



IEEE Standard Specification for Public Key Cryptographic Techniques Based on Hard Problems over Lattices

IEEE Computer Society

Sponsored by the
Microprocessor Standards Committee

1363.1TM

IEEE
3 Park Avenue
New York, NY 10016-5997, USA
10 March 2009

IEEE Std 1363.1TM-2008

IEEE Standard Specification for Public Key Cryptographic Techniques Based on Hard Problems over Lattices

Sponsor
Microprocessors and Microcomputers Committee
of the
IEEE Computer Society

Approved 10 December 2008
IEEE-SA Standards Board

Abstract: Specifications of common public key cryptographic techniques based on hard problems over lattices supplemental to those considered in IEEE Std 1363-2000 and IEEE Std 1363a-2004, including mathematical primitives for secret value (key) derivation, public key encryption, identification and digital signatures, and cryptographic schemes based on those primitives are provided. Also presented are specifications of related cryptographic parameters, public keys, and private keys. Class of computer and communications systems is not restricted.

Keywords: encryption, lattice-based cryptography, public key cryptography

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

Copyright © 2009 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 10 March 2009. Printed in the United States of America.

IEEE is a registered trademark in the U.S. Patent & Trademark Office, owned by the Institute of Electrical and Electronics Engineers, Incorporated.

PDF: ISBN 978-0-7381-5863-1 STD95858
Print: ISBN 978-0-7381-5864-8 STDPD95858

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information contained in its standards.

Use of an IEEE Standard is wholly voluntary. The IEEE disclaims liability for any personal injury, property or other damage, of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, or reliance upon this, or any other IEEE Standard document.

The IEEE does not warrant or represent the accuracy or content of the material contained herein, and expressly disclaims any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained herein is free from patent infringement. IEEE Standards documents are supplied “**AS IS**.”

The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

In publishing and making this document available, the IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Nor is the IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEEE Standards document, should rely upon the advice of a competent professional in determining the exercise of reasonable care in any given circumstances.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position, explanation, or interpretation of the IEEE.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
Piscataway, NJ 08854
USA

Authorization to photocopy portions of any individual standard for internal or personal use is granted by the Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Introduction

This introduction is not part of IEEE Std 1363.1-2008, IEEE Standard Specification for Public Key Cryptographic Techniques Based on Hard Problems over Lattices.

The IEEE P1363™ project started as the “Standard for Rivest-Shamir-Adleman, Diffie-Hellman, and Related Public Key Cryptography” with its first meeting in January 1994, following a strategic initiative by the Microprocessor Standards Committee to develop standards for cryptography. Over the next eight years, the working group produced a broad standard reflecting the state of the art in public key cryptography, including techniques from three major families of hard problems. In addition, the working group drafted an addendum that provides additional techniques from those three major families. A more thorough history of the IEEE P1363 working group and its contributions beyond IEEE Std 1363™-2000 are given in the Introduction to IEEE Std 1363-2000.

At the same time, new cryptographic research was producing additional families of cryptographic techniques such as the family of techniques based on hard problems over lattices. These techniques enjoy operating characteristics that make them attractive in certain implementation environments, and thus they have received considerable scrutiny and deployment.

As a result, the working group proposed a new project to standardize techniques from this family. This project was approved by the Microprocessors and Microcomputers Standards Committee, and this current standard is the result of this project.

Notice to users

Laws and regulations

Users of these documents should consult all applicable laws and regulations. Compliance with the provisions of this standard does not imply compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Copyrights

This document is copyrighted by the IEEE. It is made available for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making this document available for use and adoption by public authorities and private users, the IEEE does not waive any rights in copyright to this document.

Updating of IEEE documents

Users of IEEE standards should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect. In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the IEEE Standards Association web site at <http://ieeexplore.ieee.org/xpl/standards.jsp>, or contact the IEEE at the address listed previously.

For more information about the IEEE Standards Association or the IEEE standards development process, visit the IEEE-SA web site at <http://standards.ieee.org>.

Errata

Errata, if any, for this and all other standards can be accessed at the following URL: <http://standards.ieee.org/reading/ieee/updates/errata/index.html>. Users are encouraged to check this URL for errata periodically.

Interpretations

Current interpretations can be accessed at the following URL: <http://standards.ieee.org/reading/ieee/interp/index.html>.

Patents

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

Participants

At the time this standard was submitted to the IEEE-SA Standards Board for approval, the P1363 Working Group had the following membership:

William Whyte, *Chair*
Don Johnson, *Vice Chair*

Matthew Ball
Xavier Boyen
Mike Brenner
Daniel Brown
Mark Chimley

Andy Dancer
David Jablon
Satoru Kanno
David Kravitz
Michael Markowitz
Luther Martin

Jim Randall
Roger Schlafly
Ari Singer
Terence Spies
Yongge Wang

The following members of the individual balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Ed Addario
Butch Anton
Matthew Ball
H. Stephen Berger
Martin J. Bishop
Juan Carreon
Keith Chow
Kevin Coggins
Geoffrey Darnton
James Davis
Thomas Dineen
Andrew Fieldsend

Michael Geipel
Randall Groves
Werner Hoelzl
Atsushi Ito
Mark Jaeger
Susan Land
David J. Leciston
Daniel Lindberg
Edward McCall
Avygdor Moise
Michael S. Newman
Ulrich Pohl

Robert Robinson
Randall Safier
Bartien Sayogo
Thomas Starai
Walter Struppler
Gerald Stueve
Mark Sturza
Vincent Tume
William Whyte
Paul Work
Oren Yuen
Wenhao Zhu

When the IEEE-SA Standards Board approved this standard on 10 December 2008, it had the following membership:

Robert M. Grow, *Chair*
Thomas Prevost, *Vice Chair*
Steve M. Mills, *Past Chair*
Judith Gorman, *Secretary*

Victor Berman
Richard DeBlasio
Andy Drozd
Mark Epstein
Alexander Gelman
William R. Goldbach
Arnold M. Greenspan
Kenneth S. Hanus

Jim Hughes
Richard H. Hulett
Young Kyun Kim
Joseph Koepfinger*
John Kulick
David J. Law
Glenn Parsons
Ronald C. Petersen

Chuck Powers
Narayanan Ramachandran
Jon Walter Rosdahl
Robby Robson
Anne-Marie Sahazizia
Malcolm V. Thaden
Howard L. Wolfman
Don Wright

*Member Emeritus

Also included are the following nonvoting IEEE-SA Standards Board liaisons:

Satish K. Aggarwal, *NRC Representative*
Michael Janezic, *NIST Representative*

Don Messina
IEEE Standards Program Manager, Document Development

Malia Zaman
IEEE Standards Program Manager, Technical Program Development

Contents

1. Overview	1
1.1 Scope	1
1.2 Purpose	1
2. Normative references.....	2
3. Definitions, acronyms, and abbreviations	2
3.1 Definitions	2
3.2 Acronyms and abbreviations	9
4. Types of cryptographic techniques.....	11
4.1 General model.....	11
4.2 Schemes.....	11
4.3 Additional methods.....	12
4.4 Algorithm specification conventions	12
5. Mathematical notation	13
6. Polynomial representation and operations.....	15
6.1 Introduction	15
6.2 Polynomial representation	15
6.3 Polynomial operations	15
6.3.1 Polynomial multiplication.....	15
6.3.2 Reduction of a polynomial mod q	15
6.3.3 Inversion in $(\mathbb{Z}/q\mathbb{Z})[X]/(X^N - 1)$	15
7. Data types and conversions	18
7.1 Bit strings and octet strings.....	18
7.2 Converting between integers and bit strings (I2BSP and BS2IP).....	18
7.2.1 Integer to bit string primitive (I2BSP)	18
7.2.2 Bit string to integer primitive (BS2IP).....	19
7.3 Converting between integers and octet strings (I2OSP and OS2IP).....	19
7.3.1 Integer to octet string primitive (I2OSP).....	19
7.3.2 Octet string to integer primitive (OS2IP).....	19
7.4 Converting between bit strings and right-padded octet strings (BS2ROSP and ROS2BSP)	20
7.4.1 Bit string to right-padded octet string primitive (BS2ROSP)	20
7.4.2 Right-padded octet string to bit string primitive (ROS2BSP).....	20
7.5 Converting between ring elements and bit strings (RE2BSP and BS2REP)	21
7.5.1 Ring element to bit string primitive (RE2BSP).....	21
7.5.2 Bit string to ring element primitive (BS2REP)	21
7.6 Converting between ring elements and octet strings (RE2OSP and OS2REP)	22
7.6.1 Ring element to octet string primitive (RE2OSP).....	22
7.6.2 Octet string to ring element primitive (OS2REP)	22
8. Supporting algorithms	22
8.1 Overview	22
8.2 Hash functions	23
8.3 Encoding methods	23
8.3.1 General.....	23
8.3.2 Blinding polynomial generation methods (BPGM)	23
8.4 Supporting algorithms	24
8.4.1 Mask generation functions.....	24
8.4.2 Index generation function	25
9. Short vector encryption scheme (SVES).....	28
9.1 Encryption scheme (SVES) overview	28
9.2 Encryption scheme (SVES) operations.....	28
9.2.1 Key generation	28
9.2.2 Encryption operation.....	29
9.2.3 Decryption operation.....	31
9.2.4 Key pair validation methods	33
9.2.5 Public key validation.....	33

Annex A (informative) Security considerations	35
A.1 Lattice security: background	35
A.1.1 Lattice definitions	35
A.1.2 Hard lattice problems	36
A.1.3 Theoretical complexity of hard lattice problems	36
A.1.4 Lattice reduction algorithms	36
A.1.5 The Gaussian heuristic and the closest vector problem	37
A.1.6 Modular lattices: definition	38
A.1.7 Modular lattices and quotient polynomial rings	38
A.1.8 Balancing CVP in modular lattices	38
A.1.9 Fundamental CVP ratios in modular lattices	39
A.1.10 Creating a balanced CVP for modular lattices containing a short vector	39
A.1.11 Modular lattices containing (short) binary vectors	40
A.1.12 Convolution modular lattices	41
A.1.13 Heuristic solution time for CVP in modular lattices	41
A.1.14 Zero-forcing	42
A.2 Experimental solution times for NTRU lattices—full key recovery	42
A.2.1 Experimental solution times for NTRU lattices using BKZ reduction	42
A.2.2 Alternative target vectors	44
A.3 Combined lattice and combinatorial attacks on LBP-PKE keys and messages	44
A.3.1 Overview	44
A.3.2 Lattice strength	44
A.3.3 Reduced lattices and the “cliff”	45
A.3.4 Combinatorial strength	48
A.3.5 Summary	50
A.4 Other security considerations for LBP-PKE encryption	50
A.4.1 Entropy requirements for key and salt generation	50
A.4.2 Reduction mod q	50
A.4.3 Selection of N	50
A.4.4 Relationship between q and N	50
A.4.5 Form of q	50
A.4.6 Leakage of $m'(1)$	51
A.4.7 Relationship between p , q , and N	51
A.4.8 Adaptive chosen ciphertext attacks	51
A.4.9 Invertibility of g in R_q	52
A.4.10 Decryption failures	52
A.4.11 OID	52
A.4.12 Use of hash functions by supporting functions	53
A.4.13 Generating random numbers in $[0, N - 1]$	53
A.4.14 Attacks based on variation in decryption times	53
A.4.15 Choosing to attack r or m	54
A.4.16 Quantum computers	54
A.4.17 Other considerations	54
A.5 A parameter set generation algorithm	54
A.6 Possible parameter sets	55
A.6.1 Size-optimized	55
A.6.2 Cost-optimized	57
A.6.3 Speed-optimized	60
A.7 Security levels of parameter sets	62
A.7.1 Assumed security levels versus current knowledge	62
A.7.2 Potential research	63
Annex B (informative) Bibliography	64

IEEE Standard Specification for Public Key Cryptographic Techniques Based on Hard Problems over Lattices

IMPORTANT NOTICE: *This standard is not intended to ensure safety, security, health, or environmental protection in all circumstances. Implementers of the standard are responsible for determining appropriate safety, security, environmental, and health practices or regulatory requirements.*

This IEEE document is made available for use subject to important notices and legal disclaimers. These notices and disclaimers appear in all publications containing this document and may be found under the heading “Important Notice” or “Important Notices and Disclaimers Concerning IEEE Documents.” They can also be obtained on request from IEEE or viewed at <http://standards.ieee.org/IPR/disclaimers.html>.

1. Overview

1.1 Scope

This standard provides specifications of common public key cryptographic techniques based on hard problems over lattices supplemental to those considered in IEEE Std 1363™-2000 [B47]¹ and IEEE Std 1363a™-2004 [B48], including mathematical primitives for secret value (key) derivation, public key encryption, identification and digital signatures, and cryptographic schemes based on those primitives. Specifications of related cryptographic parameters, public keys, and private keys are also presented. Class of computer and communications systems is not restricted.

1.2 Purpose

The transition from paper to electronic media brings with it the need for electronic privacy and authenticity. Public key cryptography offers fundamental technology addressing this need. Many alternative public key techniques have been proposed, each with its own benefits. IEEE Std 1363-2000 [B47] and IEEE Std 1363a-2004 [B48] have produced a comprehensive reference defining a range of common public key

¹ The numbers in brackets correspond to those of the bibliography in Annex B.

techniques covering key agreement, public key encryption, and digital signatures from several families, namely the discrete logarithm, integer factorization, and elliptic curve families.

This standard specifies cryptographic techniques based on hard problems over lattices. These techniques may offer tradeoffs in operating characteristics when compared with the methods already specified in IEEE 1363-2000 and IEEE Std 1363a-2004. This standard also provides a second-generation framework for the description of cryptographic techniques, as compared to the initial framework provided in IEEE Std 1363-2000 and IEEE Std 1363a-2004.

It is not the purpose of this project to mandate any particular set of public key techniques or security requirements (including key sizes) for this or any family. Rather, the purpose of this standard is to provide the following:

- a) A reference for specification of a variety of techniques from which applications may select
- b) The relevant number-theoretic background
- c) Extensive discussion of security and implementation considerations so that a solution provider can choose appropriate security requirements for itself

2. Normative references

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used, so each referenced document is cited in text and its relationship to this document is explained). For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

FIPS 180, *Secure Hash Standard*, Federal Information Processing Standards Publication 180, U.S. Department of Commerce/National Institute of Standards and Technology, National Technical Information Service, Springfield, Virginia.²

3. Definitions, acronyms, and abbreviations

3.1 Definitions

For the purposes of this standard, the following terms and definitions apply. *The Authoritative Dictionary of IEEE Standards* [B46] should be referenced for terms not defined in this clause.

- 3.1.1 algorithm:** A clearly specified mathematical process for computation; a set of rules that, if followed, give a prescribed result.
- 3.1.2 asymmetric cryptographic algorithm:** A cryptographic algorithm that uses two related keys, a public key and a private key; the two keys have the property that, given the public key, it is computationally infeasible to derive the private key.
- 3.1.3 authentication (of a message):** The act of determining that a message has not been changed since leaving its point of origin. The identity of the originator is implicitly verified.

²FIPS 180 current version as of 2008 is FIPS 180-2, August 26, 2002, available at <http://csrc.nist.gov/CryptoToolkit/Hash.html>.

- 3.1.4 authentication of ownership:** The assurance that a given, identified party intends to be associated with a given public key. May also include assurance that the party possesses the corresponding private key (see IEEE Std 1363-2000, Annex D.3.2, for more information).
- 3.1.5 big modulus:** The big modulus q is used to define the larger polynomial ring. The modulus q can generally be taken to be any value that is relatively prime in the ring to the small modulus p .
- 3.1.6 birthday paradox:** For a category size of 365 (the days in a year), after only 23 people are gathered, the probability is greater than 0.5 that at least two people have a common birthday (month and day). The reason is that among 23 people, there are $23 \times (23 - 1)/2 = 253$ pairs of people, each with a $1/365$ chance of having matching birthdays. The chance of no matching birthday is therefore $(364/365)^{253} \sim 0.4995$. In general, any case where the criterion for success is to find a collision (two matching values) rather than a hit (one value that matches a pre-selected one) displays this pairing property, so that the size of the space to be searched for success is about the square root of the size of the space of all possible value.
- 3.1.7 bit length:** *See: length.*
- 3.1.8 bit string:** An ordered sequence of zeroes and ones. The left-most bit is the most-significant bit of the string. The right-most bit is the least-significant bit of the string. A bit and a bit string of length 1 are equivalent for all purposes of this standard.
- 3.1.9 blinding polynomial:** In this standard, the ciphertext e is generated according to the equation $e = r \times h + m'$, where h is the public key, m' is the message representative, and r is a pseudorandomly generated blinding polynomial.
- 3.1.10 blinding polynomial generation methods:** In the encryption schemes in this document, a blinding polynomial generation method (LBP-BPGM) is used to generate a blinding polynomial r from the padded message pm in order to provide plaintext awareness.
- 3.1.11 blinding polynomial space:** The space that a LBP-BPGM selects from. Usually defined implicitly by the definition of the LBP-BPGM.
- 3.1.12 certificate:** The public key and identity of an entity together with some other information rendered unforgeable by signing the certificate with the private key of the certifying authority, which issued that certificate.
- 3.1.13 ciphertext:** The result of applying encryption to a message. *Contrast: plaintext. See also: encrypt.*
- 3.1.14 composite:** An integer that has at least two prime factors.
- 3.1.15 confidentiality:** The property that information is not made available or disclosed to unauthorized individuals, entities, or processes.
- 3.1.16 conformance region:** A set of inputs to a primitive or a scheme operation for which an implementation operates in accordance with the specification of the primitive or scheme operation
- 3.1.17 cryptographic family:** A set of cryptographic techniques in similar mathematical settings. For example, this standard presents a single family of techniques based on the underlying hard problems of finding a short vector and a close vector in a lattice.
- 3.1.18 cryptographic hash function:** *See: hash function.*

- 3.1.19 cryptographic key (key):** A parameter that determines the operation of a cryptographic function such as: the transformation from plain text to cipher text and vice versa; synchronized generation of keying material; digital signature computation, or validation.
- 3.1.20 cryptography:** The discipline that embodies principles, means, and methods for the transformation of data in order to hide its information content, prevent its undetected modification, prevent its unauthorized use, or a combination thereof.
- 3.1.21 data integrity:** A property whereby data has not been altered or destroyed.
- 3.1.22 decrypt:** To produce plaintext (readable) from ciphertext (unreadable). *Contrast:* **encrypt**. *See also:* **ciphertext; encrypt; plaintext**.
- 3.1.23 dimension:** The dimension N identifies the dimension of the convolution polynomial ring used. The dimension of the associated lattice problem is $2N$. Elements of the ring are represented as polynomials of degree $N - 1$.
- 3.1.24 domain parameters:** A set of mathematical objects, such as fields or groups, and other information, defining the context in which public/private key pairs exist. More than one key pair may share the same domain parameters. Not all cryptographic families have domain parameters. *See also:* **public/private key pair; valid domain parameters**.
- 3.1.25 domain parameter validation:** The process of ensuring or verifying that a set of domain parameters is valid. *See also:* **domain parameters; key validation; valid domain parameters**.
- 3.1.26 encrypt:** To produce ciphertext (unreadable) from plaintext (readable). *Contrast:* **decrypt**. *See also:* **ciphertext; encrypt; plaintext**.
- 3.1.27 encryption primitives:** An operation that converts a plaintext to a ciphertext, providing security according to the difficulty of solving an underlying hard problem, against a ciphertext-only attack by a passive attacker who only has a single non-chosen ciphertext. A building block for encryption schemes.
- 3.1.28 encryption scheme:** A means for providing encryption, based on an encryption primitive, that is secure against both active and passive attackers. A secure encryption scheme typically provides semantic security (an attacker who knows that one of two messages has been encrypted will find it computationally infeasible to determine which) against an attacker who can make polynomially many queries to a decryption oracle.
- 3.1.29 entity:** A participant in any of the schemes in this standard. The words “entity” and “party” are used interchangeably. This definition may admit many interpretations: it may or may not be limited to the necessary computational elements; it may or may not include or act on behalf of a legal entity. The particular interpretation chosen does not affect operation of the key agreement schemes.
- 3.1.30 exclusive OR (XOR):** A mathematical bit-wise operation, symbol $\oplus$, defined as:
$$\begin{aligned} 0 \oplus 0 &= 0, \\ 0 \oplus 1 &= 1, \\ 1 \oplus 0 &= 1, \text{ and} \\ 1 \oplus 1 &= 0. \end{aligned}$$

Equivalent to binary addition without carry. May also be applied to bit strings: the XOR of two bit strings of equal length is the concatenation of the XORs of the corresponding elements of the bit strings.
- 3.1.31 family:** *See:* **cryptographic family**.

- 3.1.32 field:** A setting in which the usual mathematical operations (addition, subtraction, multiplication, and division by nonzero quantities) are possible and obey the usual rules (such as the commutative, associative, and distributive laws).
- 3.1.33 finite field:** A field in which there are only a finite number of quantities.
- 3.1.34 first bit:** The leading bit of a bit string or an octet. For example, the first bit of 0110111 is 0. *Contrast: last bit. Syn: most significant bit; leftmost bit. See also: bit string; octet.*
- 3.1.35 first octet:** The leading octet of an octet string. For example, the first octet of 1c 76 3b e4 is 1c. *Contrast: last octet. Syn: most significant octet; leftmost octet. See also: octet; octet string.*
- 3.1.36 hash function:** A function that maps a bit string of arbitrary length to a fixed-length bit string and satisfies the following properties:
1. It is computationally infeasible to find any input that maps to any pre-specified output;
 2. It is computationally infeasible to find any two distinct inputs that map to the same output.
- 3.1.37 hash value:** The result of applying a hash function to a message.
- 3.1.38 index generation function (IGF):** An IGF is a function that is seeded once, can be called multiple times, and produces statistically independent integers modulo some number m on each call.
- 3.1.39 key:** *See: cryptographic key.*
- 3.1.40 key confirmation:** The assurance of the legitimate participants in a key establishment protocol that the intended recipients of the shared key actually possess the shared key.
- 3.1.41 key derivation:** The process of deriving one or more session keys from a shared secret and (possibly) other, public information. Such a function can be constructed from a one-way hash function such as SHA-1.
- 3.1.42 key encrypting key (KEK):** A key used exclusively to encrypt and decrypt keys.
- 3.1.43 key establishment:** A protocol that reveals a secret key to its legitimate participants for cryptographic use.
- 3.1.44 key generation primitive:** A method used to generate a key pair.
- 3.1.45 key management:** The generation, storage, secure distribution, and application of keying material in accordance with a security policy.
- 3.1.46 key pair:** When used in public key cryptography, a private key and its corresponding public key. The public key is commonly available to a wide audience and can be used to encrypt messages or verify digital signatures; the private key is held by one entity and not revealed to anyone—it is used to decrypt messages encrypted with the public key and/or produce signatures that can be verified with the public key. A public/private key pair can also be used in key agreement. In some cases, a public/private key pair can only exist in the context of domain parameters. *See also: digital signature; domain parameters; encryption; key agreement; public key cryptography; valid key; valid key pair.*
- 3.1.47 key transport:** A key establishment protocol under which the secret key is determined by the initiating party.

- 3.1.48 key validation:** the process of ensuring or verifying that a key conforms to the arithmetic requirements for such a key in order to thwart certain types of attacks. *See also:* **domain parameter validation; public/private key pair; valid key; valid key pair.**
- 3.1.49 keying material:** The data (e.g., keys, certificates and initialization vectors) necessary to establish and maintain cryptographic keying relationships.
- 3.1.50 known-key security:** Known-key security for Party U implies that the key agreed upon will not be compromised by the compromise of the other session keys. If each ephemeral key is used only to compute a single session key, then known-key security may be achieved.
- 3.1.51 last bit:** The trailing bit of a bit string or an octet. For example, the last bit of 0110111 is 1. *Contrast:* **first bit.** *Syn:* **least significant bit; rightmost bit.** *See also:* **first bit; octet.**
- 3.1.52 last octet:** The trailing octet of an octet string. For example, the last octet of 1c 76 3b e4 is e4. *Contrast:* **first octet.** *Syn:* **least significant octet; rightmost octet.** *See also:* **octet; octet string.**
- 3.1.53 lattice-based polynomial public key encryption:** An encryption mechanism where operations are based on polynomial multiplication and the security is based on the difficulty of performing high-dimension lattice reduction.
- 3.1.54 least significant:** *See:* **last bit; last octet.**
- 3.1.55 leftmost bit:** *See:* **first bit.**
- 3.1.56 leftmost octet:** *See:* **first octet.**
- 3.1.57 length:** (1) Length of a bit string is the number of bits in the string. (2) Length of an octet string is the number of octets in the string. (3) Length in bits of a nonnegative integer n is $\lfloor \log_2 (n + 1) \rfloor$ (i.e., the number of bits in the integer's binary representation). (4) Length in octets of a nonnegative integer n is $\lfloor \log_{256} (n + 1) \rfloor$ (i.e., the number of digits in the integer's representation base 256). For example, the length in bits of the integer 500 is 9, and its length in octets is 2.
- 3.1.58 mask generation function (MGF):** An MGF is a construction built around a hash function that produces an arbitrary-length output string, possibly longer than the output of the underlying hash function.
- 3.1.59 message authentication code (MAC):** A cryptographic value that is the results of passing a financial message through the message authentication algorithm using a specific key.
- 3.1.60 message length encoding length:** In SVES, the length of the message that is to be encrypted is encoded in the padded message. The length of the field that represents the length of the message, called the message length encoding length, is represented by the parameter $lLen$. For all parameter sets in this standard, $lLen$ is set to 1.
- 3.1.61 message representative:** A mathematical value for use in a cryptographic primitive, computed from a message that is input to an encryption or a digital signature scheme and uniquely linked to that message. *See also:* **encryption scheme; digital signature scheme.**
- 3.1.62 modular lattice:** A lattice in which (among other things) all values are integers reduced mod q .
- 3.1.63 most significant:** *See:* **first bit; first octet.**
- 3.1.64 norm:** A measure of the “size” of a vector or polynomial.

- 3.1.65 octet:** A bit string of length 8. An octet has an integer value between 0 and 255 when interpreted as a representation of an integer in base 2. An octet can also be represented by a hexadecimal string of length 2, where the hexadecimal string is the representation of its integer value base 16. For example, the integer value of the octet 10011101 is 157; its hexadecimal representation is 9d. Also commonly known as a byte. *See also:* **bit string**.
- 3.1.66 octet string:** An ordered sequence of octets. *See also:* **octet**.
- 3.1.67 owner:** The entity whose identity is associated with a key pair.
- 3.1.68 parameters:** *See:* **domain parameters**.
- 3.1.69 plaintext:** A message before encryption has been applied to it; the opposite of ciphertext. *Contrast:* **ciphertext**. *See also:* **encryption**.
- 3.1.70 polynomial index generation constant:** A value used when generating a random number in the range $[0, N - 1]$, to eliminate bias without impacting efficiency.
- 3.1.71 prime number:** An integer that is greater than 1 and divisible only by 1 and itself.
- 3.1.72 primitives:** Cryptographic primitives used in the SVES encryption scheme include key generation primitives, encryption primitives, and decryption primitives.
- 3.1.73 private key:** The private element of the public/private key pair. *See also:* **public/private key pair**; **valid key**.
- 3.1.74 private key space:** The space from which a key generation primitive selects the private key.
- 3.1.75 public key:** The public element of the public/private key pair. *See also:* **public/private key pair**; **valid key**.
- 3.1.76 public key cryptography:** Methods that allow parties to communicate securely without having prior shared secrets through the use of public/private key pairs. *Contrast:* **symmetric cryptography**. *See also:* **public/private key pair**.
- 3.1.77 public key space:** The space from which a key generation primitive selects the public key.
- 3.1.78 public key validation:** *See:* **key validation**.
- 3.1.79 public/private key pair:** *See:* **key pair**.
- 3.1.80 rightmost bit:** *See:* **last bit**.
- 3.1.81 rightmost octet:** *See:* **last octet**.
- 3.1.82 ring:** A setting in which addition, subtraction, and multiplication are possible, and division by a given nonzero quantity may or may not be possible. A field is a special case of a ring. *See also:* **field**.
- 3.1.83 ring element:** In general, an element in a ring. In the context of this standard, a *binary N -ring element* refers to an element in the ring $(\mathbf{Z}/2\mathbf{Z})[X]/(X^N - 1)$, which is to say a binary polynomial of degree $N-1$ or an array of N binary elements. A (q, N) -ring element refers to an element in the ring $(\mathbf{Z}/q\mathbf{Z})[X]/(X^N - 1)$, which is to say a polynomial of degree $N - 1$ with coefficients reduced mod q or an array of N elements each taken mod q .

- 3.1.84 salt:** Random bits that are used to pad the message during encryption, to provide for semantic security.
- 3.1.85 salt size:** The size of the salt. This can be expressed in bits or octets.
- 3.1.86 scheme options:** Scheme options consist of parameters and algorithms that do not affect the key space (i.e., that are not domain parameters), but that shall be agreed upon in order to implement the encryption scheme.
- 3.1.87 secret key:** A key used in symmetric cryptography; needs to be known to all legitimate participating parties involved, but cannot be known to an adversary. *Contrast:* **public/private key pair**. *See also:* **key agreement; shared secret key; symmetric cryptography**.
- 3.1.88 secret value:** A value that can be used to derive a secret key, but typically cannot by itself be used as a secret key. *See also:* **secret key**.
- 3.1.89 shared secret key:** A secret key shared by two parties, usually derived as a result of a key agreement scheme. *See also:* **key agreement; secret key**.
- 3.1.90 shared secret value:** A secret value shared by two parties, usually during a key agreement scheme. *See also:* **key agreement; secret value**.
- 3.1.91 short vector encryption scheme (SVES):** The encryption scheme defined in IEEE Std 1363.1-2008.
- 3.1.92 signature:** *See:* **digital signature**.
- 3.1.93 small modulus:** In LBP-PKE, the small modulus p is used for key generation and for modular reduction during decryption.
- 3.1.94 statistically unique:** For the generation of n -bit quantities, the probability of two values repeating is less than or equal to the probability of two n -bit random quantities repeating. More formally, an element chosen from a finite set S of n elements is said to be *statistically unique* if the process that governs the selection of this element is such that, for any integer $L \leq n$, the probability that all of the first L selected elements are different is no smaller than the probability of this happening when the elements are drawn uniformly randomly from S .
- 3.1.95 symmetric cryptographic algorithm:** A cryptographic algorithm that uses one cryptographic key. Anyone who knows the key can both encrypt and decrypt a message, and can calculate a Message Authentication Code using that key.
- 3.1.96 symmetric cryptography:** Methods that allow parties to communicate securely only when they already share some prior secrets, such as the secret key. *Contrast:* **public key cryptography**. *See also:* **secret key**.
- 3.1.97 symmetric key:** A cryptographic key that is used in symmetric cryptographic algorithms. The same symmetric key that is used for encryption is also used for decryption.
- 3.1.98 user:** A party that uses a public key.
- 3.1.99 valid domain parameters:** A set of domain parameters that satisfies the specific mathematical definition for the set of domain parameters of its family. While a set of mathematical objects may have the general structure of a set of domain parameters, it may not actually satisfy the definition (for example, it may be internally inconsistent) and thus not be valid. *See also:* **domain parameters; public/private key pair; valid key; valid key pair; validation**.

- 3.1.100 valid key:** A key (public or private) that satisfies the specific mathematical definition for the keys of its family, possibly in the context of its set of domain parameters. While some mathematical objects may have the general structure of keys, they may not actually lie in the appropriate set (for example, they may not lie in the appropriate subgroup of a group or be out of the bounds allowed by the domain parameters) and thus not be valid keys. *See also:* **domain parameters; public/private key pair; valid domain parameters; valid key pair; validation.**
- 3.1.101 valid key pair:** A public/private key pair that satisfies the specific mathematical definition for the key pairs of its family, possibly in the context of its set of domain parameters. While a pair of mathematical objects may have the general structure of a key pair, the keys may not actually lie in the appropriate sets (for example, they may not lie in the appropriate subgroup of a group or be out of the bounds allowed by the domain parameters) or may not correspond to each other; such a pair is thus not a valid key pair. *See also:* **domain parameters; public/private key pair; valid domain parameters; valid key; validation.**
- 3.1.102 validation:** *See:* **domain parameter validation; key validation.**
- 3.1.103 verify:** In relation to a Digital Signature means to determine accurately: (1) that the Digital Signature was created during the operational period of a valid Certificate by the private key corresponding to the public key listed in the Certificate; and (2) the message has not been altered since its Digital Signature was created.

3.2 Acronyms and abbreviations

BS2IP	bit string to integer conversion primitive
BS2REP	bit string to ring element conversion primitive
BS2ROSP	bit string to right-padded octet string conversion primitive
BPGM	blinding polynomial generation method
DP	decryption primitive
ES	encryption scheme
I2BSP	integer to bit string conversion primitive
I2OSP	integer to octet string conversion primitive
IGF	index generation function
IGF-MGF1	index generation function based on mask generation function 1
KGP	key generation primitive
LBP-BPGM1	blinding polynomial generation method for generating binary blinding polynomials
LBP-BPGM2	blinding polynomial generation method for generating product-form blinding polynomials
LBP-DP1	decryption primitive for use with lattice-based polynomial public key decryption

LBP-KGP1	lattice-based polynomial key generation primitive
LBP-PKE	lattice-based polynomial public key encryption
MAC	message authentication code.
MGF	mask generation function
MPM	message padding method
MRGM	message representative generation method
OS2IP	octet string to integer conversion primitive
OS2REP	octet string to ring element conversion primitive
RE2BSP	ring element to bit string conversion primitive
RE2OSP	ring element to octet string conversion primitive
ROS2BSP	right-padded octet string to bit string conversion primitive
SVDP	short vector decryption primitive
SVES	short vector encryption scheme

4. Types of cryptographic techniques

4.1 General model

As stated in Clause 1, the purpose of this standard is to provide a reference for specifications of a variety of common public key cryptographic techniques from which applications may select. Different types of cryptographic techniques can be viewed abstractly according to the following three-level general model:

- *Primitives*: Basic mathematical operations. Historically, they were discovered based on number-theoretic hard problems. Primitives are not meant to achieve security just by themselves, but they serve as building blocks for schemes.
- *Schemes*: A collection of related operations combining primitives and additional methods (see 4.4). Schemes can provide complexity-theoretic security that is enhanced when they are appropriately applied in protocols.
- *Protocols*: Sequences of operations to be performed by multiple parties to achieve some security goal. Protocols can achieve desired security for applications if implemented correctly.

From an implementation viewpoint, primitives can be viewed as low-level implementations (e.g., implemented within cryptographic accelerators, or software modules), schemes can be viewed as medium-level implementations (e.g., implemented within cryptographic service libraries), and protocols can be viewed as high-level implementations (e.g., implemented within entire sets of applications).

This standard contains only specifications of schemes.

4.2 Schemes

Schemes in this standard are presented in a general form based on certain primitives and additional methods. For example, the encryption scheme defined in this standard is based on a key generation primitive, a decryption primitive, and a blinding polynomial generation method.

Schemes also include key management operations, such as selecting a private key or obtaining another party's public key. For proper security, a party needs to be assured of the true owners of the keys and domain parameters and of their validity. Generation of domain parameters and keys needs to be performed properly, and in some cases validation also needs to be performed. While outside the scope of this standard, proper key management is essential for security.

This standard defines one type of scheme, as follows:

- Encryption schemes (ESs), in which any party can encrypt a message using a recipient's public key, and only the recipient can decrypt the message by using its corresponding private key. Encryption schemes may be used for establishing secret keys to be used in symmetric cryptography.

An *encryption scheme* is specified by providing the following:

- Name
- Type (e.g., asymmetric public key encryption scheme)
- Options (key type, primitives, parameters)

- Operations
 - Key pair generation
 - Key pair validation
 - Public key validation
 - Encryption operation
 - Input
 - Output
 - Decryption operation
 - Input
 - Output

An encryption scheme specification may also include the following:

- Security considerations
- Implementation considerations
- Related standards

The specifications are functional specifications, not interface specifications. As such, the format of inputs and outputs and the procedure by which an implementation of a scheme is invoked are outside the scope of this standard.

4.3 Additional methods

This standard specifies the following additional methods:

- Blinding polynomial generation methods, which are components of encryption schemes.
- Auxiliary Functions, which are building blocks for other additional methods.
 - Index generation functions
 - Mask generation functions
 - Hash functions, which are used as the core of index generation functions and of mask generation functions.

The specified additional methods are required for conformant use of the schemes. The use of an inadequate message encoding method, key derivation function, or auxiliary function may compromise the security of the scheme in which it is used. Therefore, any implementation that chooses not to follow the recommended additional methods for a particular scheme should perform its own thorough security analysis of the resulting scheme.

4.4 Algorithm specification conventions

When specifying an algorithm or method, this standard uses four parts to specify different aspects of the algorithm. They are as follows:

- a) **Components**, such as choice of index generation function (IGF), are parameters that are specified before the beginning of the operation and that are not specific to the particular algorithm call.

Components tend to be kept fixed for multiple users and multiple instances of the algorithm call and need not be explicitly specified if they are implicitly known [e.g., if they are defined within a selected object identifier (OID)].

- b) **Inputs**, such as keys and messages, are values that shall be specified for each algorithm call.
- c) **Outputs**, such as ciphertext, are the result of transformations on the inputs.
- d) **Operations** specify the transformations that are performed on the data to arrive at the output. Throughout the standard, the operations are defined as a sequence of steps. A conformant implementation may perform the operations using any sequence of steps that consistently produces the same output as the sequence in this standard. Caution should be taken to ensure that intermediate values are not revealed, however, as they may compromise the security of the algorithms.

5. Mathematical notation

When referring to mathematical objects and data objects in this standard, the following notation in Table 1 is used. Throughout the document, numbers at the end of variable names are used to distinguish different, but related values ($df1$, $df2$, $df3$ or $Dmin1$, $Dmin2$, etc.).

Table 1—Mathematical notation

0	Denotes the integer 0, the bit 0, or the additive identity (the element zero) of a ring.
1	Denotes the integer 1, the bit 1, or the multiplicative identity (the element one) of a ring.
$\times$	Indicates the convolution product of two polynomials and is also used to indicate multiplication of integers.
$\oplus$ or XOR	Exclusive OR function.
$\parallel$	Concatenation. $A\parallel B$ is the concatenation of the octet strings A and B where the leading octet of A is the leading octet of $A\parallel B$ and the trailing octet of B is the trailing octet of $A\parallel B$.
$:=$	Initialization. $a := b$ means initialize or set the value of a equal to the value of b .
A	Lower-bound decryption coefficient, used in decryption process to reduce into correct interval.
$\text{ceil}[\cdot]$ or $\lceil \cdot \rceil$	Ceiling function (i.e., the smallest integer greater than or equal to the contents of $[\cdot]$).
db	The number of random bits used as input for encryption.
d_f	An integer specifying the number of ones in the polynomials that comprise the private key value f (also specified as $df1$, $df2$, and $df3$, or as dF).
d_g	An integer specifying the number of ones in the polynomials that comprise the temporary polynomial g (often specified as dG).
d_r	An integer specifying the number of ones in the blinding polynomial r in SVES. (Also specified as $dr1$, $dr2$, and $dr3$.)
e	Encrypted message representative, a polynomial, computed by an encryption primitive.

E	Encrypted message, an octet string.
ES	(Asymmetric) encryption scheme.
f	Private key in SVES.
F	In SVES, a polynomial that is used to calculate the value f when $f = 1 + pF$.
$\text{floor}[\cdot]$ or $\lfloor \cdot \rfloor$	Floor function (i.e. the largest integer less than or equal to the contents of $[\cdot]$).
g	In SVES, a temporary polynomial used in the key generation process.
$\text{GCD}(a, b)$	Greatest Common Divisor of two non-negative integers a and b .
h	Public key.
$\text{Hash}(\cdot)$	A cryptographic hash function computed on the contents of $(\cdot)$.
$hLen$	Length in octets of a hash value.
i	An integer.
k	Security level in bits.
m	The message, an octet string, which is encrypted in SVES.
M	In SVES, the padded and formatted message representative octet string used during encryption and decryption.
m'	The message representative polynomial that is submitted to the encryption primitive in the SVES encryption scheme.
$\text{mod } q$	Used to reduce the coefficients of a polynomial into some interval of length q .
$\text{mod } p$	Used to reduce a polynomial to an element of the polynomial ring $\text{mod } p$.
N	Dimension of the polynomial ring used (i.e., polynomials are up to degree $N - 1$).
p	“Small” modulus, an integer or a polynomial.
q	“Big” modulus, usually an integer.
r	In LBP-PKE, the encryption blinding polynomial (generated from the hash of the padded message M in SVES).
x	The integer input to or output from integer conversion primitives.
X	The indeterminate used in polynomials.
$\mathbf{Z}$	The ring of integers.
$\mathbf{Z}_q$	The ring of integers $\text{mod } q$.