



IEEE Standard for Wireless Access in Vehicular Environments—Security Services for Application and Management Messages

IEEE Vehicular Technology Society

Developed by the
Intelligent Transportation Systems Committee

IEEE Std 1609.2™-2022
(Revision of IEEE Std 1609.2-2016)

STANDARDS

IEEE Standard for Wireless Access in Vehicular Environments—Security Services for Application and Management Messages

Developed by the

Intelligent Transportation Systems Committee
of the
IEEE Vehicular Technology Society

Approved 3 December 2022

IEEE SA Standards Board

Abstract: Secure message formats and processing for use by Wireless Access in Vehicular Environments (WAVE) devices are defined in this standard, including methods to secure WAVE management messages and methods to secure application messages. Administrative functions necessary to support the core security functions are also described.

Keywords: cryptography, IEEE 1609.2™, security, Wireless Access in Vehicular Environments (WAVE)

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

Copyright © 2023 by The Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 20 March 2023. 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 978-1-5044-9317-8 STD25867
Print: ISBN 978-1-5044-9316-1 STDPD25867

IEEE prohibits discrimination, harassment, and bullying.

For more information, visit <https://www.ieee.org/about/corporate/governance/p9-26.html>.

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.

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE Standards documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page (<https://standards.ieee.org/ipr/disclaimers.html>), appear in all standards and may be found under the heading “Important Notices and Disclaimers Concerning IEEE Standards Documents.”

Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE SA) Standards Board. IEEE develops its standards through an accredited consensus development process, which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE Standards are documents developed by volunteers with scientific, academic, and industry-based expertise in technical working groups. Volunteers are not necessarily members of IEEE or IEEE SA, and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

IEEE does not warrant or represent the accuracy or completeness of the material contained in its standards, and expressly disclaims all warranties (express, implied and statutory) not included in this or any other document relating to the standard, including, but not limited to, the warranties of: merchantability; fitness for a particular purpose; non-infringement; and quality, accuracy, effectiveness, currency, or completeness of material. In addition, IEEE disclaims any and all conditions relating to results and workmanlike effort. In addition, IEEE does not warrant or represent that the use of the material contained in its standards is free from patent infringement. IEEE Standards documents are supplied “AS IS” and “WITH ALL FAULTS.”

Use of an IEEE standard is wholly voluntary. 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.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity, nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: THE NEED TO PROCURE SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

Translations

The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE is the approved IEEE standard.

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, nor be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that the presenter's views should be considered the personal views of that individual rather than the formal position of IEEE, IEEE SA, the Standards Committee, or the Working Group.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE or IEEE SA. However, **IEEE does not provide interpretations, consulting information, or advice pertaining to IEEE Standards documents.**

Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive 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 comments, or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in evaluating comments or in revisions to an IEEE standard is welcome to join the relevant IEEE working group. You can indicate interest in a working group using the Interests tab in the Manage Profile & Interests area of the [IEEE SA myProject system](#). An IEEE Account is needed to access the application.

Comments on standards should be submitted using the [Contact Us](#) form.

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not constitute 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.

Data privacy

Users of IEEE Standards documents should evaluate the standards for considerations of data privacy and data ownership in the context of assessing and using the standards in compliance with applicable laws and regulations.

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under US and international copyright laws. They are made available by IEEE and are adopted 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 these documents available for use and adoption by public authorities and private users, IEEE does not waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate licensing fees, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400; <https://www.copyright.com/>. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Updating of IEEE Standards documents

Users of IEEE Standards documents 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.

Every IEEE standard is subjected to review at least every 10 years. When a document is more than 10 years old and has not undergone a revision process, 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 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 [IEEE Xplore](#) or [contact IEEE](#). For more information about the IEEE SA or IEEE's standards development process, visit the IEEE SA Website.

Errata

Errata, if any, for all IEEE standards can be accessed on the [IEEE SA Website](#). Search for standard number and year of approval to access the web page of the published standard. Errata links are located under the Additional Resources Details section. Errata are also available in [IEEE Xplore](#). Users are encouraged to periodically check for errata.

Patents

IEEE Standards are developed in compliance with the [IEEE SA Patent Policy](#).

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 by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE SA Website at <https://standards.ieee.org/about/sasb/patcom/patents.html>. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. 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.

IMPORTANT NOTICE

IEEE Standards do not guarantee or ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. IEEE Standards development activities consider research and information presented to the standards development group in developing any safety recommendations. Other information about safety practices, changes in technology or technology implementation, or impact by peripheral systems also may be pertinent to safety considerations during implementation of the standard. Implementers and users of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, and interference protection practices and all applicable laws and regulations.

Participants

At the time this standard was completed, the Dedicated Short Range Communications Working Group had the following membership:

Justin McNew, *Chair*
Kevin Smith, William Whyte, *Vice Chairs*

Benedikt Brecht
Adrian Buckley
Kevin Henry
Carl Kain
John Kenney
Malik Khan

Virendra Kumar
Thomas M. Kurihara
Bill Lattin
Julius Madey
Sean Maschue
Michael Montemurro
Vincent Park

Randal Roebuck
Richard Roy
Michael Stelts
Jasja Tijink
Michaela Vanderveen
Jimmy Upton

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

Iwan Adhicandra
Robert Aiello
Charles Barest
William Byrd
Pin Chang
Udhaya Kumar Dayalan
Gloria Gwynne
Marco Hernandez
Ronald Hochnadel
Werner Hoelzl
David Hunter
Brian Johnson
Piotr Karocki
John Kenney
Stuart Kerry

Virendra Kumar
Thomas Kurihara
Jeremy Landt
William Lee
Joseph Levy
Greg Luri
Julius Madey
Sean Maschue
Justin Mcnew
Michael Montemurro
Rajesh Murthy
Satoshi Oyama
Sivaraman P
Venkatesha Prasad
Randal Roebuck

Stephan Sand
Di Dieter Smely
Kevin Smith
Dorothy Stanley
Eugene Stoudenmire
Mark Sturza
Jasja Tijink
Max Turner
Dmitri Varsanofiev
John Vergis
Lisa Ward
Stephen Webb
William Whyte
Yu Yuan
Oren Yuen

When the IEEE SA Standards Board approved this standard on 3 December 2022, it had the following membership:

David J. Law, *Chair*
Ted Burse, *Vice Chair*
Gary Hoffman, *Past Chair*
Konstantinos Karachalios, *Secretary*

Edward A. Addy
Ramy Ahmed Fathy
J. Travis Griffith
Guido R. Hiertz
Yousef Kimiagar
Joseph L. Koepfinger*
Thomas Koshy
John D. Kulick

Johnny Daozhuang Lin
Kevin Lu
Daleep C. Mohla
Andrew Myles
Damir Novosel
Annette D. Reilly
Robby Robson
Jon Walter Rosdahl

Mark Siira
Dorothy V. Stanley
Lei Wang
F. Keith Waters
Karl Weber
Sha Wei
Philip B. Winston
Daidi Zhong

*Member Emeritus

Introduction

This introduction is not part of IEEE Std 1609.2-2022, IEEE Standard for Wireless Access in Vehicular Environments—Security Services for Application and Management Messages.

5.9 GHz Dedicated Short Range Communications for Wireless Access in Vehicular Environments (DSRC/WAVE, hereafter simply WAVE), as specified in a range of standards including those generated by the IEEE P1609 working group, enables vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) wireless communications. This connectivity makes possible a range of applications that rely on communications between road users and road operators, including vehicle safety, public service, commercial fleet management, tolling, and other operations.

With improved communications come increased risks, and the safety-critical nature of many WAVE applications makes it vital that services be specified that can be used to protect messages from attacks such as eavesdropping, spoofing, alteration, and replay. Additionally, the fact that the wireless technology will be deployed in personal vehicles, whose owners have a right to privacy, means that in as much as possible the security services should respect that right and not leak personal, identifying, or linkable information to unauthorized parties.

With this in mind, at the time that IEEE P1609 was established to develop the standards for the WAVE wireless networking protocols, the IEEE also established IEEE P1556™ (later renumbered as IEEE Std 1609.2) to develop standards for the security techniques that will be used to protect the services that use these protocols. These applications face unique constraints. Many of them, particularly safety applications, are time critical: It is crucial to keep the processing and bandwidth overhead due to security to a minimum, to improve responsiveness and decrease the likelihood of packet loss. For many applications, the potential audience consists of all vehicles on the road in North America; therefore, the mechanism used to authenticate messages needs to be as flexible and scalable as possible and has to accommodate the smooth removal of compromised WAVE devices from the system. Additionally, as mentioned above, it is essential to respect the privacy of privately owned and operated vehicles, as well as potentially other personal devices within the WAVE system, as far as technically and administratively feasible.

This standard is a revision of IEEE Std 1609.2™-2016 with no significant changes in functionality, but with some extensions to protocol data units (PDUs). This standard is intended and believed to be fully backward compatible with IEEE Std 1609.2-2016; that is, all valid encoded structures produced with IEEE Std 1609.2-2016 are valid encoded structures for this standard as well, and all valid encoded structures produced with this standard and not valid for IEEE Std 1609.2-2016 can be identified as such by a parser for IEEE Std 1609.2-2016.

Contents

1. Overview	11
1.1 Scope	11
1.2 Purpose	11
1.3 Word usage	12
1.4 Document organization	12
1.5 Document conventions	12
1.6 Testing considerations	13
2. Normative references	13
3. Definitions, abbreviations, and acronyms	14
3.1 Definitions	14
3.2 Abbreviations and acronyms	23
4. General description	25
4.1 WAVE protocol stack overview	25
4.2 Secure data service (SDS)	29
4.3 Security services management entity (SSME)	35
4.4 Behavior of SDEEs	37
5. Cryptographic operations and validity	37
5.1 Certificate validity	37
5.2 Signed SPDU validity	59
5.3 Cryptographic operations	80
6. Data structures	90
6.1 Presentation and encoding	90
6.2 Basic types	91
6.3 Security services protocol data units (SPDUs)	92
6.4 Certificates and other security management data structures	114
6.5 HeaderInfo extension base	138
6.6 Contributed HeaderInfo extensions	139
7. Certificate revocation lists (CRLs) and the CRL Verification Entity	140
7.1 General	140
7.2 CRL Verification Entity specification	140
7.3 Data structures	141
7.4 CRL: 1609.2 Security envelope	149
8. Peer-to-peer certificate distribution (P2PCD)	154
8.1 General	154
8.2 P2PCD operations	155
8.3 P2PCD Entity specification	169
8.4 Data structures	169
9. Service primitives and functions	171
9.1 General comments and conventions	171
9.2 Identifiers used in the interface specification	173
9.3 Sec SAP	179
9.4 SSME SAP	213
9.5 SSME-Sec SAP	236

Annex A (normative) Protocol Implementation Conformance Statement (PICS) proforma.....	242
A.1 Instructions for completing the PICS proforma.....	242
A.2 PICS proforma—IEEE Std 1609.2	244
Annex B (normative) ASN.1 modules.....	257
B.1 General.....	257
B.2 1609.2 security services	257
B.3 Certificate revocation list (CRL).....	258
B.4 Peer-to-peer certificate distribution (P2PCD)	258
B.5 ETSI TS 103097 extension module	258
Annex C (informative) Specifying the use of IEEE Std 1609.2 by SDEEs.....	259
C.1 General.....	259
C.2 IEEE 1609.2 security profiles	259
C.3 IEEE 1609.2 security profile proforma.....	280
C.4 Service Specific Permissions (SSP).....	285
C.5 Assurance level.....	286
C.6 Recommendations on certificates	286
C.7 Source of encryption keys.....	286
Annex D (informative) Examples and use cases	288
D.1 Guidance for SDEE specifiers and implementers.....	288
D.2 Processing CRLs.....	289
D.3 Constructing a certificate chain	290
D.4 Peer-to-peer certificate distribution	295
D.5 Example data structures	304
D.6 Cryptographic test vectors	309
D.7 Test vectors for Linkage Values $lv(i,j)$	320
Annex E (informative) Guidance on future HeaderInfo extensions	326
E.1 General.....	326
E.2 Steps for the IEEE P1609 working group	327
E.3 Steps for HeaderInfo extension contributor	328
Annex F (informative) Deployment considerations	330
Annex G (informative) Extending this standard.....	332
G.1 ToBeSignedCertificate extensions.....	332
Annex H (informative) Anticipated model of use for Operating Organization ID.....	334
H.1 General.....	334
H.2 Obtaining certificates containing Operating Organization ID	335
H.3 Use of Operating Organization ID by receiver	337
H.4 Entitlement of CAs to issue certificates containing particular Operating Organization ID values	339
H.5 Operating Organization ID in the context of signal prioritization	341
H.6 Adding new organizations and the semantics and hierarchy of Operating Organization ID	344
H.7 Consistency between Operating Organization ID and message payload	344
H.8 Conclusion.....	345
Annex I (informative) Bibliography.....	346

IEEE Standard for Wireless Access in Vehicular Environments—Security Services for Application and Management Messages

1. Overview

1.1 Scope

This standard defines secure message formats and processing for use by Wireless Access in Vehicular Environments (WAVE) devices, including methods to secure WAVE management messages and methods to secure application messages. It also describes administrative functions necessary to support the core security functions.

1.2 Purpose

The safety-critical nature of many WAVE applications makes it vital that services be specified that can be used to protect messages from attacks such as eavesdropping, spoofing, alteration, and replay. Additionally, the fact that the wireless technology will be deployed in communication devices in personal vehicles as well as other portable devices, whose owners have an expectation of privacy, means that in as much as possible the security services need to be designed to respect privacy and not leak personal, identifying, or linkable information to unauthorized parties. This standard describes security services for WAVE management messages and application messages designed to meet these goals.

1.3 Word usage

The word *shall* indicates mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted (*shall* equals *is required to*).^{1,2}

The word *should* indicates that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required (*should* equals *is recommended that*).

The word *may* is used to indicate a course of action permissible within the limits of the standard (*may* equals *is permitted to*).

The word *can* is used for statements of possibility and capability, whether material, physical, or causal (*can* equals *is able to*).

1.4 Document organization

Clause 1 provides an overview of the document. Clause 2 contains the normative references. Clause 3 contains definitions and abbreviations. Clause 4 provides a general description of WAVE Security Services and their use. Clause 5 specifies validity of signed security services protocol data units (SPDUs), correctness of encrypted protocol data units (PDUs), and core cryptographic operations. Clause 6 specifies the encoding and structure of messages generated and consumed by WAVE Security Services. Clause 7 provides a specification of certificate revocation lists, which are a mechanism used to distribute information about certificates that are no longer trusted. Clause 8 specifies mechanisms for peer-to-peer certificate distribution. Clause 9 defines the primitives used to communicate between WAVE Security Services and other functional entities.

Annex A provides a Protocol Implementation Conformance Statement (PICS) proforma. Annex B provides ASN.1 modules. Annex C provides a description of the IEEE 1609.2 security profile and a proforma that may be used by developers of applications (or other entities that invoke WAVE Security Services) to specify options for how those applications are to interact with WAVE Security Services. Annex D provides examples, including examples of encoded datagrams and sample process flows, from the point of view of an entity invoking WAVE Security Services rather than from the point of view of WAVE Security Services. Annex E provides guidance for external contributors on extending the security header info in the future. Annex F describes other considerations that impact the deployment of a secure communications system using the techniques of this standard. Annex G explains how the framework provided in this standard allows for the extension of different fields in the certificate. Annex H presents the anticipated model of use for Operating Organization ID. Annex I provides an informative bibliography.

1.5 Document conventions

Unless otherwise stated, conventions follow those in IEEE Std 802.11™ [B12],³ including conventions for the ordering of information elements within data streams.

Numbers are decimal unless otherwise noted. Numbers preceded by 0x are to be read as hexadecimal, so that 0xFF is equivalent to “FF hexadecimal.” Occasionally, this standard includes representations of octet strings in hexadecimal form; these strings are indicated as hexadecimal on a case-by-case basis.

Figures are used for illustration and are informative, unless otherwise noted.

¹ The use of the word *must* is deprecated and cannot be used when stating mandatory requirements; *must* is used only to describe unavoidable situations.

² The use of *will* is deprecated and cannot be used when stating mandatory requirements; *will* is only used in statements of fact.

³ The numbers in brackets correspond to those of the bibliography in Annex I.

1.6 Testing considerations

The services defined in this standard, with the exception of the peer-to-peer certificate distribution service specified in Clause 8, operate over internal interfaces that are not directly observable in normal operations. Conformance claims made about an implementation of this standard can only be fully tested by direct access to the specific interfaces of that implementation. This standard does not provide a normative interface specification, so an implementation of a test process is not guaranteed a standard interface that may be used to access the implementation under test.

2. Normative references

The following referenced documents are indispensable for the application of this standard (i.e., they must be understood and used, so each referenced document is cited in text and its relationship to this standard 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.

ANSI and FIPS Codes: United States Census, 2010 FIPS Codes for Counties and County Equivalent Entities.^{4, 5}

FIPS 180-4, Secure Hash Standard (SHS), Aug. 2015.

FIPS 186-4, Digital Signature Standard (DSS), July 2013.

FIPS 197, Advanced Encryption Standard (AES), Nov. 2001.

IEEE Std 1363™-2000, IEEE Standard Specifications for Public Key Cryptography.^{6, 7}

IEEE Std 1363a™-2004, IEEE Standard Specifications for Public Key Cryptography—Amendment 1: Additional Techniques.

IEEE Std 1609.0™, IEEE Guide for Wireless Access in Vehicular Environments (WAVE) Architecture.

IEEE Std 1609.3™, Standard for Wireless Access in Vehicular Environments (WAVE)—Networking Services.

IEEE Std 1609.12™, IEEE Standard for Wireless Access in Vehicular Environments (WAVE)—Identifiers.

IETF Request for Comments: 3610, Counter with CBC-MAC (CCM).⁸

IETF Request for Comments: 3629, UTF-8, A Transformation Format of ISO 10646.

IETF Request for Comments: 5639, Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation.

IETF RFC 4648, The Base16, Base32, and Base64 Data Encodings.

ISO/IEC 14888-3:2018, IT Security techniques—Digital signatures with appendix—Part 3: Discrete logarithm based mechanisms.⁹

⁴ ANSI publications are available from the American National Standards Institute (<https://www.ansi.org/>).

⁵ FIPS publications are available from the National Technical Information Service, U.S. Department of Commerce (<http://www.ntis.org/>).

⁶ The IEEE standards or products referred to in Clause 2 are trademarks owned by The Institute of Electrical and Electronics Engineers, Incorporated.

⁷ IEEE publications are available from The Institute of Electrical and Electronics Engineers (<http://standards.ieee.org/>).

⁸ IETF publications are available from the Internet Engineering Task Force (<https://www.ietf.org/>).

ISO/IEC 10118-3:2018: IT Security techniques—Hash-functions—Part 3: Dedicated hash-functions SM4:
ISO/IEC 18033-3:2010/AMD 1:2021 Information technology—Security techniques—Encryption algorithms—Part 3: Block ciphers—Amendment 1: SM4.

ITU-T Recommendation X.680, Information Technology—Abstract Syntax Notation One (ASN.1) & ASN.1 Encoding Rules.¹⁰

ITU-T Recommendation X.696, Information Technology—ASN.1 Encoding Rules: Specification of Octet Encoding Rules (OER).

NIMA TR8350.2, Department of Defense World Geodetic System 1984, Its Definition and Relationships with Local Geodetic Systems., Jan. 2000.¹¹

NIST SP 800-38C, Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality.¹²

SAC GB/T 32918.4-2016, Information Security Technology—Public Key Cryptographic Algorithm SM2 Based On Elliptic Curves—Part 4: Public Key Encryption Algorithm, July 2018.¹³

SAC GB/T 32918.5-2017, Information Security Technology—Public Key Cryptographic Algorithm SM2 Based On Elliptic Curves—Part 5: Parameter Definition, July 2018.

SECG, “SEC 1: Elliptic Curve Cryptography”, Version 2.0, May 21, 2009.¹⁴

SECG, “SEC 4: Elliptic Curve Qu-Vanstone Implicit Certificate Scheme (ECQV)”, Version 1.0, Jan. 24, 2013.

United Nations Statistics Division, “Composition of Macro Geographical (Continental) Regions, Geographical Sub-Regions, and Selected Economic and Other Groupings” [referred to as “UN Region Codes”], revision of 31 Oct. 2013.

3. Definitions, abbreviations, and acronyms

3.1 Definitions

For the purposes of this standard, the following terms and definitions apply. The *IEEE Standards Dictionary Online* should be consulted for terms not defined in this clause.¹⁵

advanced encryption standard (AES): Federal Information Processing Standard (FIPS) 197, specifying a symmetric block cipher; also, the block cipher specified in that standard.¹⁶

application: A higher layer entity that may make use of WAVE communication facilities.

⁹ ISO publications are available from the International Organization for Standardization (<http://www.iso.org/>) and the American National Standards Institute (<http://www.ansi.org/>).

¹⁰ ITU-T publications are available from the International Telecommunications Union (<http://www.itu.int/>).

¹¹ NIMA publications are available from the National Imagery and Mapping Agency (<https://www.nga.mil/index.html/>).

¹² NIST special publications are available from <http://csrc.nist.gov/publications/nistpubs/>.

¹³ SAC publications are available from the Standardization Administration of the People's Republic of China (<http://www.sac.gov.cn/sacen/>).

¹⁴ SECG publications are available from the Standards for Efficient Cryptography Group (<https://www.secg.org/>).

¹⁵ *IEEE Standards Dictionary Online* is available at: <http://dictionary.ieee.org>. An IEEE Account is required for access to the dictionary, and one can be created at no charge on the dictionary sign-in page.

¹⁶ Information on references can be found in Clause 2.

application permissions: The actions a certificate holder is allowed to take as stated in their certificate. Expressed in this standard using Provider Service Identifiers (PSIDs).

asserted data: Data sent in a signed security services protocol data unit (SPDU), where the SPDU is created by a *carrying secure data exchange entity* (SDEE) and the data will be consumed by a separate *included SDEE*. The signature on the SPDU is generated by the carrying SDEE and indicates that a sending instance of the included SDEE asserts that the data are true.

associated certificate (of a private key): The certificate used to verify signatures generated by that private key.

associated public key (of a certificate): The public key that is used to verify signatures associated with a certificate.

asymmetric cryptographic algorithm: A cryptographic algorithm that uses two related keys, a public key and a private key, such that the public key is derived from the private key. However, given only the public key, it is computationally infeasible to derive the private key.

authenticated channel: A logical communications channel such that the receiver has assurance that the sender is who they claim to be, and that modifications to the data can be detected. A channel may be authenticated by applying cryptographic mechanisms to the channel itself, or by checking the transmitted protocol data units (PDUs) by some out-of-band mechanism.

authentication: A cryptographic service that provides assurance that the sender of a protocol data unit (PDU) is who they claim to be.

authorization: A cryptographic service that provides assurance that the sender of a protocol data unit (PDU) is entitled to certain permissions.

authorization certificate: A certificate that is used to validate application protocol data units (PDUs) other than certificate requests.

block cipher: A symmetric encryption algorithm that processes data in blocks, typically of 8 or 16 octets.

carrying secure data exchange entity (SDEE): A secure data exchange entity that creates signed security services protocol data units (SPDUs) that contain *asserted data* associated with a separate secure data exchange entity, the *included secure data exchange entity*.

certificate: *See digital certificate.*

certificate authority (CA) certificate: A certificate that is used to verify other certificates.

certificate authority (CA): An entity that issues certificates to entities that are entitled to them.

certificate chain: An ordered set of certificates such that each certificate (except for the top) was issued by the certificate above it in the list. *See also: complete certificate chain; partial certificate chain.*

certificate holder: The entity authorized to use a particular digital certificate to establish trust. The certificate holder can carry out operations using the private key corresponding to the certificate's public key. A certificate holder's certificate is referred to as a *locally held certificate*.

certificate management information: Information that allows the secure data service to determine the trustworthiness of certificates and received data.

certificate revocation list (CRL) distribution center: An entity that stores and distributes certificate revocation lists (CRLs).

certificate revocation list (CRL) series (CRL series): An integer used to assign different certificates issued by the same certificate authority (CA) to distinct sets, such that the certificates in different sets appear on different revocation lists if revoked.

certificate revocation list (CRL): A list identifying certificates that have been revoked. *See:* **revocation**.

certificate revocation list (CRL) signer: An entity authorized to sign certificate revocation lists (CRLs).

certificate signing request (CSR): A protocol data unit (PDU) sent from an entity to a certificate authority (CA), requesting that the CA issues a certificate on behalf of the entity.

chains to: A digital certificate *A chains to* another certificate *B* if *B* is above *A* in the certificate chain (q.v.) from *A* to the root.

complete certificate chain: A certificate chain in which the top certificate is a root certificate and the bottom certificate is an end entity certificate.

confidentiality: A cryptographic service that provides assurance that only the intended recipients of a protocol data unit (PDU) can read it.

consistency conditions: Criteria for validity of a signed protocol data unit (PDU) that depend only on the contents of the signed security services protocol data unit (SPDU) and not on the state of the receiver.

counter mode with cipher block chaining message authentication code (CCM): A mode of operation of a block cipher where the data is encrypted with a keystream, which in turn is generated by encrypting an incrementing counter, and in turn authenticated with a message authentication code calculated using cipher block chaining mode.

critical information field: An information field necessary to establish the validity of a signed security services protocol data unit (SPDU).

cryptographic type (of a certificate): How a certificate transfers information about its associated public key (q.v.). Cryptographic types of certificate are **implicit certificate**; **explicit certificate**.

cryptographic verification: The process of determining whether a signature on a signed security services protocol data unit (SPDU) is consistent with the SPDU and the private key.

cryptographically secure hash function: A function that maps an arbitrary-length input into a fixed-length output (the hash value) such that (1) it is computationally infeasible to find an input that maps to a specific hash value and (2) it is computationally infeasible to find two inputs that map to the same hash value. All hash functions used in this standard are cryptographically secure hash functions.

Cryptomaterial Handle (CMH): A reference to a private key and the associated public key or certificate, used to indicate to the secure data service that the referenced key is to be used in a particular operation.

cryptomaterial: A private key, a public key, or a certificate.

data encapsulation approach: How a verifying entity is made aware of the data signed by a signing entity.

NOTE—This approach is supported in this standard using explicit payloads, external data, omitted data, or a combination of the above.¹⁷

data input: The input to a hash operation for signing or verifying that represents the payload provided by the invoking secure data exchange entity (SDEE).

data plane: A component of the abstract architecture containing entities that exchange user data.

decode: To convert an array of octets into a data structure. *Contrast:* **decrypt, encode.**

decrypt: To convert unreadable, encrypted data to readable, decrypted data using a decryption algorithm and a key. *Contrast:* **decode, encrypt.**

decryption algorithm: An algorithm that takes as input ciphertext and a key and (if the correct key is provided) produces the original plaintext.

delta certificate revocation list (delta CRL): A certificate revocation list that carries information only about certificates that were revoked within a certain time period. *Contrast:* **full certificate revocation list.**

digital certificate: A digitally signed document binding a public key to an identity and/or a set of permissions.

direct hashing: Creating a hash of particular data by passing that data through a hash function without including any additional data or processing.

direct issuing certificate: The certificate authority (CA) certificate that is the issuer for a subordinate certificate, that is, the certificate identified by the issuer identifier in that subordinate certificate.

dubious certificate: A certificate for which the most recent certificate revocation list (CRL) is overdue; that is, it is scheduled to be issued and has not yet been received.

elliptic curve cryptography (ECC): A form of public-key cryptography based on the problem of finding discrete logarithms in a group defined over elliptic curves.

Elliptic Curve Digital Signature Algorithm (ECDSA): A digital signature mechanism based on the elliptic curve discrete logarithm problem and standardized in Federal Information Processing Standard (FIPS) 186-4.

Elliptic Curve Integrated Encryption Scheme (ECIES): A public-key encryption mechanism based on the elliptic curve discrete logarithm problem.

encode: To convert a data structure into an array of octets. *Contrast:* **decode, encrypt.**

encrypt: To convert readable data to unreadable, encrypted data using an encryption algorithm and a key. *Contrast:* **decrypt, encode.**

encryption algorithm: An algorithm that takes as input plaintext and a key and produces ciphertext.

encryption certificate: A certificate that contains an encryption key.

end entity: An entity requesting certificates or signing Protocol Data Units. *Contrast:* **certificate authority.**

¹⁷ Notes in text, tables, and figures of a standard are given for information only and do not contain requirements needed to implement this standard.

NOTE—An entity may act as an end entity in one context and as a Certificate Authority in another context.

end entity certificate: A certificate used to validate application PDUs or certificate requests.

enrollment certificate: A certificate used to validate a certificate request (CSR). *Contrast:* **authorization certificate, certificate authority (CA) certificate.**

explicit certificate: A certificate that contains a public key and the certificate authority's signature.

explicit payload: Data included in the calculation of a signature for a security services protocol data unit (SPDU) and that is transmitted within the body of the SPDU. *Contrast:* **external data.**

external data: Data included in the calculation of a signature for a security services protocol data unit (SPDU) and that is not transmitted within the body of the SPDU. *Contrast:* **explicit payload.**

full certificate revocation list (full CRL): A certificate revocation list that carries information about certificates that were revoked and have not expired, regardless of when the revocation took place. *Contrast:* **delta certificate revocation list.**

global consistency conditions: Consistency conditions (q.v.) that apply to all protocol data units (PDUs) regardless of the application area in which they are used.

hash function: *See:* **cryptographically secure hash function.**

hash ID-based revocation: Revocation (q.v.) which identifies certificates to be revoked via their cryptographic hash.

hash value: The output of a hash function.

IEEE 1609.2 security profile: A means for specifying options and parameters that are provided to the IEEE 1609.2 security services by a particular consumer of those services. May be used as part of the specification of a *secure data exchange entity* (q.v.).

implicit certificate: A digital certificate that allows the associated public key to be reconstructed from a reconstruction value and the certificate authority's public key rather than directly providing the associated public key.

included Secure Data Exchange Entity (SDEE): A secure data exchange entity that creates or consumes *asserted data* included within signed security services protocol data units (SPDUs), where the signed SPDUs are signed by a separate secure data exchange entity, the *included secure data exchange entity*.

indirect issuing certificate: Any certificate authority (CA) certificate in the issuing chain of a certificate under consideration other than the CA certificate that directly issued that certificate.

information flow: A transfer of data with at least one sending role defined, designed to accomplish a functional goal.

information subflow: An information flow with a single sending role or, for an information flow where there are multiple sending roles, a subset of the information flow associated with a single sending role.

inherited permissions: Permissions within a subordinate certificate that are communicated by reference to the issuing certificate, rather than stated explicitly within the subordinate certificate.

integrity: A cryptographic service that provides assurance that any changes to a protocol data unit (PDU) made after it is validly created will be detected.

issuing certificate: The certificate that issues a **subordinate certificate**. See: **directly issuing certificate**, **indirectly issuing certificate**

length of certificate chain (In a certificate chain from any certificate authority (CA) certificate to a final certificate, with each certificate in the chain issuing the one after it): The number of certificates in the chain, except for the first one; in other words, the number of certificates in the chain minus 1.

linkage-based revocation: Revocation (q.v.) which identifies certificates to be revoked via a linkage value (q.v.) included in the certificate.

linkage seed: A value used to derive the linkage value for the current time period and linkage seeds for future time periods.

linkage value: A value included in a certificate that enables that certificate to be revoked via linkage-based revocation (q.v.).

linkage value generation function: A function used to calculate one or more linkage values from one or more linkage seeds.

locally held certificate: A certificate on a device such that WAVE Security Services on that device may generate a signature on a security services protocol data unit (SPDU) which verifies correctly with that certificate.

management plane: A component of the abstract architecture containing functions that manage the entities in the data plane.

misbehavior: Behavior that results in Secure Data Exchange Entities (SDEEs) receiving information that could cause them to take incorrect actions.

misbehavior authority: A management component on the network with responsibility for determining which Secure Data Exchange Entities (SDEEs) are responsible for misbehavior.

misbehavior reporting: The activity of providing information to some authority, known as the misbehavior authority, about misbehavior within a particular application area. The subsequent actions taken by the misbehavior authority may be application area specific.

network byte order: An ordering of the bytes of an integer such that the byte transmitted first is the byte containing the most significant bit, and the most significant bit is first in that byte.

non-repudiation (of origin): A cryptographic service whereby the origin of a message can be demonstrated to a third party, preventing the sender from denying that they produced the message.

omitted data: A type of external data, that is, data included in the calculation of a signature for an security services protocol data unit (SPDU) that is not transmitted within the body of the SPDU, where the SPDU contains neither the data itself nor a hash of the data.

off-cycle certificate revocation list (CRL): A certificate revocation list (CRL) that is issued before the “next CRL” time indicated in the previous CRL in the CRL series.

partial certificate chain: A certificate chain that is not a complete certificate chain, that is, the top certificate is not a root certificate or the bottom certificate is not an end entity certificate.

peer-to-peer certificate distribution: A mechanism for allowing Wireless Access in Vehicular Environments (WAVE) Security services to learn certificates from peer instances of WAVE Security Services.

plaintext: Unencrypted data.

properly formed certificate: A certificate that can be parsed correctly according to the data structures and encoding defined in this standard.

Provider Service Identifier (PSID): An identifier of a higher layer service provided by a higher layer entity. (IEEE Std 1609.12™)

Provider Service Identifier (PSID) derived from context: A Provider Service Identifier (PSID) associated with a protocol data unit (PDU) received by a secure data exchange entity (SDEE), where the SDEE does not obtain the PSID from the PDU itself, the SDEE obtains the PSID by some other means.

pseudonym certificate: An authorization certificate (q.v.) that indicates its holder's permissions and not its holder's identity.

pseudonymity: A property wherein an entity's permanent or long-lived identities, and its long-term patterns of behavior, cannot be deduced from its network traffic and are only observable by appropriately authorized parties.

public-key digital signature: A cryptographically secure checksum that is generated using a private key and verified using a public key.

raw public key: A public encryption key that was obtained by the encrypter from a source other than a certificate or the `encryptionKey` field of the `HeaderInfo` in a signed security services protocol data unit (SPDU).

reconstruction value: A value in an implicit certificate that allows the associated public key to be recovered.

recognize an identifier for a region: Can determine whether a two-dimensional location on the surface of the earth lies inside or outside the borders of the identified region.

referenced for use: A certificate is referred to as being referenced for use by a secure data exchange entity (SDEE) if it is in the chain of an end entity certificate used by that SDEE to sign a security services protocol data unit (SPDU).

relevance area: An area to which the information in a security services protocol data unit (SPDU) is relevant, which may be required by the secure data exchange entity (SDEE) specification to be consistent with geographic constraints in the sender's signing certificate.

relevance conditions: Criteria for validity of a signed security services protocol data unit (SPDU) that depend on the local state of the receiver.

relevant geographic information: Information about the geographic applicability of a security services protocol data unit (SPDU) that is used when a receiving secure data exchange entity (SDEE) determines whether a received SPDU is valid.

replay attack: An attack in which an attacker retransmits, possibly with a delay, a message that was originally validly transmitted, with the intent that the receiver takes some action based on the received message.

request secure data exchange entity [request secure data exchange entity (SDEE)]: An SDEE that sends or receives peer-to-peer certificate distribution (P2PCD) requests in its security services protocol data units (SPDUs).

request-sending secure data exchange entity [request-sending secure data exchange entity (SDEE)]: An SDEE that sends peer-to-peer certificate distribution (P2PCD) requests in its secure protocol data units (SPDUs).

request-receiving secure data exchange entity [request-receiving secure data exchange entity (SDEE)]: An that sends or receives peer-to-peer certificate distribution (P2PCD) requests in its secure protocol data units (SPDUs).

request secure protocol data unit [request security services protocol data unit (SPDU)]: An SPDU that contains a peer-to-peer certificate distribution (P2PCD) learning request.

revocation: The publication by a relevant authority of the information that a particular certificate is no longer to be trusted.

revocation information set: Revocation information about a collection of certificates that all belong to the same *revocation set*.

revocation set: A set of certificates that all have their revocation status contained in the same *revocation information set*.

root certificate: A self-signed certificate authority certificate that is used as a trust anchor to verify other certificates.

secure data service: A subset of Wireless Access in Vehicular Environments (WAVE) Security Services providing services that allow secure data service entities to request communications security services to be applied to protocol data units (PDUs).

secure data exchange entity (SDEE): An entity that uses IEEE 1609.2 services to secure any communications.

secure data exchange entity (SDEE)-specific consistency checks: Consistency checks (q.v.) which are specific to an SDEE (q.v.).

secure data exchange entity (SDEE)-verified relevance checks: Relevance checks (q.v.) which cannot be verified within the security services.

security services protocol data unit (SPDU): A protocol data unit which has been processed by Wireless Access in Vehicular Environments (WAVE) Security Services before transmission.

secure session: A communications session providing authenticity, integrity, replay prevention, and (as necessary) confidentiality services to valid participants.

security envelope: The additional data added to a protocol data unit by the security services when transforming it into a security services protocol data unit (SPDU).

security services management entity (SSME): A management entity responsible for managing certificate management information.

security management message: A protocol data unit (PDU) used to manage certificates or information about certificates.

security management: Operations that support acquiring or establishing the validity of the IEEE 1609.2 certificates.

security service-verified relevance conditions: Relevance conditions (q.v.) that can be tested within the security services.

seed evolution function: A function used to derive the linkage seed for the next time period from the linkage seed for the current time period.

self-signed certificate: A certificate whose signature can be verified with the public key in the certificate.

Service Specific Permissions (SSP): A field that indicates the permissions of a particular certificate holder with respect to a particular application area.

signature: *See: public key digital signature.*

subordinate certificate: The certificate that was issued by an *issuing certificate*.

symmetric cryptographic algorithm: A cryptographic algorithm that uses a single key. Knowledge of a symmetric encryption key allows both encryption and decryption. Knowledge of a symmetric authentication key allows generation and verification of message authentication codes.

Symmetric Cryptomaterial Handle (SCMH): A reference to a symmetric key, used to indicate to the secure data service that the referenced key is to be used in a particular operation.

symmetric key: The key for a *symmetric cryptographic algorithm* (q.v.).

Transport Layer Security (TLS): An Internet Engineering Task Force (IETF) protocol, specified in Request for Comments (RFC) 5246 [B16], providing for secure communications over Transmission Control Protocol/Internet Protocol (TCP/IP).

trigger secure data exchange entity [trigger secure data exchange entity (SDEE)]: An SDEE that sends or receives a security services protocol data unit (SPDU) that triggers peer-to-peer certificate distribution.

trigger-sending secure data exchange entity [trigger-sending secure data exchange entity (SDEE)]: An SDEE that sends a security services protocol data unit (SPDU) that triggers peer-to-peer certificate distribution.

trigger-receiving secure data exchange entity [trigger-receiving secure data exchange entity (SDEE)]: An SDEE that receives a security services protocol data unit (SPDU) that triggers peer-to-peer certificate distribution.

trigger security services protocol data unit [trigger security services protocol data unit (SPDU)]: An SPDU that triggers peer-to-peer certificate distribution.

trust anchor: A certificate whose validity does not depend on the validity of other certificates.

unlinkability: A property of a communications system that improves the privacy of its users by making it hard for an outside observer to associate multiple protocol data units (PDUs), sent by a particular user at different locations and times, with each other and/or with that user.

valid certificate: A certificate that is correctly formed, that has not been revoked or expired, and for which a certificate chain to a trust anchor exists.

verification type [of a signed security services protocol data unit (SPDU)]: An indication of whether a signed SPDU is to be verified with a public key associated with a certificate, or a public key included in the protocol data unit (PDU) payload. Takes the values *certificate* or *self-signed*.

whole-certificate hash algorithm: The algorithm used to calculate the hash of a certificate for purposes of identifying that certificate.

Wireless Access in Vehicular Environments (WAVE) device: A device that is compliant to IEEE Std 1609.3™, IEEE Std 1609.4™, IEEE Std 1609.12™, and IEEE Std 802.11™, communicating outside the context of a basic service set. (IEEE Std 1609.3™)

Wireless Access in Vehicular Environments (WAVE) Short Message (WSM): A packet consisting of a WSMP header and WSM Data. (IEEE Std 1609.3™)

Wireless Access in Vehicular Environments (WAVE) Short Message Protocol (WSMP): The protocol specified in IEEE Std 1609.3™ that minimizes communications overhead. (IEEE Std 1609.3™)

3.2 Abbreviations and acronyms

AES	Advanced Encryption Standard
APDU	application protocol data unit
ASN.1	Abstract Syntax Notation 1
BSM	Basic Safety Message
CA	certificate authority
CCM	counter mode with cipher block chaining message authentication code
CMH	Cryptomaterial Handle
COER	Canonical Octet Encoding Rules
CRACA	Certificate Revocation Authorizing Certificate Authority
CRL	certificate revocation list
CSR	certificate signing request
CTL	certificate trust list
DSRC	dedicated short range communications
ECC	elliptic curve cryptography
ECDSA	Elliptic Curve Digital Signature Algorithm
ECIES	Elliptic Curve Integrated Encryption Scheme
EDCA	enhanced distributed channel access
FIPS	Federal Information Processing Standard
GNSS	Global Navigation Satellite System

GPS	Global Positioning System
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPv6	Internet Protocol version 6
ITS	intelligent transportation systems
MAC	medium access control, or message authentication code
NIST	National Institute for Standards and Technology
OBU	on-board unit
P2PCD	peer-to-peer certificate distribution
PDU	protocol data unit
PHY	physical layer
PSID	Provider Service Identifier
RSU	roadside unit
RFC	Request for Comments
SAE	Society of Automotive Engineers
SAP	Service Access Point
SCMH	Symmetric Cryptomaterial Handle
SCMS	Security Credential Management System
SDEE	secure data exchange entity
SDS	secure data service
SM2	Chinese National Public-Key Cryptographic Algorithm
SM3	Chinese National Hash Algorithm
SM4	Chinese National Symmetric Encryption Algorithm
SP	special publication
SPaT	signal phase and timing
SPDU	security services protocol data unit
SSME	security services management entity
SSP	Service Specific Permissions

TLS	Transport Layer Security
TAI	International Atomic Time
TTP	trusted third party
UTC	Coordinated Universal Time
UTF	Unicode Transformation Format
V2I	vehicle-to-infrastructure
V2V	vehicle-to-vehicle
V2X	vehicle-to-everything
WAVE	Wireless Access in Vehicular Environments
WGS	World Geodetic System
WME	WAVE Management Entity
WSM	WAVE Short Message
WSMP	WAVE Short Message Protocol

4. General description

4.1 WAVE protocol stack overview

4.1.1 General

Wireless Access in Vehicular Environments (WAVE) provides a communication protocol stack optimized for the vehicular environment, employing both customized and general-purpose elements as shown in Figure 1. WAVE supports both IP- and non-IP-based data transfers, although individual devices might support only one networking stack. Non-IP-based data transfers are supported through the WAVE Short Message Protocol (WSMP) specified in IEEE Std 1609.3. Channel coordination is a collection of extensions to the IEEE 802.11 medium access control (MAC) specified in IEEE Std 1609.4 [B14]. The WAVE Management Entity (WME) and corresponding network services are specified in IEEE Std 1609.3. WAVE Security Services are specified in this standard. IEEE Std 1609.0 provides a description of the WAVE system architecture and operations.

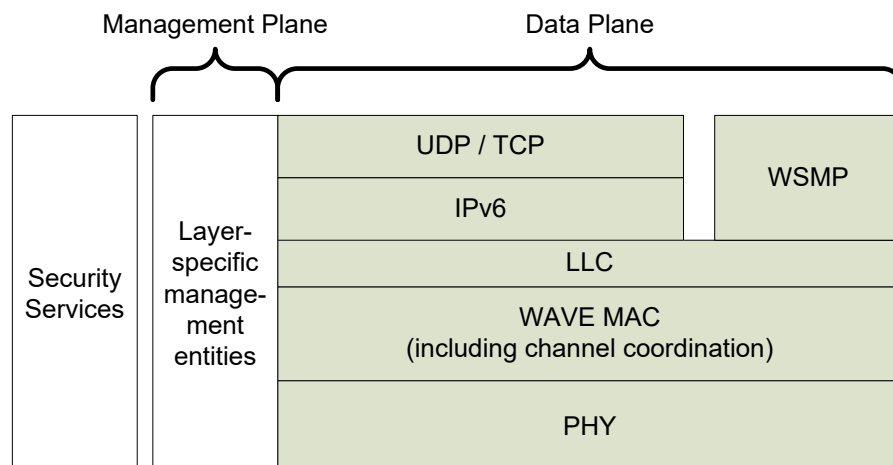


Figure 1—WAVE reference model

This standard specifies a collection of WAVE Security Services available to processes running on platforms, such as WAVE devices. The WAVE Security Services are shown in Figure 2 and consist of WAVE Internal Security Services and WAVE Higher Layer Security Services.

WAVE Internal Security Services are:

- *Secure data service (SDS)*: Transforming unsecured protocol data units (PDUs) into security services protocol data units (SPDUs) to be transferred between entities, and processing SPDUs on reception, including transforming SPDUs into unsecured SPDUs. The additional data added to a PDU when it is transformed into an SPDU is referred to as the *security envelope*. An entity that uses the secure data service is referred to as a *secure data exchange entity (SDEE)*.
- *Security management*: Managing information about certificates as specified in 4.3.

WAVE Higher Layer Security Services are:

- *Certificate revocation list (CRL) verification entity (CRLVE)*: Validates incoming CRLs and passes the related revocation information to the SSME for storage as specified in 5.1.3 and Clause 7.
- *Peer-to-peer certificate distribution (P2PCD) entity (P2PCDE)*: Enables peer-to-peer certificate distribution as specified in Clause 8.

The services and entities within the WAVE Security Services are shown in Figure 2, which also shows Service Access Points (SAPs) that support communications between WAVE Security Services entities and other entities. This standard specifies information element exchanges to support security processing via primitives defined at these SAPs. The Sec-SAP is used by higher layer entities and by the WME. Additionally, the certain SDS operations involve invoking certain primitives across the SSME-SAP as specified in 4.3.

Information elements passed across the SAPs in this standard are assumed to be secure and trustworthy. This standard does not provide mechanisms to ensure the trustworthiness of these information elements.

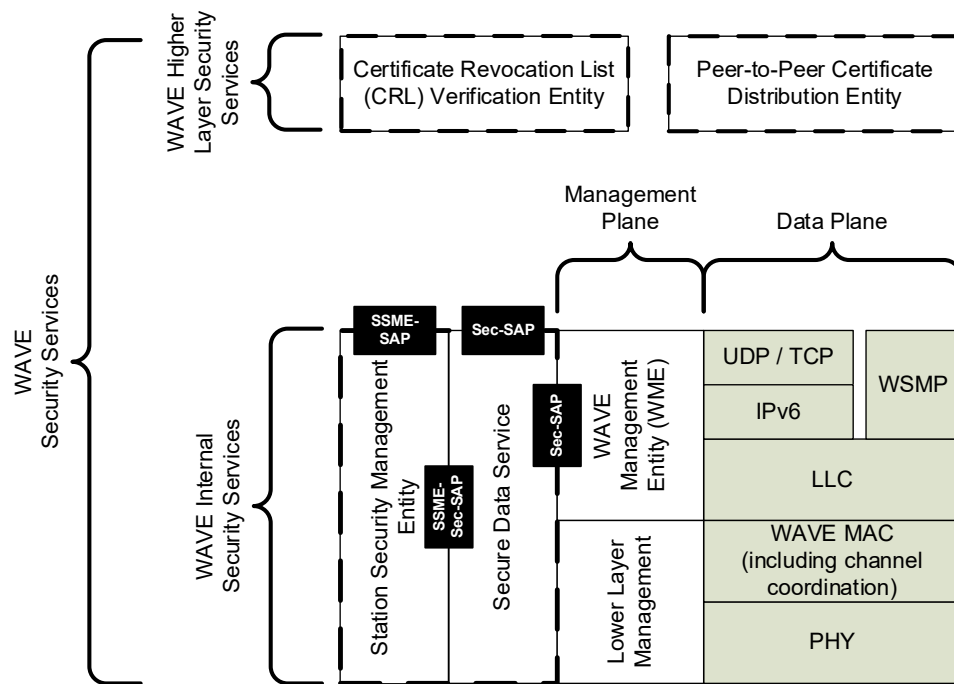


Figure 2—WAVE protocol stack showing detail of WAVE Security Services

Figure 3 shows the general model for security processing using the SDS. The SDS is invoked by a secure data exchange entity (SDEE) with a request to process data; the resulting processed data is returned to the invoking SDEE. A secure data exchange involves two SDEEs, one sending and one receiving. Transmission and reception of the SPDU are not specified in this standard.

The sending SDEE invokes the secure data service to perform sender-side security processing. The result of the processing is an SPDU, which is returned to the sending entity. The sending SDEE invokes the secure data service at least once, and possibly multiple times, prior to transmission of an SPDU.

The receiving SDEE invokes the secure data service to perform security processing on the contents of a received SPDU. The results of the processing, which may include an SPDU and may include additional information about the SPDU, are returned to the receiving SDEE. Complete processing of a received SPDU might require multiple invocations of the SDS.

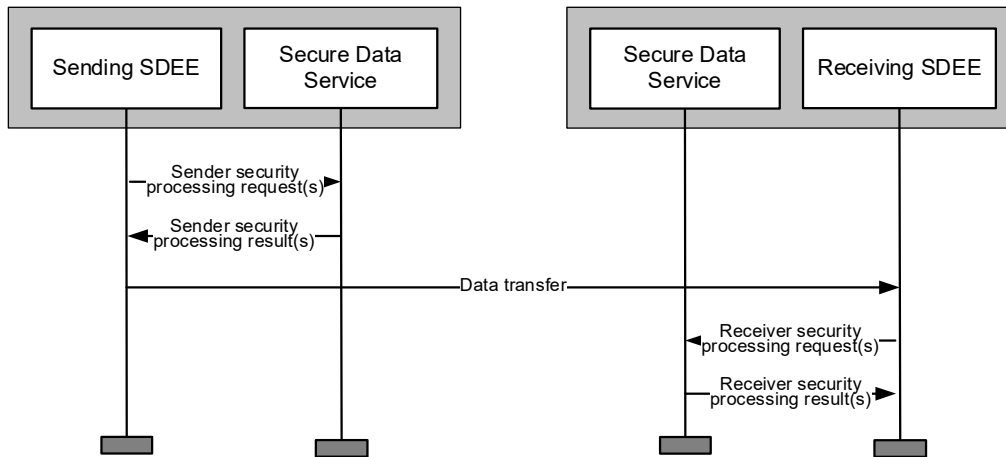


Figure 3 —Process flow for use of IEEE 1609.2 secure data service

At a minimum, an implementation of WAVE Security Services shall support at least one of the following:

- Generate signed SPDU (see 4.2.2.2.3)
- Verify signed SPDU (see 4.2.2.3.2)
- Generate encrypted SPDU (see 4.2.2.2.4)
- Decrypt encrypted SPDU (see 4.2.2.3.3)
- The CRL Verification Entity (see Clause 6.5)
- The Peer-to-Peer Certificate Distribution Entity (P2PCDE) (see 8.3)

4.1.2 Terminology—information flow and information subflow

The security services defined in this standard operate “memory to memory”; in other words, they are invoked by an SDEE and return the result to the same SDEE independent of how the SPDU will be or was transferred from a sender to (a) recipient(s). However, when an SDEE specifier specifies the use of security services, the decision about what services to use will be influenced by the content and function of the “information flows” that transfer the SPDU. In this standard, an *information flow* refers to a transfer of data with at least one node in a sending role, designed to accomplish a functional goal. An *information subflow* is an information flow with a single sending role or, for an information flow with multiple sending roles, a subset of the information flow associated with a single sending role. For example:

- A broadcast application has a single information flow with a single sending role.
- An application where a client requests data from a server has a single information flow with two subflows: the subflow where the client is the sender of the request, and the subflow where the server is the sender of the response. This is important from the security point of view because the security services (or security service options) applied to the request may be different from those applied to the response.
- An application where a client provides data to a server, and the server provides data to the client, may be considered to have two distinct information flows (client to server and server to client) or one information flow with two subflows. The choice of which perspective to use does not have security implications.

4.2 Secure data service (SDS)

4.2.1 Security services protocol data units (SPDUs)

The SDS operations create or process SPDUs. SPDUs are the datagrams that are exchanged between instances of SDEEs that make use of the security services. SPDUs are used for two purposes:

- a) To provide cryptographic protection of the contents.
- b) To provide security management information to be exchanged to allow the correct processing of other SPDUs.

SPDUs may be of type *unsecured*, *signed*, *encrypted*, or *signed certificate request*. An SPDU may contain another SPDU of the same or different type.

4.2.2 Secure data service

4.2.2.1 SDEE identifier

If an implementation of the secure data exchange services supports being invoked by multiple SDEEs, the SDS distinguishes between different SDEEs that invoke them for purposes of replay detection when verifying signed messages (see 5.2.7.2.6), and peer-to-peer certificate distribution (see Clause 8). The mechanism by which this is supported is implementation specific. In the primitives defined in Clause 9, SDEEs are distinguished by an SDEE identifier value which is distinct for distinct SDEEs.

NOTE—One primitive in this standard, SSME-P2pcdRequesterSupport.request, assumes that one SDEE can know properties of other SDEEs that use the same instance of WAVE Security Services, including the SDEE ID among other properties. This standard does not provide primitives to support discovery by one SDEE of the properties of the other; those mechanisms are implementation specific.

4.2.2.2 Generate SPDUs

4.2.2.2.1 Types of SPDUs

This standard defines the following types of SPDUs: *unsecured*, *signed*, and *encrypted*.

4.2.2.2.2 Unsecured SPDUs

An unsecured SPDU is an encoded `Ieee1609Dot2Data` indicating content of type `unsecuredData` as defined in Clause 6. The SDS interface specified in this standard does not provide a primitive to create an unsecured SPDU as this can be implemented trivially: see 6.3.41 for encoding details.

4.2.2.2.3 Signed SPDUs

A signed SPDU is an encoded `Ieee1609Dot2Data` indicating content of type `signedData` as defined in Clause 6. The SDS may provide the service of generating a signed SPDU. This service provides:

- a) *Authenticity*—assurance that the sender is who they claim to be

- b) *Authorization*—assurance that the sender is entitled to the privileges requested
- c) *Integrity*—assurance that any changes to the SPDU after it is signed can be detected
- d) *Non-repudiation (of origin)*—the ability to demonstrate authenticity, authorization, and integrity to a third party

The SDS is requested to generate a signed SPDU via `Sec-SignedData.request`. The SDS returns the result of the request to the requesting SDEE via `Sec-SignedData.confirm`. The result shall be one of the following:

- On success, an octet string containing an encoded `Ieee1609Dot2Data` as defined in Clause 6 containing a signed SPDU that is valid by the criteria specified in 5.2
- On failure, because it was not possible for the SDS to generate a valid signed SPDU, an indication of the reason for failure (or one of the reasons for failure, if there were multiple reasons)

The authorization of a signer of a signed SPDU to create that signed SPDU is established using the signer's certificate. See the specification of `ToBeSignedCertificate` for more details. A signer's permissions include:

- Permissions to sign information flows specified in a particular application specification, indicated by the Provider Service Identifier (PSID) associated with the application specification in the `appPermissions` field of the certificate.
 - This standard provides a mechanism to subset these permissions to allow only particular activities from the application (e.g., only particular information flows, only activities associated with particular participant roles, or only particular values of certain fields) using the Service Specific Permissions (SSP) associated with the PSID. The SSP is included in the `appPermissions` field of the certificate along with the PSID with which it is associated.
 - The SPDU design also supports the case when data for SDEE *A* is sent within a signed SPDU for SDEE *B*, and a single signature on the SPDU is considered to provide proof of authorization to both receiving SDEEs. In this case, the SPDU for *B* is referred to as containing *asserted data* for *A*. For more details, see 5.2.6.4.2.
- Permissions to sign information flows with a specific point or area of interest (e.g., the location where the contents of the SPDU were generated). These permissions are included in the `geographicRegion` field of the certificate
- Permissions to sign information flows within a particular time period, that is, the validity period of the certificate. These permissions are included in the `validityPeriod` field of the certificate.

When generating a signed SPDU, the SDS interacts with the SSME to carry out activities to support peer-to-peer certificate distribution (P2PCD) as defined in Clause 8, if the invoking SDEE so requests.

4.2.2.2.4 Encrypted SPDUs

An encrypted SPDU is an encoded `Ieee1609Dot2Data` indicating content of type `encryptedData` as defined in Clause 6. The SDS may provide the service of generating an encrypted SPDU. This service provides:

- *Confidentiality*—assurance that only the intended recipient(s) can read the contents of the SPDU

The SDS is requested to generate an encrypted SPDU via `Sec-EncryptedData.request`. The SDS returns the result of the request to the requesting SDEE via `Sec-EncryptedData.confirm`. The inputs shall be both of the following:

- An octet string containing an encoding of an `Ieee1609Dot2Data`.
- One or more *encryption key sources*, from which the SDS obtains encryption keys to carry out the encryption. The encryption keys may be public or symmetric. Each encryption key is referred to as a *recipient key*, and the holder of the corresponding decryption key is referred to as a *recipient*.

The input `Ieee1609Dot2Data` is encrypted using the encryption keys in such a way that the holder or holders of a decryption key corresponding to any of the encryption keys can decrypt the SPDU as specified in 5.3.4. The result shall be one of the following:

- On success, an octet string containing an encoding of an `Ieee1609Dot2Data` containing an encrypted PDU that is valid by the criteria specified in 5.3.4.
- On failure, an indication of the reason for failure.

4.2.2.2.5 Signed certificate request

A signed certificate request SPDU is an encoded `Ieee1609Dot2Data` indicating content of type `signedCertificateRequest` as defined in Clause 6. The SDS may provide the service of generating a signed certificate request. This service provides:

- a) *Authenticity*—assurance that the sender is who they claim to be
- b) *Authorization*—assurance that the sender is entitled to the privileges requested
- c) *Integrity*—assurance that any changes to the SPDU after it is signed can be detected
- d) *Non-repudiation (of origin)*—the ability to demonstrate authenticity, authorization, and integrity to a third party

The authorization of a signer of a signed certificate request to create that signed certificate request is established using the `certRequestPermissions` field in the signer's certificate. See the specification of `ToBeSignedCertificate` for more details. (Contrast with signed SPDU, where the signer's authorization is established using the `appPermissions` field).

This standard does not provide interfaces for generating signed certificate requests. Further material about the use of signed certificate requests is in IEEE Std 1609.2.1.

4.2.2.2.6 Signed X.509 certificate request

A signed X.509 certificate request SPDU is an encoded `Ieee1609Dot2Data` indicating content of type `signedX509CertificateRequest` as defined in Clause 6. The SDS may provide the service of generating a signed X.509 certificate request. This service provides:

- a) *Authenticity*—assurance that the sender is who they claim to be
- b) *Authorization*—assurance that the sender is entitled to the privileges requested
- c) *Integrity*—assurance that any changes to the SPDU after it is signed can be detected
- d) *Non-repudiation (of origin)*—the ability to demonstrate authenticity, authorization, and integrity to a third party

The authorization of a signer of a signed X.509 certificate request to create that signed X.509 certificate request is established using mechanisms out of scope for this standard.

This standard does not provide interfaces for generating signed X.509 certificate requests. Further material about the use of signed X.509 certificate requests is in IEEE Std 1609.2.1.

4.2.2.2.7 SPDU with multiple layers of cryptographic protection

To create an SPDU with multiple layers of cryptographic protection (signed and then encrypted, for example), an SDEE invokes the SDS multiple times, passing the output of one invocation of the SDS as the payload to the next invocation. The SPDU produced by each step contains the SPDU from the previous step (or the original PDU) as the payload and is suitable for processing by either the security services (if an SPDU) or a peer SDEE (if the original PDU).

4.2.2.3 Processing received SPDUs

4.2.2.3.1 Preprocessing

The SDS may provide the service of preprocessing a received SPDU. This enables peer SDSs to exchange management information and to extract information that can be used by the invoking SDEE to decide whether to process the SPDU further.

The SDS shall provide this service if it supports P2PCD as defined in Clause 8.

The SDS shall provide this service if it allows a signer identifier to be of type digest as defined in 6.3.31.

Otherwise, provision of this service is optional.

The SDS is requested to preprocess a received SPDU via `Sec-SecureDataPreprocessing.request`. The SDS returns the result of the request to the requesting SDEE via `Sec-SecureDataPreprocessing.confirm`. The result is:

- The type of the SPDU
- If the SPDU was of type `signedData`:
 - The Service Specific Permissions of the signer as defined in 5.2.6.4.1
 - The geographic validity region of the signer's certificate as defined in 6.4.17
 - The assurance level of the signer's certificate as defined in 6.4.27
 - The earliest Next CRL Time of any certificate in the chain as defined in 5.1.3.6

When providing this service, the SDS extracts security management information and passes it to the SSME to support P2PCD and digest-form `SignerIdentifier` structures.

4.2.2.3.2 Verifying signed SPDUs

The SDS may provide the service of verifying a signed SPDU.

The SDS is requested to verify a signed SPDU via `Sec-SignedDataVerification.request`. The SDS returns the result of the request to the requesting SDEE via `Sec-SignedDataVerification.confirm`. The result shall be a correct indication of whether a signed SPDU is valid, meaning whether it meets the three validity conditions specified in 5.2.1. An SPDU that cannot be determined to be valid is considered invalid.

A conformant implementation shall indicate whether a verified SPDU was determined to be valid or invalid.

A conformant implementation may provide more information than whether the SPDU was valid or invalid. In particular, a conformant implementation may provide information about why an SPDU was determined to be invalid if this was the case. For example, an implementation could distinguish between “determined to be invalid” and “not determined to be valid.” The approach taken by the informative primitive `Sec-SignedDataVerification.confirm` in this standard is to give one reason why the SPDU was determined to be invalid.

If a received SPDU violates multiple validity conditions, that is, if there are multiple reasons why the SPDU could be determined to be invalid, a conformant implementation of verification may provide only one of those reasons or may provide more than one. The result returned via `Sec-SignedDataVerification.confirm` gives information about one and only one of those reasons, and for this primitive, the choice of which reason is provided is implementation specific. An implementation may take a different approach; for example, it might provide all the reasons why an SPDU is invalid, or it might provide only one reason and require that it is the most critical reason (for some ranking of criticality).

Verifying asserted data: The SPDU design supports the case when data for SDEE *A* is sent within a signed SPDU for SDEE *B*, and a single signature on the SPDU provides proof of authorization to both receiving SDEEs. In this case, the SPDU for *B* is referred to as containing *asserted data* for *A*. For more details, see 5.2.6.4.2. For an SPDU containing asserted data to be valid, it is required that the PSID for *B* and the PSID for *A* appear in the `appPermissions` field of the certificate. `Sec-SignedDataVerification.request` provides the capability to check that this condition is met.

NOTE—The decision as to whether to submit an SPDU to the SDS for verification is made by the SDEE, subject to any constraints in the SDEE specification. How an SDEE determines whether to request verification of an SPDU is out of scope for this standard.

4.2.2.3.3 Decrypting encrypted SPDUs

The SDS may provide the service of decrypting an encrypted SPDU.

The SDS is requested to decrypt an encrypted SPDU via `Sec-EncryptedDataDecryption.request`. The SDS returns the result of the request to the requesting SDEE via `Sec-EncryptedDataDecryption.confirm`. The result shall be:

- On success, an SPDU that is the correct decryption of the ciphertext within the encrypted SPDU. This will always be an `Ieee1609Dot2Data`.
- On failure, an indication of the reason for failure. A decryption attempt might fail because the data was not validly encrypted, because valid encrypted data was not received correctly, or because the relevant decryption key is not known to the SDEE.

4.2.3 Cryptomaterial

Cryptographic operations for SDS make use of data of the following types:

- Symmetric keys.
- Private keys and associated certificates. The *associated certificate* for a private key is the certificate for which the associated public key verifies signatures generated with that private key and which was valid at the time of signing with the private key.
- Digital certificates held by peer entities, for which no private key is stored by the SDS.

Symmetric keys, and private keys and associated certificates, are referred to as *cryptomaterial* in this standard. Cryptomaterial is used by the SDS in the following operations:

- Generate signed SPDUs (private keys and certificates only)
- Decrypt encrypted PDUs (private keys or symmetric keys)

The interfaces to the SDS in this standard do not pass secret or private keys, and represent them via the Cryptomaterial Handle (CMH). The CMH is an abstraction of private/secret key storage used in the definition of the interfaces and is defined in 9.2.2.

4.2.4 Peer-to-peer certificate distribution

The SDS may also support peer-to-peer certificate distribution operations as specified in Clause 8.

An SDS implementation may support the requester role functionality specified in 8.2.5.1.

An SDS implementation may support the responder role functionality specified in 8.2.5.2.

4.2.5 IEEE 1609.2 security profile

The information elements used by the SDS operations are specified in 9.2.2 as (sometimes optional) parameters to primitives; the SPDU data structures and their encodings are specified in Clause 6. The *IEEE 1609.2 security profile* (occasionally referred to in this standard simply as the “security profile”) specified in Annex C is a format suggested for use by the specifier of an SDEE as a compact way to specify which SDS parameters are used and which values are taken for that particular SDEE. Additionally, the security profile allows the SDEE specifier to specify other aspects of the security behavior of the SDEE; see Annex C for more information.

Examples of security profiles are in Clause 7 of this standard (for use with CRLs) and in IEEE Std 1609.3 (for use with WAVE Service Advertisements).

4.2.6 Support for privacy via distinct signing certificates

Deployers of the security mechanisms defined in this standard are expected to have an interest in preserving the privacy of system users. One approach that contributes to improving privacy is to make it hard for an outside observer to associate multiple PDUs, sent by a particular user at different locations and times, with each other and/or with that user. This property is referred to as the *unlinkability* of the PDUs. Within a given deployment, there is likely going to be a policy determining which system user types require privacy and so make use of unlinkability. It is natural for such a policy to determine that end-users in private vehicles or using personal devices require privacy and use unlinkability. For other user types, it might be different; for example, in some systems, privacy may be granted to emergency vehicles carrying out privileged activities (to protect the operators from retaliation), while in others, emergency vehicles may be required to operate without privacy (to provide greater accountability to the broader community).

In the case of the WAVE system, many messages are broadcast and might contain static information that could be used to associate the messages from a single user with each other, especially if that information is unique to a particular user. For signed SPDUs, the signing certificate is information that is visible to all receivers and so potentially a field that can be used for linking. If certificates are unique to an individual user (i.e., no certificate can be associated with two different users), then SPDUs that are signed by the same certificate can be easily associated with each other and potentially with that user. This is true even if the certificate used is a pseudonym certificate; that is, the certificate does not contain any identifying information about the user.

The interfaces in this standard allow a signing SDEE to indicate at signing time which certificate is to be used to sign. An SDEE that has access to multiple, concurrently valid, pseudonym certificates can choose to use different certificates at different times to achieve unlinkability and thus preserve its privacy. The use of different certificates, and unlinkability in general, creates a potential issue for misbehavior detection and revocation; if it is necessary to revoke all of an SDEE's certificates, the infrastructure needs to determine which certificates belong to that SDEE (necessarily breaking unlinkability) and a CRL containing all the certificates might become impractically large even with low levels of revocation. The linkage value based CRLs specified in this standard allow efficient revocation of those SDEEs while preserving unlinkability against an eavesdropper (i.e., the fact that the certificates are linked cannot be determined by an eavesdropper prior to revocation). The linkage value based CRLs with two independent linkage seeds additionally provide protection against linkability by an insider at the SCMS. For further discussion, see Annex F of IEEE Std 1609.2.1.

A full treatment of privacy enhancements based on changing certificates needs to take many details into account, such as how many different concurrently valid certificates are needed, what effect this has on the ability of an attacker to mount a Sybil attack (i.e., an attack where an attacker who has compromised a single device can masquerade as multiple devices), how certificate change interacts with other identifiers up and down the stack (e.g., the Medium Access Control [MAC] address or equivalent), and how two or more SDEEs hosted on the same device coordinate certificate changes among them. These issues are out of scope for this standard.

4.3 Security services management entity (SSME)

4.3.1 General

The SSME stores certificates and information about certificates. The information stored by the SSME relates to both certificates for which the corresponding private key is stored by the SDS (i.e., locally held certificates) and certificates for which the corresponding private key is not stored by the SDS, (e.g., those belonging to peer SDEEs and to Certificate Authorities [CAs]).

The SSME stores the following information relating to each certificate that it manages:

- The certificate data.
- The last time relevant revocation information was received, if any (see 5.1.3).
- The next time revocation information is expected to be received, if any (see 5.1.3). This time may be in the past.
- The certificate's verification status, which is one of the following:
 - Verified and trusted, meaning that it satisfies all the validity conditions of 5.1.
 - Chain does not end in trust anchor (see 5.1.2.1).
 - Chain too long for implementation (see 5.1.2.3).
 - Not cryptographically valid (see 5.1.2.3).
 - Not yet cryptographically validated.
 - Inconsistent permissions in chain (see 5.1.2.4).
 - Revoked (see 5.1.3).
 - Dubious (see 5.1.3.6).
 - Certificate or chain contains unsupported critical information fields (see 5.2.8).

- Invalid encoding (certificate or certificate in its chain is not a valid encoding of the data structures in Clause 6).
- Whether or not the certificate is a trust anchor.

This standard refers to certificates whose management information is available within the SSME as certificates that are “known to” or “managed by” the SSME. When certificate information is added to an instance of an SSME, the SSME makes it available to all SDEEs and instances of the SDS that have access to that SSME.

As illustrated in Figure 2, the SSME has two Service Access Points (SAPs) through which other entities communicate with it to obtain and update security management information: one for use primarily by the SDS (SSME-Sec-SAP), and one for use by the SDS and by other entities (SSME-SAP).

The SSME-SAP is used:

- To add information about certificates via SSME-AddCertificate.request and SSME-AddCertificate.confirm
- To provide information about managed certificates via SSME-CertificateInfo.request and SSME-CertificateInfo.confirm
- To request that a certificate is verified via SSME-VerifyCertificate.request and SSME-VerifyCertificate.confirm
- To request deletion of information about a certificate via SSME-DeleteCertificate.request and SSME-DeleteCertificate.confirm
- To add a certificate to its list of trust anchors via SSME-AddTrustAnchor.request and SSME-AddTrustAnchor.confirm
- To update revocation information for a certificate via SSME-AddHashIdBasedRevocation.request, SSME-AddHashIdBasedRevocation.confirm, SSME-AddIndividualLinkageBasedRevocation.request, SSME-AddIndividualLinkageBasedRevocation.confirm, SSME-AddGroupLinkageBasedRevocation.request, and SSME-AddGroupLinkageBasedRevocation.confirm
- To update information about CRLs relevant to managed certificates via SSME-AddRevocationInfo.request and SSME-AddRevocationInfo.confirm
- To provide information about CRLs relevant to managed certificates via SSME-RevocationInformationStatus.request and SSME-RevocationInformationStatus.confirm
- To enable application processes associated with peer-to-peer certificate distribution via SSME-P2pcdResponseGenerationService.request, SSME-P2pcdResponseGenerationService.confirm, and SSME-P2pcdResponseGeneration.indication
- To enable configuration of peer-to-peer certificate distribution via SSME-P2pcdConfiguration.request, SSME-P2pcdConfiguration.confirm, SSME-P2pcdRequesterSupport.request and SSME-P2pcdRequesterSupport.confirm.

The SSME-Sec-SAP is used:

- To provide information about replayed PDUs via SSME-Sec-ReplayDetection.request and SSME-Sec-ReplayDetection.confirm
- To provide information to enable peer-to-peer certificate distribution as specified in Clause 8 via SSME-Sec-IncomingP2pcdInfo.request, SSME-Sec-IncomingP2pcdInfo.confirm, SSME-Sec-OutgoingP2pcdInfo.request, and SSME-Sec-OutgoingP2pcdInfo.confirm

In addition to the SSME-Sec-SAP primitives, the SDS operations in this standard invoke the following SSME-SAP primitives:

- SSME-AddCertificate.request
- SSME-CertificateInfo.request
- SSME-VerifyCertificate.request
- SSME-RevocationInformationStatus.request

The SSME-SAP specification in this standard assumes that all information provided to the SSME is trustworthy; how this is ensured is outside the scope of this standard.

The interfaces defined in this standard make use of the SDEE ID described in 4.2.2.1.

4.3.2 Peer-to-peer certificate distribution

The SSME may also support peer-to-peer certificate distribution operations as specified in Clause 8.

An SSME implementation may support the requester role functionality specified in 8.2.5.1.

An SSME implementation may support the responder role functionality specified in 8.2.5.2.

4.4 Behavior of SDEEs

This standard specifies WAVE Security Services and does not specify SDEE behavior. However, WAVE Security Services protect SDEEs most effectively if used appropriately. D.1 provides guidance for implementers of SDEEs.

5. Cryptographic operations and validity

5.1 Certificate validity

5.1.1 Certificate contents

A certificate is a data structure used to transport the following information:

- A public key used to verify digital signatures (the “verification key”)
- The permissions associated with that public key
- Optionally, a public key that may be used to encrypt data
- An identifier for the issuer
- Information that may be used to determine whether or not the certificate has been revoked as specified in 5.1.3
- A cryptographic demonstration that the issuer authorized the linkage between the public key and the permissions

The entity that uses the private key corresponding to the public key is referred to as the *certificate holder*.

“Permissions” consist of:

- Geographic permissions: The region within which the certificate is valid, if relevant
- Validity period: The time period within which the certificate is valid
- Application permissions: The activities other than certificate request and issuance that the holder is allowed to perform
- Certificate issuance permissions: The type(s) of certificate, if any, that the holder is permitted to issue
- Certificate request permissions: The type(s) of certificate request, if any, that the holder is permitted to generate

Provider Service Identifiers (PSIDs) are used in certificates to specify permitted application areas. The PSID is defined in IEEE Std 1609.12™. Since the PSID identifies the SDEE, a new SDEE specification is likely to need a specifically allocated PSID. Details on how to apply for a PSID are available on the IEEE Registration Authority website, <https://standards.ieee.org/regauth/>.

A Service Specific Permission (SSP) is provided (explicitly or implicitly) for each PSID in the Application Permissions, identifying specific sender permissions within that PSID’s application area. The syntax and semantics of the SSP are specific to each PSID value.

When a certificate is being used to authorize application PDUs it is referred to as an *authorization certificate*.

Certificate issuance permissions, that is, the permissions that govern what certificates a CA is authorized to issue, are expressed using the following information elements (see 6.4.31 for a full specification):

- One or more PSIDs.
- For each PSID, the SSP Range, which identifies the SSPs associated with that PSID for which the CA is permitted to grant permissions.
- The permissible length(s) of the certificate chain from the certificate containing these issuance permissions to the certificate that signs the PDU. Certificate chain length is defined in 5.1.2.1.
- The end entity permissions, which indicate whether the ultimate end entity certificate permits application operations, certificate request operations, or both.

When a certificate is being used to issue certificates, it is referred to as a *CA certificate*.

Certificate request permissions are expressed using the same information elements as certificate issuance permissions. When a certificate is being used to request certificates it is referred to as an *enrollment certificate*.

The “cryptographic demonstration that the issuer authorized the linkage between the verification key and the permissions” comes in two forms, referred to as *explicit* and *implicit certificates*:

- If the verification key is explicitly given in the certificate, the certificate is an *explicit certificate*. In this case the cryptographic demonstration that the issuer authorized the linkage is provided by the issuer’s signature on the certificate.

- If the verification key is not explicitly given in the certificate, and is obtained from a *reconstruction value* in the certificate and the issuer’s public key via the reconstruction function specified in 5.3.2, the certificate is an *implicit certificate* and the corresponding verification key is referred to as the *associated public key*.¹⁸ In this case the cryptographic demonstration that the issuer authorized the linkage is provided by the fact that a signature verifies correctly with the verification key that was so derived.

The difference between implicit and explicit certificates is illustrated in Figure 4.

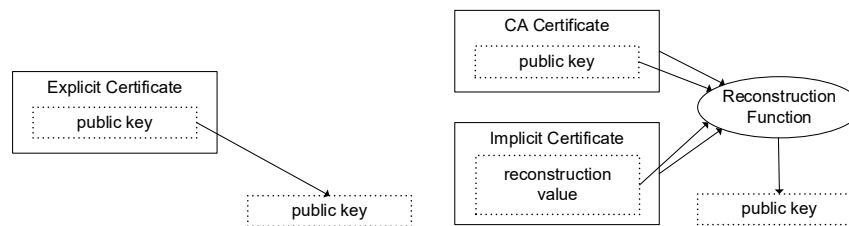


Figure 4—Implicit and explicit certificates

An explicit certificate is invalid if it has an implicit certificate as its issuer. An implicit certificate may have an implicit or an explicit certificate as its issuer.

5.1.2 Certificate chain

5.1.2.1 Certificate chain construction

A *certificate chain* is a set of certificates ordered from “top” to “bottom” (equivalently, “first” to “last” or “beginning” to “end”) such that each certificate in the chain, except the last one, is the *issuing certificate* for one below (or “after”) it and each certificate, except the first one, is the *subordinate certificate* of the certificate above (or “before”) it.

One certificate is the *issuing certificate* for a second one if the certificate holder of the first certificate used the private key of the first certificate to create the final form of the second certificate, either by signing it (in the case of an explicit certificate) or by carrying out cryptographic operations to create a reconstruction value (in the case of an implicit certificate). The counterpart of an issuing certificate is a *subordinate certificate*. If certificate B is the issuing certificate for certificate A, for compactness this standard uses the terminology “B issues A” even though it would be more correct to use the terminology “B’s holder issues A” or “the private key associated with B issues A.”

A *trust anchor* is any certificate that is established to be trustworthy by itself (e.g., by preconfiguration or independent provisioning), in other words, not by reference to any other certificate. A necessary (and not sufficient) condition for a certificate to be valid is that it is possible to construct a certificate chain from the certificate to a trust anchor. The SSME stores information about which certificates are trust anchors.

A *root certificate* is an explicit certificate that is verified with the public key included directly in the certificate, in contrast to other certificates that are verified using the verification key of the issuing certificate. There is no distinct issuing certificate for a root certificate. All trusted root certificates are by definition trust anchors. A certificate chain that starts with a root certificate is referred to in this standard as a *full certificate chain*.

¹⁸ Elliptic Curve Qu-Vanstone (ECQV) or “implicit” certificates were proposed in Brown et al. [B3] and Pintsov and Vanstone [B25], and modifications to protect against attacks were proposed in Brown et al. [B4].

The *length of a certificate chain* is defined as the number of certificates in the chain apart from the topmost one or, equivalently, as the number of intra-certificate gaps in the chain. Figure 5 illustrates these two definitions. Figure 6 illustrates that the definition also applies to a “subchain” within a longer chain; the definition does not assume that the topmost certificate is a root certificate or that the bottom certificate is an end entity certificate. Figure 7 shows a chain of length 0, that is, a single certificate.

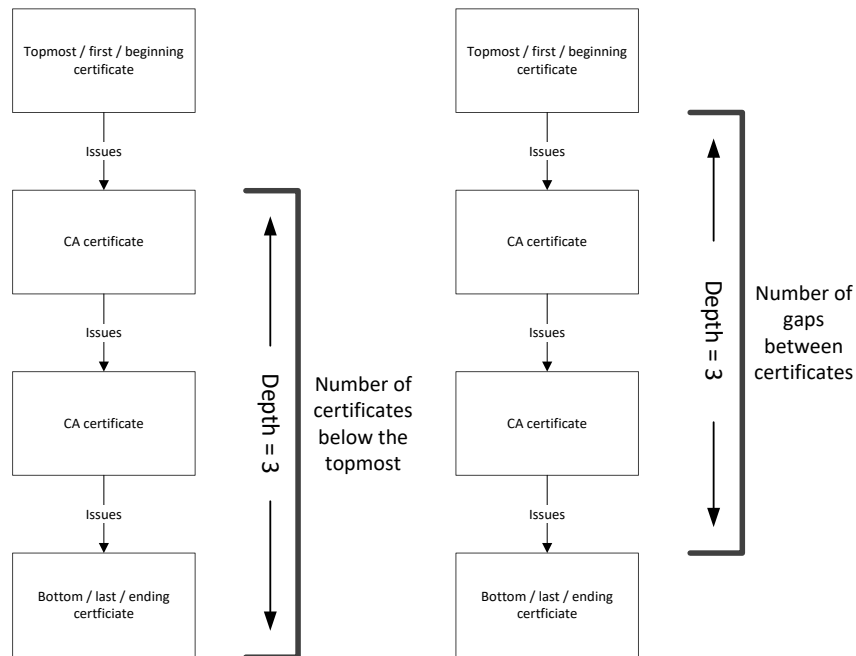


Figure 5—Certificate chain of length 3

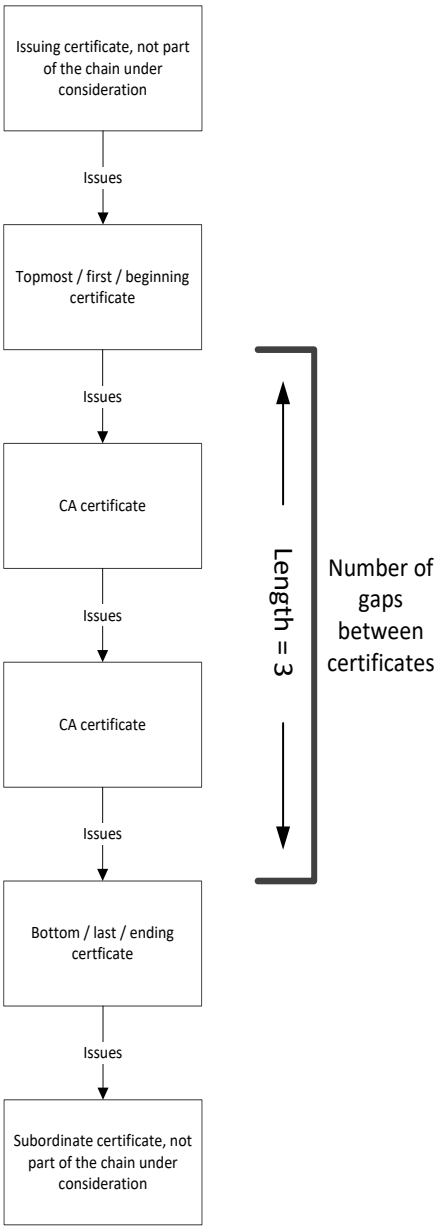


Figure 6—Subchain of length 3 within a longer chain

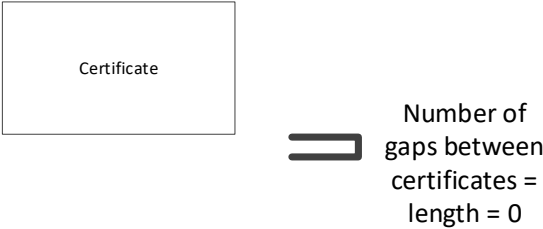


Figure 7—Chain of length 0, that is, a single certificate

To enable construction of certificate chains, each IEEE 1609.2 certificate contains an identifier for its issuer. An example algorithm for the construction of certificate chains is specified in D.3.1 and illustrated in D.3.2. In constructing the certificate chain, the receiver uses a combination of the certificates included in the signed SPDU and locally cached copies of certificates. Local copies of certificates are managed by the SSME.

The issuer identifier is obtained by calculating an eight-octet cryptographic digest of the issuing certificate. There is therefore a 2^{-64} probability that any pair of CA certificates known to the SSME have the same eight-byte digest. If this happens, then when a certificate or SPDU comes to be validated, the SDS constructs two chains, both of which are consistent with the issuer identifiers. The certificate or SPDU is considered valid if either of the chains is cryptographically valid.

5.1.2.2 Maximum supported certificate chain length

An implementation of WAVE Security Services may have a maximum length of full certificate chain that it supports. A conformant implementation shall support a maximum length of at least two, that is, a maximum total number of certificates in the chain of at least three. The Protocol Implementation Conformance Statement (PICS) proforma given in Annex A allows the vendor of an implementation of WAVE Security Services to state the maximum length of certificate chain that the implementation supports.

5.1.2.3 Cryptographic validity of a chain

A certificate chain is cryptographically valid if the following conditions hold:

- For each explicit certificate in the chain, the signature on that certificate can be cryptographically verified with the public key in the issuing certificate. See Figure 8 for an illustration.
- If the chain ends with one or more implicit certificates: the signature on a signed SPDU can be verified with the associated public key from the last certificate in the chain.

If the chain ends with one or more implicit certificates, and no signed SPDU can be verified with the associated public key from the last certificate in the chain, then the cryptographic validity of the implicit certificates is undetermined. If an implicit certificate is known to be valid, then an implementation has the option of deriving the public key and caching it, marked as valid, for faster verification of later messages signed with that certificate. If the validity of an implicit certificate is undetermined, this optimization is not available to an implementation because the derived public key cannot be marked as valid.

See Figure 8 for an illustration involving explicit certificates. See Figure 9 for an illustration involving an implicit authorization certificate.

For all signed SPDUs, the hash operation, signature, and verification are carried out as specified in 5.3.1. The encoding of data structures for input to those cryptographic operations is defined in Clause 6.

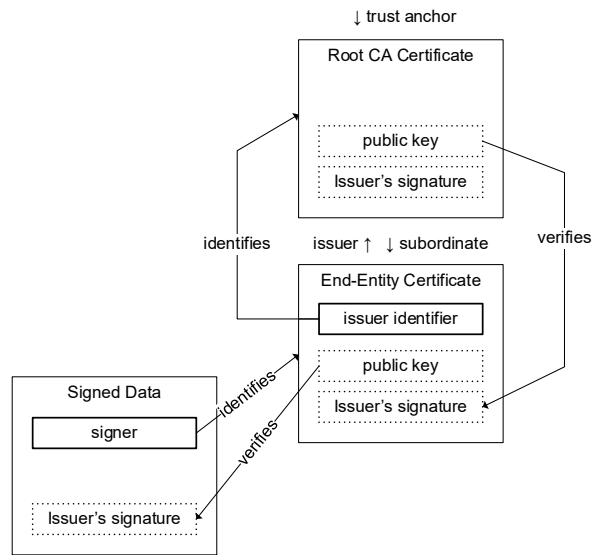


Figure 8—Cryptographic verification of a signed SPDU with a full certificate chain using explicit certificates

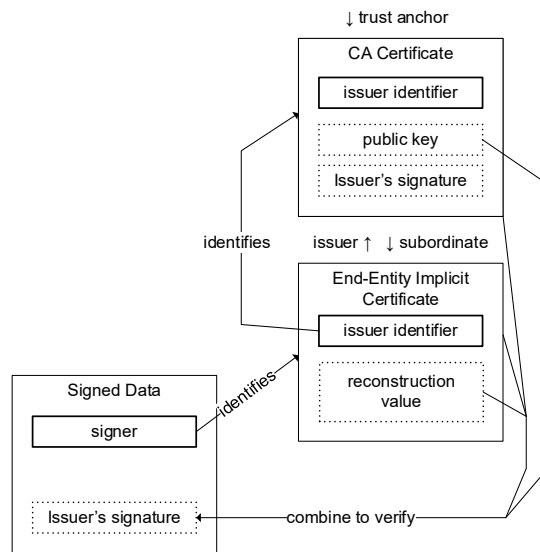


Figure 9—Cryptographic verification of a signed SPDU with a (non-full) certificate chain with implicit end entity certificate

For further illustrations, see D.3.1.

5.1.2.4 Consistency of permissions within a certificate chain

In a certificate chain associated with a valid signed SPDU, the certificate that signs the PDU includes application permissions, and all other certificates include certificate issuance permissions.

A valid certificate's permissions are consistent with its issuer's permissions. A valid certificate chain contains only valid certificates.

A self-signed certificate (i.e., a root CA certificate) is consistent with itself by definition. A subordinate certificate is consistent with its issuing certificate if the following conditions hold:

- **Geographic consistency:** No point in the subordinate certificate's validity region is outside the issuing certificate's validity region.
- **Validity period consistency:** The subordinate certificate's validity period is within the issuing certificate's validity period.
- **Consistency of application/issuance permissions:** The subordinate certificate's application or certificate issuance permissions are consistent with the issuing certificate's certificate issuance permissions; that is, for every (PSID, SSP) entry in the subordinate certificate's application or certificate issuance permissions (the "subordinate entry"), there is an entry in the issuing certificate's certificate issuance permissions (the "issuing entry") such that:
 - The PSID in the subordinate entry is the same as the PSID in the issuing entry.
 - The SSP or SSP Range in the subordinate entry is equal to or a subset of the SSP Range in the issuing entry. See 6.4.35 for a full definition of consistency of SSP with SSP Range, or of one SSP Range with another.
 - The permitted length of the chain in the subordinate entry is consistent with the permitted length of the chain in the issuing entry. Specifically:
 - If the subordinate entry is for certificate issuance, then, denoting the fields *minChainLength* and *chainLengthRange* in the subordinate entry and issuing entry by mcl_s , clr_s , mcl_i , clr_i , respectively, the validity condition is $mcl_i \leq mcl_s + 1$ and $(mcl_i + clr_i) \geq (mcl_s + clr_s + 1)$.
 - If the subordinate certificate is an end entity certificate, then, denoting the fields *minChainLength* and *chainLengthRange* in the issuing entry by mcl_i , clr_i , the validity condition is $mcl_i \leq 1$ and $(mcl_i + clr_i) \geq 1$.
 - The eeType field in the issuing entry permits the subordinate entry, that is:
 - If the subordinate entry is in the application permissions field, then:
 - The eeType field in the issuing entry includes the value *app*.
 - If the subordinate entry is in the certificate issuance permissions field, then:
 - All the values of eeType in the subordinate entry also appear in the issuing entry.
- **Consistency of extensions:**
 - For every appExtension field in the subordinate certificate, one of the following holds:
 - The definition of the appExtension in Clause 6 states that there is no requirement for consistency with the issuing certificate.
 - The definition of the appExtension in Clause 6 specifies conditions for consistency with the certIssueExtension field in the issuing certificate, and the appExtension is consistent with the issuing certificate's certIssueExtension per that specification.
 - Every certIssueExtension field in the subordinate certificate is consistent with the certIssueExtension field in the issuing certificate, as specified in the definition of the certIssueExtension in Clause 6.

NOTE—In this version of this standard, no certIssueExtensions are defined; these consistency conditions are provided because this approach will be followed if future appExtensions require permissions in the issuing certificate.

— **Cryptographic consistency:**

- If the issuing certificate is an explicit certificate: the verification key from the issuing certificate can be used to verify the subordinate certificate.
- If the issuing certificate is an implicit certificate: the cryptographic material in the certificates can be used to verify a signature on an SPDU.

Figure 10 and Figure 11 illustrate the definition of consistency of a subordinate certificate with its issuing certificate and capture the internal consistency within a certificate as defined in Clause 6. For clarity, only the relevant fields within the data structures are shown. Clause 9 specifies example processing steps that correctly carry out this check.

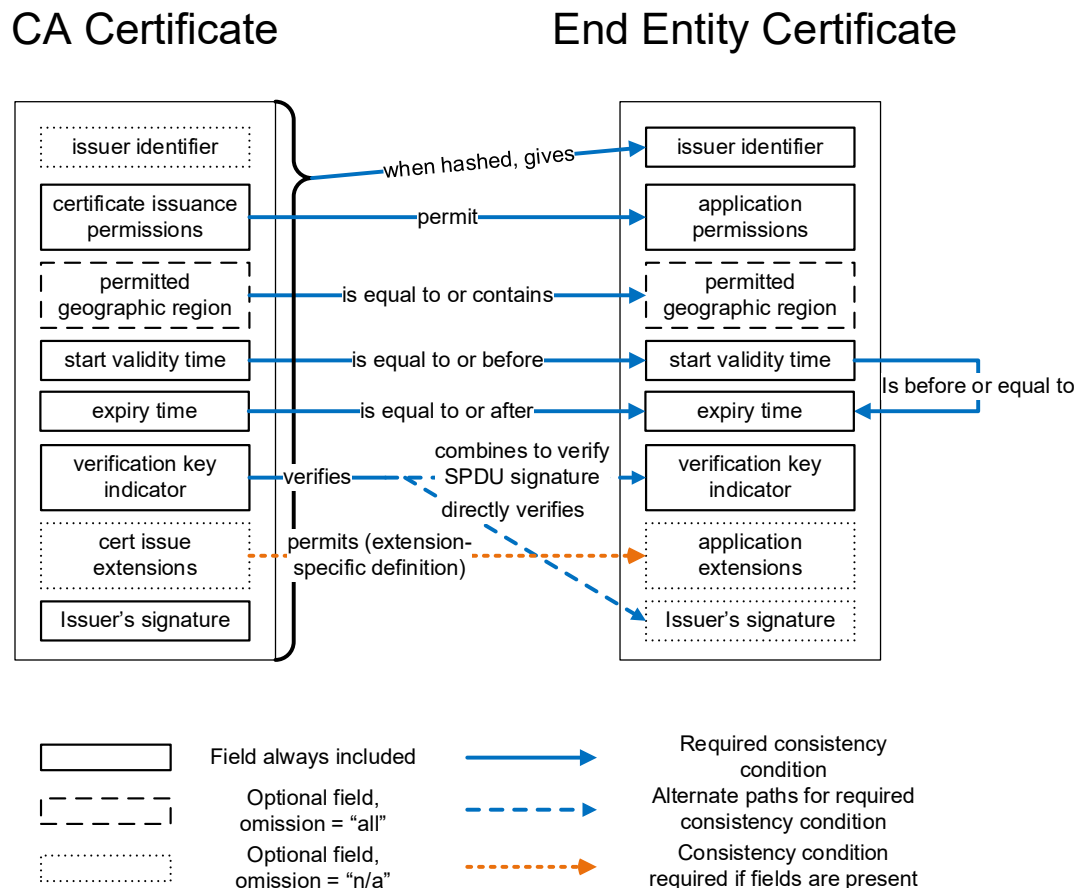


Figure 10—Consistency of permissions between an issuing CA certificate and a subordinate end entity certificate

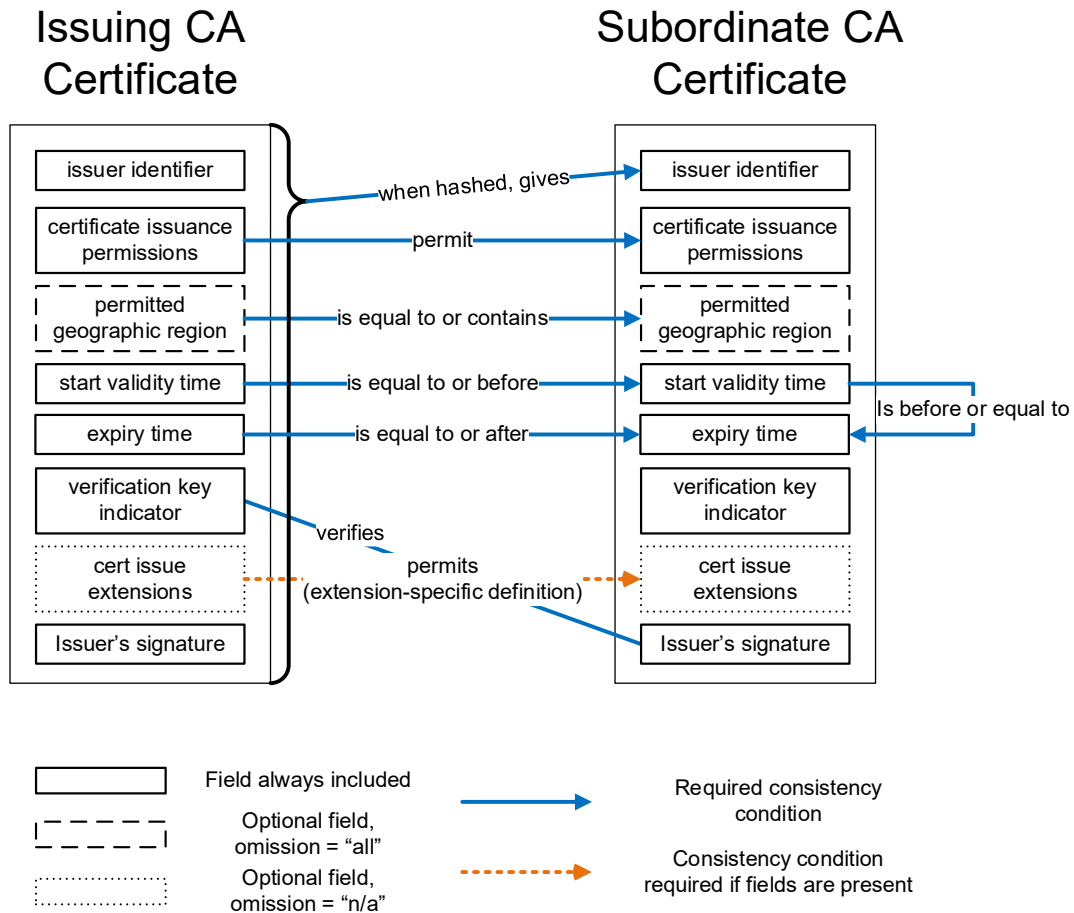


Figure 11—Consistency of permissions between an issuing CA certificate and a subordinate CA certificate

5.1.2.5 Trustworthiness of a certificate chain

A certificate chain is trustworthy if the following hold:

- Each subordinate certificate is consistent with its issuing certificate.
- The chain begins with a trust anchor.
- None of the certificates in the chain have been revoked, as discussed in 5.1.3.
- (Optional) none of the certificates in the chain have expired at the time of chain verification, as discussed in 5.2.6.2.3, 5.2.6.3.3, and 5.2.7.2.7.

Whether the expiry test is applied is specified as part of the SDEE specification, as discussed in 5.2.7.2.7. It is strongly recommended that chains with expired certificates are treated as untrustworthy.

5.1.3 Revocation and expiry

5.1.3.1 General

A certificate is said to be revoked if an appropriately authorized entity states that that certificate is known not to be trustworthy. A signed SPDU is invalid if its signer is revoked. A certificate is invalid if its (direct or indirect) issuing CA certificate is revoked.

Every certificate in the IEEE 1609.2 system is either revocable or not revocable. The certificate contains information stating whether or not it is revocable as specified in 5.1.3.3.

NOTE—If a certificate is marked as not revocable, that means it is not revocable with the mechanisms defined in this standard. A deployment has the ability to specify and use different revocation means that are applied to certificates marked as not revocable. An example of this is the approach taken in ETSI TS 102 941 [B7] and TS 103 097 [B8]. How an implementation recognizes that certificates are revocable through means other than the ones of this standard, and how it maintains revocation information about those certificates, is out of scope for this standard.

The set of revocable certificates is partitioned into different *revocation sets*. The collection of information about whether each certificate in the revocation set is revoked or not is called a *revocation information set*. Each certificate contains information allowing its revocation information set to be identified.

Therefore, the process of determining whether or not a certificate is revoked involves two steps:

- a) Determine which set of revocation information applies to the certificate.
- b) Determine whether any individual data item within the relevant revocation information indicates that the certificate is revoked.

For any given certificate, the set of revocation information that applies to that certificate is identified with the following information, which the verifier of a certificate obtains from that certificate or by other appropriate means:

- The *Certificate Revocation Authorizing CA* (CRACA) certificate
- The *CRL Series* value

For any certificate *C*, there is at most one authority with authorization to issue revocation information for that certificate (each such authority might, however, be entitled to issue and keep up to date more than one set of revocation information).

The rules used to determine whether a set of revocation information applies to a given certificate are defined in 5.1.3.2.

There are two forms of revocation information. The type of revocation information that applies to a certificate is indicated by the `CertificateId` field in the certificate:

- *Linkage-based*: If the `CertificateId` field indicates the choice `linkageData`, the certificate is revoked by publishing the *linkage seed* value corresponding to the `linkageData` value. See 5.1.3.4 for a full description.
- *Hash ID-based*: If the `CertificateId` field indicates the choice `name`, `binaryId`, or `none`, the certificate is revoked by publishing a hash of the certificate. See 5.1.3.5 for a full specification.

If a certificate is revoked, the SDS shall consider all SPDUs signed by that certificate and received after the issue date of the revocation statement to be invalid even if their stated generation time is before the issue date of the revocation statement.

If a CA certificate is revoked, the SSME shall indicate that any certificates issued by that CA certificate and first received after the issue date of the revocation statement are revoked, even if their stated start of validity period is before the issue date of the revocation statement. This applies to any certificate that chains back to the revoked CA certificate.

Information about revoked certificates is stored by the SSME via the SSME-AddRevocationInfo.request and SSME-AddRevocationInfo.confirm primitives. The SSME provides the revocation status of certificates via the SSME-CertificateInfo.request and SSME-CertificateInfo.confirm primitives. Revocation information consists of a series of individual data items and information allowing the SSME to associate the revocation information with specific certificates.

The *certificate revocation list* (CRL) is a data structure for distribution of CRL information. A CRL contains one or more revocation information items. The CRL structure and a specification of how to secure CRLs using the mechanisms of this standard are defined in 6.5.

NOTE—It is the intent of this standard that once a certificate is revoked, it remains revoked for the rest of its lifetime. CRL signers are expected to include a revoked certificate on all CRLs issued between the certificate's revocation and its expiry.

5.1.3.2 Determining which revocation information applies to a given certificate

Revocation information applies to a given certificate if it:

- Indicates that certificate's *Certificate Revocation Authorizing CA* (CRACA) certificate, and
- Indicates that certificate's *CRL Series* value, and
- Is of the appropriate type (linkage-based or hash ID-based)

A CRACA is a CA that has authority to authorize the issuance of revocation information for a particular group of other certificates. The CRACA certificate for a certificate *C* is only valid if it is one of the certificates in *C*'s full chain. Likewise, when the revocation information is transported in the form of a signed CRL, the CRACA certificate is only valid if it either signed the CRL itself, or issued the certificate that signed the CRL.

The CRL Series value is an integer that allows a CA to partition its issued certificates into groups that appear on different CRLs.

A certificate indicates the relevant CRACA certificate, CRL series value, and revocation type as specified in 6.4.8. If revocation information is received via a CRL, that CRL indicates the relevant CRACA certificate, CRL series value, and revocation type as specified in 7.3. The primitives SSME-AddRevocationInfo.request, SSME-AddRevocationInfo.confirm, SSME-CertificateInfo.request and SSME-CertificateInfo.confirm allow the CRACA certificate and CRL series to be associated with certificates and revocation information managed by the SSME.

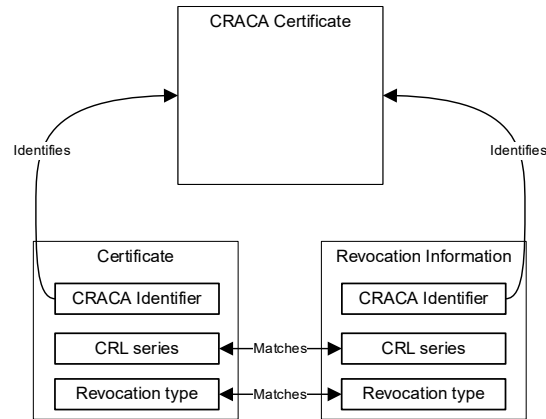


Figure 12—Relationship between certificate and revocation information

When revocation information for a given certificate is transported in a signed CRL, the CRL is validly authorized only if one of the following conditions holds:

- The certificate’s CRACA signed the CRL.
- The certificate’s CRACA issued a certificate which signed the CRL (referred to as a *CRL signer* certificate).

Examples of the two types of acceptable relationship between the CRL and the CRACA are given in Figure 13.

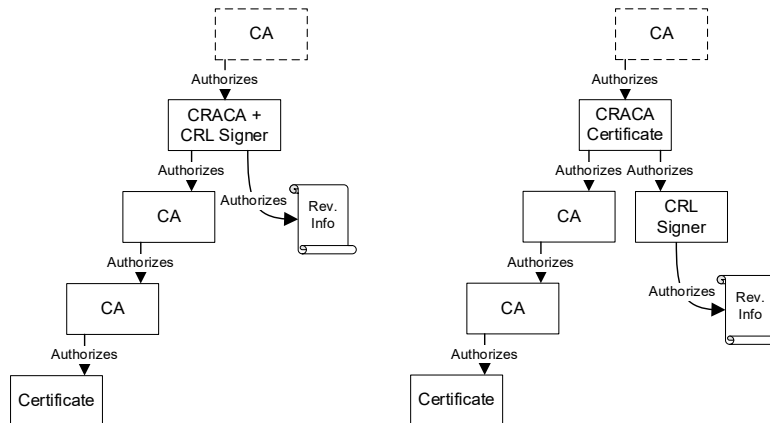


Figure 13—Revocation information: issued by a CRACA (on the left) or by a CRL signer directly authorized by the CRACA (on the right)

5.1.3.3 Identification of CRACA certificate

A certificate contains a *cracaId* field as specified in 6.4.8. This is an octet string of length 3. The relevant CRACA certificate is the certificate in the full chain for which the low-order 3 bytes of its hash are equal to the *cracaId*. The hash of the certificate is obtained as specified in 6.4.3; this specification includes a determination of which hash algorithm to use to obtain the hash. The low-order 3 bytes of the hash are referred to as the *HashedId3*.