

Public-Key Cryptography with Arbitrary Finite Fields

Contribution to IEEE P1363a

Daniel V. Bailey and Christof Paar

bailey@wpi.edu, christof@wpi.edu

Worcester Polytechnic Institute

February 18, 2000

Abstract. This contribution proposes text for possible inclusion in IEEE P1363a specifying support for additional finite fields in the DL and EC settings. In particular, this contribution generalizes IEEE P1363 to support all finite fields. Like IEEE P1363a, it is written as updates to the IEEE P1363 document. It is intended for discussion and review at the March 16-17, 2000, IEEE P1363 working group meeting. The contribution has not yet been approved by the working group.

Copyright © 2000 by the Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street
New York, NY 10017, USA
All rights reserved.

This is a contribution to an unapproved draft of a proposed IEEE Standard, subject to change. Permission is hereby granted for IEEE Standards Committee participants to reproduce this document for purposes of IEEE standardization activities. If this document is to be submitted to ISO or IEC, notification shall be given to IEEE Copyright Administrator. Permission is also granted for member bodies and technical committees of ISO and IEC to reproduce this document for purposes of developing a national position. Other entities seeking permission to reproduce portions of this document for these or other uses must contact the IEEE Standards Department for the appropriate license. Use of information contained in the unapproved draft is at your own risk.

IEEE Standards Department
Copyright and Permissions
445 Hoes Lane, P. O. Box 1331
Piscataway, NJ 08855-1331, USA

Contents

3. DEFINITIONS (UPDATED)	3
5. MATHEMATICAL CONVENTIONS (UPDATED)	4
5.3.2.3 <i>Composite Basis over $GF(2^r)$ (new)</i>	4
5.3.2.3.1 <i>Polynomial Basis over $GF(2^r)$ (new)</i>	4
5.3.2.3.2 <i>Normal Basis over $GF(2^r)$ (new)</i>	4
5.3.3 <i>Odd Characteristic Extension Fields (new)</i>	5
5.5.6 <i>Converting between Odd Characteristic Extension Field Elements and Octet Strings (OCEFE2OSP and OS2OCEFEP) (new)</i>	6
6. PRIMITIVES BASED ON THE DISCRETE LOGARITHM PROBLEM (UPDATED)	7
7. PRIMITIVES BASED ON THE ELLIPTIC CURVE DISCRETE LOGARITHM PROBLEM (UPDATED)	8
ANNEX A (INFORMATIVE) NUMBER-THEORETIC BACKGROUND (UPDATED)	9
A.5.8 <i>Checking Binomials over $GF(q^m)$ for Irreducibility (new)</i>	9
A.10.1 <i>Full Addition and Subtraction ($p > 3$) (updated)</i>	10
A.10.9 <i>Full Addition and Subtraction ($p = 3$) (new)</i>	10
A.11.5 <i>Curve Orders over Extension Fields (updated)</i>	11
A.16.6 <i>Algorithms for Validating DL Public Keys (updated)</i>	11
A.16.10 <i>Algorithms for Validating EC Public Keys (updated)</i>	11
A.16.13 <i>An Algorithm for Validating DL Parameters (prime-power case) (new)</i>	12
A.17 ODD CHARACTERISTIC EXTENSION FIELDS (NEW)	12
A.17.1 <i>Basic Arithmetic in an Odd Characteristic Extension Field</i>	13
A.17.1.1 <i>Addition and Subtraction</i>	13
A.17.1.2 <i>Extension Field Multiplication</i>	13
A.17.2 <i>Optimal Extension Fields</i>	13
A.17.2.1 <i>Type I Optimal Extension Fields</i>	14
A.17.2.2 <i>Type II Optimal Extension Fields</i>	14
A.17.3 <i>Optimal Extension Fields: Algorithms</i>	14
A.17.3.1 <i>Subfield Multiplication</i>	14
A.17.3.2 <i>Extension Field Modular Reduction</i>	15
A.17.3.3 <i>Extension Field Inversion</i>	15
A.17.3.5 <i>Subfield Inversion</i>	16
A.17.3.6 <i>Construction</i>	16
A.17.3.8 <i>Table of Type I OEFs</i>	17
ANNEX F (INFORMATIVE) BIBLIOGRAPHY (UPDATED)	18

3. Definitions (updated)

binomial: a polynomial of the form $t^m - w$.

characteristic: the largest prime divisor of the integer p for the field $GF(p^m)$.

Frobenius map: the map $\mathbf{a} \rightarrow \mathbf{a}^p$, for $\mathbf{a} \in GF(p^m)$, for prime p .

Hamming weight: the number of 1s in the binary representation of an integer. Represented with the notation H_w .

Norm: the function $\text{Norm}(\mathbf{a}) = \mathbf{a} \times \mathbf{a}^q \times \dots \times \mathbf{a}^{q^{(m-1)}} = \mathbf{a}^r$, $r = (q^m - 1)/(q - 1)$, $\alpha \in GF(p^m)$.

pseudo-Mersenne prime: a positive rational integer of the form $2^n \pm c$, $\log_2 c \leq \lfloor n/2 \rfloor$

Optimal Extension Field: a finite field $GF(p^m)$ with p a pseudo-Mersenne prime and an irreducible binomial as the field polynomial.

5. Mathematical Conventions (updated)

5.3.2.3 Composite Basis over $GF(2^r)$ (new)

This representation is possible for fields $GF(2^m)$ if m is a composite integer. The representation is determined by choosing a subfield $GF(2^r)$ of $GF(2^m)$, where $1 < r < m$, and $m = rs$. Elements of $GF(2^m)$ are then represented as vectors of s elements from $GF(2^r)$ using any basis representation, including those in Section 5.3.2.3.1 or 5.3.2.3.2. Elements of $GF(2^r)$ may in turn be represented in any basis, including those in Section 5.3.2.3.1 or 5.3.2.3.2, or with another composite basis if r is a composite integer. This process of choosing representations for subfields of $GF(2^r)$ may be repeated for each of the divisors of r .

5.3.2.3.1 Polynomial Basis over $GF(2^r)$ (new)

This representation is similar to that defined in Section 5.3.2.1 with the difference that $r \geq 1$. This representation is determined by choosing an irreducible polynomial $p(t)$ over $GF(2^r)$ of degree $s = m/r$. Then $GF(2^m)$ is isomorphic to $GF(2^r)[t]/p(t)$.

Then if the polynomial basis representation over $GF(2^r)$ is used, for purposes of conversion, the string

$$(a_{s-1} \dots a_2 a_1 a_0)$$

where each a_i is a bit string of length r shall be taken to represent the polynomial

$$a_{s-1}t^{s-1} + \dots + a_2t^2 + a_1t + a_0$$

where the coefficients a_i are elements of $GF(2^r)$. The coefficients a_i will in turn be represented with some basis over some subfield of $GF(2^r)$.

As in Section 5.3.2.1, the additive identity (zero) element of the field $GF(2^m)$ is represented by a string of s representations of the zero element in the chosen representation for $GF(2^r)$. Thus, if $GF(2^r)$ is represented with a polynomial or normal basis, the zero element of $GF(2^m)$ is represented by a string of all zero bits. Similarly, the multiplicative identity (one) element of $GF(2^m)$ is represented by a string of $(s-1)$ representations of the zero element of $GF(2^r)$ followed by the representation of the one element of $GF(2^r)$. Thus, if $GF(2^r)$ is represented with a polynomial basis, the one element of $GF(2^m)$ is represented by a string of $m-1$ zero bits followed by a one bit. If $GF(2^r)$ is represented by a normal basis, the one element of $GF(2^m)$ is represented by a string of $(s-1)r$ zero bits followed by r one bits.

The arithmetic in this basis is identical to that of Section 5.3.2.1 with the exception that all coefficient operations are performed in the field $GF(2^r)$.

5.3.2.3.2 Normal Basis over $GF(2^r)$ (new)

This representation is similar to that defined in Section 5.3.2.2 with the difference that $r \geq 1$. This representation is determined by choosing a normal polynomial $p(t)$ of degree $s = m/r$. Then $GF(2^m)$ is isomorphic to $GF(2^r)[t]/p(t)$.

Then if the normal basis representation over $GF(2^r)$ is used, for purposes of conversion, the string

$$(a_0 a_1 \dots a_{s-1})$$

where each a_i is an bit string of length r shall be taken to represent the element

$$a_0\mathbf{q} + a_1\mathbf{q}^2 + a_2\mathbf{q}^{2^2} + \dots + a_{s-1}\mathbf{q}^{2^{s-1}},$$

where \mathbf{q} is a root of $p(t)$ in $GF(2^s)$. The coefficients a_i will in turn be represented with some basis over some subfield of $GF(2^r)$.

As in Section 5.3.2.2, the additive identity (zero) element of the field $GF(2^m)$ is represented by a string of s representations of zero in $GF(2^r)$. For example, if $GF(2^r)$ is represented with a polynomial or normal basis, a string of zero bits represents zero in $GF(2^m)$. The multiplicative identity (one) element of the field is represented by a string of s representations of one in $GF(2^r)$. For example, if $GF(2^r)$ is represented with a polynomial basis, the string consists of the pattern: $(r - 1)$ zero bits followed by a one bit. The pattern repeated s times represents one in $GF(2^m)$. If $GF(2^r)$ is represented with a normal basis, the one element of $GF(2^m)$ is represented with a string of all one bits.

The arithmetic in this basis is identical to that of Section 5.3.2.1 with the exception that all coefficient operations are performed in the field $GF(2^r)$.

5.3.3 Odd Characteristic Extension Fields (new)

An *odd characteristic extension field* is a finite field whose number of elements is a power of an odd prime. If $m \geq 1$, then there is a unique field $GF(p^m)$ with p^m elements. For purposes of conversion, the elements of $GF(p^m)$ shall be represented in a polynomial basis. This representation is determined by choosing an irreducible polynomial $p(t)$ over $GF(p)$. Then $GF(p^m)$ is isomorphic to $GF(p)[t]/p(t)$. This interpretation shall be the bit string formed by concatenating the values of the coefficients represented as integers. Thus the polynomial

$$a_{m-1}t^{m-1} + \dots + a_2t^2 + a_1t + a_0$$

is represented by the bit string

$$(a_{m-1} \dots a_2 a_1 a_0)$$

where each of the a_i are positive integers less than p , padded with leading 0 bits so that each a_i is represented with $\lceil \log_2 p \rceil$ bits.

A description of the arithmetic of $GF(p^m)$ is given in A.17.

Update Section 5.4 of IEEE P1363 as follows:

Add the following before the final paragraph:

If q is a power of 3, then b shall be nonzero in $GF(q)$, and every point $P = (x_P, y_P)$ on E (other than the point O) shall satisfy the following equation in $GF(q)$:

$$y_P^2 = x_P^3 + a x_P^2 + b.$$

Update Section 5.5.4 of IEEE P1363 as follows:

Change the first paragraph to read:

An element x of a finite field $GF(q)$, for the purposes of this standard, is represented by an integer if q is an odd prime (see Section 5.3.1); by a bit string if q is a power of 2 (see Section 5.3.2); and a vector of coefficients if q is an odd prime-power (see Section 5.3.3). If q is an odd prime, then to represent x as an octet string, I2OSP shall be used with the integer value of x and the length $\lceil \log_{256} q \rceil$ as inputs. If q is a power of 2, then to represent x as an octet string, BS2OSP shall be applied to the bit string representing x . If q is an odd prime-power, OCFE2OSP shall be used with the element x , p , and m where $q = p^m$.

Change the second sentence of the second paragraph to read:

It takes a field element x , the field size q , and both the field characteristic p and the extension degree m if q is an odd prime-power as inputs and outputs the corresponding octet string.

Add the following after the second sentence of the third paragraph:

If q is an odd prime-power, OS2OCFEP shall be used with the octet string, p , and m where $q = p^m$.

Update Section 5.5 of IEEE P1363 as follows:

Add the following new section:

5.5.6 Converting between Odd Characteristic Extension Field Elements and Octet Strings (OCFE2OSP and OS2OCFEP) (new)

To represent an element of $GF(p^m)$, $p > 2$, $m > 1$, as an octet string, represent the polynomial

$$a_{m-1}t^{m-1} + \dots + a_2t^2 + a_1t + a_0$$

by the integer

$$a_{m-1}p^{m-1} + \dots + a_2p^2 + a_1p + a_0.$$

The resulting octet string is obtained from this integer using the I2OSP method in Section 5.5.3.

6. Primitives Based on the Discrete Logarithm Problem (updated)

Update Section 6.1.2 of IEEE P1363 as follows:

Replace the second sentence with the following:

A set of DL domain parameters specifies a field $GF(q)$, where q is a positive odd prime integer p , 2^m for some positive integer m , or p^m for odd p and positive integer m ; a positive prime integer r dividing $(q - 1)$; and a field element g of multiplicative order r (g is called the *generator* of the subgroup of order r).

7. Primitives Based on the Elliptic Curve Discrete Logarithm Problem (updated)

Update Section 7.1.2 of IEEE P1363 as follows:

Replace the second sentence with the following:

A set of EC domain parameters specifies a field $GF(q)$, where q is a positive odd prime integer p , 2^m for some positive integer m or p^m for odd p and some positive integer m ; two elliptic curve coefficients a and b , elements of $GF(q)$, that define an elliptic curve E ; a positive prime integer r dividing the number of points on E ; and a curve point G of order r (G is called the *generator* of a subgroup of order r).

Annex A (Informative) Number-Theoretic Background (updated)

Update Section A.3.1 of IEEE P1363 as indicated below:

Add a new item at the bottom of the bulleted list:

— When $q = p^m$ for some $m > 1$ and some odd prime p , the field $GF(p^m)$ is called an *odd characteristic extension field*. The field $GF(p^m)$ is typically represented by polynomials modulo an irreducible field polynomial.

Update Section A.5 of IEEE P1363 as indicated below:

Add a new subsection:

A.5.8 Checking Binomials over $GF(q^m)$ for Irreducibility (new)

Let $f(t)$ be a polynomial with coefficients in the field $GF(q)$ and let $f(t)$ have the form $t^m - w$ for $w \in GF(q)^*$ and $m > 1$. Let w have order e in $GF(q)^*$. Then $f(t)$ is irreducible if and only if [LN94]:

1. Each prime factor of m divides e , but not $(q - 1)/e$
2. If $m \equiv 0 \pmod{4}$, then $q \equiv 1 \pmod{4}$.

These conditions imply that m divides $(q - 1)$. In addition, they imply that no irreducible binomials exist over $GF(2)$.

Update Section A.9 of IEEE P1363 as indicated below:

Change the first bulleted item to read:

- For the finite fields $GF(p^m)$ with $p > 3$, $m \geq 1$, the Weierstrass equation is

Add the following bulleted item:

- For the finite fields $GF(p^m)$ with $p = 3$, $m \geq 1$, the Weierstrass equation is [Kob98]

$$y^2 = x^3 + ax^2 + b$$

Update Section A.9.2 of IEEE P1363 as indicated below:

Update the equation in the second paragraph to read:

$$-P = \begin{cases} (x, -y) & \text{if } q = p^m, p \neq 2 \\ (x, x + y) & \text{if } q = 2^m. \end{cases}$$

Update Section A.10.1 of IEEE P1363 as indicated below:

Change the A.10.1 heading to read:

A.10.1 Full Addition and Subtraction ($p > 3$) (updated)

Change the first line to read:

The following algorithm implements a full addition (on a curve over $GF(p^m)$, $p > 3$) in terms of affine coordinates.

Change the Input: line of the algorithm to read:

Input: prime $p > 3$, $m \geq 1$, where the field is $GF(p^m)$; coefficients a, b for an elliptic curve $E: y^2 = x^3 + ax + b$ over $GF(p^m)$; points $P_0 = (x_0, y_0)$ and $P_1 = (x_1, y_1)$ on E .

Change line 3.1 of the algorithm to read:

3.1 set $I \leftarrow (y_0 - y_1) / (x_0 - x_1)$, $I \in GF(p^m)^*$

Change the second to last line to read:

The above algorithm requires 3 or 4 multiplications and an inversion in $GF(p^m)$.

Update Section A.10 of IEEE P1363 by adding the following:

A.10.9 Full Addition and Subtraction ($p = 3$) (new)

The following algorithm implements a full addition (on a curve over $GF(3^m)$) in terms of affine coordinates. All arithmetic operations are performed in the field.

Input: a field $GF(3^m)$; coefficients a, b for an elliptic curve $E: y^2 = x^3 + ax^2 + b$ over $GF(3^m)$; points $P_0 = (x_0, y_0)$ and $P_1 = (x_1, y_1)$ on E .

Output: the point $P_2 := P_0 + P_1$.

1. If $P_0 = \mathcal{O}$ then output $P_2 \leftarrow P_1$ and stop
2. If $P_1 = \mathcal{O}$ then output $P_2 \leftarrow P_0$ and stop
3. If $x_0 \neq x_1$ then
 - 3.1 set $I \leftarrow (y_0 - y_1) / (x_0 - x_1)$, $I \in GF(3^m)^*$
 - 3.2 go to step 8
4. If $y_0 \neq y_1$ then output $P_2 \leftarrow \mathcal{O}$ and stop
5. If $y_1 = 0$ then output $P_2 \leftarrow \mathcal{O}$ and stop
6. Set $I \leftarrow (ax_0 - b) / y_0$
7.
 - 7.1 set $x_2 \leftarrow I^2 - a - 2x_0$
 - 7.2 go to step 9
8. Set $x_2 \leftarrow I^2 - x_0 - x_1 - a$
9. Set $y_2 \leftarrow (x_1 - x_2) I - y_0$
10. Output $P_2 \leftarrow (x_2, y_2)$

The above algorithm requires 3 or 4 field multiplications and a field inversion.

To subtract the point $P = (x, y)$, one adds the point $-P = (x, -y)$.

Replace A.11.5 with the following:

A.11.5 Curve Orders over Extension Fields (updated)

Given the order of an elliptic curve E over a finite field $GF(p^d)$, the following algorithm computes the order of E over the extension field $GF(p^{de})$.

Input: a prime $p \geq 2$, positive integers d and e , an elliptic curve E defined over $GF(p^d)$, and the order w of E over $GF(p^d)$.

Output: the order u of E over $GF(p^{de})$.

1. Set $P \leftarrow p^d + 1 - w$ and $Q \leftarrow p^d$.
2. Compute via A.2.4 the Lucas sequence element V_e .
3. Compute $u := p^{de} + 1 - V_e$.
4. Output u .

Replace A.16.6 of IEEE P1363 with the following:

A.16.6 Algorithms for Validating DL Public Keys (updated)

The following algorithm verifies if a DL public key is valid.

Input: valid DL parameters q , r and g ; the public key w .

Output: “True” or “False.”

1. If $q = p$ is an odd prime, check that w is an integer such that $1 < w < p$.
2. If $q = p^m$ for $m > 1$, $p \geq 2$, check that w is an element of $GF(p^m)$ and that $w \neq 0$ and $w \neq 1$ in $GF(p^m)$.
3. Check that $w^r = 1$ in $GF(q)$.
4. Output “True” if the checks work, and “False” otherwise.

The following algorithm does not verify the validity of a DL public key, but merely checks if it is in the multiplicative group of $GF(q)$. It may be used in conjunction with DLSVDP-DHC and DLSVDP-MQVC primitives.

Input: valid DL parameters q , r and g ; the public key w .

Output: “True” or “False.”

1. If $q = p$ is an odd prime, check that w is an integer such that $0 < w < p$.
2. If $q = p^m$ for $m > 1$, $p \geq 2$, check that w is an element of $GF(p^m)$ and that $w \neq 0$ in $GF(p^m)$.
3. Output “True” if the checks work, and “False” otherwise.

Update A.16.10 of IEEE P1363 as indicated below:

A.16.10 Algorithms for Validating EC Public Keys (updated)

The following algorithm verifies if an EC public key is valid.

Input: valid EC parameters q, a, b, r, G and k such that r does not divide k ; a public key W .

Output: “True” or “False.”

1. Check that $W \neq \mathcal{O}$. Let $W = (x, y)$.
2. If $q = p^m, p > 3$, check that x and y are elements of $GF(p^m)$ and $y^2 = x^3 + ax + b \in GF(p^m)$.
3. If $q = 2^m$, check that x and y are elements of $GF(2^m)$ and that $y^2 + xy = x^3 + ax^2 + b \in GF(2^m)$.
4. If $q = 3^m$, check that x and y are elements of $GF(3^m)$ and that $y^2 = x^3 + ax^2 + b \in GF(3^m)$.
5. Check that $rW = \mathcal{O}$.
6. Output “True” if the checks work, and “False” otherwise.

The following algorithm does not verify the validity of an EC public key, but merely checks if it is a non-identity point on the elliptic curve specified by the parameters. It may be used in conjunction with ECSVDP-DHC and ECSVDP-MQVC primitives.

Input: valid EC parameters q, a, b, r , and G ; a public key W .

Output: “True” or “False.”

1. Check that $W \neq \mathcal{O}$. Let $W = (x, y)$
2. If $q = p^m, p > 3$, check that x and y are elements of $GF(p^m)$ and that $y^2 \equiv x^3 + ax + b \in GF(p^m)$.
3. If $q = 2^m$, check that x and y are elements of $GF(2^m)$ and that $y^2 + xy = x^3 + ax^2 + b \in GF(2^m)$.
4. If $q = 3^m$, check that x and y are elements of $GF(3^m)$ and that $y^2 = x^3 + ax^2 + b \in GF(3^m)$.
5. Output “True” if the checks work, and “False” otherwise.

Update Section A.16 of IEEE P1363 by adding the following new section:

A.16.13 An Algorithm for Validating DL Parameters (prime-power case) (new)

Input: DL parameters p, m , for the field $GF(p^m)$; the field polynomial $p(t)$; r, g ; the cofactor k (optional); and whether or not the parameters will be used for DLSVDP-DHC or DLSVDP-MQVC.

Output: “True” or “False.”

1. Check that p is an odd integer and $p > 2$. Check primality of p via A.15.3.
2. Check that r is an odd integer and $r > 2$. Check primality of r via A.15.3.
3. Check that g is an element of $GF(p^m)$ and that $g \neq 0$ and $g \neq 1$ in $GF(p^m)$.
4. Check that $g^r \equiv 1 \pmod{p(t)}$.
5. Check that $r \mid (p^m - 1)$.
6. Check that m is a positive integer and that $p(t)$ is of degree m . Check that $p(t)$ is irreducible.
7. If k is supplied, check that k is an integer such that $kr = (p^m - 1)$.
8. If the parameters will be used for DLSVDP-DHC or DLSVDP-MQVC, check that r does not divide k (if k is not supplied, first set $k \leftarrow (p - 1)/r$).
9. Output “True” if all checks work, and “False” otherwise.

A.17 Odd Characteristic Extension Fields (new)

As defined in A.3.1, a *finite field* (or *Galois field*) is a set with finitely many elements in which the usual algebraic operations (addition, subtraction, multiplication, division by nonzero elements) are possible, and in which the usual algebraic laws (commutative, associative, distributive) hold. The *order* of a finite field

is the number of elements it contains. A finite field is identified with the notation $GF(p^m)$ for a prime p and positive integer m . When $m > 1$, we represent field elements as polynomials with coefficients from $GF(p)$. As in Section 5.3.2.3, if m is composite, the field elements may be represented with a composite basis. In this case, a subfield $GF(p^r)$ of $GF(p^m)$ is chosen where $1 < r < m$ and $r \mid m$. Elements of $GF(p^m)$ are then represented as vectors of elements from $GF(p^r)$, as described in [AHK99]. Arithmetic is performed exactly as described below, except that arithmetic in $GF(p^m)$ is implemented using operations from $GF(p^r)$. This process of choosing representations for subfields of $GF(p^m)$ may be repeated for each of the divisors of r .

A.17.1 Basic Arithmetic in an Odd Characteristic Extension Field

This section describes the basic method for arithmetic in fields $GF(p^m)$, for an odd prime p , $m > 1$, of which an Optimal Extension Field (OEF) is a special case. The material of this section is described in [BP98].

An extension field $GF(p^m)$ is isomorphic to $GF(p)[t]/(p(t))$, where $p(t)$ is a monic irreducible polynomial of degree m over $GF(p)$. In the following, a residue class will be identified with the polynomial of least degree in this class. We consider a polynomial basis representation of a field element $A(t) \in GF(p^m)$:

$$A(t) = a_{m-1} t^{m-1} + \dots + a_1 t + a_0, a_i \in GF(p)$$

All arithmetic operations are performed modulo the field polynomial. The choice of field polynomial determines the complexity of the modular reduction.

A.17.1.1 Addition and Subtraction

Addition and subtraction of two field elements is implemented in a straightforward manner by adding or subtracting the coefficients of their polynomial representation and if necessary, performing a reduction modulo p by subtracting or adding p once from the intermediate result.

A.17.1.2 Extension Field Multiplication

Field multiplication can be performed in two stages. First, we perform an ordinary polynomial multiplication of two field elements $A(t)$ and $B(t)$, resulting in an intermediate product $C'(t)$ of degree less than or equal to $2m-2$:

$$C'(t) = A(t) \times B(t) = c_{2m-2}' t^{2m-2} + \dots + c_1' t + c_0', c_i' \in GF(p).$$

The schoolbook method to calculate the coefficients c_i' , $i = 0, 1, \dots, 2m-2$, requires m^2 multiplications and $(m-1)^2$ additions in the subfield $GF(p)$.

In Section A.17.3.2 we present an efficient method to calculate the residue $C(t) \equiv C'(t) \pmod{p(t)}$, $C(t) \in GF(p^m)$.

Squaring can be considered a special case of multiplication. The only difference is that the number of coefficient multiplications can be reduced to $m(m+1)/2$.

In order to perform coefficient multiplications, we must multiply in the subfield. A method for fast subfield multiplication is found in A.17.3.1. Certain choices for p can result in additional computational savings in the subfield multiplication.

A.17.2 Optimal Extension Fields

Given the foregoing, we observe that performance improvement can result from the selection of particular finite fields in which the algorithms for extension field arithmetic have an especially efficient implementation.

In the following, we define a class of finite field, which we call an *Optimal Extension Field* (OEF).

An *Optimal Extension Field* is a finite field $GF(p^m)$ such that:

1. p is a pseudo-Mersenne prime,
2. An irreducible binomial $P(t) = t^m - w$ exists over $GF(t)$.

Section A.5.8 gives an algorithm to check if a binomial is irreducible.

We observe that there are two special cases of OEF which yield additional arithmetic advantages, which we call Type I and Type II.

A.17.2.1 Type I Optimal Extension Fields

A *Type I Optimal Extension Field* has $p = 2^n \pm 1$.

A Type I OEF allows for subfield modular reduction with very low complexity. An OEF may be of both Type I and Type II.

A.17.2.2 Type II Optimal Extension Fields

A *Type II Optimal Extension Field* has an irreducible binomial $p(t) = t^m - 2$.

A Type II OEF allows for a reduction in the complexity of extension field modular reduction since the multiplications by w in Algorithm A.17.3.2 can be implemented using shifts instead of explicit multiplications. An OEF may be of both Type I and Type II.

A.17.3 Optimal Extension Fields: Algorithms

A.17.3.1 Subfield Multiplication

The following algorithm implements subfield multiplication in an OEF.

Input: a prime $p = 2^n \pm c$, $\log_2 c \leq \lfloor n/2 \rfloor$, integers $0 \leq a, b < p$.

Output: the integer $r \equiv ab \pmod{p}$

1. Set $x \leftarrow ab$
2. Set $q_0 \leftarrow x \gg n$
3. Set $r_0 \leftarrow x - (q_0 \ll n)$
4. Set $r \leftarrow r_0$
5. Set $i \leftarrow 0$
6. While $q_i > 0$
 - 6.1 $q_{i+1} \leftarrow q_i c \gg n$
 - 6.2 $r_{i+1} \leftarrow q_i c - (q_{i+1} \ll n)$
 - 6.3 $i \leftarrow i + 1$
 - 6.4 $r \leftarrow r + r_i$

7. While ($r \geq p$)
 - 7.1 $r \leftarrow r - p$

The above algorithm requires a maximum of two iterations of the first while loop, so we require at the most two multiplications by c .

If $c = 1$, this algorithm executes the first while loop only once and multiplication by c is an identity map.

A.17.3.2 Extension Field Modular Reduction

The following algorithm performs extension field modular reduction in the field $GF(p^m)$ modulo an irreducible binomial.

Input: a polynomial $C'(t)$, of degree up to $(2m - 2)$, an integer w such that $P(t) = t^m - w$ is an irreducible binomial over $GF(p)$.

Output: a polynomial $C(t) \equiv C'(t) \pmod{P(t)}$, where $C(t)$ is of degree less than m .

1. Let $C(t) = c_{m-1}t^{m-1} + \dots + c_0$ and $C'(t) = c_{2m-2}'t^{2m-2} + \dots + c_0'$
2. Set $c_{m-1} \leftarrow c_{m-1}'$
3. For i from $m - 2$ downto 0 , j from $2m - 2$ downto m
 - 2.1 Set $c_i \leftarrow wc_j' + c_i'$

The above algorithm requires a maximum of $(m - 1)$ subfield multiplications by w . If $w = 2^i$, these multiplications may be implemented as bitwise shifts.

A.17.3.3 Extension Field Inversion

This algorithm [ITT86, BP00] computes the multiplicative inverse \mathbf{b}^{-1} of an element \mathbf{b} such that $\mathbf{b}\mathbf{b}^{-1} \equiv 1 \in GF(p^m)$. An analogous algorithm for fields $GF(2^m)$ is in Section A.4.4.

Input: a field $GF(p^m)$ and a nonzero field element \mathbf{b} .

Output: the reciprocal \mathbf{b}^{-1}

1. Set $r \leftarrow (p^m - 1) / (p - 1)$
2. Set $\mathbf{b} \leftarrow \mathbf{b}^{r-1}$
3. Set $\mathbf{c} \leftarrow \mathbf{b}\mathbf{b}$
4. Set $\mathbf{c} \leftarrow \mathbf{c}^{-1}$
5. Set $\mathbf{b}^{-1} \leftarrow \mathbf{b}\mathbf{c}$

The above algorithm requires an exponentiation to the r -th power and an inversion in $GF(p)$, since $\mathbf{c} = \mathbf{b}^r = \text{Norm}(\mathbf{b}) \in GF(p)$.

To quickly perform the exponentiation, we observe the following power series representation for r :

$$r = p^{m-1} + p^{m-2} + \dots + p + 1$$

Thus, we have the p -adic representation $(r - 1) = (11\dots10)_p$. Exponentiation, then, may be performed by repeatedly multiplying and taking p -th powers, in an analogous manner to the algorithm in A.5.1. Since $(r - 1)$ will be fixed for a given field, we can use an addition chain to further reduce the computations required. Using such an addition chain constructed from the p -adic representation of $(r - 1)$ requires:

$(\lfloor \log_2(m-1) \rfloor + H_w(m-1) - 1)$ general multiplications + $(\lfloor \log_2(m-1) \rfloor + H_w(m-1))$ exponentiations

Each exponentiation is to a p^i -th power and is thus an iterate of the Frobenius map. Since an OEF has an irreducible binomial as the field polynomial, we have the following algorithm for each exponentiation.

The following algorithm computes \mathbf{b}^{p^i} in an OEF.

Input: An OEF $GF(p^m)$, $\mathbf{b} = \sum \mathbf{b}_j t^j \in GF(p^m)$, an integer i

Output: \mathbf{b}^{p^i}

1. For j from m down to 1
 - 1.1 Set $\mathbf{b}_j \leftarrow \mathbf{b}_j t^{jp^i}$

Since the t^{jp^i} values will be fixed for a particular field, they may be precomputed and stored. Thus, this algorithm requires $(m-1)$ subfield multiplications by fixed constants.

A.17.3.5 Subfield Inversion

To compute the subfield inverse required in the algorithm in 17.3.3, one may use the algorithm in A.2.2.

A.17.3.6 Construction

This algorithm finds an irreducible binomial to construct an Optimal Extension Field, given an approximate subfield order and extension degree.

Input: an integer n , where $2^n \pm c$ will be the characteristic of the field; a positive integer m , the extension degree.

Output: \mathbf{w} where $t^m - \mathbf{w}$ is an irreducible binomial over $GF(p)$.

1. Set $c \leftarrow 1$
2. While $\log_2 c \leq \lfloor n/2 \rfloor$
 - 2.1 Set $p \leftarrow 2^n \pm c$
 - 2.2 If p is prime and $m \mid (p-1)$
 - 2.2.1 Set $\mathbf{w} \leftarrow 2$
 - 2.2.2 While $\mathbf{w} < p$
 - 2.2.3.1 If $t^m - \mathbf{w}$ is irreducible by Algorithm A.5.8, output ω
 - 2.3 Set $c \leftarrow c + 2$

A.17.3.8 Table of Type I OEFs

The following table provides Type I OEFs with $2^{160} \leq p^m \leq 2^{256}$.

<i>n</i>	<i>c</i>	<i>m</i>	<i>mn</i>	<i>w</i>
7	-1	27	189	3
8	1	32	256	2
13	-1	13	169	2
13	-1	14	182	17
13	-1	15	195	17
13	-1	18	234	17
16	1	16	256	2
17	-1	10	170	3
17	-1	15	255	3
19	-1	9	171	3
31	-1	6	186	7
31	-1	7	217	7
61	-1	3	183	5

Annex F (Informative) Bibliography (updated)

Add the following new references:

[AHK99] Kazumaro Aoki, Fumitaka Hoshino, Tetsutaro Kobayashi, “OEF Using a Successive Extension,” presented at the rump session of *CRYPTO '99*.

[BP98] D. V. Bailey and C. Paar, “Optimal Extension Fields for Fast Arithmetic in Public-Key Algorithms,” *Advances in Cryptography – Crypto '98, Lecture Notes in Computer Science* (1998), Springer-Verlag, Berlin.

[BP00] D. V. Bailey and C. Paar, “Efficient Arithmetic in Finite Field Extensions with Application in Elliptic Curve Cryptography,” submitted to *Journal of Cryptology*, 1999.

[Kob98] N. Koblitz, *Algebraic Aspects of Cryptography*, Springer-Verlag, 1998.

[LN94] R. Lidl and H. Niederreiter, *Introduction to Finite Fields and their Applications*, Cambridge University Press, 1994.