



IEEE Standard for Wireless Access in Vehicular Environments (WAVE)— Certificate Management Interfaces for End Entities

IEEE Vehicular Technology Society

Developed by the
Intelligent Transportation Systems Committee

IEEE Std 1609.2.1™-2020

STANDARDS

IEEE Standard for Wireless Access in Vehicular Environments (WAVE)— Certificate Management Interfaces for End Entities

Sponsor

**Intelligent Transportation Systems Committee
of the
IEEE Vehicular Technology Society**

Approved 24 September 2020

IEEE-SA Standards Board

Abstract: Certificate management protocols are specified in this document to support provisioning and management of digital certificates, as specified in IEEE Std 1609.2™, to end entities, that is, an actor that uses digital certificates to authorize application activities.

Keywords: authorization, certificate, end entity, enrollment, IEEE Std 1609™, IEEE Std 1609.2.1™, interface, Security Credential Management System (SCMS)

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

Copyright © 2020 by The Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 30 December 2020. 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 978-1-5044-6974-6 STD24360
Print: ISBN 978-1-5044-6975-3 STDPD24360

IEEE prohibits discrimination, harassment, and bullying.

For more information, visit <http://www.ieee.org/web/aboutus/whatis/policies/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.

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

IEEE Standards documents (standards, recommended practices, and guides), both full-use and trial-use, are developed within IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (“IEEE SA”) Standards Board. IEEE (“the Institute”) develops its standards through a consensus development process, approved by the American National Standards Institute (“ANSI”), which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE Standards are documents developed through scientific, academic, and industry-based technical working groups. Volunteers in IEEE working groups are not necessarily members of the Institute 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 Standards do not guarantee or ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. 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.

IEEE does not warrant or represent the accuracy or content 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. 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: PROCUREMENT OF 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 should be considered 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, or 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 his or her views should be considered the personal views of that individual rather than the formal position of IEEE.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE. However, IEEE does not provide 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 revisions to an IEEE standard is welcome to join the relevant IEEE working group.

Comments on standards should be submitted to the following address:

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

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

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under U.S. 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 fee, 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. 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 at <https://ieeexplore.ieee.org> or contact IEEE at the address listed previously. For more information about the IEEE SA or IEEE's standards development process, visit the IEEE SA Website at <http://standards.ieee.org>.

Errata

Errata, if any, for IEEE standards can be accessed via <https://standards.ieee.org/standard/index.html>. 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: <https://ieeexplore.ieee.org/browse/standards/collection/ieee/>. Users are encouraged to periodically check for errata.

Patents

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken 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.

Participants

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

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

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 balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention:

Benedikt Brecht
Demetrio Bucaneg Jr.
William Byrd
Marc Emmelmann
Kevin Henry
Ronald Hochnadel
Werner Hoelzl
Piotr Karocki
John Kenney
Virendra Kumar

Thomas M. Kurihara
Hyeong Ho Lee
James Lepp
Stephen McCann
Justin McNew
Michael Montemurro
Satoshi Oyama
R. K. Rannow
Randal Roebuck
Richard Roy
Kevin Smith

Manikantan Srinivasan
Jasja Tijink
Jimmy Upton
John Vergis
George Vlantis
Lisa Ward
Karl Weber
Hung-Yu Wei
William Whyte
Yu Yuan

When the IEEE-SA Standards Board approved this standard on 24 September 2020, it had the following membership:

Gary Hoffman, *Chair*
Jon Walter Rosdahl, *Vice Chair*
John D. Kulick, *Past Chair*
Konstantinos Karachalios, *Secretary*

Ted Burse
Doug Edwards
J. Travis Griffith
Grace Gu
Guido R. Hiertz
Joseph L. Koepfinger*

David J. Law
Howard Li
Dong Liu
Kevin Lu
Paul Nikolich
Damir Novosel
Dorothy Stanley

Mehmet Ulema
Lei Wang
Sha Wei
Philip B. Winston
Daidi Zhong
Jingyi Zhou

*Member Emeritus

Introduction

This introduction is not part of IEEE Std 1609.2.1-2020, IEEE Standard for Wireless Access in Vehicular Environments (WAVE)—Certificate Management Interfaces for End Entities.

In intelligent transportation systems (ITS), participants exchange messages to improve safety, efficiency, and mobility for travelers and other participants in the transportation environment. These messages have security requirements that are in many cases addressed by signing the messages using digital certificates. A certificate design optimized for ITS is specified in IEEE Std 1609.2™.

In security terminology, the system participants that use certificates to authorize their application messages are known as “end entities” (EEs). In order for the system to operate, these end entities must be provisioned with certificates for their applications. Crash Avoidance Metrics Partners LLC (CAMP) under a cooperative agreement with the United States Department of Transportation (USDOT) developed a design for a Security Credential Management System (SCMS) to provide security credentials to ITS end entities. This document uses the CAMP design as a starting point to specify certificate management protocols that can be used for all types of ITS end entities to provide efficient, robust, scalable certificate management services.

The EEs receive two types of certificates: enrollment and authorization, issued by an enrollment certificate authority (ECA) and an authorization certificate authority (ACA), respectively. The enrollment certificate that is provisioned during the bootstrapping of an EE certifies that the EE meets all the security requirements for participating in the ITS communications and, hence, enables it to request authorization certificates for use in the actual ITS messages/applications. There are two types of authorization certificates, namely, identification and pseudonym, where the exact type of certificate depends on the EE and the ITS messages/applications used by that EE; for example, for sending Basic Safety Messages, Cooperative Awareness Messages, or Decentralized Environmental Notification Messages, the vehicle onboard unit (OBU) uses pseudonym certificates.

For identification and pseudonym certificate types, a “top-off” system is supported, where the EEs need to make the certificate request only once, and the SCMS automatically generates the certificates at a fixed interval and makes them available for download by the EE at the EE’s convenience. All the certificates generated by the SCMS have unique and random verification keys, despite there being a single request per EE. This is enabled by the butterfly key mechanism, a core part of the SCMS design. Additionally, for the pseudonym certificate type, an EE can request multiple simultaneously valid certificates that are generated in a privacy-preserving manner to help protect the EE’s privacy from both SCMS-insider and -outsider attacks. Finally, for efficient revocation of pseudonym certificates that do not have any identifying information in them, that is, any two pseudonym certificates belonging to an EE look completely random and unrelated to each other, linkages values are used that link all the pseudonym certificates of an EE through a common and secret seed.

This document specifies interfaces only and does not address how devices are to be certified as suitable to receive certificates, nor what the requirements for that certification might be. Those are matters for different standards, or for policy.

Contents

1. Overview	11
1.1 Scope	11
1.2 Purpose	11
1.3 Word usage	11
2. Normative references.....	12
3. Definitions, acronyms, and abbreviations	14
3.1 Definitions	14
3.2 Acronyms and abbreviations	19
4. Architecture	20
4.1 Security Credential Management System (SCMS).....	20
4.2 Interface approach	35
4.3 System parameters	36
5. Secure session.....	43
5.1 General	43
5.2 Physical security	44
5.3 Transport Layer Security (TLS)	44
5.4 ISO/TS 21177	46
6. Web API.....	47
6.1 General	47
6.2 OAuth 2.0 bearer authorization	48
6.3 SCMS REST API v2	51
7. Data structures: ASN.1	77
7.1 Presentation and encoding	77
7.2 Data structures from IEEE Std 1609.2.....	77
7.3 SCMS protocol data unit structures	81
7.4 Secured protocol data unit structures.....	98
7.5 Parameterized types	106
7.6 Misbehavior reporting upload to RA	110
7.7 Certificate profiles	111
8. Data structures—Files	118
8.1 General	118
8.2 Authorization certificate download files.....	118
8.3 Successor enrollment certificate download files.....	121
8.4 Certificate chain files.....	121
8.5 Composite CRL files	122
8.6 CTL files.....	122
8.7 Certificate management information status files	122
9. Cryptographic constructions.....	123
9.1 General	123
9.2 Butterfly keys and SCMS architecture	123
9.3 Butterfly key mechanism.....	129
9.4 ACPC and SCMS architecture.....	136
9.5 Binary hash tree for ACPC activation codes	139

10. Validity conditions for particular SPDUs	150
10.1 Validity of SignedCertificateRequest	150
10.2 Validity of multisigned CTLs	157
11. Extensions and modifications of IEEE Std 1609.2	158
11.1 General	158
11.2 Extensions and modifications of, and new data structures for, IEEE Std 1609.2	158
11.3 Identification of encoded data structures by their hash value	161
Annex A (normative) Protocol Implementation Conformance Statements (PICS) proforma	164
A.1 Instructions for completing the PICS proforma	164
A.2 PICS proforma—IEEE Std 1609.2.1	166
Annex B (normative) ASN.1 modules.....	193
B.1 General.....	193
B.2 End entity interfaces	193
Annex C (informative) Implementation profiles	248
C.1 General.....	248
C.2 Profile template.....	248
C.3 Additional information.....	249
C.4 Use case specific profile contents	250
Annex D (informative) Privacy-preserving generation of linkage values with two linkage authorities	251
Annex E (informative) ACA – RA interface	254
E.1 General	254
E.2 AcaRaInterfacePdu	254
E.3 RaAcaCertRequest	254
E.4 RaAcaCertRequestFlags.....	256
E.5 LinkageInfo	256
E.6 EncryptedIndividualPLV	256
E.7 PreLinkageValue.....	257
E.8 AcaRaCertResponse.....	257
E.9 AcaResponse	257
E.10 RaAcaCertRequestSpdu	258
E.11 AcaRaCertResponseSpdu.....	258
Annex F (informative) CAM – RA interface	259
F.1 General	259
F.2 CamRaInterfacePdu.....	259
F.3 RaCamBatchRequest.....	259
F.4 CamRaBatchResponse	260
Annex G (informative) Mapping enrollment certificate permissions to authorization certificate permissions	261
Annex H (informative) Root CA management and failure recovery	263
H.1 Elector replacement	263
H.2 Revoking a root CA certificate	268
H.3 Adding a root CA certificate.....	271
Annex I (informative) Network communications architecture	277
I.1 General	277
I.2 Location obscurer proxy.....	280

Annex J (informative) CTL design and electors.....	281
J.1 Policy assumptions.....	281
J.2 Comparison of IEEE CTL and ETSI CTL	281
J.3 Comparison of IEEE CTL and CAMP elector Ballots.....	282
J.4 Electors across multiple regions and domains	282
Annex K (normative) Registered values for ElectorGroupId and AcpcTreeId	284
K.1 ElectorGroupId	284
K.2 AcpcTreeId	284
Annex L (informative) Bibliography.....	285

IEEE Standard for Wireless Access in Vehicular Environments (WAVE)— Certificate Management Interfaces for End Entities

1. Overview

1.1 Scope

This standard specifies certificate management protocols to support provisioning and management of digital certificates, as specified in IEEE Std 1609.2™, to end entities, that is, actors that use digital certificates to authorize application activities.¹

1.2 Purpose

IEEE Std 1609.2 specifies services that can be used to protect messages from attacks such as eavesdropping, spoofing, alteration, and replay. These services make use of digital certificates. Standardized interfaces for requesting, obtaining, and managing these certificates significantly assist the deployment of the system. This standard provides those interfaces for use by end entities, that is, entities other than certificate authorities.

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*).^{2, 3}

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*).

¹ Information on references can be found in Clause 2.

² The use of the word *must* is deprecated and cannot be used when stating mandatory requirements, *must* is used only to describe unavoidable situations.

³ The use of *will* is deprecated and cannot be used when stating mandatory requirements, *will* is only used in statements of fact.

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*).

2. Normative references

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

NOTE—This list of normative references includes the “base standard” IEEE Std 1609.2-2016, and its amendments IEEE Std 1609.2a-2017 and IEEE Std 1609.2b-2019. Where the current document refers to “IEEE Std 1609.2”, this is shorthand for the combination of the base standard and its amendments.⁴

FIPS 180-4, Secure Hash Standard (SHS), Aug. 2015.⁵

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

IEEE Std 1609.2™-2016, IEEE Standard for Wireless Access in Vehicular Environments—Security Services for Applications and Management Messages.^{6, 7}

IEEE Std 1609.2a™-2017, IEEE Standard for Wireless Access in Vehicular Environments—Security Services for Applications and Management Messages—Amendment 1.

IEEE Std 1609.2b™-2019, IEEE Standard for Wireless Access in Vehicular Environments—Security Services for Applications and Management Messages—Amendment 2.

IEEE Std 1609.12™-2019, IEEE Standard for Wireless Access in Vehicular Environments (WAVE)—Identifiers.

IETF RFC 4648, The Base16, Base32, and Base64 Data Encodings.⁸

IETF RFC 5246, The Transport Layer Security (TLS) Protocol Version 1.2.

IETF RFC 5280, Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile.

IETF RFC 5289, TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM).

IETF RFC 5480, Elliptic Curve Cryptography Subject Public Key Information.

IETF RFC 6749, The OAuth 2.0 Authorization Framework.

⁴ Notes in text, tables, and figures of a standard are given for information only and do not contain requirements needed to implement this standard.

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/>).

IETF RFC 6750, The OAuth 2.0 Authorization Framework: Bearer Token Usage.

IETF RFC 6960, X.509 Internet Public Key Infrastructure Online Certificate Status Protocol—OCSP.

IETF RFC 6961, The Transport Layer Security (TLS) Multiple Certificate Status Request Extension.

IETF RFC 7230, Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing.

IETF RFC 7231, Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content.

IETF RFC 7232, Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests.

IETF RFC 7233, Hypertext Transfer Protocol (HTTP/1.1): Range Requests.

IETF RFC 7515, JSON Web Signature (JWS). IETF RFC 7519. JSON Web Token (JWT).

IETF RFC 7523, JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants.

IETF RFC 7662, OAuth 2.0 Token Introspection.

IETF RFC 8446, The Transport Layer Security (TLS) Protocol Version 1.3.

ISO 8601-1, Date and time—Representations for information interchange—Part 1: Basic rules.⁹

ISO/TS 21177, Intelligent transport systems—ITS station security services for secure session establishment and authentication between trusted devices.

ITU-T Recommendation X.509, Information Technology—Open Systems Interconnection—The Directory: Public-Key and Attribute Certificate Frameworks.¹⁰

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

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

PKWARE.ZIP File Format Specification, published as APPNOTE.TXT—ZIP File Format Specification version 6.3.6.¹¹

SECG, “SEC 4: Elliptic Curve Qu-Vanstone Implicit Certificate Scheme (ECQV),” Version 1.0, Jan. 24, 2013.¹²

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 PKWARE publications are available from PKWARE, Inc. (<https://www.pkware.com/>).

¹² SECG publications are available from the Standards for Efficient Cryptography Group (<https://www.secg.org/>).

3. Definitions, acronyms, and abbreviations

3.1 Definitions

For the purposes of this document, the following terms and definitions apply. The *IEEE Standards Dictionary Online* should be consulted for terms not defined in this clause.¹³

Since this document specifies functionality that supports the functionality defined in IEEE Std 1609.2, definitions in IEEE Std 1609.2 also apply to this document. In particular, IEEE Std 1609.2 should be consulted for the definitions of the following terms: asymmetric cryptographic algorithm, authentication, authorization, certificate authority, certificate holder, certificate revocation list (CRL), certificate revocation list (CRL) signer, digital certificate, decrypt, elliptic curve cryptography, enrollment certificate, explicit certificate, hash value, implicit certificate, misbehavior, nonrepudiation, peer-to-peer certificate distribution, Provider Service Identifier, reconstruction value, revocation, root certificate, secure session, security management, self-signed certificate, Service Specific Permissions, symmetric cryptographic algorithm, valid certificate.

The following definitions update the corresponding definitions in IEEE Std 1609.2: authorization certificate, end entity, enrollment certificate, misbehavior authority, pseudonym certificate, Transport Layer Security.

application activities: The activities that are carried out to achieve the business or operational goals of a distributed application.

application domain: The collection of *application instances* and management services that carry out and support a specific collection of *application activities*.

application instance: A single instance of an implementation of a component of a distributed application, running on a single device (or perceived as running on a single device from the perspective of other participants in the *application domain*).

authorization certificate: A certificate that is used to authorize *application activities*. *Contrast: enrollment certificate.*

authorization certificate authority (ACA): A certificate authority (CA) whose main responsibility is to issue *authorization certificates*.

binary hash tree: A data structure in which each node at level $l + 1$ has its value derived by applying a hash function to its parent node at level l , such that the publication of one node value at level $l + 1$ allows the derivation of all node values at levels l and below.

blocked enrollment certificate: An *enrollment certificate* that has been determined to be no longer eligible to authorize certificate requests or certificate download requests.

butterfly key: The final cryptographic *public key* or *private key* produced by the *butterfly key process*.

butterfly key certificate request: A request created by an *end entity (EE)* that is intended to result in the issuance of multiple certificates, with the keys in those certificates created via the *butterfly key process*.

¹³*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.

butterfly key expander (BKE): A component of the *Security Credential Management System (SCMS)* that adds an additional random elliptic curve point to each *cocoon public key* to create the *butterfly public key* (or, the implicit certificate) for an explicit certificate.

butterfly key parameters: The *caterpillar public key* and the *expansion function* used in the *butterfly key process*.

butterfly key process: A process used in certificate generation where an initial *caterpillar public key* is modified using an *expansion function* by the *cocoon key expander (CKE)* to create a *cocoon public key*, and further modified by a *butterfly key expander (BKE)* to produce a *butterfly public key* (or, an implicit certificate) for an explicit certificate, in such a way that only the holder of the original caterpillar private key can derive the butterfly private key corresponding to the butterfly public key (or, the implicit certificate). It is infeasible for a party that does not know the caterpillar private key to derive the corresponding butterfly private key.

butterfly private key: The final cryptographic *private key* produced by the *butterfly key process*.

butterfly public key: The final cryptographic *public key* produced by the *butterfly key process*.

canonical identifier: A device identifier used to look up the device's *canonical key*.

canonical key: A device key with a long lifetime, used to request *enrollment certificates*.

canonical key acceptance policy: A set of conditions applied to a *canonical key* and its metadata to determine whether that key is acceptable to authorize an *enrollment certificate* request received by a particular *enrollment certificate authority (ECA)*.

caterpillar key: The initial cryptographic *public key* or *private key* input to the *butterfly key process*.

caterpillar private key: The initial cryptographic *private key* input to the *butterfly key process*.

caterpillar public key: The initial cryptographic *public key* input to the *butterfly key process*.

certificate acceptance policy: A policy setting constraints on the digital certificates (ITU-T Recommendation X.509 or IEEE Std 1609.2 based) that may be used to authorize certain activities.

certificate acceptance policy: A statement of properties that a *Security Credential Management System (SCMS)* component certificate is required to have when it is used to authenticate that SCMS component in the context of a *Transport Layer Security (TLS)* session.

certificate trust list (CTL): A list of the *electors* and the *root certificate authorities (root CAs)* that are trusted by a particular *Security Credential Management System (SCMS) Manager*, signed by the eligible electors.

characterization parameters: Parameters used to indicate properties of a protocol mechanism (secure session or web API) specified in this document, with the purpose of making the properties of a composite protocol (secure session + web API) clear.

client (of a registration authority (RA)): Any entity within the system that uses a particular *registration authority (RA)* for certificate management activities.

cocoon key expander (CKE): A component of the *Security Credential Management System (SCMS)* that uses the *expansion function* to create a series of statistically uncorrelated *cocoon public keys*.

cocoon key: The intermediate cryptographic *public key* or *private key* produced by applying an *expansion function* to a *caterpillar key* in the *butterfly key process*.

cocoon private key: The intermediate cryptographic *private key* produced by applying an *expansion function* to a *caterpillar private key* in the *butterfly key process*.

cocoon public key: The intermediate cryptographic *public key* produced by applying an *expansion function* to a *caterpillar public key* in the *butterfly key process*.

delegated registration authority: An organization responsible for assigning identifiers, or identifier sets, from a designated range of values of the identifier *ElectorGroupId* defined in this standard. The name indicates that the authority to assign from the indicated range has been assigned to the delegated registration authority by the IEEE Registration Authority.

derivable node: A node in a *binary hash tree* whose value can be derived from *published node* values.

device configuration manager (DCM): A component of the *Security Credential Management System (SCMS)* that is responsible for bootstrapping an *end entity (EE)* and providing secure connection between the EE and the *enrollment certificate authority (ECA)*.

direct authorization (for enrollment certificate request): A mode of authorization for *enrollment certificate* request where the enrollment certificate request generated by an *end entity (EE)* device contains a proof that the device is entitled to that enrollment certificate.

distribution center (DC): A component of the *Security Credential Management System (SCMS)* that distributes public information such as certificates and certificate revocation lists. *Contrast:* **registration authority (RA)**.

elector: A component of the *Security Credential Management System (SCMS)* that manages trust of *root certificate authority (root CA)* certificates and peer *elector* certificates.

end entity (EE): An actor that uses digital certificates to authorize *application activities*. *Contrast:* **certificate authority (CA)**.

end entity node: A bottom-layer node in a *binary hash tree* used to calculate an Activation Codes for Pseudonym Certificates (ACPC) private activation value (APrV).

enrollment certificate: A certificate that is used to request *authorization certificates* and to manage other interactions between an *end entity (EE)* and the *Security Credential Management System (SCMS)*. *Contrast:* **authorization certificate**.

enrollment certificate authority (ECA): A certificate authority (CA) that issues *enrollment certificates*.

expansion function: A function used to produce *cocoon keys* from a *caterpillar key* in the *butterfly key process*.

identification certificate: An *authorization certificate* that is constructed so as not to deliberately obscure the real-world identity of the certificate holder. *Contrast:* **pseudonym certificate**.

identifying uniform resource locator (URL): A uniform resource locator (URL) that acts as a long-lived identifier for an SCMS component.

IEEE Registration Authority (IEEE RA): A unit of IEEE that assigns unambiguous names to objects in a way that makes the assignment available to interested parties.¹⁴

indirect authorization (for enrollment certificate request): A mode of authorization for *enrollment certificate* request where the enrollment certificate request generated by an *end entity (EE)* device does not contain a proof that the device is entitled to that enrollment certificate, and the proof is instead provided to the *enrollment certificate authority (ECA)* by some other means.

individual certificate request: A request for a certificate authority (CA) to issue a single certificate. The request may come from the relevant *end entity (EE)* or from the *registration authority (RA)*. *Contrast:* **butterfly key certificate request.**

intermediate certificate authority (ICA): A certificate authority (CA) whose certificate was issued by another CA and whose main responsibility is to issue certificates to another CA, that is, an *authorization certificate authority (ACA)*, an *enrollment certificate authority (ECA)*, or another ICA.

i-period: A *validity period* for a certificate, identified by an *i-value* to simplify management of temporal sequences of certificates issued to an *end entity (EE)*.

i-period epoch: The date at which *i-periods* of the indicated length started.

i-period length: The length of time that an *i-period* lasts.

i-period series: A series of temporal intervals, each of the same length, identified by an *i-value* that increases by one for each successive interval.

ITU-T X.509 certificate: A *digital certificate* following the format specified in ITU-T Recommendation X.509.

i-value: An integer identifying an *i-period*.

linkage authority (LA): A component of the *Security Credential Management System (SCMS)* that provides inputs to the linkage value calculation process to enable efficient revocation of *pseudonym certificates* while preserving the privacy of an *end entity (EE)* against the *authorization certificate authority (ACA)*.

location obscurer proxy (LOP): A component of the *Security Credential Management System (SCMS)* that is responsible for hiding location information of an *end entity (EE)* from the *registration authority (RA)*.

misbehavior authority (MA): A component of the *Security Credential Management System (SCMS)* that that receives reports of malicious or potentially malicious *application activities*, analyzes them, and determines whether or not to take mitigating actions.

omitted node: A node in a *binary hash tree* whose value is omitted from the encoding of the binary hash tree and whose value cannot be derived from *published nodes*. *Contrast:* **published node.**

parent enrollment certificate: An *enrollment certificate* that maintains continuity of ownership with a subsequent enrollment certificate (a *successor enrollment certificate*), such that if a certificate management activity could be authorized with the parent enrollment certificate that same activity can also be authorized with the successor enrollment certificate.

¹⁴ IEEE Registration Authority (<https://standards.ieee.org/products-services/regauth/index.html>).

physically secure session: A communications session in which security is provided by the fact that both endpoints of the session are in the same physically secure environment.

privacy against insiders: A property of a system such that the system protects users of the system from having personal information revealed even to privileged actors within that system.

private key: A cryptographic key, used for key exchange, decryption, and/or signature generation, that has a corresponding *public key* such that the private key cannot feasibly be derived from the public key using public information.

pseudonym certificate: An *authorization certificate* that is designed to help protect the privacy of an *end entity (EE)*. This is achieved using mechanisms such as linkage valued-based revocation. An EE that uses pseudonym certificates will typically have multiple certificates valid at the same time to allow that EE to use different certificates at different times and locations, disrupting an eavesdropper's ability to track them. *Contrast: identification certificate.*

public key: A cryptographic key, used for key exchange, encryption, and/or signature verification, that has a corresponding *private key* such that the private key cannot feasibly be derived from the public key using public information.

public key infrastructure (PKI): A system of certificate authorities and supporting entities to support the management of digital certificates and *public keys*.

published node: A node in a *binary hash tree* whose value is published in the encoding of the binary hash tree or can be derived from the values of other published nodes. *Contrast: omitted node.*

registration authority (RA): A component of the *Security Credential Management System (SCMS)* that is the main point of contact for an *end entity (EE)*, and is responsible for provisioning the EE with authorization and successor enrollment certificates. *Contrast: Distribution Center (DC), IEEE Registration Authority (IEEE RA).*

root certificate authority (root CA): A certificate authority (CA) that issues certificates for other entities and whose certificate was issued by itself.

security credential management system (SCMS): A system of certificate authorities and supporting entities to support distribution of trust in a system based on IEEE 1609.2 digital certificates.

security credential management system (SCMS) manager: A component of the *Security Credential Management System (SCMS)* whose role is to govern the entire SCMS, including defining and enforcing the certificate and security policies to be applied to *electors* and *root CAs*.

successor enrollment certificate: An *enrollment certificate* that maintains continuity of ownership with a previous enrollment certificate (a *parent enrollment certificate*), such that if a certificate management activity could be authorized with the parent enrollment certificate that same activity can also be authorized with the successor enrollment certificate.

Transport Layer Security (TLS): A security protocol developed and maintained by the Internet Engineering Task Force (IETF) providing confidentiality, integrity, and authentication services.

validity period (of a certificate): The time period during which a certificate is to be trusted. In the IEEE 1609.2 system, this is indicated by the *validityPeriod* field in the certificate, that is, the time period starting at *validityPeriod.start* and ending at (*validityPeriod.start* + *validityPeriod.duration*).

3.2 Acronyms and abbreviations

ACPC	Activation Codes for Pseudonym Certificates
ACA	authorization certificate authority
AES	Advanced Encryption Standard
APDU	application protocol data unit
API	application programming interface
APrV	Activation Codes for Pseudonym Certificates (ACPC) private activation value
APuV	Activation Codes for Pseudonym Certificates (ACPC) public activation value
ASD	aftermarket safety device
AT	access token
CA	certificate authority
CAM	certificate access manager
CAMP	Crash Avoidance Metrics Partners LLC
CAL	certificate access list
CCF	certificate chain file
CCG	client credentials grant
C-OER	canonical octet encoding rules
CRACA	certificate revocation authorizing certificate authority
CRL	certificate revocation list
CTL	certificate trust list
DC	distribution center
DCM	device configuration manager
DER	distinguished encoding rules
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
ECA	enrollment certificate authority
EE	end entity
HTTP	Hypertext Transfer Protocol
I2V	infrastructure to vehicle
IETF	Internet Engineering Task Force
IP	Internet Protocol
ITS	intelligent transportation systems
JSON	JavaScript Object Notation
JWKS	JavaScript Object Notation (JSON) web key set
JWT	JavaScript Object Notation (JSON) web token
LA	linkage authority
LOP	location obscurer proxy
M2M	machine to machine
NAT	Network Address Translation
OAS	OAuth authorization server
OBU	onboard unit
OCSP	Online Certificate Status Protocol
OER	octet encoding rules
P2PCD	peer to peer certificate distribution
PKI	public key infrastructure
PSID	Provider Service Identifier
RA	registration authority
REST	representational state transfer
RFC	Request for Comments
RSU	roadside unit
SAS	Supplementary Authorization Server
SCMS	Security Credential Management System
SPDU	secured protocol data unit
SSME	security services management entity
SSP	service specific permissions

TLS	Transport Layer Security
URL	uniform resource locator
USDOT	United States Department of Transportation
UTC	Coordinated Universal Time
V2I	vehicle to infrastructure
V2V	vehicle to vehicle
V2X	vehicle to everything
WAVE	Wireless Access in Vehicular Environments
WSA	Wireless Access in Vehicular Environments (WAVE) Service Advertisements

4. Architecture

4.1 Security Credential Management System (SCMS)

4.1.1 General

The SCMS, for which this document defines interfaces, is a PKI that creates and manages digital certificates as specified in IEEE Std 1609.2. The primary functional goal of the certificates is to enable authenticated application interactions between EEs. An end entity is an application instance running on a computing platform, which can be an RSU, an OBU, an ASD, or some other computing system. IEEE 1609.2 certificates encode permissions indicating which application activities the certificate holder can carry out, with the application being indicated by the PSID, as defined in IEEE Std 1609.12-2019, and the specific activities or roles within the application being indicated by the SSP, as defined in IEEE Std 1609.12-2019. See IEEE Std 1609.2 for further discussion. Certificates used in application interactions are referred to as authorization certificates and are issued by ACAs in response to a request from an end entity. The ACA follows policies put in place by the SCMS manager that provide assurance that the end entity is in fact entitled to the permissions indicated in the certificate.

To protect the privacy of system participants, the ACA may provide an end entity with several simultaneously valid authorization certificates so that the end entity may use different certificates at different times for the same activity. These certificates are known as “pseudonym certificates.” If certificates are issued such that there is only one certificate valid at a given time for a given end entity (except perhaps for a brief overlap period as discussed in 4.3.2), then the certificates are known as “identification certificates” (for applications hosted on mobile devices) or as “application certificates” (for applications hosted on nonmobile devices). Pseudonym certificates, identification certificates, and application certificates are the only subtypes of authorization certificate identified in this document.

The conceptual SCMS architecture is shown in Figure 1. A previous version of this architecture was developed by CAMP for V2V, V2I, and I2V communications (together referred to as V2X communications), in a series of projects under a cooperative agreement with the USDOT. The purpose of the SCMS is to provision EEs in the system with the certificates that they use to send SPDUs, in such a way that the recipients of the SPDU can have assurance that the certificate reflects the true authorizations of the sender. The boxes in the architectural diagram represent functional entities, that is, different boxes represent distinct collections of functionality. In a real-world deployment, there may be a single organization that operates more than one functional entity. The architecture shown in the architecture diagram permits a particularly high level of privacy protection against both eavesdroppers and SCMS component operators when two linkage authorities are used and the four components {authorization CA, registration authority, linkage authority 1, linkage authority 2} are operated by different organizations with limited sharing of information. See Annex D for more details.

This document specifies two main use cases of the SCMS: 1) enrollment certificate provisioning, where an ECA provisions an end entity with an enrollment certificate that it uses for communications with the

SCMS, and 2) authorization certificate provisioning, where an end entity with a valid enrollment certificate is provisioned by an ACA with authorization certificates.

The SCMS components are as follows:

- **SCMS manager:** A component of the SCMS whose role is to govern the entire SCMS, including defining and enforcing the certificate and security policies to be applied to electors and root CAs.
- **Elector:** A component of the SCMS that manages trust of root CA certificates and peer elector certificates. Electors do this by signing CTLs containing identifiers for the trusted certificates. The contents of the CTLs, that is, the set of trusted certificates, is set by the SCMS manager. The system design allows the SCMS manager to specify various electors that are active at any one time and a quorum value such that a CTL is trusted only if it is signed by a quorum or more of trusted electors. For example, (number of electors, quorum) may be (1, 1), or (3, 2), or (5, 3), or any other pair of values as set by the SCMS manager. How the quorum and the number of electors are managed and communicated to system participants are outside the scope of this document. For further discussion, see 4.1.8 and Annex J.
- **Root CA:** A CA that issues certificates for other entities and whose certificate was issued by itself. Root CAs in general are brought online rarely because any root CA compromise has a significant impact.
- **Intermediate CA (ICA):** A CA whose certificate was issued by another CA and whose main responsibility is to issue certificates to other CAs, that is, an ACA, an ECA, or other ICAs.
- **Authorization CA (ACA):** A CA whose main responsibility is to issue authorization certificates.
- **Enrollment CA (ECA):** A CA whose main responsibility is to issue enrollment certificates.
- **Certificate revocation list (CRL) signer:** A component of the SCMS that is authorized to sign CRLs. Each CRL Signer is associated with a single CA known as a CRACA. The CRL Signer may be the same as its CRACA (i.e., the CRL signing certificate is the same as the CA's certificate issuing certificate), or it may be a separate component. This allows for the CRL signing operations to be online more than the certificate issuing operations, improving the responsiveness and robustness of the system. Every IEEE 1609.2 certificate that can be revoked by an IEEE 1609.2 CRL is associated with a single CRACA that is in the issuance chain of that certificate, and it can only be revoked by a CRL Signer associated with that CRACA.
- **Registration authority (RA):** A component of the SCMS that is the main point of contact for an EE and is responsible for provisioning the EE with certificates. The RA receives certificate requests from EEs, determines whether the EE is entitled to the certificate, and passes authorized requests to the ACA or the ECA for the certificates to be issued. The CAs trust the RAs to carry out the authorization checks and do not carry out checks of their own, except to determine that the RA has authorized the certificate request. The RA also provides support services around load management, support for privacy, and provisioning of security management information like CTLs and CRLs to EEs. The RA manages a group of EEs, in the sense that those EEs are configured to use that particular RA for certificate management in a particular application domain or a set of application domains. The RA may choose to restrict its support services to the EEs that it manages, or may provide them to a broader population of system participants. RAs are identified by a URL known as the *RA identifying URL*, which is used to locate them on the network and for authentication. EEs are configured with the identifying URL of the RA that they are to contact to carry out EE-RA activities. How this configuration is carried out is out of scope of this document.
- **Supplementary authorization server (SAS):** A component of the SCMS that provides authorization tokens to EEs for use in authorizing operations that are not critical for security. In this document, the supplementary authorization service is instantiated with OAuth 2.0 bearer authorization, as described in 6.2.
- **Distribution center (DC):** A component of the SCMS that distributes public information such as certificates and CRLs, similar to the RA, except that unlike the RA it does not act as a trusted entity for approving certificate requests. Like the RA, it may make the information available to all system participants or to a subset of system participants.
- **Device configuration manager (DCM):** A component of the SCMS that is responsible for bootstrapping an EE and providing secure connection between the EE and the ECA.

- **Location obscurer proxy (LOP):** A component of the SCMS that is responsible for hiding location information of an EE from the RA when the EE requests or attempts to download certificates. This improves the privacy of the EE against being tracked by its RA. The interfaces in this document are written so that the existence of the LOP is transparent to the EE.
- **Linkage authority (LA):** A component of the SCMS that provides inputs to the linkage value calculation process to enable efficient revocation of pseudonym certificates while preserving the privacy of an EE against the ACA. The SCMS architecture allows multiple LAs. In the current system design, one or two LAs provide input into the generation of each certificate that uses linkage values. Using one LA enables efficient revocation of an arbitrary number of certificates, while using two LAs enables efficient revocation and also allows the system to be run so that no one SCMS component stores information that allows a collection of certificates to be identified as all belonging to the same end entity.
- **Misbehavior authority (MA):** A component of the SCMS that receives reports of malicious or potentially malicious application activities, analyzes them, and determines whether or not to take mitigating actions. The mitigating actions could include a recommendation that the misbehaving actor is revoked. CRL Signers trust the MA's recommendation on revocation, so the revocation is effectively triggered by the MA alone. The mitigating actions could also include a recommendation to the RA that the relevant EE is no longer permitted to be issued with authorization certificates, that is, that the EE's enrollment certificate is blocked. In this case, the RA is assumed to trust the MA's recommendation.

The enrollment certificate provisioning use case involves the EE and up to two SCMS components, namely the DCM and the ECA. Either the DCM on behalf of the EE or the EE directly sends an enrollment certificate request to the ECA. If the request is valid, the ECA responds with an enrollment certificate. Whether the DCM is used and what are the security requirements are outside the scope of this document. This document only specifies the messages exchanged between the EE and the ECA; the security requirements on the EE and the interface are outside the scope of this document. End entity interfaces to support enrollment certificate provisioning are given in 6.3.4.2.

The authorization certificate provisioning use case involves the EE and up to six SCMS components, namely the ACA, the LA1, the LA2, the LOP, the RA, and optionally a SAS. The EE connects with the RA, either via the LOP for location privacy or directly if the LOP is not available, and sends a request for authorization certificates. If the request is valid, the RA responds with an acknowledgment containing the time and location where the EE should connect for downloading the certificates. The EE downloads the certificates from the location on or after the time specified in the RA's acknowledgment. After receiving a valid request from an EE, the RA starts the process of generating the requested certificates by communicating with the ACA, and additionally with the LA1 and the LA2 for authorization certificates of type pseudonym. End entity interfaces to support authorization certificate provisioning are specified in 6.3.5.2 and 6.3.5.3.

This document supports two different approaches to authorization certificate provisioning. In the first, an end entity requests and is issued a single authorization certificate. In the second, the end entity uses a mechanism known as "butterfly key expansion" to create a single request that the SCMS components use to generate a set of certificates, covering multiple time periods, and with (potentially) multiple certificates within each time period. These two different approaches are indicated by different parameters in the authorization certificate provisioning interfaces specified in 6.3.5.2 and 6.3.5.3.

NOTE—In (nonstandard) publications preceding this document, the ACA was previously called the PCA because it was originally designed to issue only pseudonym certificates to vehicles. As additional certificate types were identified, that is, applications for road-side units and identification for vehicles that do not necessarily need privacy protection, new types were added to the SCMS use cases (and they were also issued by the PCA). The name of the PCA was never updated to reflect the additional certificate types it issues because these additional certificate types required only a subset of mechanisms that were developed for pseudonym certificates. In this document, the above three certificate types have been consolidated into two types, namely, identification and pseudonym certificates that are collectively called "authorization certificates." Therefore, the PCA has been renamed to the ACA.

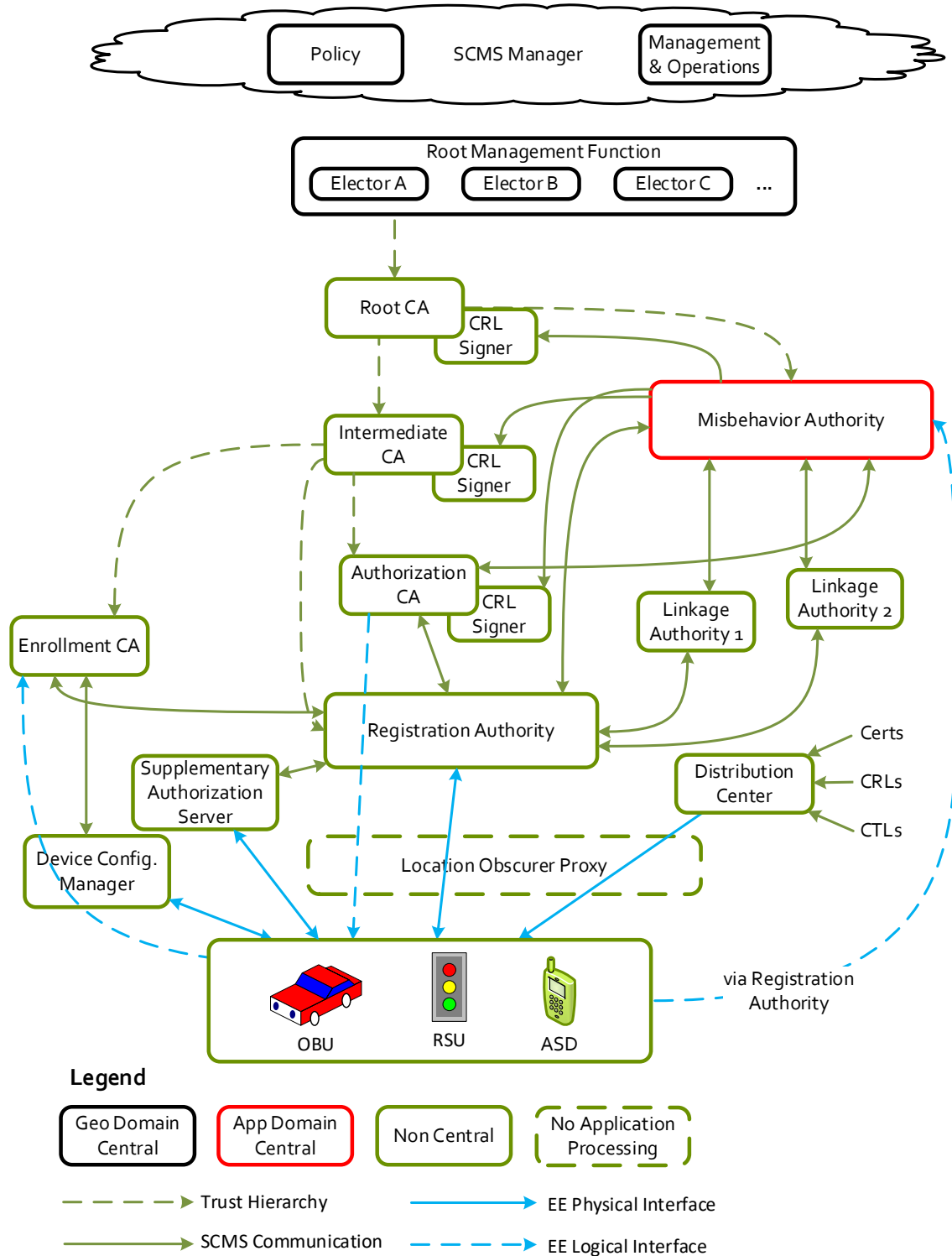


Figure 1—Conceptual SCMS architecture

4.1.2 Supported interfaces

4.1.2.1 General

This document specifies interfaces between an EE and the SCMS that are used to provision the EE with digital certificates and supporting system management information. The architecture supported by this document is shown in Figure 1. In this architecture, there are the following four such interfaces:

- Authorization certificate authority – end entity (ACA – EE)
- Distribution center – end entity (DC – EE)
- Enrollment certificate authority – end entity (ECA – EE)
- End entity – registration authority (EE – RA)

In addition, this document also provides for informational purposes the specification of an interface between two SCMS components, the ACA and the RA; see Annex E.

An identifier for the endpoints of the message flow is embedded in the ASN.1 definition of each message to help ensure that all messages can be parsed unambiguously (i.e., that no matter which physical interface a message is sent over, it can always be interpreted the same way). This provides logical separation of the functional use of a message from the details of how it is passed between the generator and the receiver.

4.1.2.2 ACA – EE interface

The ACA – EE interface is used to provision the EE with authorization certificates. It is mainly a logical interface in that there is no direct message flow between the ACA and the EE: The messages from the ACA are stored by the RA and subsequently forwarded to the EE. This document does not provide a mechanism for the EE to know in advance which ACA(s) will provide its certificates or to request that certificates are provided by a particular ACA. The assumption is that the EE's relationship is with a particular RA, and the RA is responsible for deciding which ACA service providers to use.

4.1.2.3 DC – EE interface

The DC is an optional component of the SCMS. The DC – EE interface is used to provide the following system information to the end entity. This interface is identical to the EE – RA interface for the same operations:

- CCF
- CRL
- Individual CA certificate
- Individual CRL
- CTL
- Status (freshness) of the above information

The system may contain multiple DCs, and an EE may be configured to use one or more DCs for the use cases listed above. A DC is identified by a URL known as the *DC identifying URL*. The EE can use this URL both to locate the DC on the network (via DNS lookup or some other mechanism to map URLs to network addresses) and to obtain assurance that it, the EE, is communicating with the correct DC (via certificates that contain that URL). How the URL is configured, managed, and (if necessary) changed on the EE is out of scope for this document.

4.1.2.4 ECA – EE interface

The ECA – EE interface is used to provision the EE with enrollment certificates.

If the EE communicates directly with the ECA over a secure session as defined in 4.2, the ECA may be identified by a URL, referred to as the *ECA identifying URL*. The EE can use this URL both to locate the RA on the network (via DNS lookup or some other mechanism to map URLs to network addresses) and to obtain assurance that it, the EE, is communicating with the correct RA (via certificates that contain that URL). How the URL is configured, managed, and (if necessary) changed on the EE is out of scope for this document.

4.1.2.5 EE – RA interface

The EE – RA interface is used as follows:

- Support authorization certificate provisioning
- Support successor enrollment certificate provisioning
- Provide the following system information:
 - 1) CCF
 - 2) CRL
 - 3) Individual CA certificates
 - 4) Individual CRLs
 - 5) The CTL
 - 6) The RA certificate
 - 7) Status (freshness) of the above information
- Upload misbehavior reports (though the contents of the misbehavior reports are outside the scope of this document)

The RA is identified by a URL, referred to as the *RA identifying URL*. The EE can use this URL both to locate the RA on the network (via DNS lookup or some other mechanism to map URLs to network addresses) and to obtain assurance that it, the EE, is communicating with the correct RA (via certificates that contain that URL). How the URL is configured, managed, and (if necessary) changed on the EE is out of scope for this document. An EE is assumed to be bound to a single RA for all of the certificates with a particular set of permissions, although nothing in this document requires it to be so bound.

4.1.3 Types of enrollment certificate

An enrollment certificate is a certificate used to authenticate interactions with the SCMS. The most important use of enrollment certificates in this document is to authenticate a request for authorization certificates, which are the certificates used in application interactions.

This document supports the following two types of enrollment certificate to authenticate a request for authorization certificates:

- Enrollment certificates following the IEEE 1609.2 format. In this case, the enrollment certificate is identifiable as an enrollment certificate because it contains the `certRequestPermissions` field and is owned by an EE. Validity conditions for an IEEE 1609.2 enrollment certificate are given in 7.7.3.5. Validity conditions for an authorization certificate request signed with an IEEE 1609.2 certificate are given in 10.1.5 and 10.1.6.
- Enrollment certificates following the ITU-T Recommendation X.509 format per IETF RFC 5280 and (if elliptic curve cryptographic keys are used) IETF RFC 5480. Validity conditions for an ITU-T X.509 enrollment certificate are given in 7.7.3.6. Validity conditions for an authorization certificate request signed with an ITU-T X.509 certificate are given in 10.1.7.

This document specifies mechanisms for managing (requesting, updating) IEEE 1609.2 enrollment certificates. Mechanisms for managing ITU-T X.509 enrollment certificates are out of scope of this document. Standardized mechanisms for managing ITU-T X.509 enrollment certificates are specified in IETF RFC 5272 [B8].¹⁵

4.1.4 Assumed lifecycle and use cases

4.1.4.1 General

This document defines interfaces to support the following use cases:

- Initial enrollment certificate request/download (single round-trip)
- Authorization certificate request
- Authorization certificate download
- Successor enrollment certificate request
- Successor enrollment certificate download
- Misbehavior report submission
- Download of security management and configuration, including the following:
 - 1) RA (IEEE 1609.2) certificate download
 - 2) MA certificate download
 - 3) CRL download (composite CRL file and individual CRLs)
 - 4) CCF download (composite certificate chain file and individual CA certificates)
 - 5) CTL download
 - 6) Status (freshness) of the above information as stored by the source from which the EE is downloading it

NOTE—Unlike other SCMS components, the RA may have certificates of type other than the IEEE 1609.2 one. The above use case “RA (IEEE 1609.2) certificate download” applies only to RA’s certificate of type IEEE 1609.2.

The use cases support provisioning an end entity with its own certificates and with security management information throughout its lifecycle. An overview of the use cases follows. A detailed description of the interactions for each use case is given in Clause 6.

4.1.4.2 Initial enrollment certificate request/download

In this use case, an end entity that does not yet have an IEEE 1609.2 enrollment certificate requests and receives an enrollment certificate from the ECA, as follows:

- a) This enrollment certificate may contain the following constraints on the permissions the end entity can request in the `toBeSigned.certRequestPermissions` field:
 - 1) The mechanism by which the ECA determines whether the end entity is entitled to the permissions requested for the enrollment certificate is outside the scope of this document.
 - 2) The mechanism by which the end entity is configured to know which ECA to apply to is outside the scope of this document.
- b) The end entity is assigned at least one RA, as follows, to manage authorization certificate requests:
 - 1) The end entity may be assigned different RAs for different sets of applications.
 - 2) The RA certificate is provided to the end entity.
 - 3) The mechanism by which the end entity is configured with this information is outside the scope of this document.

¹⁵ The numbers in brackets correspond to those of the bibliography in Annex J.

When an end entity applies for an enrollment certificate, it may indicate that it is entitled to the certificate *directly* or *indirectly*.

In the *indirect authorization* case, the ECA is given assurance that the end entity is entitled to the enrollment certificate by means outside the scope of the enrollment certificate request. For example, the enrollment request could happen in an environment that is trusted by the ECA and the request could be transmitted over a secure connection.

In the *direct authorization* case, before enrollment happens, the end entity generates a keypair known as the *canonical keypair*, and is assigned (or generates) a globally unique identifier known as the *canonical identity*. The device undergoes inspection to determine that it meets security and other requirements. If the device passes these tests, the public key of this keypair is stored, associated with the canonical identifier, in a data store accessible by the ECAs. At enrollment time, the end entity includes its canonical identity in the enrollment certificate request and signs the enrollment certificate request with the private key from the canonical keypair. The ECA can use the canonical identity to look up the public key and uses that to verify the signature; this provides assurance that the device is entitled to an enrollment certificate.

The ECA follows a *canonical key acceptance policy* to determine whether a canonical key, referenced in an enrollment certificate request, is valid to authorize that request. The canonical key acceptance policy may include properties of the key such as the algorithm that it is for, the hardware security properties of the device it was generated in, and so on. The acceptance policy may be enforced by the data store, so that only keys that meet the acceptance policy are stored in the data store; or the data store may store metadata about the keys so that the ECA can apply an ECA-specific acceptance policy to that metadata. Further details about the canonical key acceptance policy, including how it is established and communicated to suppliers of end entities that use canonical keys, are out of scope for this document.

NOTE 1—The direct authorization approach, using canonical keys and identities, is specified in ETSI TS 102 941 [B4]. A version of this approach, which is functionally identical, but which uses incompatible data types is supported by this document. The use of canonical keys and identifiers for any use case other than initial enrollment certificate request is not supported by this document.

NOTE 2—As discussed in 4.1.3, this document supports enrollment certificates of two types, IEEE 1609.2 and ITU-T X.509. This document specifies PDUs for provisioning with IEEE 1609.2 enrollment certificates but not for provisioning with ITU-T X.509 enrollment certificates. Provisioning with ITU-T X.509 enrollment certificates is supported by mechanisms out of the scope of this document, for example, those specified in IETF RFC 5272 [B8].

4.1.4.3 Use cases authenticated with the enrollment certificate

4.1.4.3.1 General—Enrollment certificate registration and blocking—Relationship between EE and RA

The assumption in this document is that the enrollment certificate is used as the primary identifier of an EE to the SCMS, in particular to the RA. To prevent an EE from using its enrollment certificate to request certificates via multiple RAs and so receiving more certificates than it is entitled to, the assumption is that RAs will only process requests from EEs whose enrollment certificates are registered with that RA. If access to the RA is authorized using OAuth ATs as specified in 6.2, it is assumed that the OAS also has access to a list of registered EEs and will only issue ATs to EEs on that list. The list may use the enrollment certificate or any other appropriate identifier to identify registered EEs. This document does not specify interfaces to be used to register the EE or the enrollment certificate with an RA or other SCMS component. An enrollment certificate that is not registered with an RA is referred to as an “unregistered enrollment certificate.”

If an EE is determined by the MA to be misbehaving as described in 4.1.1, or otherwise to need certificate privileges to be withdrawn, the MA may instruct the RA to “block” that EE by marking the EE’s enrollment certificate as blocked. If an EE is blocked, the RA shall deny certificate requests and certificate

download requests for authorization and successor enrollment certificates. The RA may also restrict the EE from other activities. In this case, the EE's enrollment certificate is referred to as a "blocked" enrollment certificate.

4.1.4.3.2 Authorization certificate request

In this use case, an end entity that has an enrollment certificate requests authorization certificates from the RA.

NOTE—As discussed in 4.1.3, this document supports enrollment certificates of two types, IEEE 1609.2 and ITU-T X.509. This document specifies mechanisms for authorization certificate provisioning with authentication provided by either IEEE 1609.2 or ITU-T X.509 certificates.

An end entity that has an enrollment certificate can use that enrollment certificate to request authorization certificates from the RA. Authorization certificates are of type implicit. An implicit certificate, as specified in SEC 4, is a digital certificate that allows the associated public key to be reconstructed from a reconstruction value and the certificate issuing CA's public key rather than directly providing the associated public key; see 5.1.1 and 5.3.2 in IEEE Std 1609.2a-2017 for discussion of explicit and implicit certificates.

This document defines two types of authorization certificates, namely, identification and pseudonym. An identification certificate is a certificate that is constructed so as not to deliberately obscure the real-world identity of the certificate holder. This is in contrast to a pseudonym certificate that is designed to help protect the privacy of the end entity. Pseudonym certificates are generated using several novel mechanisms such as linkage valued-based revocation (see Annex D), butterfly keys (see 9.2 and 9.3), and ACPC (see 9.4). An EE that uses pseudonym certificates will typically have multiple certificates valid at the same time to allow that EE to use different certificates at different times and locations, disrupting an eavesdropper's ability to track them. The linkage value-based revocation mechanism allows the MA to efficiently revoke multiple pseudonym certificates of an EE by publishing linkage seeds on the CRL instead hashes of the certificates. The butterfly key mechanism allows an EE to request multiple authorization certificates with just one request via the use of expansion functions; the caterpillar key and the expansion function provided by an EE is expanded into multiple pseudorandom cocoon keys that are then used to generate pseudonym certificates that all look different from each other. Finally, the ACPC mechanism provides an alternative way of removing no-longer-trusted EEs from the system; an SCMS component locks certificates on generation and provides unlocking codes to nonrevoked EEs before it is time to use the certificates, as follows:

- a) The request is authenticated and authorized using the EE's enrollment certificate. The RA may also require that the EE provides additional authentication by other means, such as via the SAS.
- b) The request is encrypted using a currently valid RA certificate. If the end entity does not have a currently valid RA (IEEE 1609.2) certificate, it can obtain that certificate via the mechanisms specified in the description of the **RA (IEEE 1609.2) certificate download** use case.
- c) The request may indicate the use of "butterfly key" mechanism (see 9.2 and 9.3) where the end entity uploads a single request that is intended to result in multiple certificates for multiple distinct time periods, or the "conventional" process where the end entity uploads a single request and expects a single response.
- d) The RA determines whether or not the request is to be granted, as follows:
 - 1) If the enrollment certificate contains constraints, the request is only valid if the requested certificate is permitted by those constraints.
 - 2) The RA may also maintain a policy stating the authorization certificates that particular enrollment certificates or groups of enrollment certificates are entitled to. The means for maintaining and updating this policy, and any means for communicating this policy to end entities, are outside the scope of this document.

- e) The RA responds with either a failure code or information about when the EE should connect for downloading the certificates.
- f) The RA interacts with the ACA to have the appropriate certificate issued.

4.1.4.3.3 Authorization certificate download

In this use case, the end entity requests download of issued authorization certificates from the RA, as follows:

- a) The download request is authenticated and authorized using one of the following two options:
 - 1) The request is authenticated and authorized using the enrollment certificate used for the certificate request, or one of its successors (see “successor enrollment certificate request” below, and see `EeRaDownloadRequestSdu` for further discussion).
 - 2) The client is authenticated by a SAS and granted authorization to access the service through a service specific authorization token.
- b) The RA responds with either the issued certificates or an error code.

NOTE—As discussed in 4.1.3, this document supports enrollment certificates of two types, IEEE 1609.2 and ITU-T X.509. This document specifies mechanisms for authorization certificate download with authentication provided by IEEE 1609.2 certificates, or by other mechanisms, but not by ITU-T X.509 certificates.

4.1.4.3.4 Successor enrollment certificate request

In this use case, an end entity that has an IEEE 1609.2 enrollment certificate requests a new IEEE 1609.2 enrollment certificate from the RA.

NOTE—As discussed in 4.1.3, this document supports enrollment certificates of two types, IEEE 1609.2 and ITU-T X.509. This document specifies mechanisms for successor enrollment certificate provisioning with IEEE 1609.2 enrollment certificates but not for successor enrollment certificate provisioning with ITU-T X.509 enrollment certificates. Successor enrollment certificate provisioning with ITU-T X.509 enrollment certificates is supported by mechanisms out of the scope of this document, for example, those specified in IETF RFC 5272 [B8].

An enrollment certificate can be used to request a “successor” enrollment certificate. This is a certificate that indicates continuity of ownership of that enrollment certificate, in other words, that the holder of the successor enrollment certificate is entitled to access the authorization certificates that the holder of the “parent” enrollment certificate was entitled to access.

This document defines an interface that can be used to request a successor enrollment certificate and indicates which operations are authorized by either an enrollment certificate or one of its successors.

A successor enrollment certificate does not contain any fields to indicate that it is a successor. Information about successor enrollment certificates and parent enrollment certificates is maintained by the relevant SCMS components, as follows:

- a) The request is authenticated and authorized using the EE’s current enrollment certificate.
- b) The request is encrypted using a currently valid RA certificate. If the end entity does not have a currently valid RA certificate, it can obtain that certificate via the mechanisms specified in the description of the **RA (IEEE 1609.2) certificate download** use case.
- c) The RA determines whether or not the request is to be granted, as follows:
 - 1) If the enrollment certificate contains constraints, the request is only valid if the requested enrollment certificate is permitted by those constraints. However, the RA may determine that the new enrollment certificate is entitled to state either additional or a subset of permissions compared to those in the original enrollment certificate.

- 2) The RA may also apply other policies to determine the permissions to appear in the new enrollment certificate. The means for maintaining and updating these policies, and any means for communicating these policies to end entities, are outside the scope of this document.
- d) The RA responds with either a failure code or information about when the EE should connect for downloading the new enrollment certificate.
- e) The RA interacts with the ECA to have the appropriate certificate issued. The RA-ECA interface is out of scope for this document.

4.1.4.3.5 Successor enrollment certificate download

In this use case, the end entity requests download of issued successor enrollment certificates from the RA, as follows:

- a) The download request is authenticated and authorized using the EE's current enrollment certificate or one of its successors (see `EeRaDownloadRequestSpdu`).
- b) The RA responds with either the issued certificates or an error code.

4.1.4.4 Use cases authorized with authorization certificate—Misbehavior report submission

This document specifies a single use case where an EE authorizes to an SCMS component with an authorization certificate. That use case is misbehavior report submission.

Misbehavior reports are authenticated and authorized SPDUs created by valid end entities. In this use case, the end entity provides a signed encrypted misbehavior report to the RA. The signature is generated with a current authorization certificate. The RA verifies the signature and passes it on to the appropriate MA. The RA responds to the end entity with an acknowledgement (not necessarily at the application layer). See 4.1.5 for further discussion.

NOTE—Most PDUs in this document are subtypes of `ScmsPdu` and are associated with the PSID `SecurityMgmtPsid`. Misbehavior report submission is instead associated with the PSID `AnyMbrPsid`, and an encrypted misbehavior report is not a subtype of `ScmsPdu`.

4.1.4.5 Use cases that may be authorized with enrollment certificate or may be unauthorized

The following use cases may also be supported across the EE – RA interface or the DC – EE interface. They may be supported in authenticated form (in which case the end entity is required to authenticate before downloading the information) or unauthenticated form (in which case no authentication is required):

- **RA (IEEE 1609.2) certificate download:** The end entity requests the current RA IEEE 1609.2 certificate. The RA responds with that certificate.
- **CRL download:** The end entity requests the current composite CRL that is maintained by the RA, or a single CRL. The RA responds with that CRL. See 4.1.6 for further discussion.
- **CCF download:** The end entity requests the current CCF that is maintained by the RA. This is a file containing the CA certificates that the RA operator believes are useful to the RA's end entity clients in validating incoming communications. The RA responds with that file. See 4.1.7 for further discussion. A variant of this allows an end entity to request a single CA certificate.
- **CA certificate download:** The end entity requests the current certificate of a CA. The RA responds with that certificate.

- **MA certificate download:** The end entity requests the current certificate of the MA responsible for handling misbehavior reports for misbehavior related to a specific PSID. The RA responds with that certificate.
- **CTL download:** The end entity requests the current CTL. The RA responds with the CTL.

Additionally, the RA and DC interfaces support a query for security management information status, which returns information about the most recent updates to all of the information above.

NOTE—As discussed in 4.1.3, this document supports enrollment certificates of two types, IEEE 1609.2 and ITU-T X.509. One authentication mechanism supported for the use cases described in this subclause (i.e., system information requests) is the use of IEEE 1609.2 enrollment certificates. The use of ITU-T X.509 certificates for authorization of system information requests is not specified in this document.

4.1.5 Misbehavior authorities

The SCMS diagram in Figure 1 indicates that the MA is central for a particular *application domain*, in other words, that a particular application domain has a single MA. Different domains may have different MAs. Different domains may also specify different approaches for misbehavior reporting to their MAs. And different applications should have different definitions of misbehavior and therefore different misbehavior report syntax and semantics.

This document provides a framework for misbehavior reporting via the RA and otherwise does not provide any specifics about misbehavior reporting. In this document, the assumptions made about misbehavior reporting are as follows:

- Misbehavior reports are authenticated and authorized SPDUs created by valid end entities.
- Misbehavior reports are created at the time that the misbehavior is observed and are signed by a certificate that is valid and active at that time and contains a PSID associated with signing that type of misbehavior report (see discussion of misbehavior reporting PSIDs below).
- Misbehavior report creation is an activity associated with a misbehavior reporting PSID. The actual misbehavior being reported occurs in the context of an application (or some other collection of activities) that is associated with a PSID. The PSID associated with the misbehavior being reported is separate from the PSID associated with misbehavior reporting. Different MAs may handle reports for misbehavior associated with different PSIDs. (For example, misbehavior associated with local infrastructure may be handled by a different MA from misbehavior associated with passenger vehicles sending safety messages.)
- A reporting EE will obtain the certificates for the MAs responsible for the PSIDs for which the EE will generate misbehavior reports.
- Misbehavior report payloads, containing the PSID associated with the misbehavior being reported as well as evidentiary data, misbehavior type, and other information, are encrypted for the MA at the time they are created. Additional metadata required by the RA is added in cleartext. Misbehavior reports are then stored by the reporting end entity until the end entity has access to the RA. The encryption is performed to the certificate of the MA responsible for handling misbehavior reports for misbehavior relating to that PSID.
- The end entity sends its misbehavior reports to the RA. The RA "shuffles" misbehavior reports from multiple end entities together before submitting them to the MA. This shuffle creates uncertainty at the MA about which reports come from the same end entity (so long as the reports are signed with different certificates).
- When the end entity sends the misbehavior reports to the RA, the misbehavior report upload is authorized by a currently valid misbehavior reporting certificate. The RA does not send this certificate to that MA. This is intended to prevent the MA from linking certificates belonging to a particular end entity.
- When the end entity sends the misbehavior reports to the RA, the cleartext metadata in the report contains information that allows the RA to determine which MA to send the report to and otherwise does not contain information about the encrypted payload of the report. This is implemented in this

document by encrypting the payload with the MA certificate, so that the RA can recognize the MA to which the report should be forwarded by examining the RecipientInfo headers in the cleartext metadata.

Consistent with these assumptions, this document provides an interface that can be used for misbehavior report uploading (see 6.3.5.6), and data formats that can be used to encapsulate encrypted misbehavior reports for upload (see 7.6). This document does not provide misbehavior report formats themselves, as these are to be provided by the application specifications for the applications within which the misbehavior occurs.

The following PSID is currently associated with misbehavior reporting and is the only PSID that can authorize misbehavior report upload:

- 0x26 Misbehavior reporting for common applications

As more applications are added to the system, additional misbehavior reporting PSIDs may be defined for applications outside the initial set. The current list of all assigned PSIDs is maintained at <https://standards.ieee.org/products-services/regauth/psid/public.html>. The list of PSIDs entitled to authorize misbehavior reporting is identified in this document by the ASN.1 type `AnyMbrPsid`, and the PSID currently on that list is defined in this document by the ASN.1 type `BaseMbrPsid`.

NOTE 1—"Misbehavior reporting for common applications" is the description associated with the PSID in the IEEE registry. The application specification associated with this PSID has not yet been published and so there is no official list of applications for which this PSID is associated with misbehavior reports. However, it is likely to include misbehavior reports associated with applications that use Basic Safety Messages, Signal Phase and Timing and Map messages, WSA, and other applications that will be widely used in the V2X system.

NOTE 2—Most PDUs in this document are subtypes of `ScmsPdu` and are associated with the PSID `SecurityMgmtPsid`. Misbehavior report submission is instead associated with the PSID `AnyMbrPsid`, and an encrypted misbehavior report is not a subtype of `ScmsPdu`.

4.1.6 Composite CRL

As specified in IEEE Std 1609.2, there can be multiple different CRLs in a secured Connected Vehicle ecosystem, as follows:

- Each certificate can only be revoked by a CRL Signer tightly coupled to that certificate's chain, so if there are different root CA certificates, different CRLs are bound to exist.
- The CRL Signer can in principle be coupled to any CA in the chain, so different end entities can be associated with different CRL Signers.
- Each CRL Signer can issue multiple "CRL Series", indicated by the `crlSeries` field in the CRL.

Each certificate contains information that allows the identification of the CRL that the certificate would appear on if revoked, so the CRL to use to check revocation for any received signed message is unambiguous. However, the existence of multiple CRLs means that an entity that receives messages has to be able to obtain all of those CRLs. This document provides two alternative mechanisms to obtain CRLs.

First, an end entity may download a composite CRL file. This is the preferred mechanism, as follows:

- The RA is assumed to maintain a list of all CRLs that it considers might be of interest to its clients.
- The RA forms a `CompositeCrlSpdu` structure containing those CRLs and encapsulates it in a composite CRL file as specified in 8.5.
- The RA makes the composite CRL file available for download via the interface defined in 6.3.5.8.

Second, an end entity may request an individual CRL from the RA as specified in 6.3.5.10.

Other mechanisms may be specified as the system evolves.

4.1.7 Certificate chain files

Verification of incoming messages depends on the receiver of the messages knowing and trusting the CA certificates in the sender's chain. This document specifies two mechanisms for an end entity to use to request CA certificate information from the RA or DC, enabling the end entity to form complete chains using those CA certificates.

First, an end entity may download a certificate chain file. This is the preferred mechanism, as follows:

- The RA is assumed to maintain a list of all CA certificates that it considers might be of interest to its clients.
- The RA forms a `CertificateChainSpdu` containing those CA certificates and encapsulates it in a certificate chain file as defined in 8.4.
- The RA makes the certificate chain file available for download via the interface defined in 6.3.5.7.

Second, an end entity may request an individual CA certificate from the RA as specified in 6.3.5.9.

These mechanisms supplement the P2PCD mechanism specified in IEEE Std 1609.2, which allows end entities to request that neighboring end entities provide CA certificates.

Other mechanisms may be specified as the system evolves.

4.1.8 Elector and root certificate authority management

In the system specified in this document, messages are trusted because they are signed by certificates whose chain starts with a trusted root CA certificate. Root CA certificates are self-signed certificates that issue other certificates. Since in principle a root CA can be established by any organization or individual that issues a self-signed certificate that can issue other certificates, it is important for the system to provide a mechanism that informs system participants of which root CAs meet the security and policy requirements set by the SCMS manager and so can be trusted. This mechanism is provided in this document by CTLs and electors.

Electors are entities that sign CTLs. CTLs are statements indicating that the following key components in the SCMS can be trusted to carry out their activities correctly:

- Root CAs
- Other electors

The CTL allows the system to add new trusted root CAs that conform to the policies set by the SCMS manager, as well as to declare formerly trusted root CAs as untrusted before their expiry date if necessary.

The key design feature of the elector-based system is that it permits the avoidance of a single point of failure, as follows:

- There may be more than one elector.
- Entities within the system only act on statements that change the trust state (i.e., add or remove the trust for a component) if those entities receive statements signed by a *quorum* of electors.
- The set of trusted electors can be changed without affecting the trust state of the root CAs.

The number of electors and the quorum are system parameters selected by the SCMS manager, and how they are chosen is outside the scope of this document. Previous research on the system design has recommended at least three electors and a quorum of more than half of the electors. The current number of valid electors is always indicated by the most recent CTL, as it lists all the valid electors. The CTL format specified in 7.3.15 also allows the quorum for future CTLs to be specified.

The electors provide protection against a single technical point of failure, as the CTL is only valid if it is signed by a quorum of electors. The unsigned CTL is created by the SCMS manager and signed by the electors. In principle this central creation of the CTL could create a single point of process failure, compared to a potential alternative design where each elector independently decides which certificate to trust. The intent is that this potential point of failure is addressed by transparency and clear processes, both within the SCMS manager and when the SCMS manager communicates new CTL contents to the electors. These processes are outside the scope of this document.

The CTL signatures are IEEE 1609.2 signed SPDUs and are subject to the standard validity conditions. The relevant formats are specified in 7.3.11 and 7.4.7. A profile for elector certificates is given in 7.7.3.7. The validity conditions for elector signatures are given in 10.2.

Figure 2 illustrates an example of how the elector-based scheme (shown inside dotted red boundary) can be implemented in parallel with a standard PKI hierarchy that supports all SCMS components and EEs. Note that all the structures shown here can be implemented with standard IEEE 1609.2 certificates without modification. A significant advantage of the elector-based scheme is that, as new electors are added, existing root CAs can be included in the CTL without having to change their certificates.

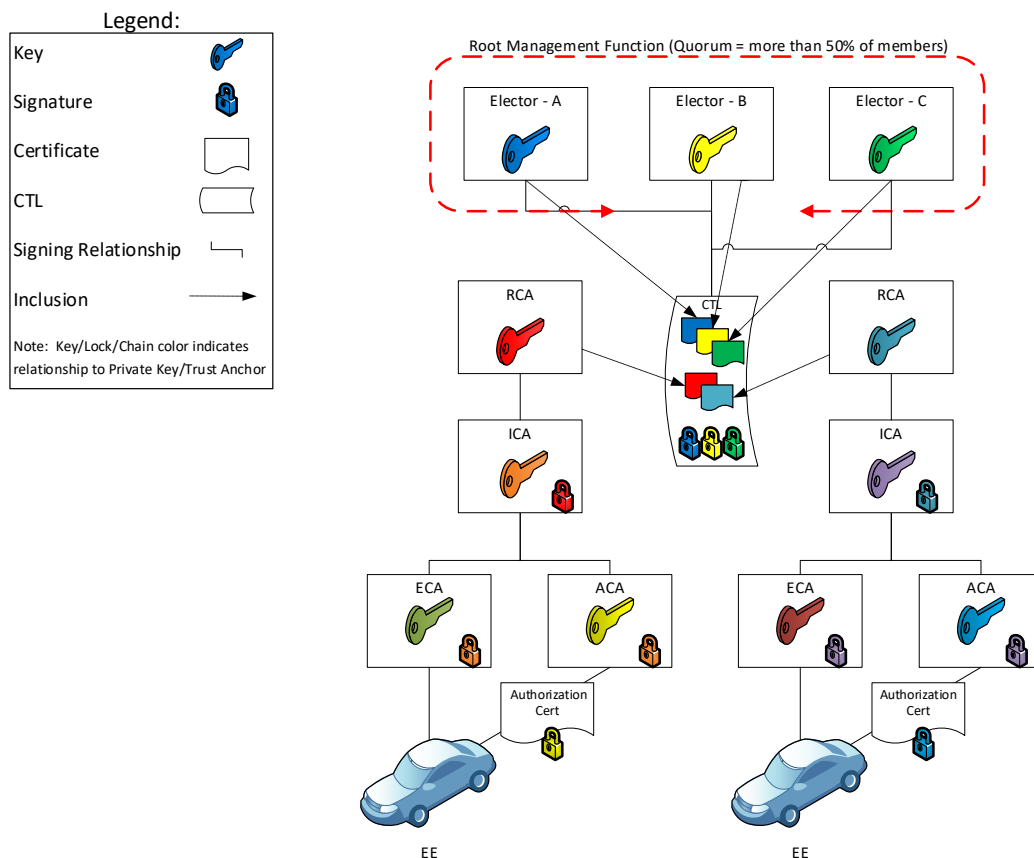


Figure 2—Elector system managing a standard PKI hierarchy

The resulting system may have multiple, self-signed root CA certificates, each of which operates at the top of their trust chain. Each root CA's certificate is endorsed by a CTL with at least a quorum of votes from nonrevoked electors. SPDU verifiers verify that a trust chain can be constructed to a trusted root CA. For SPDU verifiers operating in a system managed by electors as specified in this document, a root CA is trusted if it appears on a valid CTL.

NOTE 1—The certificate trust list (CTL) does not support linking of root CA certificates (i.e., binding a root CA certificate to its predecessor); there are no use cases within the IEEE 1609.2 system where the concept of a “successor” CA certificate is made use of. For discussion of related systems where “successor” certificates are used, see J.2.

NOTE 2—For discussion of multiple systems of electors, see J.4.

NOTE 3—IEEE Std 1609.2 supports multiple different signing algorithms and is extensible to support more. In a future version of the system, this could be made use of to improve robustness by having different electors use different algorithms, providing a hedge against potential breaks of any single algorithm. In particular, this could provide a migration path to quantum-safe cryptography in the event that quantum computing impacts the security of elliptic curve-based algorithms; see Bernstein et al. [B1], ETSI GR QSC 004 [B2], ETSI TS 102 941 [B3], and Simplicio et al. [B10]. This migration will be possible once IEEE Std 1609.2 supports quantum-safe signature algorithms, which in turn will happen once an appropriate standard specifying quantum-safe signature algorithms is available.

4.2 Interface approach

Interactions between end entities and SCMS components are specified as layered protocols in this document. A full specification of an interface consists of a reliable transport mechanism, a secure session mechanism, a Web API, and a set of APDUs. The layered approach is illustrated in Figure 3. Conceptually, the two parties 1) establish a reliable transport, 2) build a secure session on top of the reliable transport, and then 3) use the Web API to 4) exchange APDUs.

In this document, the assumption is that reliable transport is instantiated via TCP/IP. The ECA/RA <SERVER> (see 6.3) shall be configured in an EE before the EE initiates the reliable transport with the ECA/RA, communicating with it. How the EE is configured with ECA/RA <SERVER> is outside the scope of this document.

The secure session mechanism provides authentication of the SCMS component, as well as confidentiality. A list of supported secure session mechanisms is given in Clause 5.

The Web API identifies the activity that is being carried out across the interface and supports utility functionality such as download resumption for large files. This document only specifies one Web API that is provided in item b1) of 5.4.3.

The APDUs are the encoding of data structures that contain the detailed information necessary to support the end entity operations. These data structures are defined in Clause 7.

For discussion of how this layered approach may be implemented in practice, see Annex I.

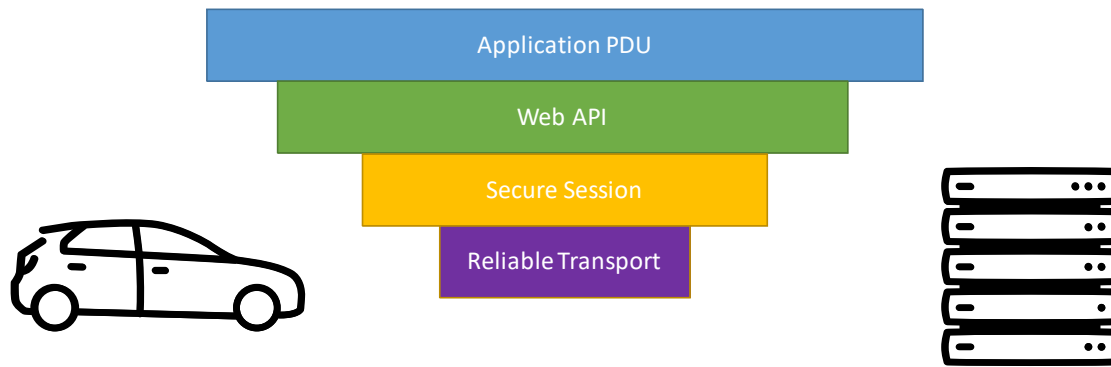


Figure 3—Layered approach to protocol definition

4.3 System parameters

4.3.1 General

The correct operation of the system depends on end entities knowing certain system parameters. These parameters are listed in this subclause. This document does not specify management mechanisms for provisioning or maintaining these parameters.

4.3.2 Time period parameters

The butterfly key mechanism (see 9.2 and 9.3) enables authorization certificates to be pregenerated by the SCMS for a series of time periods, referred to as *i*-periods.

When butterfly keys are used, the *i*-period identifying values (referred to as *i*-values) are used to derive the cocoon keys as specified in 9.2 and 9.3. The initial *i*-value is set by the RA and returned to the EE in the `RaEeCertAckSpdu` generated in response to an authorization certificate request using butterfly parameters.

The *i*-period may also be used for revocation of pseudonym certificates, that is, authorization certificates for which an EE has more than one certificate simultaneously valid and which are revoked using the `LinkageData` mechanism specified in IEEE Std 1609.2. In this case, the *i*-value appears in the `id.linkageData.iCert` field in the certificate. The `iCert` field is of type `IValue`, a 16 bit integer, and therefore can go up to 65 535.

For pseudonym certificates, it is possible that the certificates assigned to a particular *i*-period for revocation purposes may not have the same start and end validity times. This subclause specifies the mechanism for mapping from an *i*-period value to the validity periods of a set of certificates is parameterized by the parameters given in Table 1. All certificates with the same start and end validity time associated with a particular *i*-period are described in this section as belonging to the same “band” (for reasons that become visually obvious on looking at Figure 4, Figure 5, Figure 6, and Figure 7). Bands are numbered sequentially starting from 0. A batch of certificates shall always have a band 0 specified (although this may contain no certificate) and may have other bands. The certificates in band 0 always start at the start of the corresponding *i*-period and are valid until at least the end of the *i*-period.

NOTE—This document does not provide mechanisms for the EE to make requests that could be in conflict with the parameters in Table 1, and so there are no error codes (see Table 8) associated with them.

Table 1—Parameters—Mapping from time period index to validity periods

Parameter (time-[code])	Use
iPeriod	The current value of i-period.
iPeriodEpoch	The date at which i-periods of the indicated length started.
iPeriodInit	The value of i-period at <i>time-iPeriodEpoch</i> .
iPeriodLength	The length of time that an i-period lasts.
overlap	The length of time for which certificates from an i-period and certificates from the successive i-period are both valid.
baseCertNumber	The number of certificates in band 0.
overlapCertNumber	The number of certificates in band 0 that have their validity period lengthened into the overlap period.
extraBands	The number of extra bands.
Parameter (time-band[bandIndex]-[code]) (Each of these parameters is specified once for each band with band number > 0)	Use
startOffset	The time between the start of the main i-period and the start of the validity period of certificates in this band. Is less than time-iPeriodLength.
duration	The validity period of certificates in this band.
certsInBand	The number of certificates in the band.

Given these parameters, any entity can calculate the i-period that contains any date and the i-value associated with that i-period.

The mapping from these parameters to the validity period of certificates associated with the iPeriod value is as follows:

- For the time-baseCertNumber number of certificates in band 0, the validity period starts at $\text{band0-start[iPeriod]} = (\text{time-iPeriodEpoch}) + (\text{time-iPeriod} - \text{time-iPeriodInit}) \times \text{time-iPeriodLength}$ and has duration $\text{time-iPeriodLength} + \text{time-overlap}$.
- For the time-band[i]-certsInBand number of certificates in each other band i , $0 < i \leq \text{extraBands}$, the validity period starts at $\text{band0-start[iPeriod]} + \text{time-band[i]-startOffset}$ and has duration $\text{time-band[i].duration}$.

As an example, in the US Connected Vehicle Pilot Deployments, the i-period length was one week, the epoch was 4 am Eastern Time on Tuesday, January 6, 2015; the overlap period was one hour and the initial value was 0.

Components in the system make use of these parameters as follows:

- An end entity uses iPeriod to construct authorization certificate download requests as specified in 6.3.5.3.1, 7.4.17, and 8.1. The end entity obtains the iPeriod value for its initial download from the RaEeCertAckSpdu it received in response to the authorization certificate request.
- Either the RA or the SCMS manager may publish these parameters as a profile of the certificate “banding” for a particular application, to set expectations among EE implementers as to what they may implement. If these parameters are published in a profile, the “baseCertNumber” and “certsInBand” parameters may be published as a (min, max) range.
- An end entity that knows the parameters may use them to determine whether a set of downloaded certificates is consistent with the published profile. If the certificates are valid but not consistent with the profile, this indicates that the certificates can be used and also that the EE’s view of the system is out of date. In this case the EE should attempt to bring its parameters up to date. An interface for this is not provided in this version of this document.

These parameters are used to allow for a change of i-period length during the life of the system while maintaining sequential numbering of the i-period series.

The values of these parameters are a matter of policy. If the mapping from iPeriod to calendar time is consistent for all certificates that chain back to the same set of electors, it significantly reduces the complexity of calculations relating to i-periods. No mechanism is provided to change the mapping from iPeriod to calendar time, or to support having different calendar time periods for a given iPeriod, and so it is strongly recommended that the iPeriod-time mapping is a systemwide parameter used by all CAs that chain back to a given set of electors.

The overlap and offset approaches are provided to ensure that there is a smooth transition between periods even for end entities that may want to inhibit certificate change at the time when a new i-period is starting; in other words, that it is never the case that all of an end entity's certificates expire simultaneously and the next set only become valid on expiry of the previous set.

In the overlap approach, each set of certificates is valid for slightly longer than the associated iPeriodLength. This allows the use of certificates from the current set slightly into the next i-period. However, it means that for a period of time, an EE has more valid certificates than usual. This approach is therefore not appropriate for cases where there is a strict upper limit, set by policy, on the number of simultaneously valid certificates. This is illustrated in Figure 4 and Figure 5.

In the offset approach, different certificates in the set have different start validity times. This allows for the transition between i-periods to happen smoothly while managing the number of certificates that are simultaneously valid. This is illustrated in Figure 4 and Figure 5.

Figure 4 illustrates the overlap approach. This is the approach that has been used in the US Connected Vehicle Pilot Deployments (although the exact time of the change from one i-period to the next has been changed in Figure 4 for readability). In Figure 4, each device has 20 certificates, valid for a week plus a short overlap period. This would be described using the parameters in this subclause as follows:

- time-iPeriodLength: 1 week
- time-overlap: 1 hour
- time-baseCertNumber: 20
- time-overlapCertNumber: 20
- time-extraBands: 0

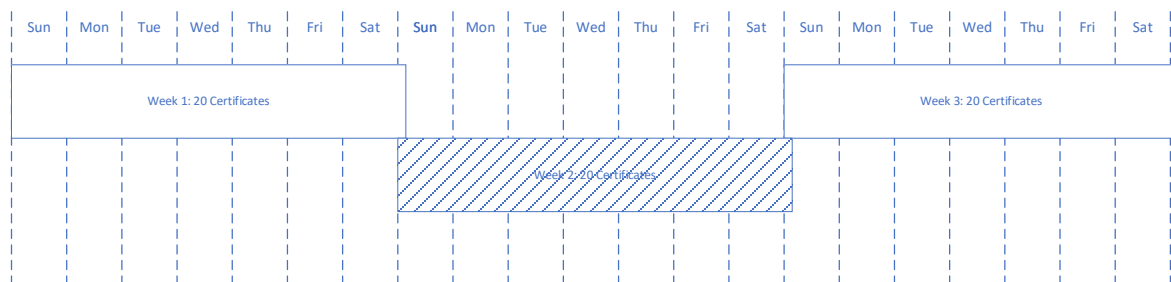


Figure 4—Overlap approach

Figure 5 illustrates the overlap approach for a subset of certificates. In Figure 5, each device has 20 certificates valid for a week, plus 2 certificates valid for a week plus a short overlap period. This would be described using the parameters in this subclause as follows:

- time-iPeriodLength: 1 week
- time-overlap: 1 hour

- time-baseCertNumber: 20
- time-overlapCertNumber: 2
- time-extraBands: 0

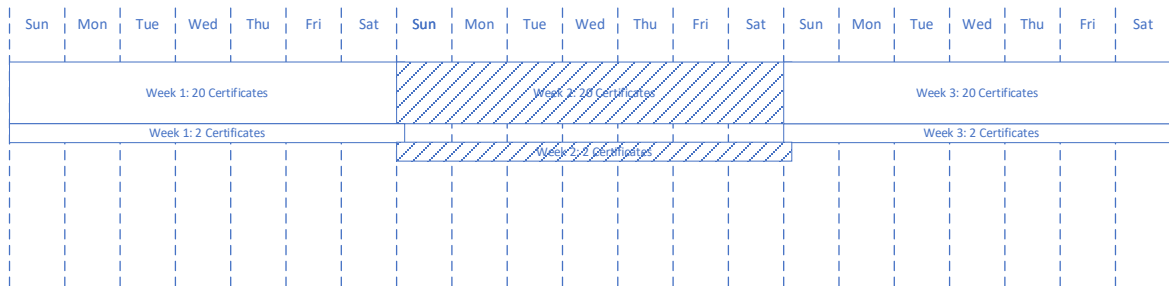


Figure 5—Overlap approach for a subset of certificates

Figure 6 illustrates one implementation of the offset approach. In Figure 6, each device has 90 certificates valid for a week, plus ten certificates that “bridge” from one week to the next. This would be described using the parameters in this subclause as follows:

- time-iPeriodLength: 1 week
- time-overlap: 0
- time-baseCertNumber: 90
- time-overlapCertNumber: 0
- time-extraBands: 1
- time-band1-startOffset: Six days, eight hours
- time-band1-duration: 32 hours
- time-band1-certsInBand: 10



Figure 6—Implementation of the offset approach

Figure 7 illustrates an alternative implementation of the offset approach. In Figure 7, each device has 98 certificates valid for a week, with 14 certificates starting their validity period on each day of that week. This would be described using the parameters in this subclause as follows:

- time-iPeriodLength: 1 week
- time-overlap: 0
- time-baseCertNumber: 14
- time-overlapCertNumber: 0
- time-extraBands: 6
- time-band1-startOffset: 1 day
- time-band1-duration: 1 week
- time-band1-certsInBand: 14
- time-band2-startOffset: 2 days
- time-band2-duration: 1 week
- time-band2-certsInBand: 14
- time-band3-startOffset: 3 days

- time-band3-duration: 1 week
- time-band3-certsInBand: 14
- time-band4-startOffset: 4 days
- time-band4-duration: 1 week
- time-band4-certsInBand: 14
- time-band5-startOffset: 5 days
- time-band5-duration: 1 week
- time-band5-certsInBand: 14
- time-band6-startOffset: 6 days
- time-band6-duration: 1 week
- time-band6-certsInBand: 14

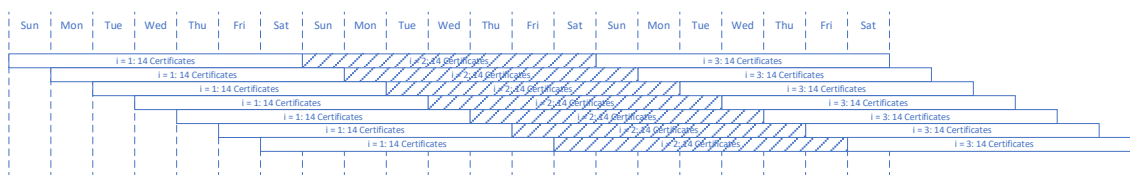


Figure 7—Alternative implementation of the offset approach

4.3.3 Session parameters

The session related parameters are specified in Table 2.

Table 2—Parameters—Session

Parameter (session-[code])	Value	Meaning
scmsAuth	iso21177	Session is protected cryptographically using ISO/TS 21177.
	physical	Session security is maintained by the fact that both endpoints of the session are in the same physically secure environment.
	tls1.2	Session is protected cryptographically using TLS 1.2.
	tls1.3	Session is protected cryptographically using TLS 1.3.
eeAuth	iso21177-authorization	Authentication by the EE at the secure session level is done within an ISO/TS 21177 TLS handshake using a valid IEEE 1609.2 authorization certificate.
	iso21177-enrollment	Authentication by the EE at the secure session level is done within an ISO/TS 21177 TLS handshake using a valid IEEE 1609.2 enrollment certificate.
	none	No authentication by the EE at the secure session level.
	physical	Authentication by the EE at the secure session level is maintained by the fact that both endpoints of the session are in the same physically secure environment.
	tls1.2-x.509	Authentication by the EE at the secure session level is done within a TLS 1.2 handshake using a valid ITU-T X.509 certificate.
	tls1.3-x.509	Authentication by the EE at the secure session level is done within a TLS 1.3 handshake using a valid ITU-T X.509 certificate.

4.3.4 Web API parameters: generic

The generic Web API-related parameters are specified in Table 3.

Table 3—Parameters—Web API (generic)

Parameter (webApi-[code])	Value	Meaning
name	scmsV2	Name of the Web API is SCMS REST API v2.
eeAuth	none	No authentication by the EE at the web API level.
	oAuth2.0	Authentication by the EE at the web API level is done using OAuth 2.0 bearer authorization.

4.3.5 Web API parameters: SCMS REST API v2

The Web API related parameters specific to SCMS REST API v2 are specified in Table 4.

Table 4—Web API Parameters—SCMS REST API v2

Parameter (scmsV2-[code])	Value	Meaning
eeAuth	authorization	Authentication by the EE at the SCMS REST API v2 level is done using a valid IEEE 1609.2 authorization certificate.
	canonical	Authentication by the EE at the SCMS REST API v2 level is done by the EE using a valid canonical key. ¹⁶
	enrollment	Authentication by the EE at the SCMS REST API v2 level is done by the EE using a valid IEEE 1609.2 enrollment certificate.
	none	No authentication by the EE at the SCMS REST API v2 level.
	x509	Authentication by the EE at the SCMS REST API v2 level is done by the EE using a valid ITU-T X.509 enrollment certificate.
error	coarse	No information is returned in case of error, just an HTTP code of 500.
	fine	Errors are returned as specified for each command.
options	supported options	A service provider may support any HTTP optional headers. Two particularly useful ones are “If-Unmodified-Since” and “Range”.

4.3.6 Use case related parameters

The use case related parameters are specified in Table 5, Table 6, and Table 7.

Table 5 provides parameters used for requests to the ECA. The only such request defined in this document is enrollment certificate request (see 6.3.4.2).

Table 5—Parameters—ECA request

Parameter (eca-[code])	PSID-specific?	Meaning
maxAge	No	Maximum age of the enrollment certificate request when received by the ECA, that is, the maximum amount of time in the past that the generationTime field in the EeEcaCertRequest can represent relative to the time that the ECA receives the request.
maxReqs	No	Maximum number of requests in a 7 × 24-hour period. The requests within a given period are determined based on the generationTime field in the request, not on the reception time at the ECA.
maxWait	No	Maximum time limit between the generationTime field and the tbsCert.validityPeriod.start field in the EeEcaCertRequest.
minWait	No	Minimum time that an EE shall wait before retrying the request.

¹⁶ This is only supported for initial enrollment certificate request.

Table 6 provides parameters used for requests to the RA. The requests defined in this document are authorization certificate request (see 6.3.5.2), successor enrollment certificate request (see 6.3.5.4), and misbehavior report submission (see 6.3.5.6). The Misbehavior report submission request is not parameterized by any of the parameters in Table 6.

NOTE—This document does not provide mechanisms for the EE to make requests that could be in conflict with the parameter `ra-maxPreloadTime` in Table 6, and so there are no error codes (see Table 8) associated with it. The maximum preload time is set by the SCMS manager for each PSID (and the actual preload time may be different for different RA instances even within that PSID). This document does not provide a mechanism for that preload time to be communicated to the EE, but implementers may wish to provide such a mechanism to allow EEs to plan storage usage.

Table 6—Parameters—RA request

Parameter (ra-[code])	PSID- specific?	Meaning	Use case
<code>acpcSupport</code>	Yes	Indicates whether or not ACPC is supported by the RA.	Authorization certificate request
<code>butterflyType</code>	Yes	Indicates which variations of the butterfly key mechanism are supported by the RA. Current options include original, unified, and compact unified.	Authorization certificate request
<code>maxAge</code>	Yes	Maximum age of the authorization certificate request for a particular set of certificate permissions (PSID/SSP) when received by the RA, that is, the maximum amount of time in the past that the <code>generationTime</code> field in the <code>EeRaCertRequest</code> can represent relative to the time that the RA receives the request.	Authorization certificate request
<code>maxGenDelay</code>	Yes	Maximum time between <code>generationTime</code> in the <code>EeRaCertRequest</code> and the requested start time of the certificates (requested using the <code>validityPeriod</code> field in the <code>tbsCert</code> field).	Authorization certificate request
<code>maxReqs</code>	Yes	Maximum number of requests in a 7 × 24-hour period for a particular set of certificate permissions (PSID/SSP) authorized by the same enrollment certificate. The requests within a given period are determined based on the <code>generationTime</code> field in the request, not on the reception time at the RA.	Authorization certificate request
<code>maxPreloadTime</code>	Yes	Maximum time from the current time that the SCMS will generate certificates for an EE, for example, up to 3 years from the current time.	Authorization certificate request
<code>minWait</code>	Yes	Minimum time that an EE shall wait before retrying the request for a particular set of certificate permissions (PSID/SSP) authorized by the same enrollment certificate.	Authorization certificate request
		Minimum time that an EE shall wait before retrying the request for a successor enrollment certificate	Successor enrollment certificate request

Table 7 provides parameters used for download services at the RA or at a DC. The download services supported at the RA are as follows: authorization certificate download (see 6.3.5.3), successor enrollment certificate download (see 6.3.5.5), CCF including CTL download (see 6.3.5.7), composite CRL including CTL download (see 6.3.5.8), individual CA certificate download (see 6.3.5.10), CTL download (see 6.3.5.11), and RA certificate download (see 6.3.5.12). The download services supported at the DC are: CCF including CTL download (see 6.3.5.7), composite CRL including CTL download (see 6.3.5.8), individual CA certificate download (see 6.3.5.10), and CTL download (see 6.3.5.11).

Table 7—Parameters—Download services

Parameter (download-[code])	PSID- specific?	Meaning
maxAge	Yes	Maximum age of a request for a particular set of certificate permissions (PSID/SSP) when received by the RA, that is, the maximum amount of time in the past that the <code>generationTime</code> field in the <code>EeRaDownloadRequest</code> can represent relative to the time that the RA receives the request.
maxReqs	Yes	Maximum number of requests in a 7×24 -hour period. The requests within a given period are determined based on the <code>generationTime</code> field in the request, not on the reception time at the RA.
minWait	Yes	Minimum time that an EE shall wait before retrying the request.

5. Secure session

5.1 General

The secure session mechanism is characterized by the *session-scmsAuth* and *session-eeAuth* parameters from Table 2. Operations at the Web API level vary depending on these parameters and otherwise do not vary depending on any characteristic of the secure session mechanism in use. The intended use of the session security parameters is twofold, as follows:

- Operations at the Web API level may include options that depend on the parameters.
- The policy for implementation of a particular deployment of a particular use case may require these parameters to have certain values. (For example, an implementation of the enrollment certificate request use case may require *session-scmsAuth* = *physical*).

This document supports the following types of secure session:

- Using physical security, indicated by *session-scmsAuth* = *physical*.
- Using TLS, as follows:
 - 1) TLS 1.2, indicated by *session-scmsAuth* = *tls1.2*.
 - 2) TLS 1.3, indicated by *session-scmsAuth* = *tls1.3*.
- Using ISO/TS 21177, indicated by *session-scmsAuth* = *iso21177*. In this case, although the parameter name includes the term “scmsAuth”, both the SCMS component and the end-entity will authenticate at the session level per the ISO/TS 21177 specification.

At the secure session level, this document supports the following types of end entity authentication:

- None, indicated by *session-eeAuth* = *none*
- Using an authorization certificate of the end entity as the client handshake certificate within an ISO/TS 21177 secure session, indicated by *session-eeAuth* = *iso21177-authorization*
- Using the enrollment certificate of the end entity as the client handshake certificate within an ISO/TS 21177 secure session, indicated by *session-eeAuth* = *iso21177-enrollment*

This model supports terminating the secure session at a different network location from the network location of the SCMS component application. If this is done, then if any security properties of the end entity are established via the secure session rather than via the Web API, the system is only secure if those properties are transmitted securely from the endpoint of the secure session to the endpoint of the Web API. An architecture to establish this is outside the scope of this document. See Annex I for more details.

5.2 Physical security

In a physically secure session, security is provided by the fact that both endpoints of the session are in the same physically secure environment. This means that each party is assumed to know all relevant properties of the other party with a high level of assurance.

The definition of a physically secure environment is outside the scope of this document.

5.2.1 Protocol parameters

The secure session based on physical security is characterized by the following values of the session security parameters:

- a) *session-scmsAuth* = *physical*
- b) *session-eeSessionAuth* = *none*

5.3 Transport Layer Security (TLS)

5.3.1 General / prerequisites

The secure session protocol may be implemented using TLS as specified in IETF RFC 5246 (for TLS 1.2) and IETF RFC 8846 (for TLS 1.3).

The prerequisites for use of TLS are as follows:

- **SCMS component certificate and certificate acceptance policy:**
 - 1) There is an *ITU-T X.509 certificate acceptance policy* that is known to both the SCMS component and the end entity:
 - i) The certificate acceptance policy for the end entity specifies a pattern (or exact value) that the `subjectAltName` field in an SCMS component X.509 certificate is required to match in order for the certificate to be accepted as coming from an SCMS component of the correct type:
 - A natural condition within an acceptance policy would be to require that the `subjectAltName` matches the identifying URL for the SCMS component.
 - The certificate acceptance policy may also include further required properties of the SCMS component certificate, so long as the end entity can verify that those properties apply. For example, the policy could identify required `subjectAltName` patterns for the CA certificates that issued the SCMS component certificate, or identify specific certificates that are required to be in the full certificate chain for that SCMS component certificate, or that the certificate is an Extended Validation certificate [B5].
 - 2) The SCMS component is the holder of an ITU-T X.509 certificate that is acceptable per the certificate acceptance policy.
- **Root CA certificate:** The end entity knows and trusts the ITU-T X.509 root CA certificate at the start of the SCMS component's full ITU-T X.509 certificate chain.
- **Online Certificate Status Protocol (OCSP) support:** The SCMS component shall have an ongoing relationship with its CA that enables the CA to issue OCSP responses regarding the SCMS component per IETF RFC 6960.

NOTE—It is assumed that the end entity supports mechanisms to update certificate acceptance policy and update X.509 root CA certificate. These mechanisms and how they are authenticated are out of scope for this document.

5.3.2 Protocol constraints

The use of the TLS protocol is subject to the following constraints:

- **SCMS component role:** The SCMS component shall take the role of the TLS server.
- **SCMS component identification:** The SCMS component shall authenticate itself to the end entity (the client) using an ITU-T X.509 certificate *X*. The end entity shall accept a connection with the SCMS component only if *X* matches an ITU-T X.509 certificate acceptance policy.
- **OCSP Stapling:** The EE shall use the TLS certificate status request extension (OCSP stapling per IETF RFC 6961 (for TLS 1.2) or IETF RFC 8446 (for TLS 1.3)) to verify SCMS component revocation status. The SCMS component shall respond to an EE request for an OCSP stapled certificate.
- **Supported ciphersuites:** Compliant implementations shall support at least the ciphersuites identified below and may support more:
 - 1) **TLS 1.2:** For TLS 1.2, both actors shall support the ciphersuite TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as specified in IETF RFC 5289.
 - 2) **TLS 1.3:** For TLS 1.3, both actors shall support the ciphersuite TLS_AES_128_CCM_SHA256, the signature algorithm ecdsa_secp256r1_sha256, and the key exchange group secp256r1 as specified in IETF RFC 8446.
- **TLS session lifetime:** An implementation that uses TLS may support session resumption. The maximum lifetime of a session, that is, the maximum time after which a client is forced to execute a new handshake rather than resuming the session, is out of scope for this document.

5.3.3 Protocol options

The protocol may optionally use client authentication. If this is the case, there shall be an ITU-T X.509 certificate acceptance policy that applies to the client. The certificate acceptance policy may include subjectAltName constraints, issuer constraints, or any other constraints that the RA or ECA operator deems appropriate.

5.3.4 Protocol parameters

The secure session based on TLS is characterized by the following values of the session security parameters:

- a) *session-scmsAuth* =
 - 1) *tls1.2*, if TLS 1.2 is used
 - 2) *tls1.3*, if TLS 1.3 is used
- b) *session-eeAuth* =
 - 1) *none*, if there is no EE authentication within the session
 - 2) *tls1.2-x.509*, if EE authentication is done using the EE's X.509 certificate within a TLS 1.2 handshake
 - 3) *tls1.3-x.509*, if EE authentication is done using the EE's X.509 certificate within a TLS 1.3 handshake

5.4 ISO/TS 21177

5.4.1 General/prerequisites

The secure session protocol may be implemented using ISO/TS 21177.

The prerequisites for use of ISO/TS 21177 are as follows:

- **SCMS component certificate and certificate acceptance policy:**
 - 1) There is an *IEEE 1609.2 certificate acceptance policy* that is known to both the SCMS component and the end entity. This policy specifies the required properties of the SCMS component certificate and the end entity certificates (ISO/TS 21177 requires client authentication so the end entity is also required to have a certificate to use this secure session mechanism):
 - i) The certificate acceptance policy may address the contents of the `CertificateId` field, the issuing CAs in the chain, or any other properties of the certificate that seem appropriate to the component operator:
 - A natural condition within an acceptance policy would be to require that the `CertificateId` matches the identifying URL for the SCMS component.
 - 2) The SCMS component and the end entity both use IEEE 1609.2 certificates that are acceptable per the certificate acceptance policy.
 - 3) The SCMS component and the end entity both use IEEE 1609.2 certificates that match the appropriate certificate profile given in 7.7.3.
- **Root CA certificate:** The end entity knows and trusts the root CA certificate at the start of the SCMS component's full certificate chain, and similarly the SCMS component knows and trusts the root CA certificate at the end of the end entity's full certificate chain.

5.4.2 Protocol constraints

The use of the ISO/TS 21177 protocol is subject to the following constraints:

- **SCMS component role:** The SCMS component shall take the role of the ISO/TS 21177 server.
- **SCMS component identification:** The SCMS component shall authenticate itself to the end entity (the client) using an IEEE 1609.2 certificate *X*. The end entity shall accept a connection with the SCMS component only if *X* matches the certificate acceptance policy.
- **End entity identification:** The end entity shall authenticate itself to the SCMS component using an IEEE 1609.2 certificate *X*. The SCMS component shall accept a connection with the end entity only if *X* matches the certificate acceptance policy.
- **Supported ciphersuites:** Compliant implementations shall support at least the ciphersuites identified in ISO/TS 21177.
- **Handshake:** The implementation of ISO/TS 21177 shall allow authentication to take place entirely on the basis of the certificates exchanged in the TLS handshake and shall not use Extended or Enhanced Authorization.
- **Private Service Discovery:** The implementation of ISO/TS 21177 shall not use the private service discovery and proxying feature.
- **Session Lifetime:** The implementation of ISO/TS 21177 shall not use the session extension mechanism defined in ISO/TS 21177 and shall not use TLS session resumption.

- **Both participants are authenticated:** Per the ISO/TS 21177 specification, both the client and the server shall authenticate if ISO/TS 21177 secure sessions are used: There is no option not to use client authentication.

5.4.3 Protocol parameters

The secure session based on ISO/TS 21177 is characterized by the following values of the session security parameters:

- a) *session-scmsAuth* = *iso21177*
- b) *session-eeAuth* =
 - 1) *authorization-iso21177*, if authentication is done by the EE using a valid IEEE 1609.2 authorization certificate
 - 2) *enrollment-iso21177*, if authentication is done by the EE using a valid IEEE 1609.2 enrollment certificate

5.4.4 Client authentication with an enrollment certificate

If a client (i.e., an end entity) authenticates with an enrollment certificate within an ISO/TS 21177 handshake, the *Ieee1609Dot2Data* within the *CertificateVerify* message shall be signed with the PSID value *SecurityMgmtPsid*. The enrollment certificate shall conform to the enrollment certificate profile provided in 7.7.3.5.

6. Web API

6.1 General

6.1.1 General

In this document, the Web API is the mechanism that delivers APDUs from a requester to a service provider and back, over a secure session as defined in Clause 5. A web API may indicate semantics (for example, indicating which service is being requested) in addition to transporting the APDUs.

The Web API mechanism is characterized by the *webApi-name* and *webApi-eeAuth* parameters from 4.3.4. Operations at the APDU level vary depending on these parameters and otherwise do not vary depending on any characteristic of the Web API mechanism in use. The intended use of the Web API security parameters is twofold, as follows:

- Operations at the APDU level may include options that depend on the parameters.
- The policy for implementation of a particular deployment of a particular use case may require these parameters to have certain values. For example, an implementation of the authorization certificate request use case may require *webApi-eeAuth* = *oAuth2.0*.

This document supports the following types of Web API:

- SCMS REST API v2, indicated by *webApi-name* = *scmsV2*

At the Web API level, this document supports the following types of end entity authentication:

- None, indicated by *webApi-eeAuth = none*
- Using OAuth 2.0 bearer authorization, indicated by *webApi-eeAuth = oAuth2.0*

A web API specification may support different options for its commands. If this is the case, service providers are expected to publish a profile of their service indicating which options are supported. The means by which the profile is communicated to clients (and updated if appropriate) is outside the scope of this document.

NOTE—The interfaces specified in this document (ACA – RA, DC – EE, ECA – EE, and EE – RA) can be implemented by the same or different hostname in the REST API procedure.

6.1.2 Protocol parameters

The Web API is characterized by the following values of the Web API security parameters:

- a) *webApi-name = scmsV2*
- b) *webApi-eeAuth =*
 - 1) *none*, if there is no authentication by the EE
 - 2) *oAuth2.0*, if authentication is done by the EE using a valid OAuth 2.0 bearer authorization token

6.2 OAuth 2.0 bearer authorization

6.2.1 General

This document specifies one mechanism to instantiate the supplementary authorization service described in 4.1.1; this mechanism is OAuth 2.0 bearer authorization. OAuth 2.0 is an optional feature of the SCMS that allows an EE to authenticate with an OAS to receive an AT. The EE then includes that Access Token in requests to the RA. The RA then uses the Access Token to authorize the EE request.

OAuth 2.0 may be implemented to provide access control at the EE – RA interface and protect the RA from unauthorized commands. OAuth 2.0 is a set of open standards provided by IETF (RFC 6749) for access delegation. It is commonly used as a way for Internet users to grant websites or applications access to their information on other websites without giving them the passwords. It also defines authentication and authorization outside the context of a user, called CCG; see section 4.4. of IETF RFC 6749. IETF RFC 6749 defines the framework for OAuth 2.0, and there are several additional RFCs that add features to this (compare <https://www.oauth.com/oauth2-servers/map-oauth-2-0-specs/>).

Using OAuth 2.0 and bearer authorization with CCG instantiates the SAS with an OAS in the SCMS architecture (see 4.1) and the following two additional policies:

- a) The OAS shall have an acceptance policy that defines the requirements a client has to fulfill to be granted authorization and get an AT issued. The OAS shall not issue an AT to an EE that is blocked or unregistered per 4.1.4.3.1.
- b) The RA shall have an AT acceptance policy that defines requirements for the AT that have to be fulfilled to give the client access to the invoked service API.

For all use cases in this document that permit the use of OAuth, OAuth is optional. Whether in practice OAuth can be used for that use case is a matter of system policy and, if permitted but optional by the system policy, of the choice of each individual provider of that use case. Providers are encouraged to publish a profile of each use case that they provide using the profile template in Annex C so that potential clients are informed as to which authentication mechanism and other optional features are supported by that particular provider for that use case.

6.2.2 Protocol flow

An EE authenticates with the OAS to obtain an AT as defined in IETF RFC 6749. The EE shall use a certificate, which may be the enrollment certificate, as a client credential to authenticate with the OAS. Other authentication mechanisms permitted by OAuth 2.0 shall not be used for this interaction.

The OAS shall grant or reject authorization based on OAS policy. The OAS shall issue an AT in the case that authorization is granted. The OAS shall not issue an AT to an EE that is blocked or unregistered per 4.1.4.3.1.

When the RA receives the AT in a request from the EE, the RA authorizes requests depending on the AT acceptance policy.

ATs shall be self-contained JWT as defined by IETF RFC 7519 and IETF RFC 7515.

The AT shall contain at least the following headers:

- "typ": type of the token - shall be "JWT"
- "alg": identifier for the signing algorithm being used in this JWT

The AT shall contain at least the following registered payload claims:

- "iss": The "iss" (issuer) claim is a case-sensitive string containing a StringOrURI value identifying the principal that issued the JWT.
- "aud": The "aud" (audience) claim identifies the recipients that the JWT is intended for. Each principal intended to process the JWT shall identify itself with a value in the audience claim. If the principal processing the claim does not identify itself with a value in the "aud" claim when this claim is present, then the JWT shall be rejected. In the general case, the "aud" value is an array of case-sensitive strings, each containing a StringOrURI value. In the special case when the JWT has one audience, the "aud" value MAY be a single case-sensitive string containing a StringOrURI value.
- "iat": The "iat" (issued at) claim identifies the time when the JWT was issued. This claim can be used to determine the age of the JWT and therefore identify replay attacks. Its value shall be a number containing a NumericDate value.
- "exp": The "exp" (expiration time) claim identifies the expiration time on or after which the JWT shall not be accepted for processing. The processing of the "exp" claim requires that the current date/time shall be before the expiration date/time listed in the "exp" claim.