may not
1404 always be expressed in terms of a Weierstrass curve with $a=-3$ via a coordinate transformation.
1405 This is, unfortunately, the case with the Montgomery curves and twisted Edwards curves
1406 specified in this recommendation.

1407 **B.3 Mapping Between Twisted Edwards Curves and Weierstrass Curves**

1408 A straightforward method to map points on a twisted Edwards curve to points on a Weierstrass
1409 curve is to convert the curve to Montgomery format first. Use the mapping described in
1410 Appendix B.1 to map points on a twisted Edwards curve to points on a Montgomery curve. Then
1411 use the mapping described in Appendix B.2 to convert points on the Montgomery curve to points
1412 on a Weierstrass curve.

1413 **B.4 4-Isogenous Mapping**

1414 The 4-isogeny map between the Montgomery curve Curve448 and the Edwards curve
1415 Edwards448 is given in [[RFC 7748](#)] to be:

1416

$$1417 \quad (u, v) = \left(\frac{y^2}{x^2}, \frac{(2 - x^2 - y^2)y}{x^3} \right)$$

$$1418 \quad (x, y) = \left(\frac{4v(u^2 - 1)}{u^4 - 2u^2 + 4v^2 + 1}, \frac{-(u^5 - 2u^3 - 4uv^2 + u)}{(u^5 - 2u^2v^2 - 2u^3v^2 + u)} \right)$$

1419

1420 The curve Edwards448 (Section 4.2.3.2) is 4-isogenous to the curve E448 (Section 4.2.3.3),
1421 where the base point of Edwards448 corresponds to the base point of E448 and where the
1422 identity element $(0, 1)$ and the point $(0, -1)$ of order two of Edwards448 correspond to the
1423 identity element $(0, 1)$ on E448. Every other point (x, y) on Edwards448 corresponds to the point
1424 on E448, where α is the element of $\text{GF}(p)$ defined in Section 4.2.2.2:

1425

$$1426 \quad (x', y') = \left(\frac{\alpha xy}{1 - d x^2 y^2}, \frac{1 + d x^2 y^2}{y^2 - x^2} \right)$$

1427

1428

1429

1430 **Appendix C – Generation Details for Recommended Elliptic Curves**1431 **C.1 General Cryptographic Criteria**

1432 All curves recommended in this specification satisfy the following general cryptographic criteria:

- 1433 1. *Underlying finite field.* The underlying finite field $\text{GF}(q)$ **shall** be either a prime number or
1434 $q=2^m$ where m is a prime number.
- 1435 2. *Curve order.* Each curve E defined over the finite field $\text{GF}(q)$ **shall** have order $|E|=h \cdot n$, where
1436 n is a large prime number, where h is co-prime with n , and where h is small (h is called the
1437 co-factor of E). Each curve **shall** have co-factor $h \leq 2^{10}$.
- 1438 3. *Base point.* Each curve E **shall** have a fixed base point G of prime order n .
- 1439 4. *Avoiding anomalous curve attack.* Each curve E defined over the finite field $\text{GF}(q)$ **shall**
1440 have order $|E| \neq q$ so as to avoid attacks using additive transfers.
- 1441 5. *Large embedding degree.* The elliptic curve discrete logarithm problem in E can be
1442 converted to an ordinary discrete logarithm problem defined over the finite field $\text{GF}(q^t)$
1443 where t is the smallest positive integer so that $q^t \equiv 1 \pmod{n}$, called the embedding degree.
1444 Each curve **shall** have embedding degree $t \geq 2^{10}$.
- 1445 6. *Endomorphism field.* For each curve E over $\text{GF}(q)$ with trace tr , the (negative) number
1446 $Disc=tr^2-4q$ is closely related to the discriminant of the endomorphism field of E . As of the
1447 publication of this document, there is no technical rationale for imposing a large lower bound
1448 on the square-free part of $|Disc|$, although—except for curves used in pairing-based
1449 cryptography—this value is often large. This recommendation does not impose restrictions
1450 on the value of the square-free part of $|Disc|$.

1451

1452 **C.1.1 Implementation Security Criteria**1453 Each field **shall** have a fixed representation.1454 **C.2 Curve Generation Details**1455 **C.2.1 Weierstrass Curves over Prime Fields**1456 **C.2.1.1 Curves P-224, P-256, P-384, P-521**

1457 Each of the curves P-224 (Section 4.2.1.2), P-256 (Section 4.2.1.3), P-384 (Section 4.2.1.4), and
1458 P-521 (Section 4.2.1.5) is a curve $W_{a,b}$ in short-Weierstrass form with prime order (and, thus, co-
1459 factor $h=1$). Each curve is defined over a prime field $\text{GF}(p)$ where the prime number is of a
1460 special form to allow efficient modular reduction (see Appendix G.1).

1461 The NIST prime curves were generated using the procedure in C.3.1 with $h_{digest} = 160$ and
1462 SHA-1 hash function. The curve parameters a and b are:

- 1463 1. The parameter a was set to $a \equiv -3 \pmod{p}$ (this allows optimizations of the group law if
1464 implemented via projective coordinates in Weierstrass form);

- 1465 2. The parameter b was derived in a hard-to-invert way using the procedure in Appendix
 1466 C.3.1 from a pseudorandom *Seed* value so that the following conditions are satisfied
 1467 simultaneously:
- 1468 a. $4a^3 + 27b^2 \neq 0$ in $\text{GF}(p)$;
 - 1469 b. The curve has prime order n (this implies that $h = 1$); and
 - 1470 c. The curve satisfies the cryptographic criteria in Appendix C.1.
- 1471 3. Select a base point $G = (G_x, G_y)$ of order n .

1472 C.2.1.2 Curves W-25519, W-448

1473 The curves W-25519 (Section 4.2.1.6) and W-448 (Section 4.2.1.7) were obtained via an
 1474 isomorphic mapping (see Appendix B.1).

1475 C.2.2 Montgomery Curves

1476 C.2.2.1 Curve25519

1477 Curve25519 was specified in IETF 7748 by the Crypto Forum Research Group (CFRG). This
 1478 curve is a Montgomery curve $M_{A,B}$ defined over the field $\text{GF}(p)$, where $p=2^{255}-19$ and where the
 1479 curve has co-factor $h=8$ and the quadratic twist E_1 has co-factor $h_1=4$. The prime number is of a
 1480 special form to allow efficient modular reduction and finite field operations that try and
 1481 minimize carry effects of operands. The curve parameters A and B are:

- 1482 1. The parameter B was set to $B = 1$.
- 1483 2. The parameter A was selected as the minimum value of $|A|$ so that the following
 1484 conditions are satisfied simultaneously:
 - 1485 a. The curve is cyclic (this implies that A^2-4 is not a square in $\text{GF}(p)$);
 - 1486 b. The curve has co-factor $h=8$ (this implies that $A+2$ is a square in $\text{GF}(p)$);
 - 1487 c. The quadratic twist has co-factor $h'=4$;
 - 1488 d. A has the form $A \equiv 2 \pmod{4}$ (this allows optimized implementations of
 1489 implementations of the group law using the Montgomery ladder); and
 - 1490 e. The curve and the quadratic twist both satisfy the cryptographic criteria in
 1491 Appendix C.1.
- 1492 3. Select the base point $G = (G_x, G_y)$ of order n , where $|G_x|$ is minimal and where G_y is odd.

1493 C.2.2.2 Curve448

1494 This curve is a Montgomery curve $M_{A,B}$ defined over the field $\text{GF}(p)$, where $p=2^{448}-2^{224}-1$ and
 1495 where the curve has co-factor $h=4$ and the quadratic twist E_1 has co-factor $h_1=4$. The prime
 1496 number is of a special form to allow efficient modular reduction and finite field operations that
 1497 try to minimize the carry effects of operands. The curve parameters A and B are:

- 1498 1. The parameter B was set to $B = 1$.
- 1499 2. The parameter A was selected as the minimum value of $|A|$ so that the following
 1500 conditions are satisfied simultaneously:
 - 1501 a. The curve is cyclic (this implies that A^2-4 is not a square in $\text{GF}(p)$);
 - 1502 b. The curve has co-factor $h = 4$ (this implies that $A+2$ is not a square in $\text{GF}(p)$);
 - 1503 c. The quadratic twist has co-factor $h' = 4$;

- 1504 d. A has the form $A \equiv 2 \pmod{4}$ (this allows optimized implementations of
1505 implementations of the group law using the Montgomery ladder); and
1506 e. The curve and the quadratic twist both satisfy the cryptographic criteria in
1507 Appendix C.1.
1508 3. Select the base point $G = (G_x, G_y)$ of order n , where $|G_x|$ is minimal and where G_y is even.

1509 C.2.3 Twisted Edwards Curves

1510 The twisted Edwards curve Edwards25519 (Section 4.2.3.1) was obtained from the Montgomery
1511 curve Curve25519 (Section 4.2.2.1) via an isomorphic mapping.

1512 The Edwards curve E448 (Section 4.2.3.3) was obtained from the Montgomery curve Curve448
1513 (Section 4.2.2.2) via an isomorphic mapping.

1514 The Edwards curve Edwards448 (Section 4.2.3.2) was obtained from the curve E448 (Section
1515 4.2.3.3) via a 4-isogenous mapping (see Appendix B.4).

1516 C.2.4 Weierstrass Curves over Binary Fields

1517 C.2.4.1 Koblitz Curves K-233, K-283, K-409, K-571

1518 Each of the curves K-233 (Section 4.3.1.2), K-283 (Section 4.3.1.3), K-409 (Section 4.3.1.4),
1519 and K-571 (Section 4.3.1.5) is a curve $B_{a,b}$ in short-Weierstrass form with co-factor $h=2$ or $h=4$.
1520 Each curve is defined over a binary field $\text{GF}(2^m)$, where m is a prime number. For Koblitz
1521 curves, the curve parameters a and b are elements of $\text{GF}(2)$, with $b = 1$. Hence, for each
1522 parameter m , there are only two Koblitz curves, viz. with $a = 0$ and with $a = 1$. Koblitz curves
1523 with $a = 0$ have order $0 \pmod{4}$, while those with $a = 1$ have order $2 \pmod{4}$.

1524 The curve parameters a and m are:

- 1525 1. The parameter a was set to $a = 0$.
1526 2. The set of integers m in the interval $[160,600]$ was determined, so that the following
1527 conditions are satisfied simultaneously:
1528 a. m is a prime number;
1529 b. The curve has co-factor $h = 4$ or the quadratic twist of this curve has co-factor $h =$
1530 2 (the latter implies that the Koblitz curve defined over the binary field $\text{GF}(2^m)$
1531 with $a = 1$ has co-factor $h = 2$); and
1532 c. The thus determined curve satisfies the cryptographic criteria in Appendix C.1.
1533 3. Select a pair (a, m) from the set determined above.
1534 4. Select an irreducible polynomial $f(z)$ of degree m , where $f(z)$ is selected of a special form
1535 so as to allow efficient modular reduction ($f(z)$ is a trinomial or pentanomial).
1536 5. Select a base point $G = (G_x, G_y)$ of order n .

1537 C.2.4.2 Pseudorandom Curves B-233, B-283, B-409, B-571

1538 Each of the curves B-233 (Section 4.3.2.2), B-283 (Section 4.3.2.3), B-409 (Section 4.3.2.4), and
1539 B-571 (Section 4.3.2.5) is a curve $B_{a,b}$ in short-Weierstrass form with co-factor $h = 2$. Each curve
1540 is defined over a binary field $\text{GF}(2^m)$, where m is a prime number, where the prime number is

1541 amongst those values for which a binary Koblitz curve exists. The NIST prime curves were
 1542 generated using the procedure in C.3.3, with $hdigest = 160$ and SHA-1 hash function. The curve
 1543 parameters a and b are:

- 1544 1. The parameter a was set to $a = 1$ (this ensures that curves with co-factor $h = 2$ may exist).
- 1545 2. The parameter b was derived in a hard-to-invert way using the procedure in Appendix
 1546 C.3.3 from a pseudorandom *Seed* value so that the following conditions are satisfied
 1547 simultaneously:
 - 1548 a. $b \neq 0$ in $\text{GF}(p)$;
 - 1549 b. The curve has co-factor $h = 2$; and
 - 1550 c. The curve satisfies the cryptographic criteria in Appendix C.1.
- 1551 3. Select a base point $G = (G_x, G_y)$ of order n .
- 1552

1553 C.3 Generation and Verification of Pseudorandom Curves

1554 C.3.1 Generation of Pseudorandom Curves (Prime Case)

1555 When generating the NIST pseudo-random curves (i.e, those in Section 4.2.1), $hdigest = 160$ and
 1556 SHA-1 hash were used.

1557

1558 **Inputs:**

- 1559 1. Positive integer l
- 1560 2. Bit-string s of length $hdigest$
- 1561 3. Approved hash function $HASH$ with output length of $hdigest$ bits and security design
 1562 strength of at least *requested_security_strength*.

1563

1564 **Output:** Coefficient b used to generate a pseudorandom prime curve.

1565

1566 **Process:**

1567

1568 Let l be the bit length of p , and define

$$1569 \quad v = \lfloor (l - 1) / hdigest \rfloor$$

$$1570 \quad w = l - hdigest * v - 1.$$

- 1571 1. Choose an arbitrary $hdigest$ -bit string s as the domain parameter *Seed*.
- 1572 2. Compute $h = HASH(s)$.
- 1573 3. Let h_0 be the bit string obtained by taking the w rightmost bits of h .
- 1574 4. Let z be the integer whose binary expansion is given by the $hdigest$ -bit string s .
- 1575 5. For i from 1 to v do:
 - 1576 5.1 Define the $hdigest$ -bit string s_i to be binary expansion of the integer
 1577 $(z + i) \bmod (2^{hdigest})$.
 - 1578 5.2 Compute $h_i = HASH(s_i)$.
- 1579 6. Let h be the bit string obtained by the concatenation of h_0, h_1, \dots, h_v as follows:

$$1580 \quad h = h_0 \parallel h_1 \parallel \dots \parallel h_v.$$

1581 7. Let c be the integer whose binary expansion is given by the bit string h .

1582 8. If $((c = 0 \text{ or } 4c + 27 \equiv 0 \pmod{p}))$, then go to Step 1.

- 1583 9. Choose integers $a, b \in \text{GF}(p)$ such that
 1584
$$c b^2 \equiv a^3 \pmod{p}.$$

 1585 (The simplest choice is $a = c$ and $b = c$. However, they may be chosen differently for
 1586 performance reasons.)
 1587 10. Check that the elliptic curve E over $\text{GF}(p)$ given by $y^2 = x^3 + ax + b$ has suitable order. If
 1588 not, go to Step 1.
 1589

1590 C.3.2 Verification of Curve Pseudorandomness (Prime Case)

1591 Given the *hdigest* domain parameter seed value s , verify that the coefficient b was obtained from
 1592 s via the cryptographic hash function *HASH* as follows.

1593
 1594 **Inputs:**

- 1595 1. Positive integer l
 1596 2. Bit-string s of length *hdigest*
 1597 3. Approved hash function *HASH* with output length of *hdigest* bits and security design
 1598 strength of at least *requested_security_strength*
 1599

1600 **Output:** Verification that the coefficient b was obtained from s via the cryptographic hash
 1601 function *HASH*.

1602
 1603 **Process:**

1604
 1605 Let l be the bit length of p , and define

$$1606 \quad v = \lfloor (l - 1) / \textit{hdigest} \rfloor,$$

$$1607 \quad w = l - \textit{hdigest} * v - 1.$$

- 1608 1. Compute $h = \textit{HASH}(s)$.
 1609 2. Let h_0 be the bit string obtained by taking the w rightmost bits of h .
 1610 3. Let z be the integer whose binary expansion is given by the *hdigest*-bit string s .
 1611 4. For $i = 1$ to v do
 1612 4.1 Define the *hdigest*-bit string s_i to be binary expansion of the integer
 1613 $(z + i) \bmod (2^{\textit{hdigest}})$.
 1614 4.2 Compute $h_i = \textit{HASH}(s_i)$.
 1615 5. Let h be the bit string obtained by the concatenation of h_0, h_1, \dots, h_v as follows:
 1616
$$h = h_0 \parallel h_1 \parallel \dots \parallel h_v.$$

 1617 6. Let c be the integer whose binary expansion is given by the bit string h .
 1618 7. Verify that $b^2 c \equiv -27 \pmod{p}$.
 1619

1620 C.3.3 Generation of Pseudorandom Curves (Binary Case)

1621 **Inputs:**

- 1622 1. Prime number m
 1623 2. Bit-string s of length *hdigest*

- 1624 3. Approved hash function *HASH* with output length of *hdigest* bits and security design
1625 strength of at least *requested_security_strength*
1626
1627

1628 **Output:** Coefficient *b* used to generate a pseudorandom binary curve.
1629

1630 **Process:**

1631 Let:
1632

$$v = \lfloor (m - 1) / hdigest \rfloor$$

$$w = m - hdigest * v.$$

- 1633 1. Choose an arbitrary *hdigest* -bit string *s* as the domain parameter seed.
1634 2. Compute $h = HASH(s)$.
1635 3. Let h_0 be the bit string obtained by taking the *w* rightmost bits of *h*.
1636 4. Let *z* be the integer whose binary expansion is given by the *hdigest*-bit string *s*.
1637 5. For *i* from 1 to *v* do:
1638 5.1 Define the *hdigest* -bit string s_i to be binary expansion of the integer
1639 $(z + i) \bmod (2^{hdigest})$.
1640 5.2 Compute $h_i = HASH(s_i)$.
1641 6. Let *h* be the bit string obtained by the concatenation of h_0, h_1, \dots, h_v as follows:
1642 $h = h_0 \parallel h_1 \parallel \dots \parallel h_v$.
1643 7. Let *b* be the element of $GF(2^m)$ which is represented by the bit string *h* in the Gaussian
1644 Normal Basis (see Appendix G.3.1).
1645 8. Choose an element *a* of $GF(2^m)$.
1646 9. Check that the elliptic curve *E* over $GF(2^m)$ given by $y^2 + xy = x^3 + ax^2 + b$ has suitable
1647 order. If not, go to Step 1.
1648
1649
1650
1651

1651 C.3.4 Verification of Curve Pseudorandomness (Binary Case)

1652 Given the *hdigest*-bit domain parameter seed value *s*, verify that the coefficient *b* was obtained
1653 from *s* via the cryptographic hash function *HASH* as follows.
1654

1655 **Inputs:**

- 1656 1. Prime number *m*
1657 2. Bit-string *s* of length *hdigest*
1658 3. Approved hash function *HASH* with output length of *hdigest* bits and security design
1659 strength of at least *requested_security_strength*
1660

1661 **Output:** Verification that the coefficient *b* was obtained from *s* via the cryptographic hash
1662 function *HASH*.
1663

1664 **Process:**

1665 Define

$$v = \lfloor (m - 1) / hdigest \rfloor$$

$$w = m - hdigest * v$$

- 1666 1. Compute $h = HASH(s)$.
1668

- 1669 2. Let h_0 be the bit string obtained by taking the w rightmost bits of h .
- 1670 3. Let z be the integer whose binary expansion is given by the $hdigest$ -bit string s .
- 1671 4. For $i = 1$ to v do
- 1672 4.1 Define the $hdigest$ -bit string s_i to be binary expansion of the integer $(z + i) \bmod (2^{160}$
- 1673).
- 1674 4.2 Compute $h_i = HASH(s_i)$.
- 1675 5. Let h be the bit string obtained by the concatenation of h_0, h_1, \dots, h_v as follows:
- 1676 $h = h_0 \parallel h_1 \parallel \dots \parallel h_v$.
- 1677 6. Let c be the element of $GF(2^m)$ which is represented by the bit string h in the Gaussian
- 1678 Normal Basis (see Section G.3.1).
- 1679 7. Verify that $c = b$.
- 1680

1681 **Appendix D — Elliptic Curve Routines**1682 **D.1 Public Key Validation**1683 **D.1.1 Non-Binary Curves in Short-Weierstrass Form**1684 **D.1.1.1 Partial Public Key Validation**1685 **Inputs:**

- 1686 1. Weierstrass curve $W_{a,b}$ defined over the prime field $GF(p)$
- 1687 2. Point $Q=(x,y)$

1688 **Output:** ACCEPT or REJECT Q as an affine point on $W_{a,b}$.

1689 **Process:**

- 1690 1. If Q is the point at infinity \emptyset , output REJECT.
- 1691 2. Verify that x and y are integers in the interval $[0, p-1]$. Output REJECT if verification
- 1692 fails.
- 1693 3. Verify that (x, y) is a point on the $W_{a,b}$ by checking that (x, y) satisfies the defining
- 1694 equation $y^2 = x^3 + ax + b$ where computations are carried out in $GF(p)$. Output REJECT
- 1695 if verification fails.
- 1696 4. Otherwise output ACCEPT.

1697 **D.1.1.2 Full Public Key Validation**

1699 **Inputs:**

- 1700 1. Weierstrass curve $W_{a,b}$ defined over the prime field $GF(p)$
- 1701 2. Point Q

1702 **Output:** ACCEPT or REJECT Q as a point on $W_{a,b}$ of order n .

1703 **Process:**

- 1704 1. Perform partial public key validation on Q using the procedure of Appendix D.1.1.1.
- 1705 Output REJECT if this procedure outputs REJECT.
- 1706 2. Verify that $nQ = \emptyset$. Output REJECT if verification fails.
- 1707 3. Otherwise, output ACCEPT.

1708 **D.1.2 Montgomery Curves**1709 **D.1.2.1 Partial Public Key Validation**1710 **Inputs:**

- 1711 1. Montgomery curve $M_{A,B}$ defined over the prime field $GF(p)$

1712 2. Point $Q=(u, v)$

1713 **Output:** ACCEPT or REJECT Q as an affine point on $M_{A,B}$.

1714 **Process:**

- 1715 1. If Q is the point at infinity \emptyset , output REJECT.
- 1716 2. Verify that both u and v are integers in the interval $[0, p-1]$. Output REJECT if
1717 verification fails.
- 1718 3. Verify that (u, v) is a point on the $M_{A,B}$ by checking that (u, v) satisfies the defining
1719 equation $v^2 = u(u^2 + Au + 1)$ where computations are carried out in $GF(p)$. Output
1720 REJECT if verification fails.
- 1721 4. Otherwise output ACCEPT.

1722 D.1.2.2 Full Public Key Validation

1723 **Inputs:**

- 1724 1. Montgomery curve $M_{A,B}$ defined over the prime field $GF(p)$
- 1725 2. Point Q

1726 **Output:** ACCEPT or REJECT Q as a point on $M_{A,B}$ of order n .

1727 **Process:**

- 1728 1. Perform partial public key validation on Q using the procedure of Appendix D.1.2.1.
1729 Output REJECT if this procedure outputs REJECT.
- 1730 2. Verify that $nQ = \emptyset$. Output REJECT if verification fails.
- 1731 3. Otherwise output ACCEPT.

1732 D.1.3 Twisted Edwards Curves

1733 D.1.3.1 Partial Public Key Validation

1734 **Inputs:**

- 1735 1. Edwards curve $E_{a,d}$ defined over the prime field $GF(p)$
- 1736 2. Point $Q=(x, y)$

1737 **Output:** ACCEPT or REJECT Q as an affine point on $E_{a,d}$.

1738 **Process:**

- 1739 1. Verify that both x and y are integers in the interval $[0, p-1]$. Output REJECT if
1740 verification fails.
- 1741 2. Verify that (x, y) is a point on the $E_{a,d}$ by checking that (x, y) satisfies the defining
1742 equation $ax^2 + y^2 = 1 + dx^2y^2$ where computations are carried out in $GF(p)$. Output
1743 REJECT if verification fails.
- 1744 3. Otherwise output ACCEPT.

1745

1746 **D.1.3.2 Full Public Key Validation**1747 **Inputs:**

- 1748 1. Edwards curve $E_{a,d}$ defined over the prime field $GF(p)$
 1749 2. Point Q

1750 **Output:** ACCEPT or REJECT Q as a point on $E_{a,d}$ of order n .1751 **Process:**

- 1752 1. Perform partial public key validation on Q using the procedure of Appendix D.1.3.1.
 1753 Output REJECT if this procedure outputs REJECT.
 1754 2. If Q is the point at identity element $(0,1)$, output REJECT.
 1755 3. Verify that $nQ = (0,1)$. Output REJECT if verification fails.
 1756 4. Otherwise output ACCEPT.

1757 **D.1.4 Binary Curves in Short-Weierstrass Form**1758 **D.1.4.1 Partial Public Key Validation**1759 **Inputs:**

- 1760 1. Weierstrass curve $B_{a,b}$ defined over the binary field $GF(2^m)$
 1761 2. Point $Q=(x, y)$

1762 **Output:** ACCEPT or REJECT Q as an affine point on $B_{a,b}$.1763 **Process:**

- 1764 1. If Q is the point at infinity \emptyset , output REJECT;
 1765 2. Verify that both x and y are binary polynomials in $GF(2^m)$ according to the field
 1766 representation indicated by the parameter FR . Output REJECT if verification fails.
 1767 3. Verify that (x, y) is a point on the $B_{a,b}$ by checking that (x, y) satisfies the defining
 1768 equation $y^2 + xy = x^3 + ax^2 + b$, where computations are carried out in $GF(2^m)$ according
 1769 to the field representation indicated by the parameter FR . Output REJECT if verification
 1770 fails.
 1771 4. Otherwise output ACCEPT.

1772 **D.1.4.2 Full Public Key Validation**1773 **Inputs:**

- 1774 1. Weierstrass curve $B_{a,b}$ defined over the binary field $GF(2^m)$;
 1775 2. Point Q .

1776 **Output:** ACCEPT or REJECT Q as a point on $B_{a,b}$ of order n .

1777 **Process:**

- 1778 1. Perform partial public key validation on Q using the procedure of Appendix D.1.4.1.
 1779 Output REJECT if this procedure outputs REJECT.
 1780 2. Verify that $nQ = \emptyset$. Output REJECT if verification fails.
 1781 3. Otherwise output ACCEPT.

1782 **D.2 Point Compression**

1783 Point compression allows a shorter representation of elliptic curve points in affine coordinates by
 1784 exploiting algebraic relationships between the coordinate values based on the defining equation
 1785 of the curve in question. Point compression followed by its inverse, “point decompression,” is
 1786 the identity map.

1787 **D.2.1 Prime Curves in Short-Weierstrass Form**

1788 Point compression for non-binary curves in short-Weierstrass form is defined as follows.

1789 **Inputs:**

- 1790 1. Weierstrass curve $W_{a,b}$ defined over the prime field $GF(p)$
 1791 2. Point P on $W_{a,b}$

1792 **Output:** Compressed point \underline{P} .

1793 **Process:**

- 1794 1. If P is the point at infinity \emptyset , set $\underline{P} = P$.
 1795 2. If $P = (x, y)$, set $\underline{P} = (x, \underline{y})$, where $\underline{y} = y \pmod{2}$.
 1796 3. Output \underline{P} .

1797 Point decompression of an object \underline{P} with respect to this Weierstrass curve is defined as follows.

1798 **Inputs:**

- 1799 1. Object \underline{P}
 1800 2. Weierstrass curve $W_{a,b}$ defined over the prime field $GF(p)$

1801 **Output:** Point P on $W_{a,b}$ or INVALID.

1802 **Process:**

- 1803 1. If \underline{P} is the point at infinity \emptyset , output $P = \underline{P}$.
 1804 2. If \underline{P} is the ordered pair (x, t) , where x is an element of $GF(p)$ and where t is an element of
 1805 $GF(2)$:
 1806 2.1. Compute $w = x^3 + ax + b$
 1807 2.2. Compute a square root y of w in $GF(p)$ using the procedure of Appendix E.3;
 1808 output INVALID if that procedure outputs INVALID
 1809 2.3. If $y = 0$ and $t = 1$, output INVALID

- 1810 2.4. If $t \neq y \pmod{2}$, set $y = p - y$
 1811 2.5. Output $P = (x, y)$
 1812 3. Output INVALID
 1813

1814 **D.2.2 Binary Curves in Short-Weierstrass Form**

1815 Point compression for binary curves in short-Weierstrass form is defined as follows.

1816 **Inputs:**

- 1817 1. Weierstrass curve $B_{a,b}$ defined over the binary field $\text{GF}(2^m)$
 1818 2. Point P on $B_{a,b}$

1819 **Output:** Compressed point \underline{P} .

1820 **Process:**

- 1821 1. If P is the point at infinity \emptyset , set $\underline{P} = P$.
 1822 2. If $P = (x, y)$ and $x=0$, set $\underline{P} = (x, \underline{y})$, where $\underline{y} = 0 \pmod{2}$.
 1823 3. If $P = (x, y)$ and $x \neq 0$:
 1824 3.1. Compute $\alpha = y/x$, where $\alpha = \alpha_0 + \alpha_1z + \dots + \alpha_{m-1}z^{m-1}$
 1825 3.2. Set $\underline{P} = (x, \underline{y})$, where $\underline{y} = \alpha_0$
 1826 4. Output \underline{P} .

1827 Consequently, for each affine point $P = (x, y)$ on the Weierstrass curve $B_{a,b}$, the compressed
 1828 point \underline{P} is an ordered pair (x, t) where x is an element of $\text{GF}(2^m)$ and where t is an element of
 1829 $\text{GF}(2)$.

1830 Point decompression of an object \underline{P} with respect to this Weierstrass curve is defined as follows.

1831 **Inputs:**

- 1832 1. Object \underline{P}
 1833 2. Weierstrass curve $B_{a,b}$ defined over the binary field $\text{GF}(2^m)$, where m is an odd integer

1834 **Output:** Point P on $B_{a,b}$ or INVALID.

1835 **Process:**

- 1836 1. If \underline{P} is the point at infinity \emptyset , output $P = \underline{P}$.
 1837 2. If \underline{P} is the ordered pair (x, t) , where x is an element of $\text{GF}(2^m)$ and where t is an element of
 1838 $\text{GF}(2)$, perform the following:
 1839 2.1. If $x = 0$, perform the following steps:
 1840 2.1.1. If $t = 1$, output INVALID
 1841 2.1.2. Set y to the square root of b in $\text{GF}(2^m)$ using the algorithm of Appendix E.1
 1842 2.2. If $x \neq 0$, perform the following steps:
 1843 2.2.1. Compute $w = (x^3 + ax^2 + b)/x^2 = x + a + b/x^2$

1844 2.2.2. Compute a solution α in $\text{GF}(2^m)$ of the equation $\alpha^2 + \alpha = w$ using the algorithm of
1845 Appendix E.2; output INVALID if that procedure outputs INVALID

1846 2.2.3. If $t \neq \alpha_0$, where $\alpha = \alpha_0 + \alpha_1 z + \dots + \alpha_{m-1} z^{m-1}$, set $\alpha = \alpha + 1$

1847 2.2.4. Set $y = \alpha x$

1848 2.3. Output $P = (x, y)$

1849 3. Output INVALID.

1850 D.3 Base Point (Generator) Selection

1851 For user-generated base points, use a verifiably random method and check the validity of the
1852 point generated. This Appendix describes these methods.

1853 D.3.1 Generation of Base Points

1854 A base point **should** be generated as follows.

1855 **Input:** Elliptic curve $E = (F_q, a, b)$, cofactor h , prime n , and, optionally, a bit string *Seed*, which
1856 indicates that verifiably random G is desired.

1857 **Output:** A base point G on the curve of order n , or FAILURE.

1858 **Process:** The following or its equivalent:

1859 1. Set $base = 1$.

1860 2. Select elements x and y in the field F_q , doing so verifiably at random using Appendix
1861 D.4.2 or by any desired method if *Seed* is not provided.

1862 Comment: The pair (x, y) **should** be chosen to lie on the curve E , or else the
1863 process could loop forever.

1864 3. Let $G = hR$, where $R = (x, y)$.

1865 4. If G is not a valid base point (see Appendix D.4.3), then increment $base$ and go back to
1866 Step 1 unless $base > 10h^2$, in which case, output FAILURE.

1867 Comment: The validity of G as a point is partially assured by R having valid
1868 coordinates and belonging to the curve. The verifiable random nature of G is also
1869 assured, so this does not need to be checked. Therefore, when validating G , it is
1870 only necessary to check that $G \neq O$ and $nG = O$.

1871 If the elliptic curve E does not have a multiple of n points, then the output will generally be
1872 FAILURE. Conversely, if the algorithm outputs FAILURE, generally the elliptic curve does not
1873 have $h \cdot n$ points. If the elliptic curve E has exactly $h \cdot n$ points but n is composite, then G is not
1874 guaranteed to have order exactly n but will have an order dividing n . The probability that G has
1875 an order exactly n depends on the factorization of n . If the elliptic curve E has $k \cdot n$ points where k
1876 $\neq h$, then the order G is not guaranteed to have order n . If n is prime, then G will generally have
1877 an order which is a multiple of n . If the elliptic curve E has exactly $h \cdot n$ points, then $base$ will
1878 generally never be incremented.

1879 **D.3.2 Verifiably Random Base Points**

1880 This procedure will generate a verifiably random candidate point.

1881 **Inputs:** Bit string *Seed*, integer counter *base*, selected hash function with output length *hashlen*
1882 bits, field size *q*, cofactor *h*1883 **Output:** Candidate point (x, y) 1884 **Process:** The following or its equivalent:

- 1885 1. Set *element* = 1.
- 1886 2. Convert *base* and *element* to octet strings *Base* and *Element*, respectively.
- 1887 3. Compute $H = \text{Hash}(\text{"Base point"} \parallel \textit{Base} \parallel \textit{Element} \parallel \textit{Seed})$.
- 1888 4. Convert *H* to an integer *e*.
- 1889 5. If $\lfloor e / 2q \rfloor = \lfloor 2^{\textit{hashlen}} / 2q \rfloor$, then increment *element* and go to Step 2.
- 1890 6. Let $t = e \bmod 2q$, so that *t* is an integer in the interval $[0, 2q - 1]$.
- 1891 7. Let $x = t \bmod q$ and $z = \lfloor t / q \rfloor$.
- 1892 8. Convert *x* to field element in F_q using the routine in Appendix F.2.
- 1893 9. Recover the field element *y* from (x, z) using an appropriate compression method from
1894 Appendix D.2.
- 1895 10. If the result is an error, then increment *element* and go to Step 2.

1896 **D.3.3 Validity of Base Points**

1897 A base point generator is valid if the following routine results in VALID.

1898 **Input:** Elliptic curve domain parameters1899 **Output:** VALID or INVALID1900 **Process:** The following or its equivalent:

- 1901 1. If $G = O$, then stop and output INVALID.
- 1902 2. If either of the base point coordinates x_G and y_G are invalid as elements of F_q (that is: if *q*
1903 is odd, then either x_G or y_G is not an integer in the interval $[0, q-1]$; or if $q = 2^m$, then
1904 either x_G or y_G is not a bit string of length *m*), then stop and output INVALID.
- 1905 3. If *G* is not on the elliptic curve, that is, $y_G^2 \neq x_G^3 + ax_G + b$ if *q* is odd, or $y_G^2 + x_G y_G \neq$
1906 $x_G^3 + ax_G^2 + b$ if *q* is even, then stop and output INVALID.
- 1907 4. If $nG \neq O$, then stop and output INVALID. A full scalar multiplication **shall** be used.

1908 Comment: Shortcuts for validating the order of point that assume a value for the
1909 cofactor would not be considered a full scalar multiplication.

- 1910 5. If the input indicates that the base point *G* is generated verifiably at random, then do the
1911 following:

1912 5.1. Set *base* = 1.1913 5.2. With *Seed* and *base* values, generate a point $R = (x, y)$, using the routine in
1914 Appendix D.4.2.

- 1915 5.3. Compute $G' = hR$.
- 1916 5.4. If $nG' \neq O$, then increment *base* and go back to Step 2.
- 1917 Comment: The counter value *base* will generally never be incremented
- 1918 5.5. If $base > 10h^2$, then stop and output INVALID.
- 1919 5.6. Compare G' with G . If not equal, then stop and output INVALID.
- 1920 6. Otherwise, output VALID.
- 1921

1922 **Appendix E Auxiliary Functions**1923 **E.1 Computing Square Roots in Binary Fields**

1924 If x is an element of $\text{GF}(2^m)$, then its square root is the element $x^{2^{m-1}}$.

1925 **E.2 Solving the Equation $x^2 + x = w$ in Binary Fields**

1926 **Input:** Field element w in $\text{GF}(2^m)$, where m is an odd integer.

1927 **Output:** Solution α in $\text{GF}(2^m)$ of the equation $\alpha^2 + \alpha = w$, or INVALID.

1928 **Process:**

- 1929 1. Compute $\text{Tr}(w) = w^{2^0} + w^{2^1} + w^{2^2} + w^{2^3} + \dots + w^{2^{m-1}}$ (the trace of w);
- 1930 2. If $\text{Tr}(w) \neq 1$, output INVALID;
- 1931 3. Compute $\alpha := \text{Hf}(x) = w^{2^0} + w^{2^2} + w^{2^4} + \dots + w^{2^{m-1}}$ (the half-trace of w);
- 1932 4. Output α .

1933 **E.3 Computing Square Roots in non-Binary Fields $\text{GF}(q)$**

1934 The Tonelli-Shanks algorithm can be used to compute a square root given an equation of the
1935 form $x^2 \equiv n \pmod{p}$ where n is an integer, which is a quadratic residue \pmod{p} , and p is an odd
1936 prime.

1937 Find Q and S (with Q odd) such that $p-1 = Q2^S$ by factoring out the powers of 2.

1938 Note that if $S = 1$, as for primes $p \equiv 3 \pmod{4}$, this reduces to finding $x = n^{(p+1)/4} \pmod{p}$

1939 Check to see if $n^Q = 1$; if so then the root $x = n^{(Q+1)/2} \pmod{p}$.

1940 Otherwise select a z which is a quadratic non-residue modulo p . The Legendre symbol $\left(\frac{a}{p}\right)$ where
1941 p is an odd prime and a is an integer can be used to test candidate values for z to see if a value of
1942 -1 is returned.

1943 Search for a solution as follows:

1944 Set $x = n^{(Q+1)/2} \pmod{p}$

1945 Set $t = n^Q \pmod{p}$

1946 Set $M = S$

1947 Set $c = z^Q \pmod{p}$

1948 While $t \neq 1$, repeat the following steps:

1949 a) Using repeated squaring, find the smallest i such that $t^{2^i} = 1$, where $0 < i < M$.

1950 For example:

1951 Let $e = 2$

1952 Loop for $i = 1$ until $i = M$:

1953 If $t^e \pmod p = 1$ then exit the loop.

1954 Set $e = 2e$

1955 b) Update values:

1956 $b = c^{2^{M-i-1}} \pmod p$

1957 $x = xb \pmod p$

1958 $t = tb^2 \pmod p$

1959 $c = b^2 \pmod p$

1960 $M = i$

1961 The solution is x and the second solution is $p - x$. If the least i found is M , then no solution exists.

1962 Square roots in a non-binary field $\text{GF}(q)$ are relatively efficient to compute if q has the special
1963 form $q \equiv 3 \pmod 4$ or $q \equiv 5 \pmod 8$. All but one of the elliptic curves recommended in this
1964 recommendation are defined over such fields. The following routines describe simplified cases to
1965 compute square roots for $p \equiv 3 \pmod 4$ or $p \equiv 5 \pmod 8$.

1966 Let $u = y^2 - 1$ and $v = dy^2 + 1$.

1967

1968 To find a square root of (u/v) if $p \equiv 3 \pmod 4$ (as in E448), first compute the candidate root x
1969 $= (u/v)^{(p+1)/4} = u^3 v (u^5 v^3)^{(p-3)/4} \pmod p$. If $v x^2 = u$, the square root is x . Otherwise, no square
1970 root exists, and the decoding fails.

1971

1972 To find a square root of (u/v) if $p \equiv 5 \pmod 8$ (as in Edwards25519), first compute the
1973 candidate root $x = (u/v)^{(p+3)/8} = u v^3 (u v^7)^{(p-5)/8} \pmod p$. To find the root, check three cases:

1974

- If $v x^2 = u \pmod p$, the square root is x .

1975

- If $v x^2 = -u \pmod p$, the square root is $x * 2^{((p-1)/4)}$.

1976

- Otherwise, no square root exists for modulo p , and decoding fails.

1977

1978 If $x = 0$ and $x_0 = 1$, point decoding fails. If $x \pmod 2 = x_0$, then the x -coordinate is x .

1979

Otherwise, the x -coordinate is $p - x$.

1980

1981 E.4 Computing Inverses in $\text{GF}(q)$

1982 If x is an element of $\text{GF}(q)$ and $x \neq 0$, its (multiplicative) inverse is the element x^{q-2} .

1983 If one is concerned about side-channel leakage, one **should** compute u^{-1} indirectly by first
1984 computing the inverse of the blinded element λu , where λ is a random nonzero element of $\text{GF}(q)$,
1985 and subsequently computing $\lambda(\lambda u)^{-1} = u^{-1}$. This yields an inversion routine where the inversion
1986 operation itself does not require side-channel protection and which may have relatively low
1987 computational complexity.

1988

1989 **Appendix F Data Conversion**1990 **F.1 Conversion of a Field Element to an Integer**1991 Field elements **shall** be converted to integers according to the following procedure.1992 **Input:** An element a of the field $\text{GF}(q)$ 1993 **Output:** A non-negative integer x in the interval $[0, q-1]$ 1994 **Process:**

- 1995 1. If q is an odd prime, a is an integer in the interval $[0, q-1]$. In this case, set $x = a$.
- 1996 2. If $q = 2^m$, a must be a binary polynomial of degree smaller than m , i.e.,
- 1997 $a = a(z) = a_{m-1} z^{m-1} + a_{m-2} z^{m-2} + \dots + a_1 z + a_0$, where each coefficient a_i is 0 or 1.
- 1998 In this case, set $x = a(2) = a_{m-1} 2^{m-1} + a_{m-2} 2^{m-2} + \dots + a_1 2^1 + a_0 2^0$;
- 1999 3. Output x .

2000
2001 **F.2 Conversion of an Integer to a Field Element**2002 Integers **shall** be converted to field elements according to the following procedure.2003 **Inputs:** Non-negative integer x and q , where q is an odd prime or $q=2^m$ 2004 **Output:** An element a of the field $\text{GF}(q)$ 2005 **Process:**

- 2006 1. Set $x = x \pmod{q}$;
- 2007 2. If q is an odd prime, x is an integer in the interval $[0, q-1]$. In this case, set $a = x$;
- 2008 3. If $q = 2^m$, x can be uniquely written as $x = a_{m-1} 2^{m-1} + a_{m-2} 2^{m-2} + \dots + a_1 2 + x_0$, where
- 2009 each coefficient x_i is 0 or 1. In this case, set $x = a(z) = a_{m-1} z^{m-1} + a_{m-2} z^{m-2} + \dots + a_1 z^1 +$
- 2010 $a_0 2^0$;
- 2011 4. Output a .

2012
2013 **F.3 Conversion of an Integer to a Bit String**2014 Integers **shall** be converted to bit strings according to the following procedure.2015 **Inputs:** Non-negative integer x in the range $0 \leq x < 2^l$ 2016 **Output:** Bit-string X of length l 2017 **Process:**

- 2018 1. The integer x can be uniquely written as $x = x_{l-1} 2^{l-1} + x_{l-2} 2^{l-2} + \dots + x_1 2 + x_0$, where
- 2019 each coefficient x_i is 0 or 1.
- 2020 2. Set X to the bit string $(x_{l-1}, x_{l-2}, \dots, x_1, x_0)$;

2021 3. Output X .

2022

2023 **F.4 Conversion of a Bit String to an Integer**

2024 Bit strings **shall** be converted to integers according to the following procedure.

2025 **Input:** Bit-string X of length l

2026 **Output:** Non-negative integer x , where $x < 2^l$

2027 **Process:**

2028 1. Let X be the bit string $(x_{l-1}, x_{l-2}, \dots, x_1, x_0)$, where each coefficient x_i is 0 or 1;

2029 2. Set x to the integer value $x = x_{l-1} 2^{l-1} + x_{l-2} 2^{l-2} + \dots + x_1 2 + x_0$;

2030 3. Output x .

2031

2032

2033

2034 **Appendix G Implementation Aspects**2035 **G.1 Implementation of Modular Arithmetic**

2036 The prime moduli of the above recommended curves are of a special type (*generalized Mersenne*
2037 *numbers* and *Crandall primes*) for which modular multiplication can be carried out more
2038 efficiently than in general. This section provides the rules for implementing this faster arithmetic
2039 for each of these recommended prime moduli.

2040 The usual way to multiply two integers (mod m) is to take the integer product and reduce it
2041 (modulo m). One, therefore, has the following problem: given an integer A less than m^2 , compute

$$2042 \quad B = A \pmod{m}.$$

2043 In general, one must obtain B as the remainder of an integer division. If m is a generalized
2044 Mersenne number, however, then B can be expressed as a sum or difference (mod m) of a small
2045 number of terms. To compute this expression, the integer sum or difference can be evaluated,
2046 and the result reduced modulo m . The latter reduction can be accomplished by adding or
2047 subtracting a few copies of m .

2048 The prime modulus p for each of the four recommended P-x curves is a generalized Mersenne
2049 number.

2050 **G.1.1 Curve P-224**

2051 The modulus for this curve is $p = 2^{224} - 2^{96} + 1$. Each integer A less than p^2 can be written as

$$2052 \quad A = A_{13} \cdot 2^{416} + A_{12} \cdot 2^{384} + A_{11} \cdot 2^{352} + A_{10} \cdot 2^{320} + A_9 \cdot 2^{288} + A_8 \cdot 2^{256} + A_7 \cdot 2^{224} + A_6 \cdot 2^{192} + \\ A_5 \cdot 2^{160} + A_4 \cdot 2^{128} + A_3 \cdot 2^{96} + A_2 \cdot 2^{64} + A_1 \cdot 2^{32} + A_0,$$

2053 where each A_i is a 32-bit integer. As a concatenation of 32-bit words, this can be denoted by

$$2054 \quad A = (A_{13} \parallel A_{12} \parallel \dots \parallel A_0).$$

2055 The expression for B is

$$2056 \quad B = T + S_1 + S_2 - D_1 - D_2 \pmod{p},$$

2057 where the 224-bit terms are given by

$$2058 \quad T = (A_6 \parallel A_5 \parallel A_4 \parallel A_3 \parallel A_2 \parallel A_1 \parallel A_0)$$

$$2059 \quad S_1 = (A_{10} \parallel A_9 \parallel A_8 \parallel A_7 \parallel 0 \parallel 0 \parallel 0)$$

$$2060 \quad S_2 = (0 \parallel A_{13} \parallel A_{12} \parallel A_{11} \parallel 0 \parallel 0 \parallel 0)$$

$$2061 \quad D_1 = (A_{13} \parallel A_{12} \parallel A_{11} \parallel A_{10} \parallel A_9 \parallel A_8 \parallel A_7)$$

2062 $D_2 = (0 \parallel 0 \parallel 0 \parallel 0 \parallel A_{13} \parallel A_{12} \parallel A_{11}).$

2063 **G.1.2 Curve P-256**

2064 The modulus for this curve is $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$. Each integer A less than p^2 can be
2065 written as

2066
$$A = A_{15} \cdot 2^{480} + A_{14} \cdot 2^{448} + A_{13} \cdot 2^{416} + A_{12} \cdot 2^{384} + A_{11} \cdot 2^{352} + A_{10} \cdot 2^{320} + A_9 \cdot 2^{288} + A_8 \cdot 2^{256} +$$

$$A_7 \cdot 2^{224} + A_6 \cdot 2^{192} + A_5 \cdot 2^{160} + A_4 \cdot 2^{128} + A_3 \cdot 2^{96} + A_2 \cdot 2^{64} + A_1 \cdot 2^{32} + A_0,$$

2067 where each A_i is a 32-bit integer. As a concatenation of 32-bit words, this can be denoted by

2068
$$A = (A_{15} \parallel A_{14} \parallel \dots \parallel A_0).$$

2069 The expression for B is

2070
$$B = T + 2S_1 + 2S_2 + S_3 + S_4 - D_1 - D_2 - D_3 - D_4 \pmod{p},$$

2071 where the 256-bit terms are given by

2072
$$T = (A_7 \parallel A_6 \parallel A_5 \parallel A_4 \parallel A_3 \parallel A_2 \parallel A_1 \parallel A_0)$$

2073
$$S_1 = (A_{15} \parallel A_{14} \parallel A_{13} \parallel A_{12} \parallel A_{11} \parallel 0 \parallel 0 \parallel 0)$$

2074
$$S_2 = (0 \parallel A_{15} \parallel A_{14} \parallel A_{13} \parallel A_{12} \parallel 0 \parallel 0 \parallel 0)$$

2075
$$S_3 = (A_{15} \parallel A_{14} \parallel 0 \parallel 0 \parallel 0 \parallel A_{10} \parallel A_9 \parallel A_8)$$

2076
$$S_4 = (A_8 \parallel A_{13} \parallel A_{15} \parallel A_{14} \parallel A_{13} \parallel A_{11} \parallel A_{10} \parallel A_9)$$

2077
$$D_1 = (A_{10} \parallel A_8 \parallel 0 \parallel 0 \parallel 0 \parallel A_{13} \parallel A_{12} \parallel A_{11})$$

2078
$$D_2 = (A_{11} \parallel A_9 \parallel 0 \parallel 0 \parallel A_{15} \parallel A_{14} \parallel A_{13} \parallel A_{12})$$

2079
$$D_3 = (A_{12} \parallel 0 \parallel A_{10} \parallel A_9 \parallel A_8 \parallel A_{15} \parallel A_{14} \parallel A_{13})$$

2080
$$D_4 = (A_{13} \parallel 0 \parallel A_{11} \parallel A_{10} \parallel A_9 \parallel 0 \parallel A_{15} \parallel A_{14})$$

2081 **G.1.3 Curve P-384**

2082 The modulus for this curve is $p = 2^{384} - 2^{128} - 2^{96} + 2^{32} - 1$. Each integer A less than p^2 can be
2083 written as

2084
$$A = A_{23} \cdot 2^{736} + A_{22} \cdot 2^{704} + A_{21} \cdot 2^{672} + A_{20} \cdot 2^{640} + A_{19} \cdot 2^{608} + A_{18} \cdot 2^{576} + A_{17} \cdot 2^{544} + A_{16} \cdot 2^{512} +$$

$$A_{15} \cdot 2^{480} + A_{14} \cdot 2^{448} + A_{13} \cdot 2^{416} + A_{12} \cdot 2^{384} + A_{11} \cdot 2^{352} + A_{10} \cdot 2^{320} + A_9 \cdot 2^{288} + A_8 \cdot 2^{256} +$$

$$A_7 \cdot 2^{224} + A_6 \cdot 2^{192} + A_5 \cdot 2^{160} + A_4 \cdot 2^{128} + A_3 \cdot 2^{96} + A_2 \cdot 2^{64} + A_1 \cdot 2^{32} + A_0,$$

2085 where each A_i is a 32-bit integer. As a concatenation of 32-bit words, this can be denoted by

$$2086 \quad A = (A_{23} \parallel A_{22} \parallel \cdots \parallel A_0).$$

2087 The expression for B is

$$2088 \quad B = T + 2S_1 + S_2 + S_3 + S_4 + S_5 + S_6 - D_1 - D_2 - D_3 \pmod{p},$$

2089 where the 384-bit terms are given by

$$2090 \quad T = (A_{11} \parallel A_{10} \parallel A_9 \parallel A_8 \parallel A_7 \parallel A_6 \parallel A_5 \parallel A_4 \parallel A_3 \parallel A_2 \parallel A_1 \parallel A_0)$$

$$2091 \quad S_1 = (0 \parallel 0 \parallel 0 \parallel 0 \parallel 0 \parallel A_{23} \parallel A_{22} \parallel A_{21} \parallel 0 \parallel 0 \parallel 0 \parallel 0)$$

$$2092 \quad S_2 = (A_{23} \parallel A_{22} \parallel A_{21} \parallel A_{20} \parallel A_{19} \parallel A_{18} \parallel A_{17} \parallel A_{16} \parallel A_{15} \parallel A_{14} \parallel A_{13} \parallel A_{12})$$

$$2093 \quad S_3 = (A_{20} \parallel A_{19} \parallel A_{18} \parallel A_{17} \parallel A_{16} \parallel A_{15} \parallel A_{14} \parallel A_{13} \parallel A_{12} \parallel A_{23} \parallel A_{22} \parallel A_{21})$$

$$2094 \quad S_4 = (A_{19} \parallel A_{18} \parallel A_{17} \parallel A_{16} \parallel A_{15} \parallel A_{14} \parallel A_{13} \parallel A_{12} \parallel A_{20} \parallel 0 \parallel A_{23} \parallel 0)$$

$$2095 \quad S_5 = (0 \parallel 0 \parallel 0 \parallel 0 \parallel A_{23} \parallel A_{22} \parallel A_{21} \parallel A_{20} \parallel 0 \parallel 0 \parallel 0 \parallel 0)$$

$$2096 \quad S_6 = (0 \parallel 0 \parallel 0 \parallel 0 \parallel 0 \parallel 0 \parallel A_{23} \parallel A_{22} \parallel A_{21} \parallel 0 \parallel 0 \parallel A_{20})$$

$$2097 \quad D_1 = (A_{22} \parallel A_{21} \parallel A_{20} \parallel A_{19} \parallel A_{18} \parallel A_{17} \parallel A_{16} \parallel A_{15} \parallel A_{14} \parallel A_{13} \parallel A_{12} \parallel A_{23})$$

$$2098 \quad D_2 = (0 \parallel 0 \parallel 0 \parallel 0 \parallel 0 \parallel 0 \parallel 0 \parallel A_{23} \parallel A_{22} \parallel A_{21} \parallel A_{20} \parallel 0)$$

$$2099 \quad D_3 = (0 \parallel 0 \parallel 0 \parallel 0 \parallel 0 \parallel 0 \parallel 0 \parallel A_{23} \parallel A_{23} \parallel 0 \parallel 0 \parallel 0).$$

2100 **G.1.4 Curve P-521**

2101 The modulus for this curve is $p = 2^{521} - 1$. Each integer A less than p^2 can be written as

$$2102 \quad A = A_1 \cdot 2^{521} + A_0,$$

2103 where each A_i is a 521-bit integer. As a concatenation of 521-bit words, this can be denoted by

$$2104 \quad A = (A_1 \parallel A_0).$$

2105

2106 The expression for B is

$$2107 \quad B = (A_0 + A_1) \pmod{p}.$$

2108 **G.1.5 Curve Curve448**

2109 The modulus for this curve is $p = 2^{448} - 2^{224} - 1$. Each integer A less than p^2 can be written

2110
$$A = A_3 \cdot 2^{672} + A_2 \cdot 2^{448} + A_1 \cdot 2^{224} + A_0,$$

2111 where each A_i is a 224-bit integer. As a concatenation of 224-bit words, this can be denoted by

2112
$$A = (A_3 \parallel A_2 \parallel A_1 \parallel A_0).$$

2113

2114 The expression for B is

2115
$$B = (S_1 + S_2 + S_3 + S_4) \pmod{p},$$

2116 where the 448-bit terms are given by

2117
$$S_1 = (A_1 \parallel A_0)$$

2118
$$S_2 = (A_2 \parallel A_2)$$

2119
$$S_3 = (A_3 \parallel A_3)$$

2120
$$S_4 = (A_3 \parallel 0).$$

2121 **G.1.6 Curve Curve25519**

2122 The modulus for this curve is $p = 2^{255} - 19$. Each integer A less than p^2 can be written

2123
$$A = A_1 \cdot 2^{256} + A_0,$$

2124 where each A_i is a 256-bit integer. As a concatenation of 256-bit words, this can be denoted by

2125
$$A = (A_1 \parallel A_0).$$

2126 The expression for B is

2127
$$B = (38 \cdot A_1 + A_0) \pmod{2p},$$

2128 where all computations are carried out modulo $2p$ rather than modulo p .

2129 This allows efficient modular reduction and finite field operations that try and minimize carry-
2130 effects of operands if each integer X less than $2p$ is represented as

2131

2132 $X = X_9 \cdot 2^{234} + X_8 \cdot 2^{208} + X_7 \cdot 2^{182} + X_6 \cdot 2^{156} + X_5 \cdot 2^{130} + X_4 \cdot 2^{104} + X_3 \cdot 2^{78} + X_2 \cdot 2^{52} + X_1 \cdot 2^{26} +$
2133 $X_0,$

2134 where each X_i is a 26-bit integer and where X_9 is a 22-bit integer. Note that in this case,
2135 multiplication by the small constant 38 does not lead to overflows if each X_i is stored as a 32-bit
2136 word. It turns out that the cost of occasional resizing of X , represented this way, is outweighed by
2137 savings due to the possibility of postponing ‘carry’ operations. This representation can also be
2138 used to efficiently compute $-X$ so that intermediate integer segments are always non-negative
2139 integers.

2140 **G.2 Scalar Multiplication for Koblitz Curves**

2141 This section describes a particularly efficient method of computing the scalar multiple $Q := kP$ on
2142 the Koblitz curve $W_{a,b}$ over $\text{GF}(2^m)$.

2143 The operation τ is defined by

$$2144 \quad \tau(x, y) := (x^2, y^2).$$

2145 When the normal basis representation is used, then the operation τ is implemented by
2146 performing right circular shifts on the bit strings representing x and y .

2147 Given m and a , define the following parameters:

- 2148 • C is some integer greater than 5.
- 2149 • $\mu = (-1)^{1-a}$.
- 2150 • For $i = 0$ and $i = 1$, define the sequence $s_i(m)$ by:

$$2151 \quad s_i(0) := 0, \quad s_i(1) := 1 - i,$$

$$2152 \quad s_i(m) = \mu \cdot s_i(m-1) - 2 \cdot s_i(m-2) + (-1)^i.$$

- 2153 • Define the sequence $V(m)$ by

$$2154 \quad V(0) := 2, \quad V(1) := \mu,$$

$$2155 \quad V(m) = \mu \cdot V(m-1) - 2 \cdot V(m-2).$$

2156 For the recommended Koblitz curves, the quantities $s_i(m)$ and $V(m)$ are as follows.

2157 Curve K-163:

$$2158 \quad s_0(163) = 2579386439110731650419537$$

$$2159 \quad s_1(163) = -755360064476226375461594$$

$$2160 \quad V(163) = -4845466632539410776804317$$

2161 Curve K-233:

2162 $s_0(233) = -27859711741434429761757834964435883$

2163 $s_1(233) = -44192136247082304936052160908934886$

2164 $V(233) = -137381546011108235394987299651366779$

2165 Curve K-283:

2166 $s_0(283) = -665981532109049041108795536001591469280025$

2167 $s_1(283) = 1155860054909136775192281072591609913945968$

2168 $V(283) = 7777244870872830999287791970962823977569917$

2169 Curve K-409:

2170 $s_0(409) = -18307510456002382137810317198756461378590542487556869338419259$

2171 $s_1(409) = -8893048526138304097196653241844212679626566100996606444816790$

2172 $V(409) = 10457288737315625927447685387048320737638796957687575791173829$

2173 Curve K-571:

2174 $s_0(571) = -3737319446876463692429385892476115567147293964596131024123406420\backslash$

2175 235241916729983261305

2176 $s_1(571) = -3191857706446416099583814595948959674131968912148564658610565117\backslash$

2177 58982848515832612248752

2178 $V(571) = -1483809269816914138996191402970514903645425741804939362329123395\backslash$

2179 34208516828973111459843

2180 The following algorithm computes the scalar multiple $Q := kP$ on the Koblitz curve $W_{a,b}$ over
 2181 $\text{GF}(2^m)$. The average number of elliptic additions and subtractions is at most $\sim 1 + (m/3)$ and is at
 2182 most $\sim m/3$ with probability at least $1 - 2^{5-C}$.

2183 1. For $i := 0$ to 1 do

2184 1.1 $k' \leftarrow \lfloor k / 2^{a-C+(m-9)/2} \rfloor$.

2185 1.2 $g' \leftarrow s_i(m) \cdot k'$.

2186 1.3 $h' \leftarrow \lfloor g' / 2^m \rfloor$.

2187 1.4 $j' \leftarrow V(m) \cdot h'$.

2188 1.5 $l' \leftarrow \text{Round}((g' + j') / 2^{(m+5)/2})$.

2189 1.6 $\lambda_i \leftarrow l' / 2^C$.

2190 1.7 $f_i \leftarrow \text{Round}(\lambda_i)$.

2191 1.8 $\eta_i \leftarrow \lambda_i - f_i$.

2192 1.9 $h_i \leftarrow 0$.

2193 2. $\eta \leftarrow 2 \eta_0 + \mu \eta_1$.

2194 3. If $(\eta \geq 1)$,

2195 then

2196 if $(\eta_0 - 3 \mu \eta_1 < -1)$

2197 then set $h_1 \leftarrow \mu$

2198 else set $h_0 \leftarrow 1$.

2199 else

2200 if $(\eta_0 + 4 \mu \eta_1 \geq 2)$

2201 then set $h_1 \leftarrow \mu$.

2202 4. If $(\eta < -1)$

2203 then

2204 if $(\eta_0 - 3 \mu \eta_1 \geq 1)$

2205 then set $h_1 \leftarrow -\mu$

2206 else set $h_0 \leftarrow -1$.

2207 else

2208 if $(\eta_0 + 4 \mu \eta_1 < -2)$

2209 then set $h_1 \leftarrow -\mu$.

2210 5. $q_0 \leftarrow f_0 + h_0$.

- 2211 6. $q_1 \leftarrow f_1 + h_1$.
- 2212 7. $r_0 \leftarrow n - (s_0 + \mu s_1) q_0 - 2s_1 q_1$.
- 2213 8. $r_1 \leftarrow s_1 q_0 - s_0 q_1$.
- 2214 9. Set $Q \leftarrow O$.
- 2215 10. $P_0 \leftarrow P$.
- 2216 11. While $((r_0 \neq 0) \text{ or } (r_1 \neq 0))$
- 2217 11.1 If $(r_0 \text{ odd})$, then
- 2218 11.1.1 set $u \leftarrow 2 - (r_0 - 2 r_1 \bmod 4)$.
- 2219 11.1.2 set $r_0 \leftarrow r_0 - u$.
- 2220 11.1.3 if $(u = 1)$, then set $Q \leftarrow Q + P_0$.
- 2221 11.1.4 if $(u = -1)$, then set $Q \leftarrow Q - P_0$.
- 2222 11.2 Set $P_0 \leftarrow \tau P_0$.
- 2223 11.3 Set $(r_0, r_1) \leftarrow (r_1 + \mu r_0 / 2, -r_0 / 2)$.
- 2224 Endwhile
- 2225 12. Output Q .

2226 **G.3 Polynomial and Normal Bases for Binary Fields**

2227 **G.3.1 Normal Bases**

2228 The elements of $\text{GF}(2^m)$, where m is odd, are expressed in terms of the type T normal² basis B for
2229 $\text{GF}(2^m)$, for some T . Each element has a unique representation as a bit string:

$$2230 \quad (a_0 a_1 \dots a_{m-1}).$$

2231 The arithmetic operations are performed as follows.

2232 Addition: Addition of two elements is implemented by bit-wise addition modulo 2. Thus, for
2233 example,

$$2234 \quad (1100111) + (1010010) = (0110101).$$

² It is assumed in this section that m is odd and T is even since this is the only case considered in this standard.

2235 Squaring: if

$$2236 \quad \alpha = (a_0 \ a_1 \ \dots \ a_{m-2} \ a_{m-1}),$$

2237 then

$$2238 \quad \alpha^2 = (a_{m-1} \ a_0 \ a_1 \ \dots \ a_{m-2}).$$

2239 Multiplication: Multiplication depends on the following function $F(\underline{u}, \underline{v})$ on inputs

$$2240 \quad \underline{u} = (u_0 \ u_1 \ \dots \ u_{m-1}) \quad \text{and} \quad \underline{v} = (v_0 \ v_1 \ \dots \ v_{m-1}),$$

2241 which is constructed s follows.

- 2242 1. Set $p = Tm + 1$;
- 2243 2. Let u be an integer having order T modulo p ;
- 2244 3. Compute the sequence $F(1), F(2), \dots, F(p-1)$ as follows:
 - 2245 a. Set $w = 1$;
 - 2246 b. For j from 0 to $T-1$ do
 - 2247 i. Set $n = w$;
 - 2248 ii. For $i = 0$ to $m-1$ do
 - 2249 1. Set $F(n) = i$;
 - 2250 2. Set $n = 2n \pmod{p}$;
 - 2251 1.2.3 Set $w = uw \pmod{p}$;
- 2252 2. Output the formulae $F(u, v)$, where

$$2253 \quad F(u, v) := \sum_{k=1}^{p-2} u_{F(k+1)} v_{F(p-k)}.$$

2254 This computation only needs to be performed once per basis.

2255 Given the function F for B , the product

$$2256 \quad (c_0 \ c_1 \ \dots \ c_{m-1}) = (a_0 \ a_1 \ \dots \ a_{m-1}) * (b_0 \ b_1 \ \dots \ b_{m-1})$$

2257 is computed as follows:

- 2258 1. Set $(u_0 \ u_1 \ \dots \ u_{m-1}) = (a_0 \ a_1 \ \dots \ a_{m-1})$;
- 2259 2. Set $(v_0 \ v_1 \ \dots \ v_{m-1}) = (b_0 \ b_1 \ \dots \ b_{m-1})$;
- 2260 3. For $k = 0$ to $m-1$ do
 - 2261 a. Compute $c_k = F(\underline{u}, \underline{v})$.
 - 2262 b. Set $u = \mathbf{LeftShift}(u)$ and $v := \mathbf{LeftShift}(v)$, where **LeftShift** denotes the circular
 - 2263 left shift operation.
- 2264 4. Output $c = (c_0 \ c_1 \ \dots \ c_{m-1})$.

2265 Example:

2266 For the type-4 normal basis for $\text{GF}(2^7)$, one has $p = 29$ and $u = 12$ or $u = 17$. Thus, the values of

2267 F are given by:

2268	$F(1) = 0$	$F(8) = 3$	$F(15) = 6$	$F(22) = 5$
2269	$F(2) = 1$	$F(9) = 3$	$F(16) = 4$	$F(23) = 6$
2270	$F(3) = 5$	$F(10) = 2$	$F(17) = 0$	$F(24) = 1$
2271	$F(4) = 2$	$F(11) = 4$	$F(18) = 4$	$F(25) = 2$
2272	$F(5) = 1$	$F(12) = 0$	$F(19) = 2$	$F(26) = 5$
2273	$F(6) = 6$	$F(13) = 4$	$F(20) = 3$	$F(27) = 1$
2274	$F(7) = 5$	$F(14) = 6$	$F(21) = 3$	$F(28) = 0$

2275

2276 Therefore,

$$2277 \quad F(\underline{u}, \underline{v}) = u_0 v_1 + u_1 (v_0 + v_2 + v_5 + v_6) + u_2 (v_1 + v_3 + v_4 + v_5) + u_3 (v_2 + v_5) +$$

$$2278 \quad u_4 (v_2 + v_6) + u_5 (v_1 + v_2 + v_3 + v_6) + u_6 (v_1 + v_4 + v_5 + v_6).$$

2279 As a result, if

$$2280 \quad a = (1\ 0\ 1\ 0\ 1\ 1\ 1) \text{ and } b = (1\ 1\ 0\ 0\ 0\ 0\ 1),$$

2281 then

$$2282 \quad c_0 = F((1\ 0\ 1\ 0\ 1\ 1\ 1), (1\ 1\ 0\ 0\ 0\ 0\ 1)) = 1,$$

$$2283 \quad c_1 = F((0\ 1\ 0\ 1\ 1\ 1\ 1), (1\ 0\ 0\ 0\ 0\ 1\ 1)) = 0,$$

$$2284 \quad \vdots$$

$$2285 \quad c_6 = F((1\ 1\ 0\ 1\ 0\ 1\ 1), (1\ 1\ 1\ 0\ 0\ 0\ 0)) = 1,$$

2286 so that $c = a * b = (1\ 0\ 1\ 1\ 0\ 0\ 1)$.2287 For the binary curves recommended in this specification, the values of T are, respectively, $T = 2$ 2288 ($m = 233$), $T = 6$ ($m = 283$), $T = 4$ ($m = 409$), and $T = 10$ ($m = 571$).

2289

2290 **G.3.2 Polynomial Basis to Normal Basis Conversion**2291 Let α be an element of the field $\text{GF}(2^m)$ with bit-string representation p with respect to a given2292 polynomial basis and bit-string representation n with respect to a given normal basis. The bit2293 strings p and n are related via

$$2294 \quad p \Gamma = n,$$

2295 where Γ is an $(m \times m)$ matrix with entries in $\text{GF}(2)$. The matrix Γ , which only depends on the

2296 bases, can be easily computed given its second-to-last row. For each conversion, that second-to-

2297 last row is given below.

2298 Degree 233:2299 `0x0be 19b89595 28bbc490 038f4bc4 da8bdfc1 ca36bb05 853fd0ed 0ae200ce`2300 Degree 283:

2301 0x3347f17 521fdabc 62ec1551 acf156fb 0bceb855 f174d4c1 7807511c 9f745382
2302 add53bc3

2303 Degree 409:

2304 0x0eb00f2 ea95fd6c 64024e7f 0b68b81f 5ff8a467 acc2b4c3 b9372843 6265c7ff
2305 a06d896c ae3a7e31 e295ec30 3eb9f769 de78bef5

2306 Degree 571:

2307 0x7940ffa ef996513 4d59dcbf e5bf239b e4fe4b41 05959c5d 4d942ffd 46ea35f3
2308 e3cdb0e1 04a2aa01 cef30a3a 49478011 196bfb43 c55091b6 1174d7c0 8d0cdd61
2309 3bf6748a bad972a4

2310 If r is the second-to-last row of Γ and represents the element β of $\text{GF}(2^m)$ with respect to the
2311 normal basis, then the rows of Γ , from top to bottom, are the bit-string representations of the
2312 elements

$$2313 \beta^{m-1}, \beta^{m-2}, \dots, \beta^2, \beta, 1$$

2314 with respect to this normal basis. (Note that the element 1 is represented by the all-1 bit string.)

2315 Alternatively, the matrix is the inverse of the matrix described in Appendix G.3.3.

2316 More details of these computations can be found in Annex A.7 of the IEEE Standard 1363-2000
2317 standard [[IEEE 1363](#)].

2318 **G.3.3 Normal Basis to Polynomial Basis Conversion**

2319 Let α be an element of the field $\text{GF}(2^m)$ with bit-string representation n with respect to a given
2320 normal basis and bit-string representation p with respect to a given polynomial basis. The bit
2321 strings p and n are related via

$$2322 n \Delta = p,$$

2323 where Δ is an $(m \times m)$ matrix with entries in $\text{GF}(2)$. The matrix Δ , which depends only on the
2324 bases, can be easily computed given its top row. For each conversion, that top row is given
2325 below.

2326 Degree 233:

2327 0x149 9e398ac5 d79e3685 59b35ca4 9bb7305d a6c0390b cf9e2300 253203c9

2328 Degree 283:

2329 0x31e0ed7 91c3282d c5624a72 0818049d 053e8c7a b8663792 bc1d792e ba9867fc
2330 7b317a99

2331 Degree 409:

2332 0x0dfa06b e206aa97 b7a41fff b9b0c55f 8f048062 fbe8381b 4248adf9 2912ccc8
2333 e3f91a24 e1cfb395 0532b988 971c2304 2e85708d

2334 Degree 571:

2335 0x452186b bf5840a0 bcf8c9f0 2a54efa0 4e813b43 c3d41496 06c4d27b 487bf107
2336 393c8907 f79d9778 beb35ee8 7467d328 8274caeb da6ce05a eb4ca5cf 3c3044bd
2337 4372232f 2c1a27c4

2338 If r is the top row of Δ and represents the element β of $\text{GF}(2^m)$, then the rows of Δ , from top to
2339 bottom, are the bit strings representing the elements

2340
$$\beta, \beta^2, \beta^{2^2}, \dots, \beta^{2^{m-1}}$$

2341 with respect to the polynomial basis. Alternatively, the matrix is the inverse of the matrix
2342 described in Appendix G.3.2.

2343 More details of these computations can be found in Annex A.7 of the IEEE Std 1363-2000
2344 standard.

2345 Appendix H – Other Allowed Elliptic Curves**2346 H.1 Brainpool Curves**

2347 This standard also allows the curves specified in *Elliptic Curve Cryptography (ECC) Brainpool*
2348 *Standard Curves and Curve Generation* [[RFC 5639](#)], which support a security strength of 112
2349 bits or higher. In particular, this includes brainpoolP224r1, brainpoolP256r1, brainpoolP320r1,
2350 brainpoolP384r1, and brainpoolP512r1. These curves were pseudorandomly generated and are
2351 allowed to be used for interoperability reasons.

2352