# IEC 62351-9

Edition 2.0 2023-06

# INTERNATIONAL STANDARD

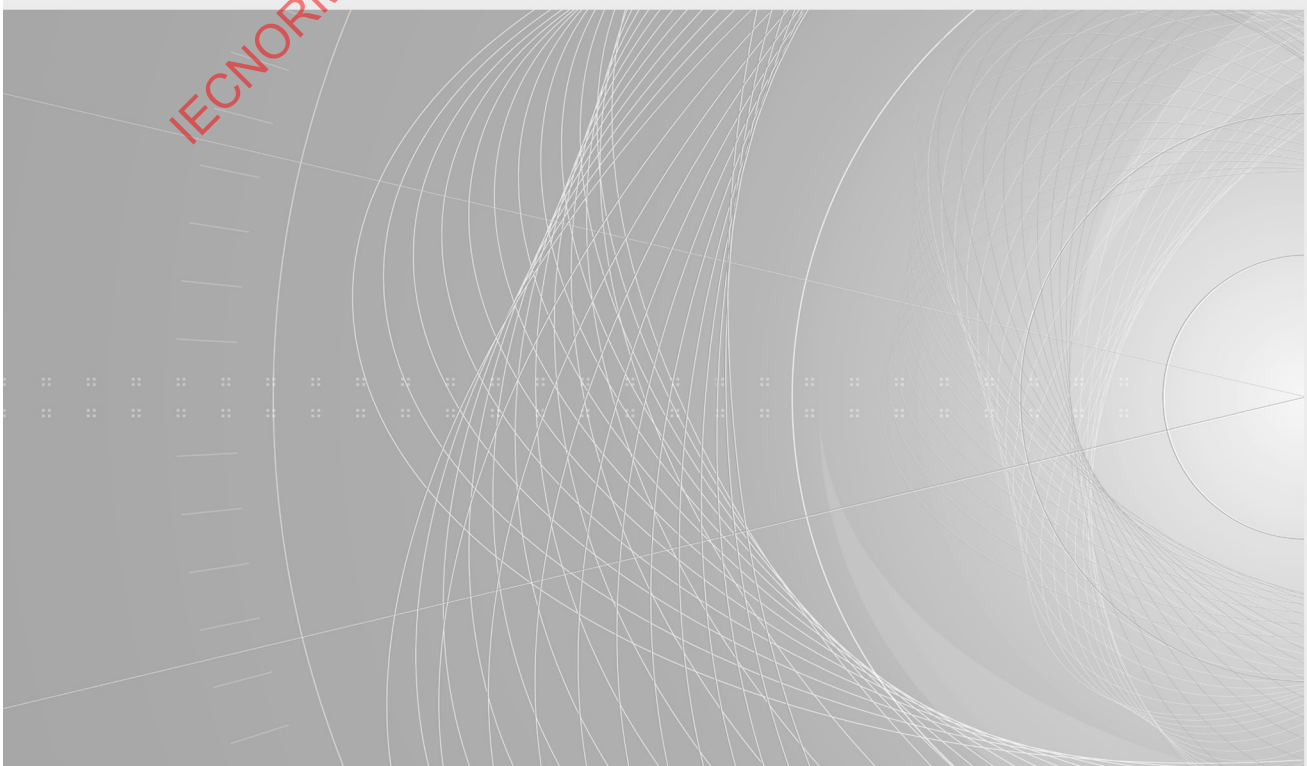
# NORME INTERNATIONALE

colour inside



## Power systems management and associated information exchange – Data and communications security – Part 9: Cyber security key management for power system equipment

## Gestion des systèmes de puissance et échanges d'informations associés – Sécurité des communications et des données – Partie 9: Gestion de clé de cybersécurité des équipements de système de puissance



IEC 62351-9:2023-06(en-fr)

IEC Secretariat
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
info@iec.ch
www.iec.ch

IEC 62351-9

Edition 2.0 2023-06

# INTERNATIONAL STANDARD

# NORME INTERNATIONALE

colour inside



## Power systems management and associated information exchange – Data and communications security – Part 9: Cyber security key management for power system equipment

## Gestion des systèmes de puissance et échanges d'informations associés – Sécurité des communications et des données – Partie 9: Gestion de clé de cybersécurité des équipements de système de puissance

INTERNATIONAL ELECTROTECHNICAL COMMISSION

COMMISSION ELECTROTECHNIQUE INTERNATIONALE

ICS 33.200

ISBN 978-2-8322-6950-3

# CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

# POWER SYSTEMS MANAGEMENT AND ASSOCIATED INFORMATION EXCHANGE – DATA AND COMMUNICATIONS SECURITY –

## Part 9: Cyber security key management for power system equipment

## FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
6) All users should ensure that they have the latest edition of this publication.
7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.



IEC 62351-9 has been prepared by WG15: Data and Communication Security, of IEC technical committee TC57: Power systems management and associated information exchange. It is an International Standard.

This second edition cancels and replaces the first edition published in 2017. This edition constitutes a technical revision.

This edition includes the following significant technical changes with respect to the previous edition:

a) Certificate components and verification of the certificate components have been added;
b) GDOI has been updated to include findings from interop tests;
c) GDOI operation considerations have been added;
d) GDOI support for PTP (IEEE 1588) support has been added as specified by IEC/IEEE 61850-9-3 Power Profile;
e) Cyber security event logging has been added as well as the mapping to IEC 62351-14;

f) Annex B with background on utilized cryptographic algorithms and mechanisms has been added.

The text of this International Standard is based on the following documents:

| Draft | Report on voting |
|---|---|
| 57/2579/FDIS | 57/2594/RVD |

Full information on the voting for its approval can be found in the report on voting indicated in the above table.

The language used for the development of this International Standard is English.

This document was drafted in accordance with ISO/IEC Directives, Part 2, and developed in accordance with ISO/IEC Directives, Part 1 and ISO/IEC Directives, IEC Supplement, available at www.iec.ch/members_experts/refdocs. The main document types developed by IEC are described in greater detail at www.iec.ch/publications.

NOTE The following print types are used:

Abstract Syntax Notation One (ASN.1): `in courier new` and **`bold courier new`** `type`.

The committee has decided that the contents of this document will remain unchanged until the stability date indicated on the IEC website under webstore.iec.ch in the data related to the specific document. At this date, the document will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

**IMPORTANT – The "colour inside" logo on the cover page of this document indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.**

# POWER SYSTEMS MANAGEMENT AND ASSOCIATED INFORMATION EXCHANGE – DATA AND COMMUNICATIONS SECURITY –

## Part 9: Cyber security key management for power system equipment

## 1 Scope

This part of IEC 62351 specifies cryptographic key management, primarily focused on the management of long-term keys, which are most often asymmetric key pairs, such as public-key certificates and corresponding private keys. As certificates build the base this document builds a foundation for many IEC 62351 services (see also Annex A). Symmetric key management is also considered but only with respect to session keys for group-based communication as applied in IEC 62351-6. The objective of this document is to define requirements and technologies to achieve interoperability of key management by specifying or limiting key management options to be used.

This document assumes that an organization (or group of organizations) has defined a security policy to select the type of keys and cryptographic algorithms that will be utilized, which may have to align with other standards or regulatory requirements. This document therefore specifies only the management techniques for these selected key and cryptography infrastructures. This document assumes that the reader has a basic understanding of cryptography and key management principles.

The requirements for the management of pairwise symmetric (session) keys in the context of communication protocols is specified in the parts of IEC 62351 utilizing or specifying pairwise communication such as:

- IEC 62351-3 for TLS by profiling the TLS options
- IEC 62351-4 for the application layer end-to-end security
- IEC TS 62351-5 for the application layer security mechanism for IEC 60870-5-101/104 and IEEE 1815 (DNP3)

The requirements for the management of symmetric group keys in the context of power system communication protocols is specified in IEC 62351-6 for utilizing group security to protect GOOSE and SV communication. IEC 62351-9 utilizes GDOI as already IETF specified group-based key management protocol to manage the group security parameter and enhances this protocol to carry the security parameter for GOOSE, SV, and PTP.

This document also defines security events for specific conditions which could identify issues which might require error handling. However, the actions of the organisation in response to these error conditions are beyond the scope of this document and are expected to be defined by the organizations security policy.

In the future, as public-key cryptography becomes endangered by the evolution of quantum computers, this document will also consider post-quantum cryptography to a certain extent. Note that at this time being no specific measures are provided.

## 2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC TS 62351-2, *Power systems management and associated information exchange – Data and communications security – Part 2: Glossary of terms*

IEC 62351-3:—[1], *Power systems management and associated information exchange – Data and communications security – Part 3: Communication network and system security – Profiles including TCP/IP*

IEC 62351-4, *Power systems management and associated information exchange – Data and communications security – Part 4: Profiles including MMS and derivatives*

IEC 62351-5, *Power systems management and associated information exchange – Data and communications security – Part 5: Security for IEC 60870-5 and derivatives*

IEC 62351-6, *Power systems management and associated information exchange – Data and communications security – Part 6: Security for IEC 61850*

IEC 62351-14:—[2], *Power systems management and associated information exchange – Data and communications security – Part 14: Cyber security event logging*

ISO/IEC 9594-8:2020, Rec. ITU-T X.509 (2019), *Information technology – Open systems interconnection – The Directory: Public-key and attribute certificate frameworks*

ISO/IEC 9594-11:2020, Rec. ITU-T X.510 (2020), *Information technology – Open systems interconnection – The Directory: Protocol specifications for secure operations*

ISO/IEC 9834-1:2012, Rec. ITU-T X.660 (2011), *Information technology – Procedures for the operation of object identifier registration authorities: General procedures and top arcs of the international object identifier tree*

IETF RFC 5272, *Certificate Management over CMS (CMC)*

IETF RFC 5755, *An Internet Attribute Certificate Profile for Authorization*

IETF RFC 5934, *Trust Anchor Management Protocol (TAMP)*

IETF RFC 6407, *The Group Domain of Interpretation*

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

IETF RFC 7030, *Enrolment over Secure Transport*

IETF RFC 8052, *Group Domain of Interpretation (GDOI) Protocol Support for IEC 62351 Security*



---

[1] Under preparation. Stage at the time of publication: IEC/RFDIS 62351-3:2023.

[2] Under preparation. Stage at the time of publication: IEC/ACDV 62351-14:2023.

IETF RFC 8263, *Group Domain of Interpretation (GDOI) GROUPKEY-PUSH Acknowledgement Message*

IETF RFC 8894, *Simple Certificate Enrolment Protocol*

## 3 Terms, definitions, and abbreviations

### 3.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at http://www.electropedia.org/
- ISO Online browsing platform: available at http://www.iso.org/obp

**3.1.1**
**asymmetric keys**
two related keys, a public key and a private key, that are used to perform complementary operations, such as encryption and decryption or signature generation and signature verification

**3.1.2**
**authorization and validation list**
AVL
signed list containing information to an AVL entity about potential communications entities and possible restrictions on the communications with such entities

[SOURCE: ISO/IEC 9594-8:2020, 3.5.9]

**3.1.3**
**authorization and validation list entity**
AVL entity
entity, when acting as a relying party, which is dependent on an AVL issued by a designated authorizer

[SOURCE: ISO/IEC 9594-8:2020, 3.5.10]

**3.1.4**
**authorizer**
entity trusted by one or more entities operating as AVL entities to create, maintain and sign authorization and validation lists

[SOURCE: ISO/IEC 9594-8:2020, 3.5.11]

**3.1.5**
**certification path**
ordered list of one or more public-key certificates, starting with a public-key certificate signed by the trust anchor, and ending with the end-entity public-key certificate to be validated

Note 1 to entry: All intermediate public-key certificates, if any, are CA certificates in which the subject of the preceding public-key certificate is the issuer of the following public-key certificate.

[SOURCE: ISO/IEC 9594-8:2020, 3.5.21]

**3.1.6**
**certification request**
request issued when a new public-key certificate or renewal of a public-key certificate is required

[SOURCE: IETF RFC 2986]

**3.1.7**
**controllership**
intersection of legal ownership, physical control, and logical control over a device or system, in which the nature of any contractual agreements between ownership and control of the device or system is not important in the context

**3.1.8**
**cryptographic binding**
use of one or more cryptographic techniques by a CKMS to establish a trusted association between a key and selected metadata elements

[SOURCE: NIST SP 800-130]

**3.1.9**
**dataset**
collection of data

**3.1.10**
**device**
component implementing specific functionalities, which may act as a client (e.g., TLS client) or a server (e.g., key distribution centre for GDOI)

**3.1.11**
**digital signature**
result of a cryptographic transformation of data that, when properly implemented, provides a mechanism for verifying origin authentication, data integrity, and signatory non-repudiation

[SOURCE: FIPS 186]

**3.1.12**
**end entity (PKI)**
entity that has been assigned an end-entity public-key certificate, where the private key cannot be used to sign other public-key certificates, but may be used for signing for other purposes

**3.1.13**
**entity**
generic term that covers human users, automation systems, software applications, communication nodes, field devices, and other types of assets

**3.1.14**
**fingerprint**
hash result ("key fingerprint") used to authenticate a public key or other data

[SOURCE: IETF RFC 4949]

**3.1.15**
**group controller/key server**
GCKS
device that defines group policy and distributes keys for that policy

[SOURCE: IETF RFC 3740]

**3.1.16**
**group domain of interpretation**
GDOI
domain that manages group security associations, which are used by IPsec and potentially other data security protocols

Note 1 to entry: These security associations protect one or more key-encrypting keys (KEK), traffic-encrypting keys (TEK), or data shared by group members. GDOI uses the notion of a group controller, which is used to support the establishment of security associations between members of a group.

[SOURCE: IETF RFC 6407]

**3.1.17**
**group member**
GM
authorized member of a secure group, sending and/or receiving IP packets related to the group

**3.1.18**
**hash function**
(mathematical) function which maps data of arbitrary size into data of a fixed size called a digest

**3.1.19**
**hash message authentication code**
HMAC
cryptographic code used for authentication with symmetric keys and for data integrity

[SOURCE: IETF RFC 2104]

**3.1.20**
**key distribution centre**
KDC
centre which, in an IEC 62351-9 context, provides a network service that supplies temporary (symmetrical) session keys to predefined set of peers after successful authentication

Note 1 to entry: This is also known as Group Controller/Key Server (GCKS) (See GDOI).

**3.1.21**
**key management system**
system for the management (e.g., generation, distribution, storage, backup, archive, recovery, use, revocation, and destruction) of cryptographic keys and their metadata

**3.1.22**
**message authentication code**
MAC
cryptographic checksum on data that uses a symmetric key to detect both accidental and intentional modification of the data

[SOURCE: SP 800-63; FIPS 201]

**3.1.23**
**object identifier**
ordered list of primary integer values from the root of the international object identifier tree to a node, which unambiguously identifies that node

[SOURCE: ISO/IEC 9834-1:2012, 3.5.11]

**3.1.24**
**online certificate status protocol**
OCSP
protocol that enables applications to determine the (revocation) state of an identified certificate

Note 1 to entry: OCSP may be used to satisfy some of the operational requirements of providing more timely revocation information than is possible with CRLs and may be used to obtain additional status information. An OCSP client issues a status request to an OCSP responder and suspends acceptance of the certificate in question until the responder provides a response.

[SOURCE: IETF RFC 6960]

**3.1.25**
**operator**
particular type of user that is usually responsible for the correct operation of the equipment under control

**3.1.26**
**pre-shared key**
PSK
secret which is shared in advanced between the two entities, such as software applications or devices, to be able to authenticate themselves or to establish a secure connection

**3.1.27**
**private key**
(in a public-key cryptosystem) that key of an entity's key pair which is known only by that entity

[SOURCE: ISO/IEC 9594-8:2020, 3.5.50]

**3.1.28**
**public-key certificate**
public key of an entity, together with some other information, rendered unforgeable by digital signature with the private key of the CA which issued it

Note 1 to entry: A public-key certificate is often called an X.509 certificate or a digital certificate. However, such terms are ambiguous, as they could also mean attribute certificates, which are also defined by ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019).

[SOURCE: ISO/IEC 9594-8:2020, 3.5.58, modified – Note 1 to entry added]

**3.1.29**
**public-key cryptography standards**
PKCS
specifications produced by RSA Laboratories in cooperation with secure systems developers worldwide for the purpose of accelerating the deployment of public-key cryptography

[SOURCE: www.rsa.com]

**3.1.30**
**random number generation**
RNG
process used to generate an unpredictable series of numbers

Note 1 to entry: Each individual value is called random if each of the values in the total population of values has an equal probability of being selected.

[SOURCE: NIST SP 800-57]

**3.1.31**
**registration authority**
RA
those aspects of the responsibilities of a certification authority that are related to identification and authentication of the subject of a public-key certificate to be issued by that certification authority

Note 1 to entry: An RA may either be a separate entity or be an integrated part of the certification authority.

Note 2 to entry: This definition is different in scope from the one defined in Rec. ITU-T X.660 | ISO/IEC 9834-1.

[SOURCE: ISO/IEC 9594-8:2020, 3.5.61]

**3.1.32**
**relying party**
entity that relies on the data in a public-key certificate in making decisions

[SOURCE: ISO/IEC 9594-8:2020, 3.5.62]

**3.1.33**
**security association**
SA
relationship established between two or more entities to enable them to protect data they exchange

[SOURCE: NIST IR 7298 Rev.1]

**3.1.34**
**security strength**
ability of the security technologies to make it infeasible for a would-be attacker to bypass or subvert

Note 1 to entry: This is often measured in bits of security.

[SOURCE: NIST SP 800-130]

**3.1.35**
**session key**
in the context of symmetric encryption, key that is temporary or is used for a relatively short period of time

[SOURCE: IETF RFC 2828]

**3.1.36**
**symmetric key**
cryptographic key that is used to perform both the cryptographic operation and its inverse, for example to encrypt and decrypt, or to create a message authentication code and to verify the code

[SOURCE: NIST SP 800-63]

**3.1.37**
**trust**
firm belief in the reliability and truth of information or in the ability and disposition of an entity to act appropriately, within a specified context

**3.1.38**
**trust anchor**
entity that is trusted by a relying party and used for validating public-key certificates in certification paths

[SOURCE: ISO/IEC 9594-8:2020, 3.5.72]

**3.1.39**
**trust anchor information**
at least the: distinguished name of the trust anchor, associated public key, algorithm identifier, public key parameters (if applicable), and any constraints on its use including a validity period

Note 1 to entry: The trust anchor information may be provided as a self-signed CA-certificate or as a normal CA-certificate (i.e., cross-certificate).

[SOURCE: ISO/IEC 9594-8:2020, 3.5.73]

**3.1.40**
**trust anchor store**
trust anchor information collection at a relying party for one or more trust anchors

[SOURCE: ISO/IEC 9594-8:2020, 3.5.74]

### 3.2 Abbreviations and acronyms

Additional abbreviations and acronyms are given in IEC TS 62351-2.

| | |
|---|---|
| AA | Attribute Authority |
| AEAD | Authenticated Encryption with Associated Data |
| ASN.1 | Abstract Syntax Notation One |
| AVL | Authorization and Validation List |
| AVMP | Authorization and Validation Management Protocol |
| BRSKI | Bootstrapping of Remote Secure Key Infrastructures |
| CA | Certification Authority |
| CASP | Certification Authority Subscription Protocol |
| CIA | Confidentiality, Integrity, and Availability |
| CMC | Certificate Management over CMS |
| CMS | Cryptographic Message Syntax |
| CPS | Certificate Practice Statement |
| CSR | Certificate Signing Request (also referred in the context of a certification request) |
| DER | Distinguished Encoding Rules |

| | |
|---|---|
| DOI | Domain of Interest |
| DSA | Digital Signature Algorithm |
| ECDSA | Elliptic Curve Digital Signature Algorithm |
| EdDSA | Edwards-curve Digital Signature Algorithm |
| EST | Enrolment over Secure Transport |
| FQDN | Fully Qualified Domain Name |
| GCKS | Group Controller/Key Server |
| GAP | Group Associated Policy |
| GCM | Galois Counter Mode |
| GDOI | Group Domain of Interpretation |
| GKDC | Group KDC |
| GM | Group Member |
| GMAC | Galois Message Authentication Code |
| GOOSE | Generic Object-Oriented Substation Event |
| HMAC | Hash-based Message Authentication Code |
| HTTP | Hypertext Transfer Protocol |
| ICV | Integrity Check Value |
| IDevID | Initial Device Identifier (IEEE 802.1AR) |
| IED | Intelligent Electronic Device |
| ID | Identity |
| IKE | Internet Key Exchange |
| ISAKMP | Internet Security Association and Key Management Protocol |
| KD | Key Download |
| KDC | Key Distribution Centre, aka GKDC |
| KDF | Key Derivation Function |
| KEK | Key Encryption Key |
| LDevID | Locally significant Device Identifier (IEEE 802.1AR) |
| MUD | Manufacturer Usage Description (RFC 8520) |
| OCSP | Online Certificate Status Protocol |
| OTP | One Time Password |
| PDU | Protocol Data Unit |
| PEM | Privacy-enhanced Electronic Mail |
| PKCS | Public-Key Cryptography Standard |
| PKI | Public-Key Infrastructure |
| RA | Registration Authority |
| RNG | Random Number Generation |
| RSA | Rivest Shamir Adleman, public-key crypto system |
| SA | Security Association |
| SCEP | Simple Certificate Enrolment Protocol |
| SPI | Security Parameter Index |
| SV | Sampled Values |
| TAMP | Trust Anchor Management Protocol |
| TEK | Traffic Encryption Key |

# 4 Security concepts applicable to power systems

## 4.1 General

This clause is intended as an introduction to cryptographic concepts, which are applicable in power systems. It provides an overview about security objectives in power systems, which cryptographic algorithms may be utilized to meet these security objectives, and how they may be applied in different deployment environments. Moreover, this clause together with Annex B provides some background information on utilized cryptographic algorithms as well as potential boundary conditions in different environments. The normative part of this document will specify their application. Also addressed is the general handling of security events.

## 4.2 Security objectives

### 4.2.1 Confidentiality

Confidentiality is needed to prevent the access to sensitive data, particularly when it is in transit.

The cryptographic goal of data confidentiality in power system implementations is typically accomplished by encrypting the message using symmetric key cryptography. The message may be encrypted individually at the application level, at the underlying communications channel, or possibly both. As stated previously, the confidentiality of this message is dependent upon the sender and receiver maintaining the secrecy of the symmetric key. Additionally, asymmetric cryptography is often used to exchange or negotiate symmetric session keys that are valid for a specified length of time, thus reducing the risk associated with keeping a specific session key secret.

To limit the damage if a symmetric key is compromised, the symmetric key used on a connection or association is recommended to be unique. This can be achieved by using an ephemeral key negotiation as described in B.8. For TLS this can be done by the selection of appropriate cipher suites (as outlined in IEC 62351-3).

### 4.2.2 Data integrity

Data integrity concerns the detection of unauthorized changes of data during transit or at rest. It enables the verifier of the integrity check value to detect any integrity violations to the related data.

Two methods are used for detecting data integrity violation. Both methods make use of cryptographic hash functions to detect any changes in received data. As discussed in Clause B.5, a hash function results in the production of a fixed length digest over some input data. This digest changes unpredictably even for a small change in the data to be hashed. The two methods used for detecting integrity violation are:

a) Use of digital signatures as discussed in B.3.6.

b) Use of integrity check values (ICVs) as discussed in Clause B.6.

Authenticated encryption is a further variant, which provides data confidentiality and data integrity in a single operation. It is described in Clause B.7.

### 4.2.3 Authentication

When an entity receives data from some other entity, it is critical that the sender of that data is securely authenticated. This document identifies the same two methods for authentication as mentioned for data integrity in 4.2.2:

a) Use of digital signature, where a verified digital signature proves with some certainty that the sender is in the possession of the private key corresponding to the public key used for verifying the digital signature. This is further discussed in B.3.6.

b) Use of integrity check value (ICV), where a verified ICV proves with some certainty that the sender is in possession of the same symmetric key as the recipient. The symmetric key used for generation of a ICV is established as part of the authentication process and thereby bound to that authentication. ICV is further discussed in Clause B.6.

#### 4.2.4 Non-repudiation

Non-repudiation refers to the ability to bind an action (e.g., a command or a message) irrefutably to an issuing entity. It may bind a sender to the action of sending a specific message or a receiver to the action of receiving a specific message.

The ISO/IEC 13888 series specifies different levels of non-repudiation and it specifies two distinct ways of how to establish non-repudiation:

a) ISO/IEC 13888-2 specifies procedures using symmetric-key cryptography. These techniques require the involvement of a trusted third party.

b) ISO/IEC 13888-3 specifies procedures for using asymmetric-key cryptography.

### 4.3 Cryptographic algorithms and concepts

The scope of this document is the key management for power system application comprising PKI based management of asymmetric key material as well as group based key management for symmetric keys. Key management is applicable to all parts of the IEC 62351 series which utilize cryptographic key material to protect information exchanged in power system automation protocols.

Annex B provides detailed explanations of different types of cryptographic algorithms and related concepts, such as trust, trust anchor, security level, random number generation, and migration from one cryptographic algorithm to a different (typically stronger) one. The latter becomes more important to address advances in quantum computing, which specifically endangers asymmetric cryptographic algorithms.

In addition to a general introduction to cryptographic algorithms, the following types of algorithms are considered to a certain extend:

- Asymmetric algorithms comprising RSA, ECDSA, and EdDSA algorithms, including general description, key generation, and security considerations for the different types of asymmetric algorithms.
- Symmetric key algorithms in different operational modes. Only algorithms based on advance encryption standard (AES) are considered. The operational modes considered are cipher block chaining (AES-CBC) and a mode based on a counter mechanism (AES-CTR). They are both defined for key sizes of 128 and 256 bits.
- Hash algorithms. The algorithms considered are so-called secure hash algorithms (SHAs) and only SHA-256 and SHA-512 are considered. The description includes a list of areas where hash algorithms are used.
- Algorithms to create Integrity Check Values (ICVs). Two types of ICV algorithms are relevant for this document. The first type is keyed-hash message authentication code (HMAC) algorithms which specify the use of a symmetric key and a hash algorithm. Two HMAC algorithms are included, one based on SHA-256 and one on SHA-512. The other type of ICV algorithm is the advance encryption standard – Galois message authentication code (AES-GMAC).
- Authenticated encryption with associated data (AEAD) algorithms such as AES-GCM or AES-CCM, are cryptographic algorithms that in addition to specifying encryption/decryption also provide integrity capabilities for encrypted data and also for some associated data. Two types of AEAD algorithms are included. They are both based on AES for encryption but use different techniques for generating the ICV part. They are both provided for AES key sizes of 128 and 256 bits.

- Key agreement using Diffie Hellman to compute a shared key based on unshared private keys. Generating symmetric keys in flight is essential for key management and the Diffie-Hellman method is a much-used technique for that purpose. Two types of Diffie-Hellman schemes are provided using two different types of mathematics (prime fields and elliptic curves). They are both defined in such a way to support migrations to more secure ones.

As these algorithms and concepts are being applied in power systems, Annex B provides an overview and also background information as well as references to further information. This annex is intended as introduction material to gain a better understanding regarding the application of these algorithms in later normative clauses of this document.

# 5 Key establishment and management techniques

## 5.1 General

This clause provides an overview about techniques for the management of symmetric and asymmetric key material, connected concepts, and necessary infrastructures. For the symmetric key material, the focus is placed on the management of group-based keys. This clause builds the background for the normative description in the following clauses.

## 5.2 Key management lifecycle

### 5.2.1 Key management in the life cycle of a device

Key management is part of the life cycle of devices, typically following the flow shown in Figure 1.

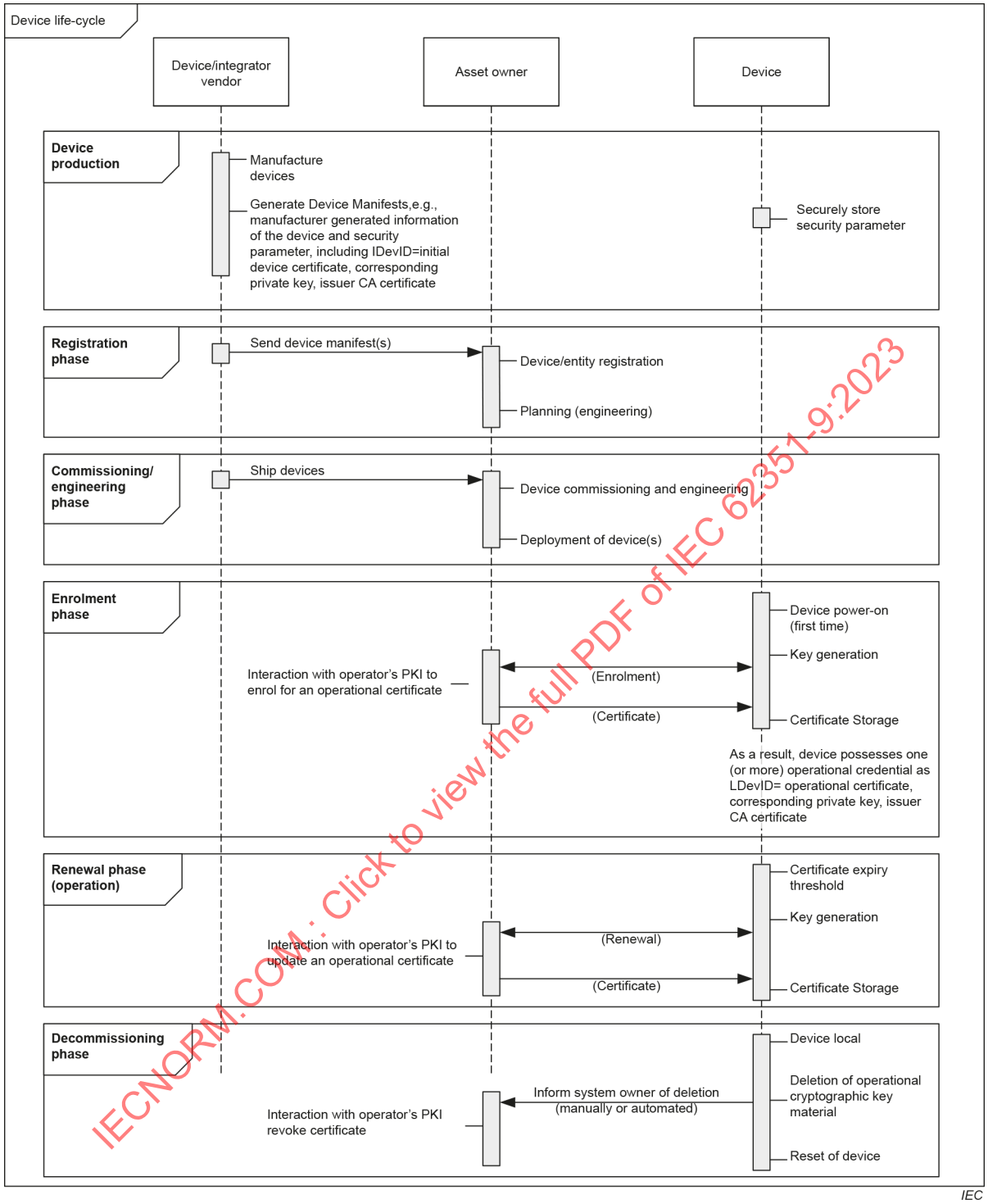





**Figure 1 – Overview key management in the life cycle of an entity**

Manufacturers can support the key management life cycle for an entity by providing a security manifest of identity information, such as a manufacturer unique identifier, or one-time-password, or certificate from a manufacturer CA, or manufacturer's entity certificate for each of their devices and systems. IEEE 802.1AR defines these initial set of credentials as IDevID (Initial Device Identifier), consisting of the initial certificate, the corresponding private key, as well as certificate chain up to the trust anchor. At each change in ownership or even status (e.g., warehoused vs. deployed), new certificates should be enrolled, providing a trusted chain of the product identity in use. When these entities are eventually engineered and deployed, their manufacturer certificates (if available) may be used to enrol the entity in operations, leading to one or more operational certificate(s) (LDevID, Local Device Identifier), which

enables them to authenticate and commence their information exchanges. Shortly before these operational certificates are due to expire, they should be renewed to avoid certificate verification errors.

Also, to be considered is the decommissioning of devices, which should go along with the deletion of any operational cryptographic credentials to ensure that this information cannot be misused once the device it taken out of operation. Note that the IDevID (if available), is typically not deleted.

Manufacturers are encouraged to make the process for securely wiping a device of any operational credentials (certificates, trust anchor database, etc.) and restoring the default initial set of credentials a simple process; or if necessarily complex, at least clearly documented. This would assist both with good end-of-service hygiene and where appropriate, confidence that any workshop/workbench testing credentials have been removed prior to deployment. System owners are involved in the decommission process as part of the lifecycle (see clause 5.2.2).

### 5.2.2 Lifecycle of a cryptographic key

Cryptographic keys in general and certificates specifically follow a life cycle. They are created, distributed, installed, applied, updated, and destroyed to meet the key management security policy requirements. As an example, the typical life cycle of keys found in PKI based systems is depicted in Figure 2. The general key life cycle management is discussed in more detail in ISO/IEC 11770 (cf. [6][3]) and NIST SP 800-130 [11]. The life cycle of a certificate that is typically managed, is depicted here on an abstract level. More details about certificate management can be found in 5.8.

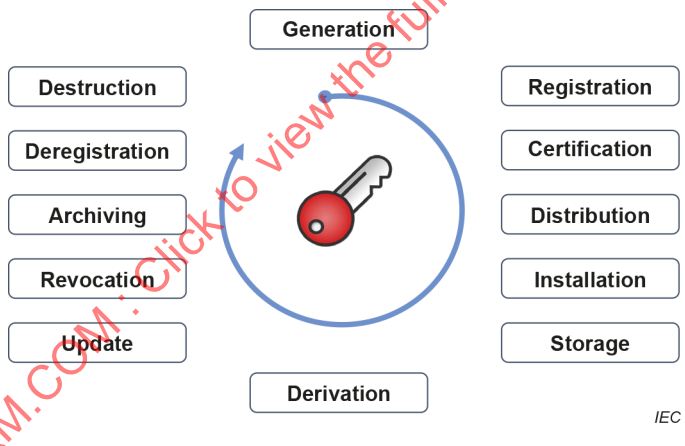



**Figure 2 – Cryptographic key life cycle**



The following list briefly describes each of the possible stages in the life cycle of a cryptographic key. All stages are required for asymmetric keys, while registration, certification, deregistration, and revocation are not required for symmetric keys:

- Generation: Cryptographic keys can be created by the entity itself if it has the appropriate capabilities. Alternatively, keys may be created externally and installed on the target entity. Often this latter approach is used if the entity cannot support one of the critical components in generating keys, namely the random number generator (RNG) which must ensure randomness to a high degree (see B.12). Keys might also need to be created externally if they must be archived (e.g., for encryption keys used for stored data). When certificates are about to expire, entities need to apply for certificate renewal.
- Registration: If certificates and the PKI process are used, registration through a Registration Authority (RA) establishes the "identity" of an entity. This is done by generating the key pair

[3] Numbers in square brackets refer to the bibliography.

either entity local or centrally. The entity provides a certification request to the RA, which is then further processed in conjunction with the Certification Authority (CA).

- Certification: After registration of an entity by an RA, the CA provides a digitally signed certificate to the entity that thus connects its public key with the private key held by the entity. Depending on the key generation, this can be part of the key generation in a trust centre or may be done using information sent in a certification request.
- Distribution: Key distribution comprises the process to transport keys as well as key management information to authorized entities in a secure manner. Private (secret) keys, although ideally generated in end devices and thus not requiring distribution, must be distributed using secure means if externally generated. Public keys, within public-key certificates, can be distributed using non-secure means since end devices can verify their authenticity directly by verifying the certificate signature.
- Installation: A key may be installed in an entity during the manufacturing and/or engineering processes (pre-shared key) or may be installed later via on-line procedures. The latter process may require communication with a security server, out-of-band using a separate communication channel, or in-band as part of a service communication.
- Storage: The private/symmetric keys are recommended to be securely stored in the entity. Secured key storage should be compliant with FIPS 140-3 [70] or ISO/IEC 19790 [71].
- Derivation: Cryptographic keys used as session keys or other short-term keys can be derived from the private or symmetric keys that are stored in the entity.
- Update: Cryptographic keys need to be regularly updated. Cryptographic keys have a dedicated lifetime, e.g., public-key certificates issued to users may have a lifetime of 2 years, while public-key certificates issued to server may be limited to 1 year or shorter, pre-shared keys to a few weeks or months, and session keys to a few hours and/or a few thousand packets. Lifetime recommendations of cryptographic keys can be found in NIST SP 800 -57 Part 1 [73] and German BSI TR 02102-1 [58].
- Revocation: Certificate revocation may occur when a certificate is no longer authorized to be used. This may be the case if a device is removed from service or changes its role, or if a private key is compromised or suspected of being compromised.
- Archiving: Symmetric keys needed to decrypt long-term stored data should be archived in secure confidential storage to enable data recovery (to allow access to store encrypted data later). In addition, to support future signature verification, public-key certificates should also be archived, although secure confidential storage is not needed.
- De-Registration: This procedure is typically provided by the registration authority to remove the association of an entity with a dedicated key. It is typically used in the key destruction phase.
- Destruction: When a cryptographic key is destroyed, it must not be able to be recovered or used. This occurs at the end of the cryptographic key life cycle.



### 5.3 Cryptographic key usages

Cryptographic keys are used for different purposes and in different phases of the product lifecycle and are applied as:

- Public-key certificates and corresponding private keys: Used to identify, authenticate, and authorize entities. Moreover, they are often employed in the negotiation or establishment of session keys.
- Pre-shared keys (e.g., for real-time communication): Used as a shared secret between entities to secure the data exchange (integrity, confidentiality) between them. This may be useful in environments not supporting a PKI. Additionally, this scheme may be used during PKI enrolment, allowing an entity to authenticate itself against the certification authority (CA), for example when processing a certification request. *Note that the security of pre-shared key relates to their complexity.* Password complexity rules are typically given by the organization security policies.
- Session keys (pair wise or group-based): Used for efficient encryption or integrity checks on communication messages.

- Session parameters (dedicated cryptographic algorithms): Used to support sessions (keys, lifetime, etc.).
- Cryptographic access tokens: Mostly used to transfer/provide authorization/access of resources to entities.
- Other relevant entities or organizational credentials.

### 5.4 Key management system security policy

Cryptographic keys need to be protected. Therefore, every organization (or group of organizations that expect to exchange data) should develop a security policy for its key management system that establishes and specifies all of the requirements for protecting the confidentiality, integrity, availability, and source authentication of all cryptographic keys and their associated metadata used by the organization. These protection requirements should cover the entire key life cycle, including when they are initially provided, operational, stored, and transported. A key management security policy should also include the selection of all cryptographic mechanisms and protocols to be used throughout the organization's systems.

Key management systems also need security administrative support to ensure the security policies are followed and the procedures are maintained. The specification of this support is out-of-scope for this document, but can be found in other documents such as ISO/IEC 11770 ([6]) or in NIST SP 800-130 ([11]) or in IEC 62443-4-2 [74].

### 5.5 Key management design principles for power system operations

Cryptographic keys are used to secure the communication between different system entities, such as users, systems, software applications, communication nodes, and the potentially large numbers of equipment and devices that are often located at remote, often untrusted sites, and increasingly owned/operated by different organizations

These cryptographic keys should be managed so that they can effectively and securely be provided to the entities that require secure exchanges of data. This key management needs to consider many issues, ranging from the capabilities of entities to the varied types of locations of these entities, to the timing for providing and revoking keys, to the varied ownership of the entities, to possible different regulatory jurisdictions, and to protecting the key management processes themselves from attacks.

For instance, many smaller devices are limited in computational power and memory capacity, while the communication networks may also be limited in available bandwidth. Therefore, some of the key management techniques used in traditional enterprise information technology system environments with powerful systems and high bandwidth communications are not well suited to power system automation and communication environments.

As another example, in groups of organizations that expect to exchange operational data, some of the organizations may not have the expertise or personnel to manage cryptographic keys without third party support. These realities should therefore be included in the design of key management processes.

To address these constraints, this document specifies different key management techniques that could be used for different requirements and constraints. Specifically, it specifies how to manage keys for each of the cryptographic functions specified in the other IEC 62351 parts. Sources of guidance on key management design principles include:

- Reference [6] ISO/IEC 11770-1:2010 Information technology – Security techniques – Key management – Part 1: Framework
- Reference [8] ISO/IEC 11770-3:2008 Information technology – Security techniques – Key management – Part 3: Mechanisms Using Asymmetric Techniques
- Reference [13] NIST 800-57, Part 1

## 5.6 Establishment of symmetric keys

### 5.6.1 Overview

Symmetric cryptographic keys are required for two reasons:

a) they are used for encryption and decryption of data during transfer; and

b) they are used for generation and verification of ICVs.

For security reasons, the keys for pair wise communication are expected to be different between any two pairs of communicating entities.. Symmetric keys between two entities may be established in different ways:

a) by means defined in other parts of IEC 62351, specifically in:
   - IEC 62351-3 for TLS, by profiling the TLS options. Here, during the TLS handshake between two entities, techniques based on public-key algorithms are used to establish a pairwise key.
   - IEC 62351-4 for the application layer end-to-end security profile. As for TLS, a pairwise key is established between two communicating entities based on public-key algorithms.
   - IEC 62351-5 for the application layer security mechanism for IEC 60870-5-101/104 and IEEE 1815 (DNP3) relying also on the application of public-key algorithms.

b) by use of key agreement for pairwise key establishment, as described in the Diffie-Hellman key agreement method (5.6.2).

c) by use of key derivation function (KDF) method (5.6.3), e.g., based on an already existing symmetric key.

d) by use of key distribution, e.g., using public key encryption as used in certain TLS cipher suites.

Symmetric keys established between two entities following a) to d) are often not directly applied to protect the communication but rather used to derive security service specific keys. In the example of IEC 62351-4 there are distinct symmetric keys available for integrity protection and also for confidentiality protection. Moreover, these keys are direction dependent resulting in multiple keys between two communicating entities. These keys also have a lifecycle requiring that the initial symmetric key is established in a secure way and also that it is managed through the lifetime of the communication connection. As outlined in the bullet list above, the establishment of symmetric keys typically supported by other methods to limit the overhead for managing pure symmetric keys between the communicating entities.

One method that can be considered for accommodating networked entities, which rely on multicast communication is group-based key management. In particular, group-based keys can be utilized in power automation systems. For this, group-based key management protocols are employed which include a key management component for managing keys and associated policies to authenticated members of a group. Group key management is described in 5.6.4.



### 5.6.2 The Diffie-Hellman key agreement method

The Diffie-Hellman key agreement method allows two parties that have to exchange so-called Diffie-Hellman public keys to jointly arrive at a shared secret in a way that does not allow an eavesdropper to learn about this shared secret.

Details on the Diffie Hellman key agreement is provided in Clause B.8 for the classical and the elliptic curve Diffie Hellman.

### 5.6.3 Key derivation function (KDF) method

A key derivation function (KDF) is a cryptographic algorithm that derives one or more symmetric keys from a secret value such as another symmetric key, a password, or a passphrase using a pseudorandom function.

Details on KDF are provided in B.9.

### 5.6.4 Group key management

#### 5.6.4.1 Purpose of group keys

For multicast interactions between entities forming a group that have stringent performance requirements, the application of a group key is more efficient than having separate peer-to-peer relationships in that group. Also, under such circumstances, the key management is likely to be performed outside the context of the actual protocol applying the keys. Group key management typically uses a combination of asymmetric and symmetric cryptography. The asymmetric part is used to identify and authenticate the entities, while the symmetric part protects the actual key and policy transport.

To realize the setup of a group-based key, one system or device typically is designated as the group controller, which in turn authenticates other entities via their certificates or pre-shared keys. After successful authentication of the entities, the group controller distributes the actual group key to them. Hence, the group controller resembles the functionality of a Key Distribution Centre (KDC) (see Figure 3).

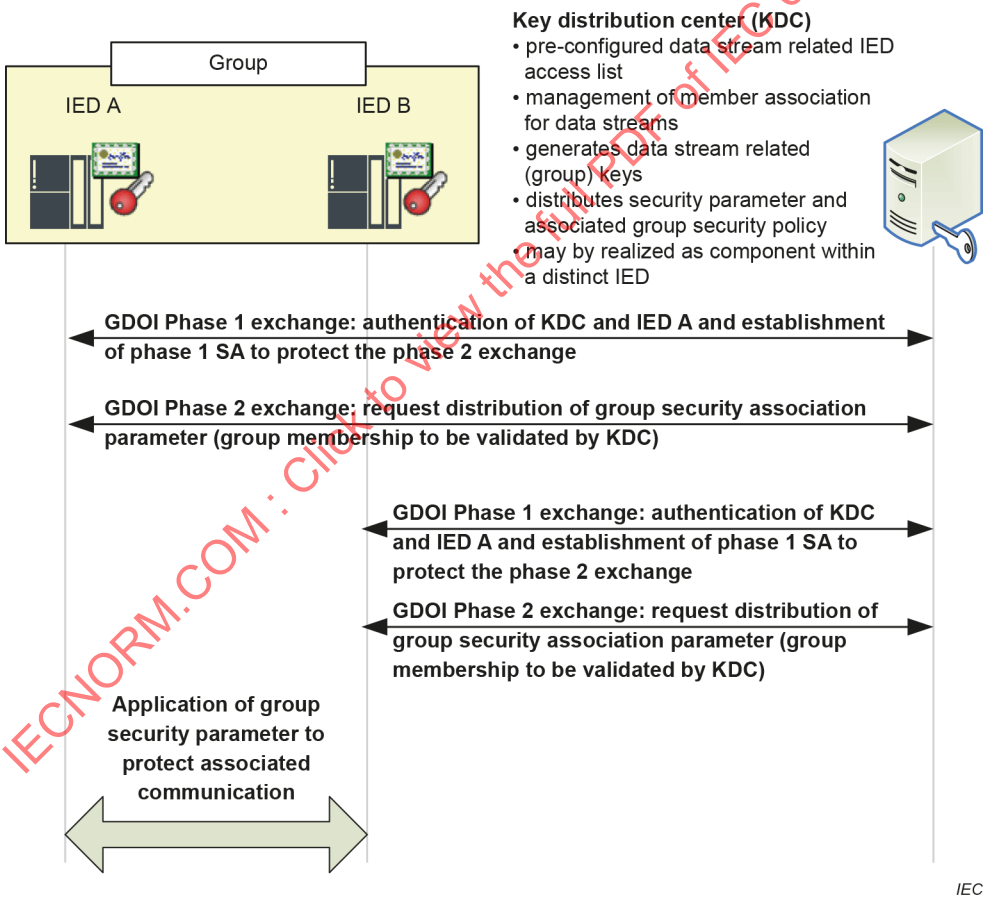



**Figure 3 – Overview of group key management on the example of GDOI**

Figure 3 shows the subscription of an Intelligent Electronic Device (IED) for the security parameters of a data stream. The data stream is associated with a communication group. Note that the group controller may be a separate entity or may be deployed within any IED. Note also that the version shown uses a certificate-based authentication, employing digital signatures.

There exist a number of different protocols for distributing group keys. However, Group Domain of Interpretation (GDOI) has been selected as the most appropriate protocol for power system automation and is further described in5.6.4.2.

#### 5.6.4.2 Group Domain of Interpretation (GDOI)

##### 5.6.4.2.1 General

The Group Domain Of Interpretation (GDOI) defined in RFC 6407 supports the distribution of symmetric group keys (i.e., Traffic Encryption Key, TEK) to all pre-configured or otherwise enrolled entities, typically devices. This method requires a Key Distribution Centre (KDC) that is responsible for distributing symmetric session key sets to enrolled entities following the GDOI process. This process uses point-to-point communications between the KDC and each group member (GM) to distribute the symmetric group keys and the associated security policy. The group key itself is then applied according to the provided security policy to protect subsequent communications within the group.

Note that a KDC failure will disrupt group communication, so KDC redundancy is imperative. However, the method for achieving KDC redundancy is outside the scope of this document.

GDOI is specified to proceed in two phases, which are described in more detail in the following sub clauses.

##### 5.6.4.2.2 GDOI Phase 1: Internet Key Exchange (IKE) Phase 1

The GDOI Phase 1 security association provides mutual authentication and authorization. The result is used by the protocol participants to execute a secured GDOI Phase 2 exchange. The GDOI RFC incorporates (i.e., uses but does not redefine) the IKEv1 Phase 1 security association definition from the Internet DOI [RFC2407], [RFC2409] as shown in Figure 4. The symmetric key transfer from the KDC to the entities is initiated after mutual authentication.

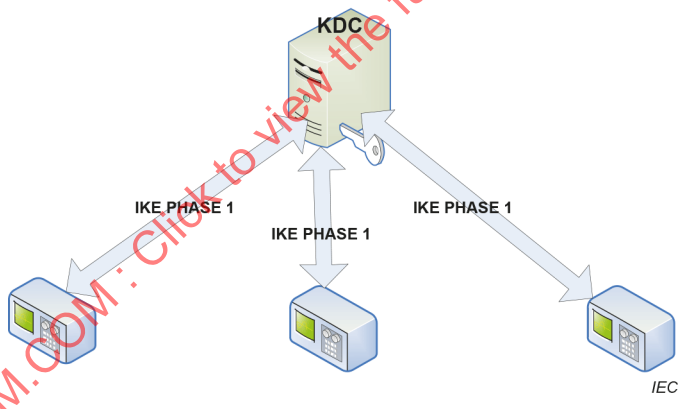



**Figure 4 – GDOI IKE Phase 1 – Authentication and securing communication channel**

NOTE Although RFCs 2407, 2408, and 2409 were obsoleted by RFC 4306 (and subsequently [RFC 5996]), they are used by GDOI because the protocol definitions remain relevant for ISAKMP protocols other than IKEv2.

##### 5.6.4.2.3 GDOI Phase 2: GDOI symmetric key distribution

###### 5.6.4.2.3.1 General

In the GDOI process (RFC 6407), two methods are defined on how to transfer the keys from the KDC into the entities. The first method is the PULL Method (keys are pulled by the entities from the KDC) and the second is the PUSH Method (the keys are pushed from the KDC into the entities) model. The GDOI phase 2 exchange is protected with the SA established in the GDOI phase 1.

NOTE The PUSH Method requires that the group members are reachable from the KDC directly. This has to be obeyed when designing the system architecture as firewalls or NAT devices may hinder this communication.

Either a GROUPKEY-PULL exchange or GROUPKEY-PUSH exchange may be used to distribute symmetric keys to the entities. GROUPKEY-PULL is required to be supported in

GDOI, as it is the only method that works in conjunction with the Phase 1 authentication and authorization. There are advantages associated with each of the two methods. GROUPKEY-PULL offers control over traffic delivery while GROUPKEY-PUSH allows for a timely revocation or eviction of an entity and better efficiency.

GDOI has been extended to support IEC 62351 Security Services within RFC 8052. These are described in 5.6.4.2.4.

#### 5.6.4.2.3.2 GROUPKEY-PULL registration protocol exchange

The GDOI Pull Method is illustrated in Figure 5.

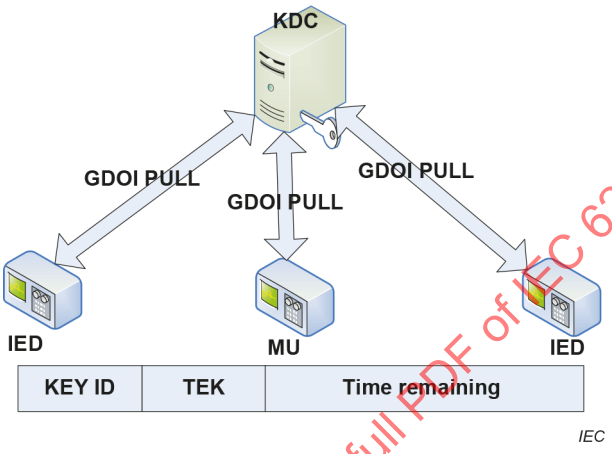



**Figure 5 – GDOI Pull Phase 2**

There are two phases of communication in the GDOI GROUPKEY-PULL method:

- GDOI Phase 1: Connection establishment and authentication of the two participating entities, a group member and KDC as described in 5.6.4.2.2.
- GDOI Phase 2: Determining the policies in use for the requesting group and downloading the keys: This is accomplished through a Group Member (GM) making a GDOI Identification Payload Request to the KDC. The KDC responds to this request with the policies (e.g., encryption and signature algorithms) that are supported. The policies are returned using GDOI SA, which returns a SA TEK payload. If the GM does not support the policies, the communications should abort. If the GM can support the policies, it issues an acknowledgement and the KDC replies with the Key Download (KD) payload.

#### 5.6.4.2.3.3 GROUPKEY-PUSH rekey protocol exchange

The KDC policy may include the use of the "push" method for rekeying, in which case a datagram initiated ("pushed") by the KDC is delivered to all group members using an IP multicast address or is sent to a specific GM using IP unicast. The GROUPKEY-PUSH method is useful for evicting group members that may have been compromised and may have had their certificates revoked. Using this method, the KDC can rekey all authorized group members while excluding the group member that is being evicted.

In case of using GROUPKEY-PUSH, RFC 8263 defines the provisioning of an acknowledgement message from the group members to inform the KDC about the reception of the pushed information. The request to provide the acknowledgement message is indicated by the KDC in the associated policy send to the group members.

#### 5.6.4.2.4 GDOI protocol support for IEC 62351 security services

RFC 8052 "GDOI Protocol Support for IEC 62351 Security Services" was developed to address the use of GDOI for the IEC 61850 power utility automation family of standards. It describes the

methods to be used to distribute security transforms that are used for the protection of messages for some IEC 61850-related security protocols: the protection of the messages is defined within IEC 62351-6 [IEC-62351-6], IEC 61850-8-1 [IEC-61850-8-1], and IEC 61850-9-2 [IEC-61850-9-2]. Protected IEC 61850 messages typically include the output of a Message Authentication Code (MAC) and may be encrypted using a symmetric cipher such as the Advanced Encryption Standard (AES).

## 5.7 Trust supported by public-key infrastructures (PKI) and privilege management infrastructures (PMI)

### 5.7.1 General

This clause provides an overview about the PKI infrastructure utilized to manage certificates through the lifecycle of the power system entities (see Figure 6). Note that entities may be devices, systems, but also human users in a power system.

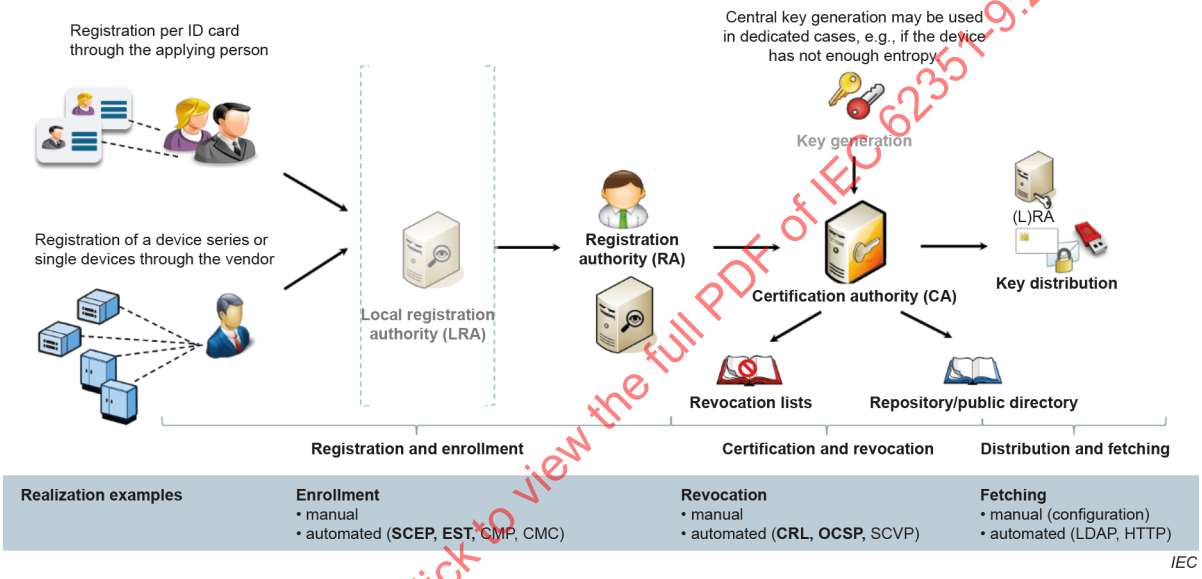



**Figure 6 – Overview of PKI infrastructure and realization examples**

Subclauses 5.7.2 to 5.7.6 describe the general functionality of the registration authority and the certification authority as well as the certificates as managed objects.

### 5.7.2 Registration authorities (RA)

In PKI-based systems, entities are required first to prove their identity through a Registration Authority (RA). For humans, RAs may determine an individual identity through birth certificates, passports, or other official documents. For systems and devices, RAs may determine an entity's identity through a manufacturer-issued digital certificate or a unique one-time-password (OTP).

The RA may either be a separate entity, or it may be part of certification authority (CA).

### 5.7.3 Certification authority (CA)

After its identity is established, a system or device needs to be bound to a certificate by a certification authority (CA). The CSR containing the public key of the generated key pair is either generated by the entity itself or by the manufacturer if the entity is not capable of key generation, e.g., due to missing entropy.. The CA verifies the CSR and issues a digital certificate based on the data included in the CSR. With the certificate, CAs establish a trusted cryptographic binding between the entity's public key and the public-key certificate component data (i.e. the CA's digital signature computed on the combination of the public key and metadata). This public-key certificate thus binds the entity's identity with its public key, so that there is now a trusted public/private key combination that can be used by the entity to exchange

information with other entities. The trust in the entity's key thus relies on one's trust in the validity of the CA's key (see 5.7.4).

CAs may be operated either by an organization itself, allowing for a closed, organization-controlled communication group, or by a third party (service provider, system operator or grid manager) that is publicly accepted and hence has a wider scope of trust. Third-party CAs require a secure method for accepting the identity of any new entity, which can range from in-person validation for humans to business-entity validation and to security chains of public-key certificates linking previously validated public-key certificates to new public-key certificates.

### 5.7.4 Public-key certificates

#### 5.7.4.1 General

A public-key certificate is a digital document that cryptographically binds the identity of an entity to a public-key of that entity. The public key has a corresponding private key. As noted in 5.7.3, this binding is verified by a digital signature by the issuing CA. In addition to the public key and identity of the owner of a public-key certificate, the public-key certificate holds verified information about validity period and the identity of the issuer. A public-key certificate may include extensions providing additional information. An extension is identified by an object identifier allocated by the organization defining the extension. A public-key certificate may be issued to a CA and is then called a CA certificate or to an end entity and is then called an end-entity public-key certificate.

In a PKI environment, public-key certificates may be digitally signed by the CA of the organization to which the entity belongs. In some cases, multiple public-key certificates are created as an entity goes through a supply chain from manufacturer, to distributer, to purchaser, to installer, etc. Root certificates of the issuing CA become the trust anchors, if used. Typically, organizations should install only CA certificates from CAs they trust, including public-key certificates issued by their own CA.

Public-key certificates may be assigned to entities like devices or human users or software applications.

Public-key certificates are defined by a base set plus extensions to the base set. Extensions are identified by object identifiers.

#### 5.7.4.2 Short-lived public-key certificates

Short-lived certificates (public-key certificates or attribute certificates) may not need to be revoked, as the possibility for compromise is quite low. Short-lived public-key certificates are probably not very useful for the power industry, as they may imply additional overhead for issuing and distributing new public-key certificates, but there may be special circumstances where they could be used.

#### 5.7.4.3 Public-key certificate renewal

When certificates are ready to expire, entities need to apply for certificate renewal using a similar process to enrolment, but simpler since the certificate update can already utilize the available certificate information.

Typically, existing enrolment protocols allow the re-enrolment utilizing the previously issued certificate. The specific process implemented depends on the security policy of the operator.

#### 5.7.4.4 Alternative process for asymmetric keys generated outside the entity

Asymmetric key pairs may be generated externally if the entity lacks good random number generation capabilities or the necessary computational power. In that case, the generation process may be delegated to a PKI compliant component, such as a PKI tool. Distribution of the keys would need to be done out-of-band using a trusted procedure. The private key may be protected with a transport key as in PEM, PKCS #8 or usually PKCS #12 which may include other objects, such as CA or root certificates. Figure 7 shows one possible option to realize centralized key and certificate generation. The distribution of the key material may be done by using a PKI tool locally.

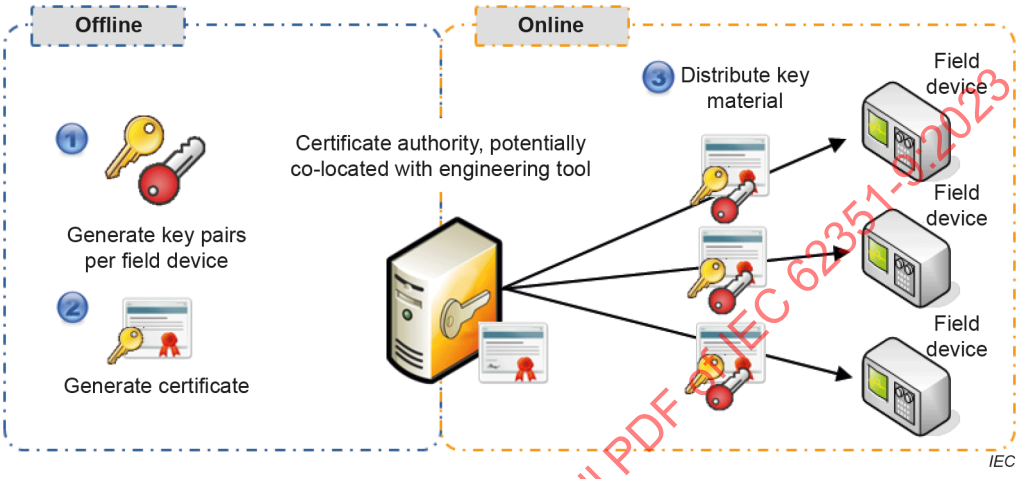



**Figure 7 – Central certificate generation**

If a manufacturer-based credential is already available in the field entity and is trusted by the infrastructure (issuing CA in the new deployment domain), this credential may be used to identify the field entity and to secure the key distribution operation. This requires that the credential be already known at the CA.

### 5.7.5 Attribute certificates

Attribute certificates provide an effective way to separate the management of identity from the management of authorizations associated with an identity. Full separation can be achieved by managing public-key certificates with a PKI and attribute certificates with a "privilege management infrastructure" (which uses the same CA technology as a PKI).

As shown in Figure 8, attribute certificates can be used to extend the information in a public-key certificate. They allow for instance for temporary enhancement of the permissions of the public-key certificate holder by specific role-based access information. This approach has been leveraged in IEC 62351-8.

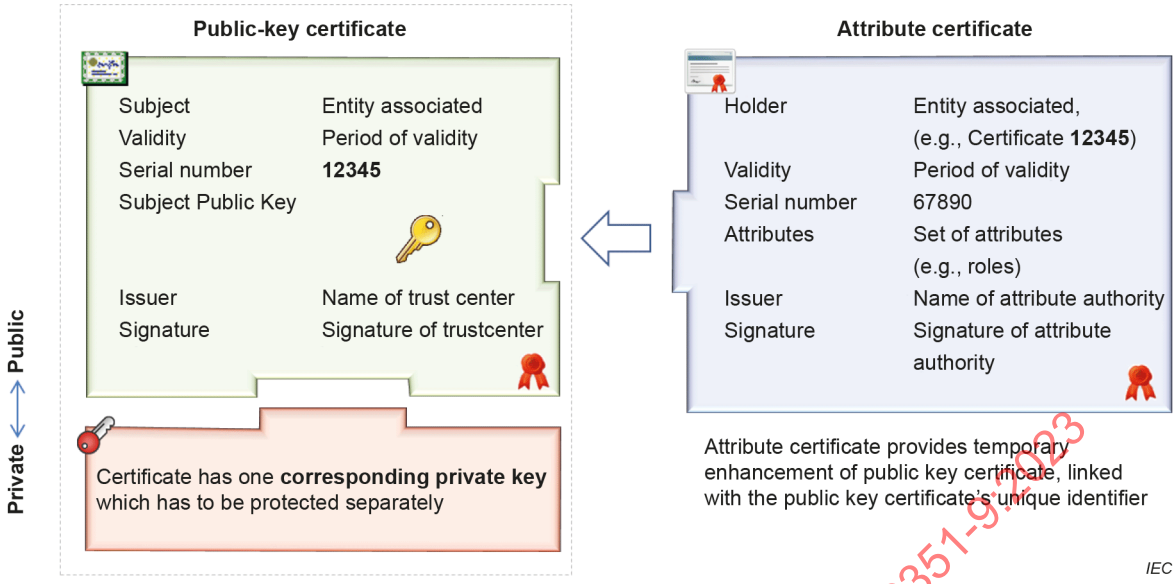



**Figure 8 – Relationship between public-key certificates and attribute certificates**

Attribute certificates may be assigned to entities like devices or human users or software applications.

Attribute certificates are defined by a base set plus extensions to the base set. Extensions are identified by object identifiers.

Attribute certificates can be seen as another approach to achieve a short-lived or temporal extension to a public-key certificate (see also 5.7.4.2). Short-lived attribute certificates can be applied to temporarily extend the permissions of an entity as defined in IEC 62351-8, e.g., for RBAC in general and in emergency cases. Such an attribute certificate should include the no revocation information extension defined in ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019).

### 5.7.6 Public-key certificate and attribute certificate extensions

Both public-key certificates and attribute certificates allow extensions to be optionally included. Each such extension provides additional information.

An extension type consists of an object identifier (see 7.6) that identifies the type of extension and specifies the syntax for the associated information. In addition, it includes a Boolean that specifies whether a particular extension is flagged as critical or as non-critical. If an extension is flagged as critical, it cannot be ignored, and the public-key or attribute certificate is considered invalid if the relying party does not support the type of extension. If the extension is flagged as non-critical, the relying party may ignore the extension if the extension type is not supported. For more details, see 7.3 of ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019).

ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019) specifies several general applicable extensions. IEC 62351-8 specifies the `IECUserRoles` extension, which has been defined for power systems to provide means for role-based access control.

## 5.8 Certificate management of public-key certificates

### 5.8.1 Certificate management process

The certificate management process may be repeated upon certificate creation, each change of ownership or use of the entity, and when the certificate is about to expire.

Beyond the utilized mechanisms and protocols used for certificate management, which are detailed in the following sub clauses, two important properties are to be obeyed in the certificate management process with respect to the requester:

- Proof of identity, to ensure that the right entity is considered in the certificate management. This can be done by using different means, such as device-related information together with an activation code (OTP) or an already available certificate and corresponding private key. Specifically, the latter allows for a higher degree of automation of the process.
- Proof of possession of the corresponding private key, to ensure that the provider of the public key also owns the private key. This is typically done by digitally signing the request information (e.g., a PKCS #10 certification request), which can be verified by the central registration authority.

To request a certificate a combination of both, proof of identity and proof of possession is required, to ensure that for the intended entity a public-key certificate will be issued in an authorized way.

### 5.8.2 Initial certificate creation

Just as a birth certificate is the proof of the identity of a person and the physical presence of that person at a registration office is required for personal identification, a device or system needs to have proof of its identity, usually when it is still on the manufacturer's premises and when received by the new owner. This is achieved by registering it with a RA and issuing a certificate by the CA of the manufacturer. This certificate can be verified against the trust anchor of that manufacturer, which therefore needs to be publicly available. Registration could be manual for individual entities or could be handled in bulk for large numbers of entities. The registration process then acts as the source of the supply chain trust or provenance of the entity.

In the context of IEEE 802.1AR, the initial credential is called IDevID for initial device identity. It is a triplet, which consists of the initial device certificate, the corresponding private key, and the certificate chain up to the trust anchor of the issuer (here the manufacturer).

### 5.8.3 Onboarding of an entity

Introducing an entity into a network requires a mutual trust establishment between the entity and the operational domain, typically resulting in the device possessing the trust anchor of the operational domain and also their own certificate and corresponding private key to authenticate other components of the same domain. Here, a similar triplet is built as for the initial device credential by having an operational credential. IEEE 802.1AR references this operational credential as LDevID for locally significant device identifier, consisting of the local device certificate, the corresponding private key, and the certificate chain to a root certificate of the local issuer. Note that a device may possess multiple LDevIDs for different purposes.

The onboarding can be a fully manual process by providing the delivery information of the vendor to the domain for further verification on the infrastructure side. Part of this is the configuring of the domain trust information into the device. This provides the necessary boundary condition for the enrolment as the next step. The latter part, establishing the trust from the entity into the operational domain, can also be automated, resulting in a zero-touch onboarding from a device perspective. To support this automation, information objects and interaction protocols defined by the IETF in the context of autonomous networking can be leveraged. Note that the following examples do not constitute a complete list of possible approaches.

RFC 8366 [44] specifies a voucher artefact that can be used to convey information (the domain certificate of the new owner) from the manufacturer to a device in a protected (signed) container. This signature can be validated by the device requiring the device to possess the root certificate of the manufacturer. Here the assumption is that the devices are shipped with an IDevID.

RFC 8572 [76] defines a mechanism to securely provision a networking device when it is booting in a factory-default state (Secure Zero Touch Provisioning, SZTP). It may be used in public or private networks and builds on the voucher as defined in RFC 8366 to facilitate trust establishment and also the enrolment of operational certificates for the domain. After finalization of the bootstrapping, the device is able to establish secure connections with other systems.

RFC 8995 [47] builds on the defined voucher process and defines an architecture and protocol (Bootstrapping of remote secure key infrastructures, BRSKI) to establish mutual trust by providing a new device with the local domain certificate information by involving the manufacturer issuing a voucher including this domain certificate. This voucher can be verified by the device as it possesses the trust anchor from the manufacturer. Once the device possesses the domain certificate, it can enrol in the target domain and receive an LDevID (local device identifier) certificate for the new domain. The LDevID is also a triplet, which consists of the local device certificate, the corresponding private key, and the certificate chain to a trust anchor of the issuer (here the operator of the target domain). Utilizing the bootstrapped information for the network level LDevID the device may be provided with further application specific LDevIDs to establish secure communication connections with other devices.

In addition, RFC 8520 [46] (Manufacturer Usage Description, MUD) defines a further approach of supporting automation when installing new devices by specifying information objects to be used by a manufacturer to provide information about the device itself as well as the communication behaviour / expectations of the device. The latter can be used to adopt the infrastructure to the communication needs according to the local security policy. This information may be used, for instance, to refine access control lists in the infrastructure or communication ports with respect to external services. As a result, MUD enables network segmentation by using fine-granular access control definitions based on the purpose and capabilities of the device.



### 5.8.4 Enrolment of an entity

Enrolment is the process by which the entity receives a signed certificate from the CA in the operational environment. Based on the existing security credentials of the entity, different enrolment scenarios can be distinguished:

- One-time unique password without certificate
- With CA-issued certificate (this may be an IDevID issued by a manufacturer CA)
- With known (authorized) self-signed certificate
- With expired or revoked certificate (this may be an IDevID issued by a manufacturer CA or an LDevID from a local CA)
- With an expired or revoked CA certificate.

There exist different protocols for enrolment and management of certificates that support different sets of functionalities. Four of them are briefly described in 5.8.6. Of these, two are focused on the applications used in the power system domain, which are SCEP (see 5.8.6.1) and EST (see 5.8.6.2). Both protocols address the main functionality seen as necessary to manage certificates from devices in a power system, while keeping simplicity, and both provide a subset of the functionality provided by the rather feature rich protocols CMP (see 5.8.6.3) and CMC (see 5.8.6.4). These latter two enrolment protocols provide better support in offline scenarios and in support of revocation handling.

An example of enrolling an entity using SCEP is shown in Figure 9. For SCEP, the activation code is an OTP (One-Time-Password) as illustrated in Figure 9 or an existing certificate. For EST, the activation code may a username and a password or an existing key pair (e.g., a manufacturer issued (imprinted) certificate with a corresponding private key and issuing CA information, IDevID).

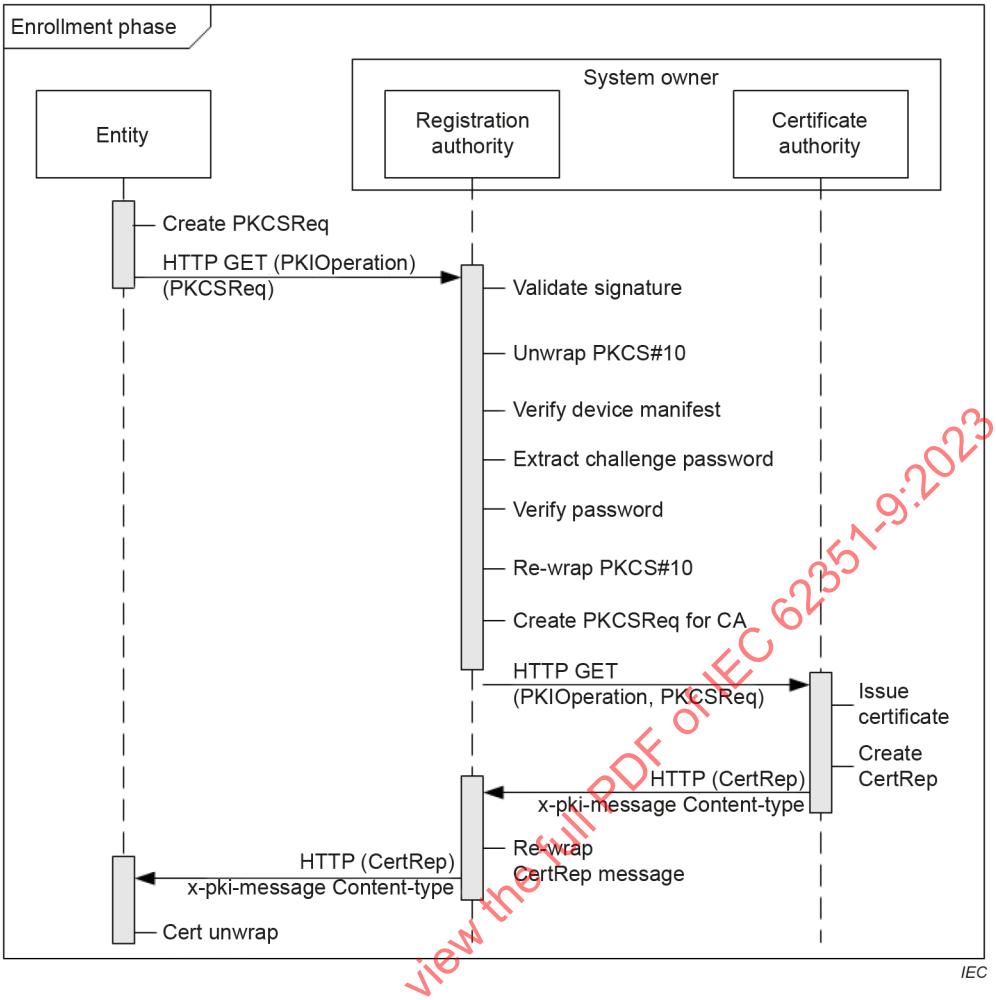



**Figure 9 – Example of the SCEP entity enrolment and CSR process**

An example of enrolling an entity using EST is shown in Figure 10.

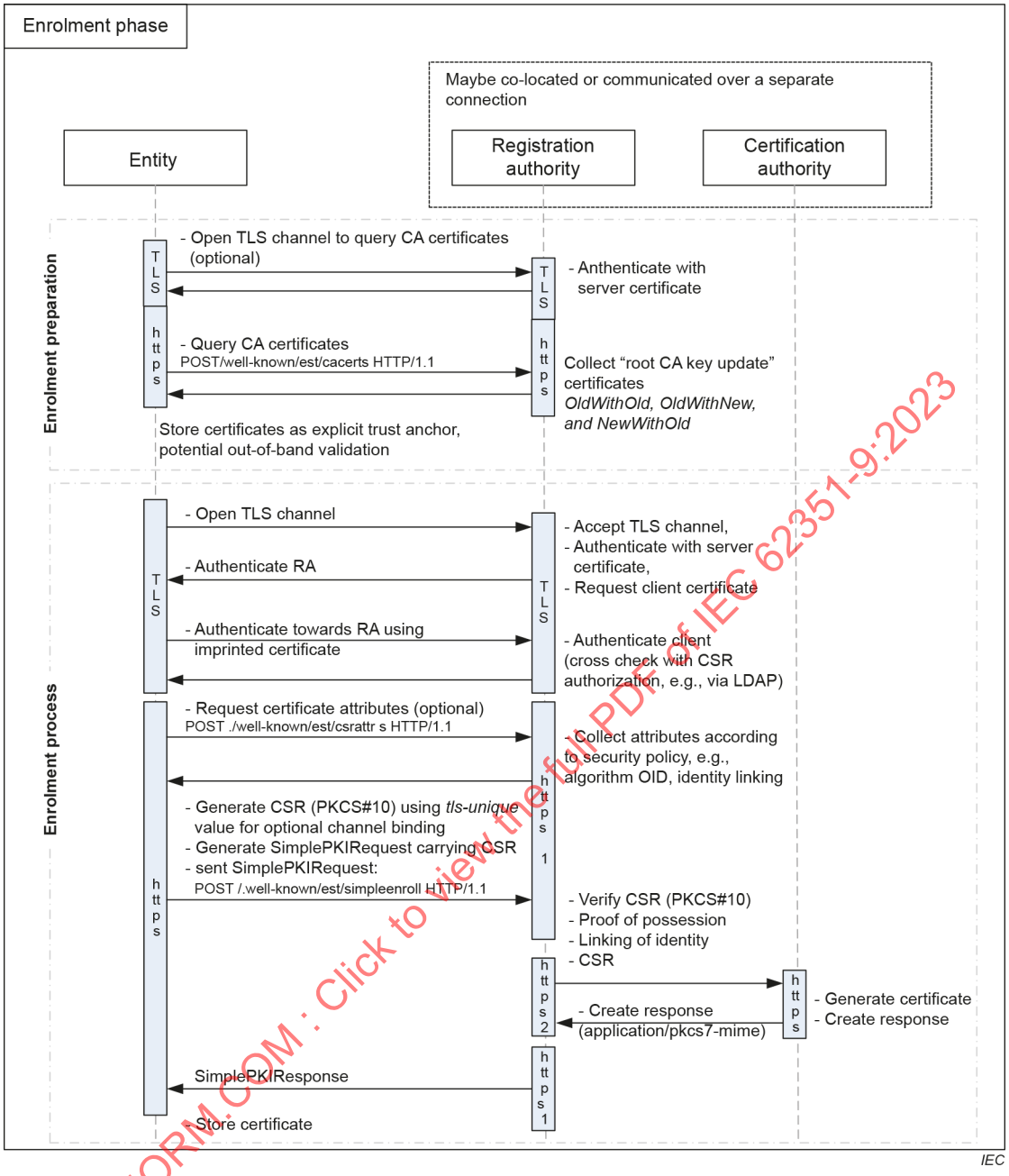



**Figure 10 – Example of the EST entity enrolment and CSR process**



Notes about Figure 9 and Figure 10:

- Prior to enrolment, devices must be configured. Besides their standard configuration, such as their own IP address(es), they must also be given their PKI enrolment data, RA/CA IP address, trust anchor certificates (CA/Root certificate), etc. As stated in 5.8.3, this process can be manual or automated. Automated provisioning can be supported by example protocols stated in 5.8.3.
- The entity checks the authenticity of communication peers based on the root certificate of the CA. This root certificate is fed to the device during the device configuration phase (see 7.3.6) or as part of an automated onboarding. The CA signs the certificates issued with the CA private key. Entities check the authenticity of the received certificate by verifying the signature in the certificates using the CA public key.

### 5.8.5 Certificate signing request (CSR) processing

#### 5.8.5.1 Enrolment process

The enrolment process involves certificate signing by a CA to verify the identity of an entity, to verify that the entity possesses the private key associated with the public key, and to issue a certificate to that effect. This certificate signing requires the entity (device) to issue a certification request to the RA/CA. The CSR processing, using device generated key material, consists of the following steps, as illustrated in Figure 11.

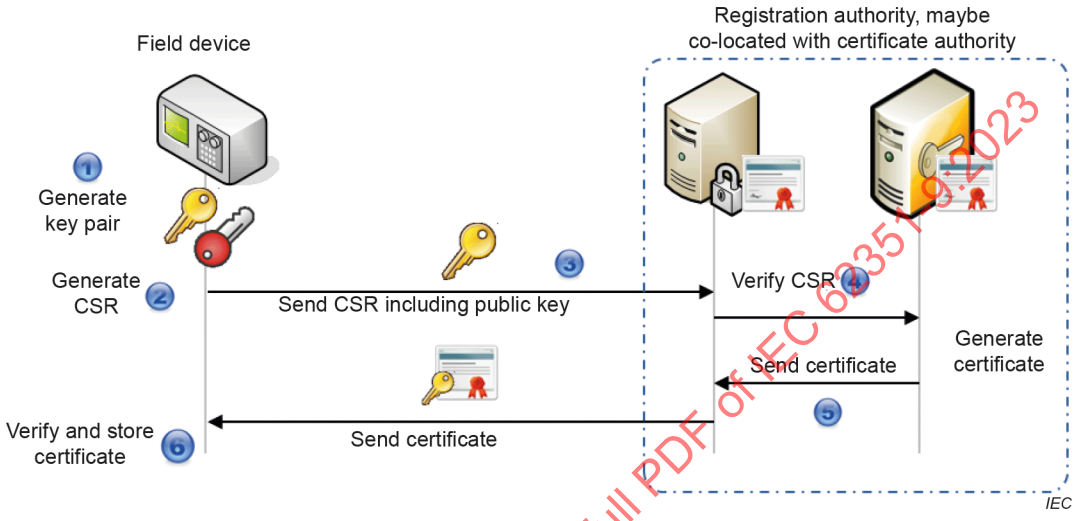



**Figure 11 – CSR processing**

1) The entity generates a pair of public and private keys.
2) The entity generates a *CertificationRequestInfo,* here using the PKCS #10 specification format [18]. This request contains a "subject distinguished name", the public key from the public/private pair it just generated (see ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019)), and optionally a set of attributes. The *CertificationRequestInfo* is signed with the entity's private key. The *CertificationRequestInfo*, a signature algorithm identifier, and the entity's signature go into a *CertificationRequest* structure. See [18].
3) The entity sends the *CertificationRequest* message to the RA/CA for authorization and certification as part of an enrolment protocol (see 5.8.6).
4) The RA validates the request by verifying the signature on the incoming request.
5) If the request is valid, the RA authenticates the requesting entity and, if valid and authorized, invokes the CA, which creates a public-key certificate from the distinguished name and public key, the issuer name, and the certification authority's choice of serial number, validity period, optional extensions and digital signature algorithm providing the information for the signature of the listed data. The CA then sends the certificate via the RA to the entity.
6) The entity performs the signature verification on the received certificate from the CA to ensure that the received certificate is issued and signed by the trusted CA. Note the assumption here is that the trusted CA certificates are distributed and installed in the entity during the commissioning/onboarding/configuration phase. The entity verifies the signature on the received certificate from the operational CA with the help of these trusted CA certificates. If the CA signature is valid, the entity stores the certificate in the implementation-preferable format and appropriate file extension when storage is file-based (e.g., .der, .pem, .cer, .crt, etc).

The CSR process may require human involvement with an administration role to activate, enable or approve the CSR process. This manual support may be necessary at the same time as the CSR application or may take place in advance, depending on the entity authentication procedure. If an entity is authenticated with a vendor-issued credential (IDevID) or with a one-time password, this information can be provided to the RA in advance.

The following two subclauses describe the certification requests formats PKCS #10 and CRMF.

#### 5.8.5.2 PKCS #10 certification request

The certification request syntax, as defined by IETF RFC 2986 [18], is used by different public-key certificate management protocols to convey a certification request from a client (end entity) to a CA or an entity interfacing to the CA, e.g., an RA or EST server (see RFC 7030).

IETF RFC 2986 does not specify any response format or how a certification request is carried out, but leaves that to a referencing specification.

The certification request is specified as PKCS #10 in RFC 2986 [18]. A certification request is sent from an entity to the registration authority (RA), which may be co-located with the CA or be physically separated.

Figure 12 illustrates the structure of a certification request. It has the following components:

a) The `Version` component indicates the version of the certification request. The version specified by IETF RFC 2986 is version 1 and is indicated by a value '0'.

b) The Subject component specifies the name to be used in the subject component of the requested public-key certificate.

c) The Subject public-key information component specifies the information to be included in the `subjectPublicKeyInfo` component of the requested public-key certificate.

d) The Attributes component holds a set of directory attributes that provides additional information for the generation of the requested public-key certificates.

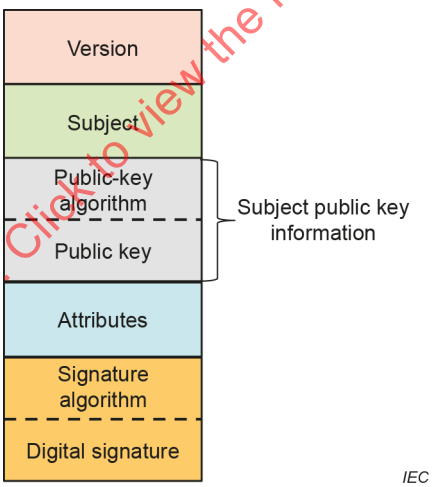



**Figure 12 – Certification request format**

The certification request is digital signed using the private key corresponding to the public key provided in certification request. If the recipient is able to validate the signature using the public key provided in the certification request, it proves that the issuer is in possession of the private key corresponding to the presented public key.

IETF RFC 2985 [19] defines two attribute types considered relevant for inclusion in a certification request:

a) An attribute of the `challengePassword` attribute type may be included and, if present, the same attribute shall be used for a revocation request.

b) An attribute of the `extensionRequest` attribute type may be included and shall then hold a sequence of public-key certificate extensions to be included in the public-key certificates.

#### 5.8.5.3 Certificate Request Message Format (CRMF)

The Certificate Request Message Format (CRMF) is specified in IETF RFC 4211 [20] and illustrated in Figure 13.

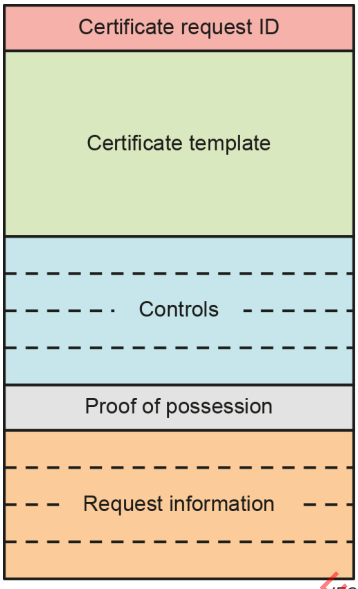



**Figure 13 – Certificate request message format**

Like the certification request described in 5.8.5.2, the CRMF does not constitute a protocol, but is a data type specification to be included in referencing protocol types.

The details of CRMF are found in IETF RFC 4211. An overview of the different parts of CRMF is given in the following:

a) The certificate request ID may be used for pairing requests and responses.

b) The certificate template allows the requestor to specify or omit content for each component of the public-key certificate being requested. It may be a little confusing that specifications for certain components are omitted although the ASN.1 allows their presence. A single component (public-key information) is present, but still listed as optional. For details, see IETF RFC 4211.

c) The controls are attributes that will not to be part of the public-key certificate but provide information about how the context in which the public key is to be issued. IETF RFC 4211 defines a number of controls, like additional authentication information about the subject of the public-key certificate, suggestions on how the public-key certificate is published, suggestions on how the private key should be archived, information about a possible old public-key certificate, and an encryption key to be used by the CA to encrypt the response.

d) The proof-of-possession, when present, provides a detailed specification for how to prove the possession of the private key, depending on whether the key is intended for digital signature encryption (RSA) or for key agreement.

e) The request information

The Certificate Request Message Format (CRMF) syntax and semantics is used to convey a request for a certificate to a CA, possibly via a Registration Authority (RA), for the purposes of public-key certificate production. The request will typically include a public key and the associated registration information.

### 5.8.6 Enrolment Protocols

#### 5.8.6.1 Simple Certificate Enrolment Protocol (SCEP)

The Simple Certificate Enrolment Protocol (SCEP) was developed to simplify the enrolment of large numbers of devices and to make the issuing of digital certificates scalable. The functionality of SCEP is limited to certificate enrolment and renewal, and the distribution of CA certificates and CRLs. Entities can use SCEP to request their digital certificate electronically using CMS and PKCS #10 over HTTP. The key material is generated only on the client side.

Client-side authentication of the enrolment request may be done by using either self-signed certificates or CA-issued certificates. For self-signed certificates, the usage of an additional secret (challenge password) is necessary to allow an authorization of the certification request. For CA-issued certificates, this can be directly linked to the CA issued certificate, if it can be validated on the RA side. SCEP binds proof of identity and proof of possession to the certification request object.

Note that SCEP utilizes encryption and digital signatures to protect the messages exchanged. The request and response messages exchanged are based on SCEP messaging objects (relying on CMS). The encryption of the messages depends on the utilized asymmetric cryptographic algorithm in the following way:

- For RSA based certificates (entity or CA side), the data is encrypted using the public key of the recipient.
- For ECDSA based certificates (entity or CA side), the data is encrypted based on the challenge password. Note that this design has an influence during operation as it requires to also distribute the challenge password before a certificate update can take place.

SCEP is defined by the IETF and is available as informational RFC 8894. Part of the revision from the initial document was the transition from PKCS #7 to CMS, which allows for broader coverage of cryptographic algorithms. Hence, it may be seen as profile of CMC.

#### 5.8.6.2 Enrolment over Secure Transport (EST)

Enrolment over Secure Transport (EST) is defined in RFC 7030 and is based on the Simple PKI handling in CMC. It defines some of the CMC functionality as optional, resulting in reduced complexity of the protocol. It may be seen as a simplified profile of CMC. The functionality supported by EST comprises the distribution of CA certificates and CRLs, retrieval of certification request attributes to query additional information or boundary conditions prior to generating a CSR, certificate enrolment, and certificate renewal. It allows for client or server-side key generation. In EST, only the simple PKI request/response interaction is mandatory, while the full PKI procedure support is optional. EST utilizes TLS as a secure channel and leverages the authentication of the TLS channel for identification and authorization of the requester by binding the CSR to the actual TLS session. Besides the proof of identity, the CMC portion provides proof-of-possession of the private key corresponding to the public key in the CSR.

In addition to the certificate enrolment and management functionalities, EST allows for the management of trust anchors on devices from the issuing CA, based on an already installed trust anchor.

EST may be terminated at the RA or at the CA directly. Terminating EST at the RA leaves the communication between the RA and the CA untouched. Terminating EST at the RA requires the RA to have a specific extension in the certificate used to authenticate towards the CA to ensure the request authorization has been done by the proper entity. Note, that it is also possible to utilize other enrolment protocols between the RA and the CA.

NOTE Implementors should take notice of IETF RFC 8951 on clarifications for the transfer encodings and ASN.1.

#### 5.8.6.3 Internet X.509 PKI Certificate Management Protocol (CMP)

The Certificate Management Protocol (CMP) is an Internet protocol used for obtaining public-key certificates in a PKI environment. It defines a protocol for the interaction between a client and the PKI components. In addition to CRL retrieval, certification request handling, identification/authorization option for the requester, and proof of possession of the associated private key, CMP also provides additional functionality like cross certification and certificate revocation. CMP supports the client and server-side generation of key material. CMP binds proof of identity and proof of possession to the certification request object, which makes it independent from the transport. CMP uses CRMF for the message format of the PKI messages. Transport of PKCS #10 requests is also supported.

CMP is defined in RFC 4210 [30]. The transport of CMP over HTTP is defined in a separate document, RFC 6712 [42].

As CMP is very feature rich, a lightweight profile is currently being standardized [48] to address industrial use cases by limiting the functionality to the necessary exchanges.

#### 5.8.6.4 Certificate Management over CMS (CMC)

Certificate management over Cryptographic Message Syntax (CMC) builds on the Cryptographic Message Syntax (CMS, defined in RFC 5652), PKCS #10 (RFC 2986) and CRMF to support the management of certificates. Similarly to CMP, CMC supports CRL retrieval, certification request handling, identification/authorization option for the requester as well as proof of possession of the associated private key. CMC also provides additional functionality like cross certification and certificate revocation. However, CMC defines both a simple and a full PKI request/response handshake, but requires both to be implemented. CMC supports the client and server-side generation of key material.

CMC binds proof of identity and proof of possession to the certification request object when using a full PKI Request. In case of the Simple PKI Request, proof-of-identity must be provided by other means.

CMC is defined in RFC 5272. The transport of CMC itself is defined in a separate document, RFC 5273 ([35]).

### 5.8.7 Trust Anchor Management Protocol (TAMP)

The enrolment protocols discussed in the previous clauses relate specifically to the handling of end entity certificates but also provide partly support for the management of the underlying root certificates as trust anchors. The Trust Anchor Management Protocol (TAMP) builds on CMS.

TAMP is specified in RFC 5934 and provides a transport independent management of trust anchors (TAs) in a device trust anchor store.

## 5.9 Revocation of public-key certificates

### 5.9.1 Certificate revocation lists (CRLs)

A CRL is a list of the serial numbers of certificates that have been revoked, along with a timestamp indicating when they were revoked, as well as a digital signature from the issuing CA. The revocation reason may also be provided as it is defined as an extension of a CRL. Revoked certificates should be considered as no longer valid and should not be relied on by any entity.

CRLs should be updated whenever a certificate is revoked. They should be made available to all affected entities in a timely manner. Adequate precision of time synchronization across entities and the system creating CRLs is necessary to ensure that the time information in the CRLs is accurate and that the entities have accurate information on revoked certificates. Support for clock synchronization and accuracy is defined in different protocols. Examples are.

- Network Time Protocol (NTPv4, IETF RFC 5905)
- Precision Time Protocol (PTP, IEEE 1588v2.1)

For both protocols, security enhancements have been defined (see also 7.7)

An example of a CRL is shown in Figure 14.

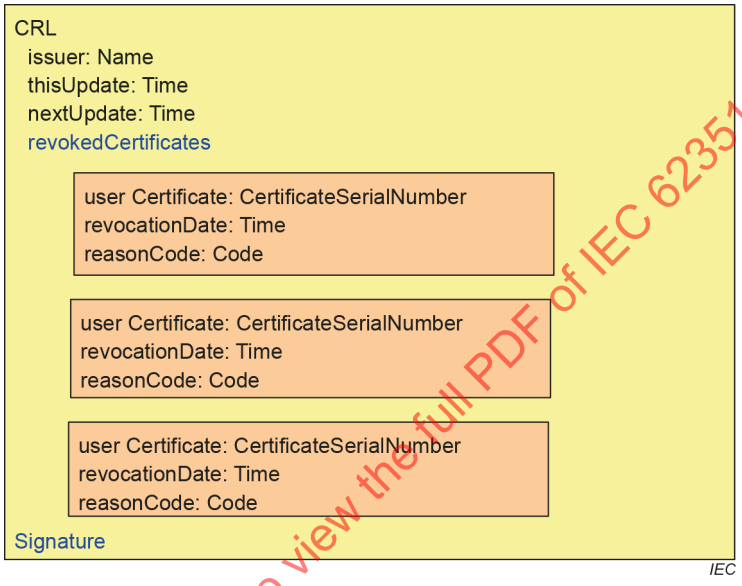



**Figure 14 – Certificate revocation list**

Since CRLs accumulate revoked certificates over time, they may grow and become very large. Large CRLs could be a problem for embedded systems, since embedded systems might not have enough available memory capacity to store them. CRLs may therefore be partitioned as a means to minimize the list for any particular entity. Alternative revocation check methods may be even more efficient, as described in 5.9.2.

Certificates should be revoked based on the following reasons and using the reason codes as defined in 9.5.3.1 of ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019).

- The private key is suspected to be compromised.
- The CA private key associated with a CA certificate is suspected to be compromised.
- The entity's affiliation has changed.
- The public-key certificate has been superseded.
- There has been a cessation of operation of the entity.
- There is a hold on the certificate.
- The privilege has been withdrawn.
- The private key for an attribute authority is suspected to be compromised.

### 5.9.2 Online certificate status protocol (OCSP)

The online certificate status protocol (OCSP), as defined by RFC 6960, is an alternative to CRL retrieval when checking the revocation status of a public-key certificate.

If a relying party is concerned whether a particular public-key certificate is still valid, it may send an OCSP revocation status request to the OCSP server (or the CA) responsible for the certificate of the entity. This OCSP request contains the protocol version, service request, the entity's certificate identifier and extensions. In order to avoid replay attacks, a "nonce" (one-time value) is mandatory to distinguish this status request from any previous status requests. The OCSP responder then validates the certificate and returns 'good', 'revoked' or 'unknown', using its own digital signature to authenticate the response.

This OCSP sequence is shown in Figure 15.

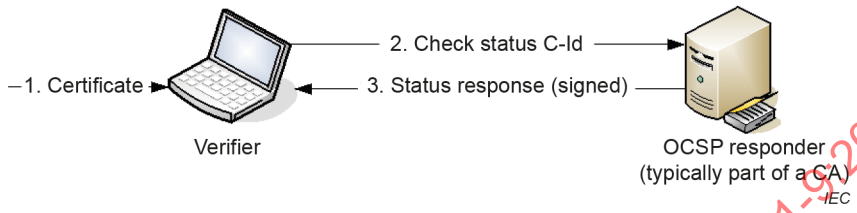



**Figure 15 – Overview of the online certificate status protocol (OCSP)**

Typically, online connectivity is required between the entity and the OCSP responder for the transmittal of the OCSP request and OCSP response. However, continuous connectivity may be challenging for some configurations of entities in the field. Additionally, the computational effort for processing the OCSP response and the communication delay may not be feasible for some entities. In addition, OCSP responses should have a short validity period since OCSP responders do not issue unsolicited status updates, even if a certificate has been revoked shortly after a status check.

OCSP caching is a further option allowing the receiver of an OCSP response to cache the response for a certain time. This avoids repeated OCSP queries during a certain timeframe. It also enables the possibility for the sender to staple the OCSP response to his certificate, avoiding the responder interaction with the OCSP server. IEC 62351-3 utilizes this option.

Therefore, depending on the system setup and the entity's capabilities, a hybrid combination of CRLs and OCSP may be utilized where one entity that normally does have connectivity acts as a proxy OCSP responder. This proxy entity fetches a CRL within a specific time period such as hourly or within 24 hours. The proxy entity (e.g., station controller) then serves as OCSP responder for other entities that do not normally have connection to the OCSP. This approach is depicted in Figure 16.

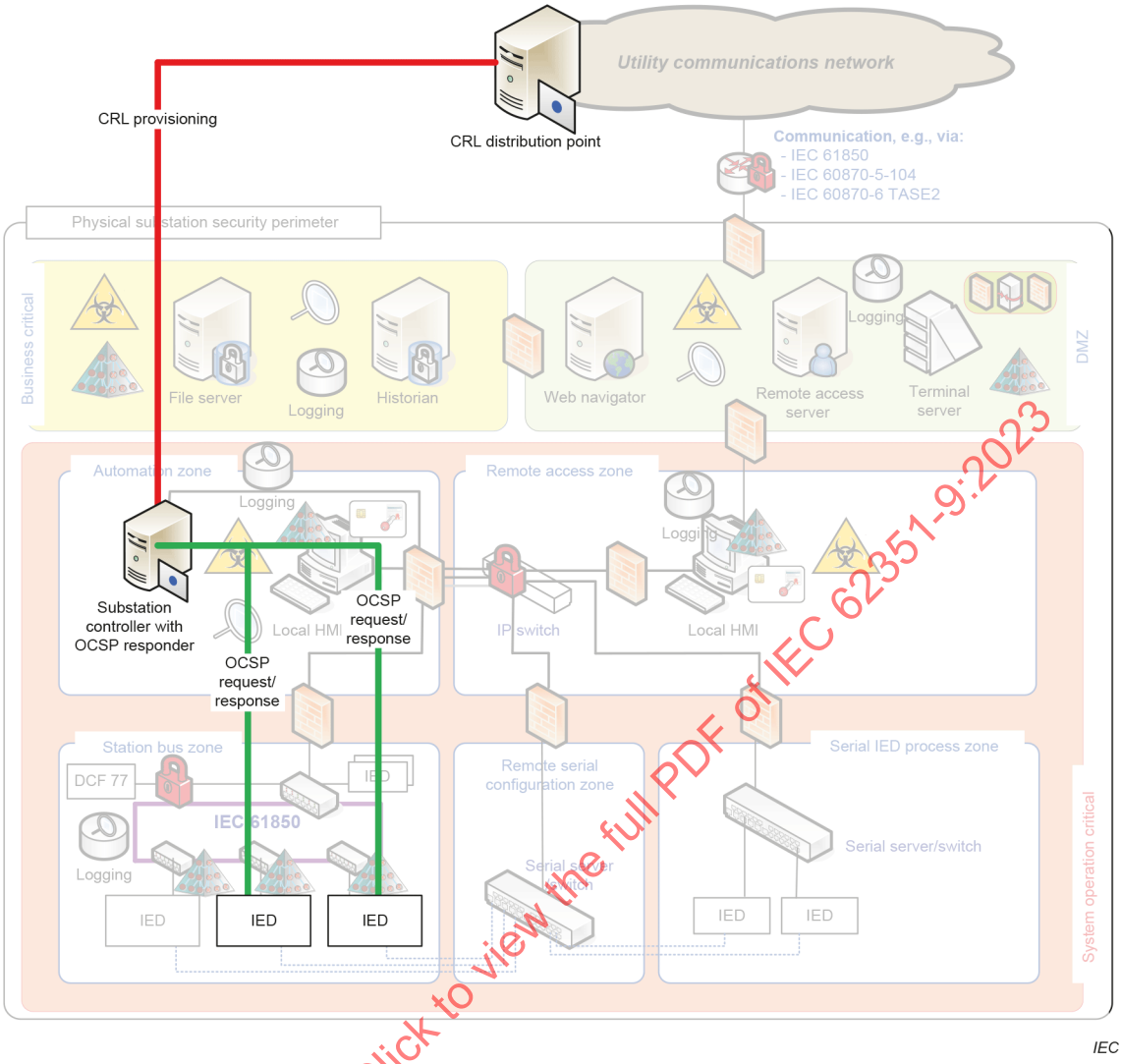



**Figure 16 – Diagram using a combination of CRL and OCSP processes**

A clear advantage of this approach is that it allows the usage of the OCSP process in local networks that lack permanent connectivity to the central CRL data. Moreover, this reduces the data traffic with the backend, which may be necessary for low bandwidth connections.

Entities should have access to a trusted real-time clock in order to check and assure the accuracy of the OCSP timestamp. If within a configurable timeout (e.g., 15 seconds) no response arrives, entities should trigger an OCSP failure alarm and assume that the certificate has not been revoked (particularly if availability is crucial).

To enable this approach, it is necessary for the proxy OCSP entity (e.g., station controller) to possess a certificate and corresponding private key to act as an OCSP responder. The OCSP responder would thus be configured as trusted responder node with a CA signed certificate.

OCSP requests may also be chained between peer OCSP responders in order to find and query the appropriate CA for the entity certificate being checked, with responders validating each other's responses against the root CA using their own OCSP requests.

OCSP requests may be performed proactively or reactively as depicted in Figure 17. In the proactive case, the entity requests its certificate status from the OCSP responder in advance and sends the OCSP response together with its certificate to the peer node that is validating the entity. This is called "OCSP stapling" and places the burden of OCSP communication on the proactive entity. In the reactive case, the receiving node performs the OCSP request to check the revocation state of the certificate presented by the peer entity. The reactive option is more commonly used, but the proactive option is supported by protocols like TLS 1.2 (RFC 5246) using an extension defined in RFC 6066 [41], allowing the TLS 1.2 server to transmit an OCSP response as part of the `ServerHello` message. This relates to the proactive server case shown in Figure 17.

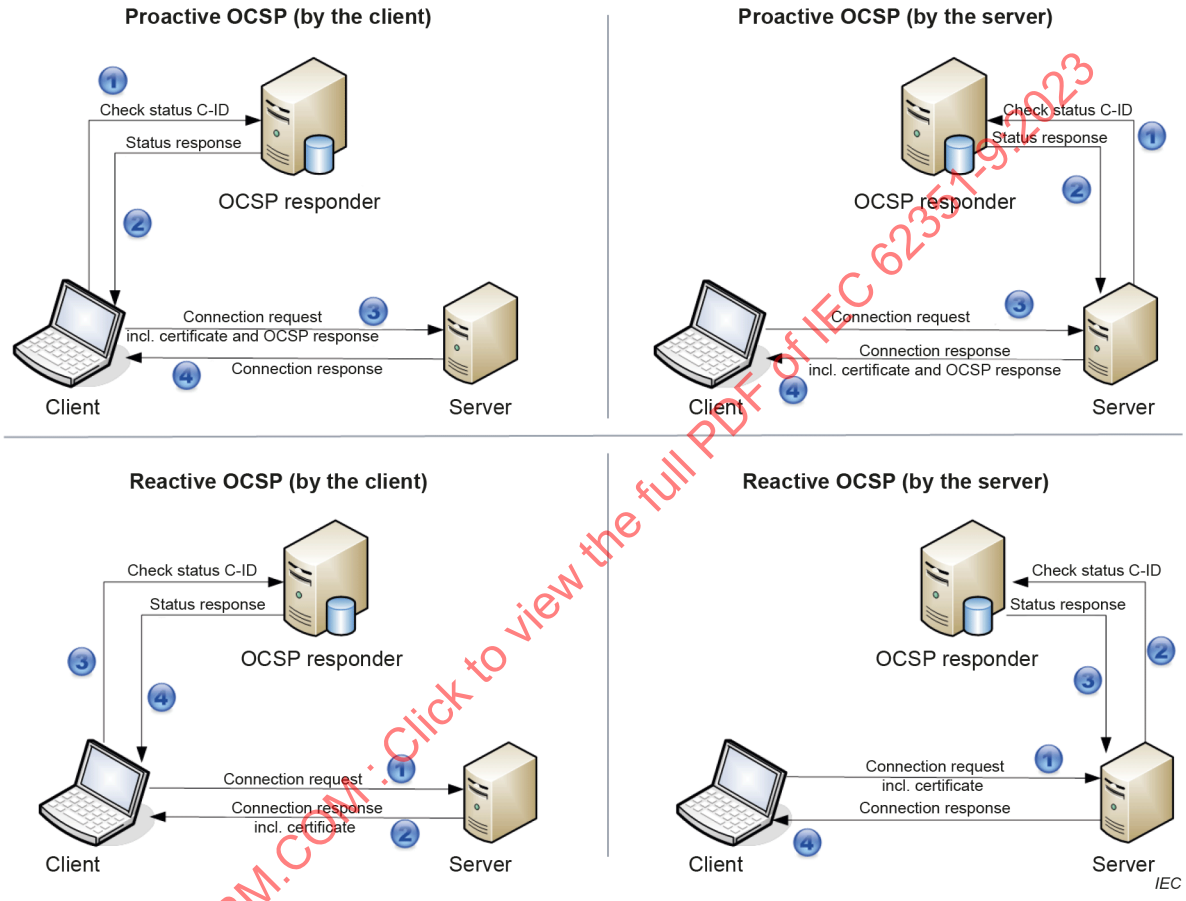



**Figure 17 – Call Flows for the Online Certificate Status Protocol (OCSP)**



Note that TLS 1.3 (RFC 8446, [45]) also allows OCSP stapling for the TLS client-side certificate.

If certificate revocation or certificate validity check fails, it is expected in most cases that the entity will continue operation after sending an OCSP alarm to warn operators about possible unauthenticated communication. This prevents denial-of-service attacks caused by making OCSP responders unavailable.

### 5.9.3 Server-based certificate validation protocol (SCVP)

The validity check of a certificate may become complex, especially if the certification path is rather long. For this case, the Server-Based Certificate Validation Protocol (SCVP) allows an entity to delegate the certification path construction and certification path validation to a "master" node. This protocol is defined in RFC 5055 ([34]). SCVP may be used in conjunction with CRL as well as with OCSP (as shown in Figure 18).

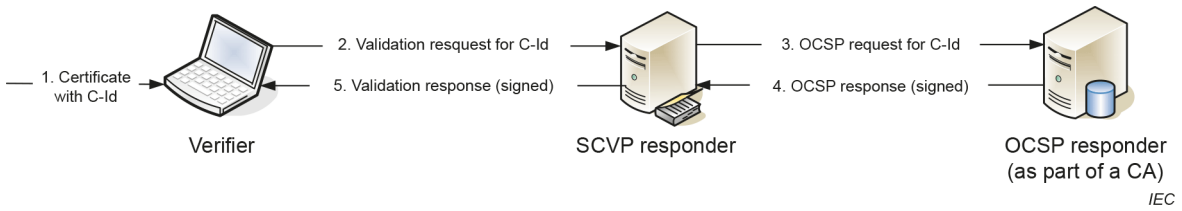



**Figure 18 – Overview Server-Based Certificate Validation Protocol using OCSP Backend**

Additionally, it is recommended to keep CA hierarchies as flat and compact as possible to reduce the performance penalties incurred when validating certificate chains.

### 5.9.4 Recovering from certificate revocation of an end entity

An end entity certificate may be revoked for different reasons as outlined in 5.9.1. To recover from this situation, further handling depends on the actual revocation reason. The evaluation of this reason and the appropriate reaction is typically defined by an organization's security policy (recovery plan) and also reflected in the CPS of the certificate issuer. Therefore, the following discussion aims at an overview of potential ways on how to get a new certificate to the device in order to keep it operational.

If the revocation was done because of a compromised private key in the devices or anything that involves physical contact with the device by an attacker, it is strongly recommended that a device local procedure be performed by a service technician to change/update the certificate and the corresponding private key. This should also involve the verification of the (physically and logical) integrity of the device and its operational settings, to ensure that it cannot be misused as an originator of an attack and also to ensure the device can be operated safely.

In general, recovering from the revocation of a certificate may involve administrative means to allow the remote renewal of a certificate. Examples include:

- The device possesses a dedicated certificate used for the management of security credentials. This certificate may be an operational certificate (LDevID) or a manufacturer provided certificate (IDevID). If not revoked, this certificate may be used to enrol for a new certificate. The device may autonomously verify the revocation state of its certificate and immediately trigger the enrolment of a new certificate. The decision to permit a new enrolment may be supported by a device specific implementation. For instance, the device could support a secure element or dedicated security hardware for storing and or managing the IDevID securely, while operational credentials (LDevIDs) could be stored in a different location. It is then the task of the operating PKI to decide upon renewing the certificate.
- The device could be triggered to renew the certificate using an administrative channel. This may be a separate connection using protocols like Web (https), SNMP, ssh, or other, for which the credential has been provided out-of-band or during the initial onboarding. Also pressing a dedicated button on the devices to initiate the bootstrapping anew may also be possible.

As stated above, there is the strong recommendation to evaluate the revocation reason to be able to plan the next steps accordingly.

## 5.10 Trust via non-PKI issued (self-signed) certificates

Self-signed certificates play an important role in a PKI environment. Trust hierarchies or trust chains in PKIs are rooted by "root certificates" (see 5.7.4). PKI root certificates may be self-signed certificates known as trust anchors, created by trusted CAs that allow this trust to be distributed.

Under certain circumstances, non-PKI issued self-signed certificates may be more easily implemented than the more complex PKI signed certificates in order to establish trust. This may especially be the case in smaller deployments with only a limited number of entities. Non-PKI issued self-signed certificates are public/private key pairs internally generated by an entity, where the private key is used to sign the entity's own public key together with additional information, such as the subject name, validity time, etc. So that they can be used in the same way as PKI based certificates, e.g., to authenticate interactions in TLS connections, self-signed public-key certificates are required to comply with the public-key certificate structure as described in ISO/IEC 9594-8.

These self-signed certificates cannot be generally accepted as trustworthy. Hence, to enable the acceptance of self-signed certificates by a limited group of entities for authentication, they should only be used in conjunction with authorization and validation lists (see 5.11).

The level of complexity of using entity-specific self-signed certificates and authorization and validation lists can be compared with the use of pre-shared keys. Self-signed certificates are issued per entity while pre-shared keys are issued per paired connection. Thus, while the number of self-signed certificates will depend upon the number of end points, the number of pre-shared keys will depend upon the number of connections between the end points.

As self-signed certificates need to be public-key certificates, their application could open the migration path to PKI-based public-key certificate implementations, so that self-signed certificates may easily be replaced with PKI-based certificates once those are available.

### 5.11 Authorization and validation lists

#### 5.11.1 General

Authorization and validation lists (AVLs) provide information about potential communications entities and possible restrictions on the communications with such entities in a particular environment. They are specified in Clause 11 of ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019).

An AVL is used by an entity when that entity is acting as a relying party. Such an entity is called an AVL entity. An AVL is placed in an entity called authorizer that is responsible for the content of the AVL, i.e., the authorizer is trusted by the AVL entity to provide valid information. An AVL is signed by the issuing authorizer.

The communication between the authorizer and the AVL entity has to be protected as any other communication with respect to integrity, authenticity and possible confidentiality. To simplify validation of the AVLs, it is recommended that an authorizer be closely related to the AVL entities it serves, e.g. being within the same organization so that a trust relationship may be established between the authorizer and the AVL entities it serves (see B.1).

AVLs may be used either in non-constrained environments or in constrained environments as detailed in 5.11.2 and 5.11.3.

#### 5.11.2 AVLs in non-constrained environments

A non-constrained environment is an environment where AVL entities are capable of performing traditional validation of received certificates.

AVLs are used to restrict communication relations within a specified set of entities. They may impose restrictions on such entities beyond path length, policy, and name restriction as detailed in 11.3 of ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019).

The entities with which communication is possible are identified in the AVL either by reference to their public-key certificates or by the name structure of the organization to which the entity belongs.

### 5.11.3 AVLs in constrained environments

An entity may be constrained in many ways:

- It is battery driven and needs to limit processing to save on the battery.
- It has little processing capabilities.
- Its communication channel has limited bandwidth.
- It has limited storage.
- It may have stringent time constraints having to respond to incidents very quickly not allowing time for external communication to validate received public-key certificates.

The AVL authorizer is responsible for verifying that all public-key certificates, represented in the AVL are valid and may be trusted at the time of issuing an AVL. It is also the responsibility of the authorizer to be updated on the validity of the public-key certificates in their AVLs. The authorizer also makes the judgement as to when an expired or revoked public-key certificate should or should not be used in cases where its withdrawal might affect a critical entity. An AVL entity assumes that a listed public-key certificate is valid to use.

This mode of operation requires communication between the authorizer and the CAs that issued the public-key certificates listed in the AVL.

## 6 Key management (normative)

### 6.1 General

Key management applies to the handling of asymmetric key material, symmetric key material, and a combination of both. For the handling of asymmetric key material, specifically the management of certificates and corresponding private keys through the device and key material life cycle, Clause 7 shall apply. For the management of symmetric keys, specifically for the management of group keys, Clause 8 shall apply.

### 6.2 Handling of security events

Throughout the document security events are defined. These security events are intended to support the error handling and thus to increase system resilience. Implementations should provide a mechanism for announcing security events.

The information about the security events and potential detailed information can only be provided by the entity based on the availability of this information through the underlying platform or utilized components.

It is strongly recommended that the security events defined throughout the document are made available to the operational infrastructure by cyber security events as specified in IEC 62351-14 and/or by monitoring objects as specified in IEC 62351-7. Annex D provides a mapping of the defined events in this document to the notion of IEC 62351-14.

Note that notice, warnings, errors, and alarms are used to indicate the severity of an event from a security point of view. The following notion from IEC 62351-14 is used:

– A *notice* refers to a cyber security related activity during the routine use or maintenance of an entity. It does not relate to a cyber security breach or attack or deviation from the normal operating condition of an entity.
– A *warning* is a deviation from the normal operating condition of an entity but not necessary a cyber-attack.
– An *error* describes an unforeseen condition, which may indicate unauthorized activity. It may not require immediate action.

– An *alarm* is an indication of a serious problem, which may indicate unauthorized activity. Action is recommended to be taken immediately.

In any case, it is expected that an organization's security policy determines the final handling of events based on the operational environment. For instance, the assessment of one of more alarms could rise to the level of an incident.

### 6.3 Required cryptographic material

Standards that reference this document shall specify the required cryptographic materials that shall be present to establish secure communications, as per the Protocol Implementation Conformance Statement (PICS) form in Clause 9.

### 6.4 Random Number Generation

The generation of any random value related to key management shall follow ISO/IEC 18031 [87]. Key pair generators shall be responsible for providing statistically adequate random number generators (RNG) and utilizing them appropriately.

NOTE Guidance can be found in NIST SP 800-90A [61] as well as in IETF RFC 4086 [62].

Keys shall be generated externally for entities which are not capable of generating keys internally. CAs or other authorized systems may provide this external key generation facility. The keys shall then be installed securely in those entities. See also Clause B.12.

### 6.5 Object identifiers

#### 6.5.1 Concept of object identifiers

An object identifier (OID, developed by ITU-T and ISO/IEC) is an identification mechanism to associate any type of object with a persistent name. It is typically structured in a hierarchical way (as OID tree). See also ISO/IEC 9834-1 | Rec. ITU-T X.660.

OIDs are allocated by various international standards, including ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019), and may be registered for private use. The OID namespace (root) value for IEC 62351 is 1.0.62351 and 1.2.840.10070. IEC TS 62351-8 utilizes the latter approach to define access tokens (authorizations). This document and every new upcoming part of IEC 62351 should use the first OID (1.0.62351).

#### 6.5.2 Use of object identifiers by this document

The principle for allocation of object identifiers (OIDs) is specified in ISO/IEC 9834-1 | Rec. ITU-T X.660. The OID allocation for this document is defined as:

```
id-IEC62351-9 OBJECT IDENTIFIER::= { 1 0 62351 9 }
```

The following branch is allocated for defining OIDs for AVL extensions:

```
avl62351Extion OBJECT IDENTIFIER::= { id-IEC62351-9 1 }
```

The following branch is allocated for defining OIDs for AVL entry extensions:

```
avl62351EntryExt OBJECT IDENTIFIER::= { id-IEC62351-9 2 }
```

The following branch is allocated for defining OIDs for protocol identifiers:

```
id-62351prot OBJECT IDENTIFIER::= { id-IEC62351-9 3 }
```

# 7 Asymmetric key management (normative)

## 7.1 General

This clause describes the normative requirements for the handling of asymmetric key material. Asymmetric key material in the context of this specification is considered as the three elements of a certificate: the public key, the private key, and the information about the issuing certificate authorities (trust anchor, certificate chain). Note that in the context of IEEE 802.1AR this is called DevID (Device Identity). This clause defines the generation of asymmetric key pairs on the end entity, their certification through a PKI as well as the lifecycle handling through the PKI.

## 7.2 Certificate components

### 7.2.1 Public-Key certificate components

The public-key certificates utilized for asymmetric key management shall be public-key certificates as defined in ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019). Most general extensions are specified by ISO/IEC 9594-8 | REC. ITU-T X.509. More specific extensions are specified by other documents. As an example, IEC 62351-8 specifies an extension for support of user roles for access control.

Table 1 comprises the mandatory and optional components of public-key certificates which shall be recognizable and processable by power system entities:

**Table 1 – Public-key certificate components**

| Component name | Support | Content |
|---|---|---|
| Version | M | Supported version of the certificate: see 7.4.4.2 |
| Serial Number | M | Integer value, defined by the issuing CA. It shall be unique for each certificate issued by a given CA |
| Signature | M | Signature algorithm used for generating the certificate, determined by an OID, see 7.4.4.4 |
| Issuer | M | Distinguished name of the Issuing CA, see 7.4.4.3 |
| Validity | M | Certificate validity time, see 7.4.4.5 |
| Subject | M | Identifies the entity associated with the public key, see 7.4.4.6. |
| Subject Public Key Info | M | Algorithm Identifier of the public key and the public-key itself, see 7.4.4.7 |
| **Extensions** | | |
| Authority Key Identifier | M | Key Identifier of the Issuing CA: determines the key used to verify the certificate signature, see 7.4.4.10.2 |
| Subject Key Identifier | C1 | Key Identifier of the entities public key, see 7.4.4.10.3 |
| Key Usage | M (c) | Identifies the intended usage of the public-key certificate, see 7.4.4.10.6 |
| Extended Key Usage | C | Identifies additional purposes for the usage of the public-key certificate (selection is conditional, depending on use case), see 7.4.4.10.7 |
| Certificate Policy | O | For an end entity certificate, it indicates the policy under which the certificate has been issued and the purposes for which the certificate may be used by an OID [36]. |
| Subject Alternative Name | C | Contains one or multiple alternative names for the certificate subject. See 7.4.4.10.4 |
| Basic Constraints | C (c) | To be set for issuing CA certificates otherwise CA=false (or empty/missing); additional component `pathLenConstraint` may be used to limit the certification path, depending on the organization's security policy, see 7.4.4.10.5 |
| CRL Distribution Point | C2 | Location of CRL for http or/and LDAP, see 7.4.4.10.9 |

| Component name | Support | Content |
|---|---|---|
| Authority Information Access | C2 | Indicates how to access information and services of the issuer of the certificate: if revocation information is provided via OCSP, it contains the location of the OCSP responder (`id-ad-ocsp`), see 7.4.4.10.10 |
| Role-based Access control | C3 | Enables role -based access control according to IEC 62351-8 A-Profile. See 7.4.4.10.11. |
| Authorization Validation | O (c) | If used it shall be set to "c" critical to ensure that the relying party verifies an AVL before accepting the end entity certificate, see also 7.4.4.10.8.. |
| SOA identifier | C4 | Source of Authority in issuing AA certificate. See 7.4.4.10.13 |

C1: Shall be set in CA certificates

C2: Setting this extension is in the responsibility of the issuing CA. This allows a CA to also issue short-term certificates, which do not require a revocation check. The validity period of short-term certificate should be carefully evaluated and chosen by the operator, specifically if CRLDP or OCSP responder information is not included in the issued certificate. Without this information the revocation of a public-key certificate may not be recognized by the relying party. As the infrastructure must support CRL and OCSP, both extensions can be provided to allow the relying party to check the revocation state.

C3: Shall be set in end entity certificate when supporting IEC 62351-8 Profile A.

C4: Shall be set in issuing AA certificates to enable identification of an AA by the issuing CA.

The following notation is used in the table:

M Required: shall be included and processed

O Optional: can be included.

C Conditional use (see remarks at the bottom of Table 1)

(c) This extension is critical. If an implementation recognizes that a "critical" extension is present, but the implementation cannot interpret the extension, the implementation shall reject the certificate.

The certificate content is checked during the verification of certificates. Error handling will be according to identified problems during the verification. Certificate verification is described in 7.4.

### 7.2.2 Attribute certificate components

If attribute certificates are used, they shall be attribute certificates as defined in ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019). Extensions to the certificate may be defined in other documents.

Table 2 comprises the mandatory and optional components of attribute certificates, which should be recognizable and processable by power system entities utilizing attribute certificates

**Table 2 – Attribute certificate components**

| Component name | Support | Content |
|---|---|---|
| Version | M | Supported version of the certificate: see 7.4.5.2 |
| Holder | M | Contains a reference to the holder either to a public-key certificate or another entity name as outlined in 7.4.5.3 |
| Issuer | M | Contains the distinguished named of the issuing AA, see 7.4.5.4 |
| Signature | M | Signature algorithm used for generating the certificate, determined by an OID, see 7.4.5.5 |
| Serial Number | M | Integer value, defined by the issuing AA. It shall be unique for each certificate issued by a given AA |
| attributes | M | See 7.4.5.6. If role-based access control according to IEC 62351-8 B-Profile is used, the defined attribute shall be used for attribute certificates. |
| attrCertValidityPeriod | M | Contains the validity period of the attribute certificate, see 7.4.5.7 |
| **Extensions** | | |
| Authority Key Identifier | O | Key Identifier of the Issuing AA. See 7.4.5.8.2 |
| CRL Distribution Point | C1 | Same extension as used in public-key certificates. In the context of attribute certificates, it is used to provide the location of the ACRL, see 7.4.5.8.3 |
| Authority Information Access | C1 | Indicates how to access information and services of the issuer of the attribute certificate, see 7.4.5.8.3 |
| No Revocation Available (noRevAvail) | C1 | Indication that revocation status information is not provided for this attribute certificate, see 7.4.5.8.3 |
| C1 If `noRevAvail` is provided, CRLDP/AIA shall not be provided. If CRLDP or AIA is provided, `noRevAvail` shall not be provided to avoid inconsistencies. Setting these extensions is in the responsibility of the issuing AA. This allows an AA to also issue attributes certificates with a very short validity period, which do not require a revocation check. | | |
| The following notation is used in the table:<br>M Required: shall be included and processed.<br>O Optional: can be included.<br>C Conditional use (see remarks at the bottom of Table 2) | | |



The certificate content is checked during the verification of certificates. Error handling shall be according to identified problems during the verification. Certificate verification is described in 7.4.

Note that functional support in IEC 62351 for attribute certificates is only specified for RBAC. Other usages are out of scope of this document.

## 7.3 Certificate generation and installation

### 7.3.1 Private and public key generation and installation

Entities performing asymmetric cryptographic functions shall possess at least one pair of asymmetric keys. An entity shall either generate its own asymmetric cryptographic key pair and contact a PKI infrastructure for certification or shall be provided with externally generated cryptographic key pair including the certificate from an issuing CA. The private key of the key pair shall be stored securely. The provisioning of the centrally generated cryptographic key pair should be performed in a protected location.

Entities either shall generate or be provided with new key pairs under the following conditions:

- No key pair is present at start-up time (if not present or the associated public-key certificate has expired)
- Change in controllership of the entity (change in ownership, control authority, and/or reconfiguration of the entity)
- By command from an authorized entity (e.g., request for certificate renewal)
- Entity's private key has been compromised
- The IP address (or FQDN) of the entity has changed.

It is strongly recommended that client-side key generation is supported by devices in the power system, to avoid transport of the private key outside the device.

The utilized format for the key provisioning is described in 7.3.2.

### 7.3.2 Cryptographic key protection

Private keys and long-term symmetric keys (including pre-shared keys) of each entity shall be protected against unauthorized modification, retrieval, and cloning.

During transport, the private key shall be protected against eavesdropping and tampering by being encrypted by a transport key such as is defined in PEM (RFC 7468), PKCS #8 (RFC 7468), and PKCS #12 (RFC 7292). All entities shall support the handling of PKCS #12. The PKCS #12 container should also include all issuing (Sub-)CA certificates.

The inability to handle PKCS #12 objects due to formatting problems shall raise a security event ("warning: PKCS #12 format mismatch"). Implementations should provide a mechanism for announcing security warnings.

If PKCS #8 objects are supported but could not be processed due to formatting problems, a security event ("warning: PKCS #8 format mismatch") shall be raised. Implementations should provide a mechanism for announcing security warnings.

Any attempt to compromise a key shall be detectable if technically possible. Such an event shall be logged and shall trigger an alarm.

NOTE Guidance on cryptographic key protection can be found in NIST SP 800-57, Part 1 [57] and in FIPS 140-2 [53]. Moreover, IEEE 1686 [5], IEEE c37.240 [75], and IEC 62443-4-2 [74] provide requirements on protecting of sensitive information.

### 7.3.3 Use of existing security key management infrastructure

The use of an existing security public key management infrastructure (PKI) shall be allowed, as long as the issuing CAs provide complete chain of trust (intermediate CAs) certificates all the way to the root CA certificate and support the required certificate management / enrolment protocols as stated in 7.3.7. to interface with devices / entities.

It is strongly recommended to only install root CA certificates which are necessary in the operational environment and for the task the device fulfils. The selection of the supported root CA certificate(s) is typically part of an organization’s security policy.

### 7.3.4 Certificate policy

It is strongly recommended to create a certificate policy (see RFC 3647 ([26]) for guidance).

### 7.3.5 Entity registration for identity establishment

All entities to be commissioned in an organization shall be registered in at least one registration authority (RA), which may be co-located with the certification authority (CA) that is approved by the organization. This RA shall be able to verify the entity's identity on a certificate signing request (CSR). Registration may be done manually (e.g., for a small number of entities), or automatically by running scripts or configured positive list that are generated from engineering data. Registration may take place out-of-band, off-site or on-site. The CSR can be exported and imported manually or transported via an enrolment protocol like described in 7.3.7.

Registration data shall include at least one of the following elements:

- The entity's subject, which identifies the entity and the unique name that will appear in the entity's certificate or another unique identifier of the subject, e.g., `serialNumber`. Note that the certificate may include the physical serial number of the device as `X520SerialNumber` with the OID tag `id-at-serialNumber`. See Table 1.
- If SCEP is used for certificate enrolment a unique one-time activation code (or OTP), which allows the entity to authenticate itself to the RA, when performing a certification request (CSR), shall be provided as registration data. Note: How the OTPs are created and distributed to the enrolling entities is out of the scope of this document.
- A certificate serial number and issuer of a manufacturer build-in public-key certificate, which allows the entity to already be authenticated using this public-key certificate.
- A fingerprint of the public-key certificate used to authenticate the enrolling entity.
- A trusted issuing CA, which enables the enrolment with arbitrary certificates from that issuing CA.

### 7.3.6 Entity configuration

In addition to the basic certificate parameters as defined in ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019), entity configuration data shall include the following:

- CA certificate(s) of the organization(s), which the entity shall trust and communicate with (see 5.6.4). An implementation claiming conformance to this document shall support a minimum of 5 Certification Authority related trust anchors. The actual number depends on the target use case. Note that this requirement is aligned with IEC 62351-3:—[4].
- RA IP address of the organization or a domain name (such as IEC62351.LocalCA) that shall be allowed to perform PKI operations, such as CSR processing.
- Part of the entity's certificate is the subject containing the device's subject (e.g., Common Name (CN) or other subject values), which identifies the entity uniquely. This entity name appears in the entity's certificate. See also 7.2.

The entity may query its DNS name using its IP address. An entity may list more than one `dnsName` and the corresponding IP addresses. IP addresses may be used in environments without DNS services.

The CSR timeout parameter is a local implementation issue, and the device will re-enrol if it does not get a certificate. It is expected to be handled by the security policy of the operator.

The registration data shall be installed and configured individually into each entity to ensure that the RA can authenticate the entity when it performs a CSR.



4 Under preparation. Stage at the time of publication: IEC/RFDIS 62351-3:2023.

Either an activation code (a onetime unique password) or a manufacturer provided certificate shall be supported for the enrolment in an operators PKI infrastructure network, which allows the entity to authenticate itself with the CA. The authentication method used during enrolment will depend on the system (deployed PKI) capabilities.

Note that the entity configuration may be performed manually using a configuration or engineering tool. The configuration may also be supported by onboarding protocols as outlined for security parameters in 5.8.3.

### 7.3.7 Entity enrolment

#### 7.3.7.1 General

Once entities have been configured with the required registration data and have generated their own asymmetric key pair, they shall perform a CSR procedure before becoming operational, based on the organization's certificate security policies. An online connection to the organization's RA/CA is required for certificate enrolment, unless out-of-band enrolment is performed.

Only registered entities shall be allowed to be enrolled at the RA/CA (see 7.3.5).

For manual enrolment entities shall generate the certificate signing request (CSR) using the PKCS #10 ([18]) format. For automated enrolment, the entity shall provide the CSR to the responsible RA specified during configuration. For out-of-band enrolment, the CSR shall be transported using arbitrary means to the responsible RA/CA. The RA shall check the validity of the request by verifying the proof of possession of the corresponding private key by verifying the PKCS #10 (CSR) signature.

If the request is valid, the RA shall send a request to the respective CA. The CA shall generate a public-key certificate and shall provide it to the RA for further distribution to the requesting entity. The protocol utilized for the communication between the RA and the CA is out-of-scope of this specification.

If the request is invalid, the RA shall not send any request to the CA.

NOTE The RA and CA might be deployed together as one entity. The process described above still applies.

For automated enrolment, the infrastructure (RA/CA) shall support the following enrolment protocols for interaction with the end entities (enrolling clients):

- Simple Certificate Enrolment Protocol (SCEP, RFC 8894) for backward compatibility focusing on RSA based public-key certificates.
- Enrolment over Secure Transport, (EST, RFC 7030) for support of RSA or ECC-based public-key certificates as the preferred enrolment protocol.

Using automated enrolment allows for proof of identity by utilizing either the activation code (OTP) or an already available certificate and corresponding private key in conjunction with the registration data on the RA.

A client entity (enrolling entity) may support at least one of the enrolment protocols.

If the enrolment is protected with TLS, it is recommended to utilize the defined TLS profile in IEC 62351-3. If the functional interface of a PKI using TLS to protect the enrolment communication can support IEC 62351-3 it is strongly recommended that it is to be used.

Certificates shall be transported as defined in the enrolment protocols. For certificate storage see 7.3.2.

#### 7.3.7.2 Entity enrolment using SCEP

The application of SCEP shall follow the mandatory requirements described in RFC 8894. Entities supporting SCEP shall support the mandatory to implement functionality as described in section 2.9 of RFC 8894:

- Querying CA capabilities (`GetCACaps`).
- Distribution of CA certificates (`GetCACert`).
- Certificate enrolment (`PKCSReq`).
- Certificate renewal (`RenewalReq`).

If the entity enrolment using SCEP was successfully performed, a security event shall be raised ("notice: enrolment using SCEP successfully performed").

If the entity enrolment using SCEP was cancelled due to an error, a security event shall be raised ("error: enrolment using SCEP cancelled with error."). Detailed information should be provided according to RFC 8894.

#### 7.3.7.3 Entity enrolment using EST

The mandatory required functional support of EST aligns with the mandatory requirements described in RFC 7030 with the addition of mandatory support of certificate attribute retrieval. Entities using EST shall support:

- The distribution of CA certificates using the */cacerts* endpoint to be able to query a CA certificate based on an implicit trust anchor. This functionality allows for the distribution of CA certificates in the initial case to build a trust anchor database, but also the update of the CA certificates
- The *SimplePKIRequest* / *Response* (simple enrol) exchange via the */simpleenroll* endpoint at the RA and may support the *FullPKIRequest* / *Response* exchange via the */fullcmc* endpoint at the RA.
- The re-enrolment (renewal or re-keying of certificates) using the */simplereenroll* endpoint at the RA.

Optionally, the *csrattrs* (certificate attribute request) message may be supported to be able to query CSR certificate attributes to be used in the CSR to allow for explicit requirements from a RA/CA side. This message can be omitted if the enrolling entity has received this information by other means, e.g., by configuration.

CSR attribute request/response handling is required to support the selection of signing algorithms and associated parameters (key length, curve selection for elliptic curve-based algorithms).

Implementors should take notice of IETF RFC 8951 on clarifications for the transfer encodings and ASN.1.

If the entity enrolment using EST was successfully performed, a security event shall be raised ("notice: enrolment using EST successfully performed").

If the entity enrolment using EST was cancelled due to an error, a security event shall be raised ("error: enrolment using EST cancelled with error."). Detailed information should be provided according to RFC 7030.

### 7.3.8 Trust anchor information update

If automated updating of trust anchor information is supported, updating of trust anchor certificates shall be performed using EST (see 7.3.7.3) based on an already existing trust anchor using the */cacerts* endpoint. Trust Anchor Update may optionally be performed using the Trust Anchor Management Protocol (TAMP), RFC 5934.

If TAMP is used, the following TAMP messages shall be supported:

- TAMP Status Query: The TAMP Status Query message is used to request information about the trust anchors that are currently installed in a trust anchor store, and for the list of communities to which the store belongs.
- TAMP Status Query Response: The TAMP Status Response message is a reply by a trust anchor store to a valid TAMP Status Query message. The TAMP Status Response message provides information about the trust anchors that are currently installed in the trust anchor store and the list of communities to which the trust anchor store belongs, if any.
- Trust Anchor Update: The Trust Anchor Update message is used to add, remove, and for change management and identity trust anchors. The Trust Anchor Update message cannot be used to update the apex trust anchor.
- Trust Anchor Update Confirm: The Trust Anchor Update Confirm message is a reply by a trust anchor store to a valid Trust Anchor Update message. The Trust Anchor Update Confirm message provides success and failure information for each of the requested updates.
- Apex Trust Anchor Update: The Apex Trust Anchor Update message replaces the operational public key and, optionally, the contingency public key associated with the apex trust anchor. Each trust anchor store has exactly one apex trust anchor. No constraints are associated with the apex trust anchor. The public key identifier of the operational public key is used to identify the apex trust anchor in subsequent TAMP messages. The digital signature on the Apex Trust Anchor Update message is validated with either the current operational public key or the current contingency public key per the RFC.
- Apex Trust Anchor Update Confirm: The Apex Trust Anchor Update Confirm message is a reply by a trust anchor store to a valid Apex Trust Anchor Update message. The Apex Trust Anchor Update Confirm message provides success or failure information for the apex trust anchor update.
- TAMP Error: The TAMP Error message is a reply by a trust anchor store to any invalid TAMP message. The TAMP Error message provides an indication of the reason for the error.

## 7.4 Certificate components and certificate verification

### 7.4.1 General

Entity certificates, including public-key certificates, attribute certificates, and CA certificates, shall be verified by the relying party. All certificate components shall be followed in conformance with ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019). The following subclauses describe the certificate components and their verification. Based on the verification result of the certificate, specific security events shall be generated in case of an error.

It is recommended to validate self-signed certificates using certificate authorization and validation lists, as described in 7.8.

### 7.4.2 Certificate format and encoding

The formal structure of the certificate shall be verified. For this the relying party shall validate the ASN.1 DER encoding as well as the validity of the ASN.1 certificate structure.

If a certificate cannot be processed due to decoding errors, a security event ("warning: certificate format mismatch. Failed verification.") shall be raised.

### 7.4.3 Certificate signature verification

The certificate signature verification process for public-key certificates and for attribute certificates is similar. In both cases the signature is calculated over the certificate components based on the `signatureAlgorithm` identified by the `AlgorithmIdentifier`. See also 7.4.4.4 regarding signature algorithm support.

If the signature verification of a public-key certificate over the `TBSCertificate` fails, a security event shall be raised ("alarm: public-key certificate signature could not be verified").

If the signature verification of an attribute certificate over the `TBSAttributeCertificate` fails, a security event shall be raised ("alarm: attribute certificate signature could not be verified").

### 7.4.4 Public-key certificate components

#### 7.4.4.1 General

Public-key certificate components are based on the content stated in Table 1.

#### 7.4.4.2 Version

The `version` number of the certificate shall be 2 (indicating X.509v3).

If the certificate version does not match the expected version, a security event ("error: wrong certificate version error.") shall be raised.

#### 7.4.4.3 Issuer

The `issuer` component identifies the CA that issued the certificate. It contains the distinguished name of the issuing CA. The distinguished name of the issuer consists of attributes as the `subject` component (see 7.4.4.6). The `issuer` component is used in the certification path validation process (see 7.4.4.8).

NOTE Annex M of ISO/IEC 9594-8 | Rec. ITU-T X.509 has a short introduction to distinguished names.

If the issuer does not match a known and trusted issuer, a security event ("error: public-key certificate issuer not trusted.") shall be raised.

#### 7.4.4.4 Signature

Implementations claiming conformance to this specification shall support the handling of the following signing algorithms for certificates:

- RSA (see also B.3.2)
- ECDSA (see also B.3.4)

An Implementation may support EdDSA (see also B.3.5). Note that at the time of publication EdDSA is not used in IEC 62351 specification. Optional support may be seen in the context of crypto agility, if EdDSA is considered in IEC 6351 specifications.

Associated parameters are key length and selected elliptic curves and are explained below.

RSA with 2048-bit keys shall be supported. The key length of 2048 is the minimum key length to be supported for RSA signatures. Based on recommendations from NIST SP 800-131A Rev.2 [89] or BSI TR01202-1 [58], it is strongly recommended to also support RSA key length of 3072 bit and higher to address advances in cryptography. The support of RSA with 1024-bit keys is deprecated. Its use is limited to backward compatibility. The selection of the signature algorithm depends on the organization’s security policy.

NOTE 1 Recommendations regarding required key length for signature algorithms are reviewed constantly and can be found in NIST SP 800-57 or BSI TR01202-1.

NOTE 2 Processing long RSA keys is process demanding, which may be a problem for constrained devices. Use of elliptic curve cryptography might then be considered (see also B.3).For ECDSA, the minimum key length is 256 bits (in combination with SHA-256). The OID for `ecdsa-with-SHA256` to be used is: 1.2.840.10045.4.3.2 (iso(1) member-body(2) us(840) ansi-X9-62(10045) signatures(4) ecdsa-with-SHA2(3) 2).

The curve to be used for ECDSA shall be at minimum `secp256r1`. The OID for this curve is: 1.2.840.10045.3.1.7 (iso(1) member-body(2) us(840) ansi-X9-62(10045) curves(3) prime(1) prime256v1(7).)

Optional support for ECDSA using the following curve may be provided:

- `brainpoolP256r1` (as defined in RFC 5639): The OID for this curve is: 1.3.36.3.3.2.8.1.1.7 (iso(1) identified-organization(3) teletrust(36) algorithm(3) signatureAlgorithm(3) ecSign(2) ecStdCurvesAndGeneration(8) ellipticCurve(1) versionOne(1) brainpoolP256r1(7)).

IEC 62351-5:20xx (planned to be released in 2022) requires the support of three further curves:

- Curve 22519 (as defined in RFC 7748 [43])
- Curve 448 (as defined in RFC 7748 [43])
- Secp256k1 (as defined in [60]): The OID for this curve is: 1.3.132.0.10 (iso(1) identified-organization(3) certicom(132) curve(0) ansip256k1(10))

Note that the support of Curve 22519 and Curve 448 at the time of publication of this document is only intended in IEC 62351 in the context of ECDH key agreement (see IEC 62351-5), not in the context of digital signatures. Note also that if used for digital signatures, Curve 22519 and Curve 448 can only be used in EdDSA.

Documents referencing this document may specify additional curves to be supported.

If the specified `signatureAlgorithm` is not supported, a security event ("error: unsupported signature algorithm.") shall be raised.

#### 7.4.4.5 Validity

The validity period of a public-key certificate depends on the organization's security policy for operational certificates (LDevID certificates) and on the manufacturer's security policy for initial device certificates (IDevID certificates). The validity period is determined by `notBefore` and `not After` values. The relying party shall check that the current date and time is between the `notBefore` and `notAfter` values of the certificate.

- If the current time is after the `notAfter` value, a security event ("error: public-key certificate expired.") shall be raised.
- If the current time is before the `notBefore` value, a security event ("error: certificate not yet valid.") shall be raised.

According to ITU-T X.509, the values for the certificate validity (not before and not after) may be specified as `UTCTime` until end of 2049. Starting from 2050 these values shall be specified as `GeneralizedTime`. Note that RFC 5280 requires that the validity shall be specified as `UTCTime` until the end of 2049

NOTE To indicate that a certificate has no well-defined expiration date, RFC 5280 uses the `GeneralizedTime` value of 99991231235959 as the `notAfter` value.

Invalid certificates shall not be accepted by entities, e.g., during the establishment of TLS sessions as required in IEC 62351-3.

Note that access to a clock that is synchronized to Time Atomic International (TAI) or UTC is required in order to assure accurate assessment of the certificate's expiration date. Accurate time and clock synchronization in a network is typically provided by the NTP or PTP protocols, which are defined respectively in RFC 5905 [38] or IEEE 1588 [4]. Note that for NTP, security options are currently revised by the IETF as NTS (Network Time Security, RFC 8915) (see also 7.7). For PTP, IEEE 1588v2.1 defines security options to ensure an integrity protected provisioning of timing information. Requirements for clock synchronization are outlined in 7.7.

#### 7.4.4.6 Subject

The `subject` component holds information about the distinguished name (DN) of the subject. This component shall be supported.

If the `subject` is not contained or contains only unknown attributes, a security event ("warning: subject not included in public-key certificate.") shall be raised.

If checked, the verification of the whole distinguished name or only specific fields shall match allowed identifiers defined by the organization's security policy. Cases in which the verification of the subject may be omitted relate to RBAC as described in IEC 62351-8 Profile A. In this case the subject is only used for logging purposes.

The content of the subject component is expected to be different for operational certificates (LDevID certificates) or manufacturer provided certificates (IDevID certificate). The content of the attributes used in the `subject` for operational certificates are defined by an operator. An implementation shall be able to validate and process at least the following:

- `CN`: common name, unique identifier for the device. Note if a device has more than one certificate, the purpose may be reflected in the CN.
- `O`: organization name to which the device may be assigned (e.g., operator, manufacturer)
- `OU`: organization unit name includes further details of the organization
- `C`: country code (ISO 3166) of the organization responsible
- `serialNumber`= serial number of the entity (not to be confused with the certificate serial number) as defined in IEEE 802.1AR (as `X520SerialNumber`)

Additional attributes may be specified, e.g., by a manufacturer for a manufacturer certificate (IDevID certificate) containing specific device information, e.g., product type information.

#### 7.4.4.7 Subject Public Key Info

The `subjectPublicKeyInfo` contains two subcomponents providing information about the algorithm (OID) as well as the public key of the certificate.

The failure of finding a matching algorithm for the public key shall raise a security event ("error: native public key algorithm in public-key certificate not supported").

#### 7.4.4.8 Unique identifiers

The `issuerUniqueIdentifier` and `subjectUniqueIdentifier` components shall be absent. These components are deprecated in X.509.

#### 7.4.4.9 Certification path validation

The relying party shall validate the certification path starting with the CA certificate issued by the trust anchor and ending with the end-entity public-key certificates. As a special case, the end-entity public-key certificate may be issued directly by the trust anchor (typically referred to as root CA certificate).

The failure of finding the corresponding trust anchor to a certificate in the local configuration shall raise a security event ("error: trust anchor not supported.").

NOTE Often the intermediate certificates are transferred inside protocol messages like in TLS. Nevertheless, the relying party shall be able to find root certificates contained in the local configuration.

All certificates shall be properly signed by the respective issuing CA, while the root certificate has to be self-signed. The `issuer` and the `authorityKeyIdentifier` have to be equal to the subject name and the `subjectKeyIdentifier`, respectively, of the next certificate in the path.

The failure of validating the certificates along the certification path shall raise a security event ("error: certification path could not be verified.").

#### 7.4.4.10 Certificate Extensions

##### 7.4.4.10.1 General

All extensions shall be followed in conformance to ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019):

- If the extension is flagged as critical, then the public-key certificate shall be used only for one of the purposes indicated.
- If the extension is flagged as non-critical, then it indicates the intended purpose or purposes of the key. If this extension is present, and the relying party recognizes and processes the extension type, then the relying party shall ensure that the public-key certificate shall be used only for one of the purposes indicated. If the extension is non-critical and not known, it may be neglected.

Moreover, certificates including unknown critical extension or critical extensions containing information that cannot be processed shall be rejected.

If the certificate contains an unrecognized extension marked as critical, a security event ("error: unknown critical extension in public-key certificate.") shall be raised.

If the certificate contains unrecognized information in an extension marked as critical, a security event ("error: unknown information in critical extension.") shall be raised.

##### 7.4.4.10.2 Authority Key Identifier

The `authorityKeyIdentifier` contains the `subjectKeyIdentifier` of the issuing CA certificate.

If the `authorityKeyIdentifier` is not contained in the certificate, a security event ("error: authority key identifier not included in public-key certificate.") shall be raised.

##### 7.4.4.10.3 Subject Key Identifier

The `subjectKeyIdentifier` extension allows the identification of certificates that contain a particular public key. According to RFC 5280 it is mandatory for issuing CA certificates and may be optionally supported in end entity certificates.

If the `subjectKeyIdentifier` is not contained in a CA certificate, a security event ("error: subject key identifier not included in public-key certificate.") shall be raised.

##### 7.4.4.10.4 Subject Alternative Name

If the `subjectAltName` extension is available in the certificate, it shall be processed. It may contain one or multiple alternative names for the certificate subject.

For TLS server certificates at least one `subjectAltName` is required to match the dNSName (e.g. FQDN) or IPAddress of the TLS server presenting the certificate.

If the certificate authenticating a TLS server does not contain a `subjectAltName`, a security event ("error: subject alternative name not included.") shall be raised.

#### 7.4.4.10.5 Basic Constraints

The `basicConstraints` extension identifies whether the certificate is a CA certificate. If the certificate is a CA certificate, the extension shall be marked as critical. For end entities it is recommended to omit the extension.

For issuing CAs two components shall be provided:

- `cA`: shall be set to true. Note that this also requires to key usage to be set to `keyCertSign`: see 7.4.4.10.6.
- `pathLenConstraint` may be used to limit the length of the certification path, depending on the organization's security policy.

If the `basicConstraints` extension is not provided in a CA certificate, a security event ("error: basic constraints not included in CA certificate.") shall be raised.

If the `pathLenConstraint` component is provided and used in a CA certificate along the certification path and is set to zero, the following certificate shall be an end-entity certificate. If the following certificate is not an end-entity certificate, a security event ("error: path length constraints in CA certificate violated.") shall be raised.

Implementations conforming to this document shall support a minimum `pathLenConstraint` of 2, allowing two intermediate CAs.

#### 7.4.4.10.6 Key usage

The `keyUsage` extension is a critical extension and shall be verified. It identifies the intended usage of the public-key certificate, like `digitalSignature`, `keyAgreement` or `keyEncipherment`.

Key usage in the context of TLS:

- `digitalSignature`
  - shall be set for TLS client certificates and,
  - shall be set for TLS server certificates if signature-based cipher suites are used

  If the certificate authenticating a TLS client or a TLS server is used in conjunction with a cipher suite requiring digital signatures in the TLS handshake for key agreement does not contain the key usage for `digitalSignature`, a security event ("error: key usage for digital signatures not included.") shall be raised.
- `keyEncipherment` shall be set for TLS server certificates if key transport-based cipher suites are used

  If the certificate authenticating a TLS server is used in conjunction with a cipher suite using public-key encryption in the TLS handshake for key agreement does not contain the key usage for `keyEncipherment`, a security event ("error: key usage for key encipherment not included.") shall be raised.

Key usage in the context of PKI operation:

- `keyCertSign` shall be set for issuing certificates for an issuing CA. If the issuing certificate does not contain `keyCertSign` information, a security event ("error: key usage not included for signing certificates.") shall be raised.

- `cRLSign` shall be set for CRL signing. If the certificate used to sign a CRL does not contain `cRLSign` usage information, a security event ("error: key usage not included for signing CRLs.") shall be raised.
- `digitalSignature` shall be set for issuing AAs, if the AA certificated was issued by a CA. It shall also be set for CA certificates used in the context of SCEP as in the context of SCEP the response messages are signed by the SCEP CA.

#### 7.4.4.10.7 Extended key usage

If available, the `extendedKeyUsage` extension shall be verified. It identifies additional purposes for the usage of the public-key certificate (support of this component is optional, depending on use case).

Extended key usage in the context of TLS certificates:

- `serverAuth` shall be set for TLS server certificates. If the `serverAuth` information is not contained in a TLS server certificate, a security event ("error: extended key usage not included for TLS server.") shall be raised.
- `clientAuth` shall be set for TLS client certificates. If `clientAuth` information is not contained in a TLS client certificate, a security event ("error: extended key usage not included for TLS client.") shall be raised.

Extended key usage in the context of PKI operation:

- `OCSPSigning` shall be set for signing OCSP responses (1.3.6.1.5.5.7.3.9 – OCSPSigning). If the `OCSPSigning` information is not contained in a certificate used to sign OCSP responses, a security event ("error: extended key usage not included for OCSP signing.") shall be raised.

Extended key usage in the context of system operation:

- AVL Signing (`avlSign`): If set, the corresponding private key may be used to sign AVLs. The authorizer uses its private key for signing an AVL to be submitted to an AVL entity. The corresponding public-key certificate shall include the extended key usage extension with the value id-`avlSign` (= 2.5.38.2). This extended key usage extension and the `id-avlSign` value is defined in 9.2.2.4 of ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019). If `id-avlSign` information is not contained in certificate used to sign an AVL, a security event ("error: extended key usage not included for signing an AVL.") shall be raised.

#### 7.4.4.10.8 Authorization and validation extension

The authorization and validation extension is specified in 9.2.2.8 of ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019). The presence of this extension in a public-key certificate signals that this public-key certificate shall only be considered valid if it can be successfully checked against a particular AVL. The handling of an AVL is in the responsibility of an operator.

If used, this extension shall always be flagged as critical.

This extension should only be used if all entities having to validate this public-key certificate are managed by the same authorizer (this limitation is not mentioned in ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019)).

#### 7.4.4.10.9 CRL Distribution Point

The `cRLDistributionPoints` extension identifies the CRL distribution point under which the relying party can access the CRL distribution list. The retrieval of the CRL is specified by the protocol stated in the CRL distribution list and may be HTTP or LDAP in the context of this specification.

Issuing CAs conforming to this document must provide CRL distribution point information in this extension. End entity implementations conforming to this document supporting CRLs shall be able to utilize the contained information to fetch the revocation information from a CRL distribution point.

Implementations (issuing CA and end-entity supporting CRLs) conforming to this document shall support the retrieval of CRLs using HTTP as transport protocol. The default method is HTTP GET.

Note that optional LDAP may be used in environments already applying LDAP for certificate related operations. This may be the case, when RBAC using the PULL model for Profile A or Profile B as defined in IEC 62351-8 is supported. If CRL fetching is supported by the utilized enrolment protocol and the operators PKI, it may also be used.

NOTE The CRL itself is self-contained (digitally signed) and can be transmitted independent of transport security.

If CRLs are used for certificate revocation checks by the relying party and if the CRL distribution point (CDP) information is not contained in the received certificate, a security event ("warning: CRL distribution point not contained in public-key certificate") shall be provided.

#### 7.4.4.10.10 Authority Information Access

The `authorityInformationAccess` extension indicates how to access information and services of the issuer of the certificate. Specifically, this extension may contain access information for an OCSP responder responsible for providing information about the revocation state of the certificate containing the extension. In the context of this specification the support information for the OCSP responder is required as specified in RFC 5280.

Issuing CAs conforming to this document must provide information about the OCSP distribution point indicated by the access method `id-ad-ocsp`.

End entity implementations conforming to this document supporting OCSP shall be able to utilize the contained information to fetch the revocation information from a OCSP responder using OCSP.

If the OCSP responder information is not contained in the received certificate, a security event ("warning: OCSP responder information not contained in public-key certificate") shall be provided.

#### 7.4.4.10.11 RBAC extension

IEC 62351-8 defines the extension `IECUserRoles` for carrying information related to the subject concerning the role the subject may act in as well as additional attributes providing more information about using the role, like the area of responsibility. See IEC 62351-8 Profile A for the definition of the extension as well as the validation.

#### 7.4.4.10.12 AVL distribution point extension

The use of the public-key certificate extension defined in Edition 1 is deprecated. AVLs are a local issue concerning the operator and may be unknown to the issuing CA. Therefore, the handling of an AVL distribution point is a local implementation issue and may be done for instance through configuration. The protocols for managing AVLs are given in ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019).

#### 7.4.4.10.13 SOA identifier extension

The `sOAIdentifier` extension is described in in ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019) and indicates that the public-key certificate subject may act as source of authority (SOA). As such, the public-key certificate subject may use the private key to issue attribute certificates that assign privileges to holders.

Setting this extension is in the responsibility of the issuing CA. If the CA issues AA certificates, this extension shall be included. It allows to have a close binding between the CA and the AA.

In addition to the `sOAIdentifier` extension, an AA also needs the key usage `digitalSignature` assigned to issue attribute certificates.

In the context of this specification, only a CA is allowed to issue an AA certificate including the SOA identifier.

### 7.4.5 Attribute certificate components

#### 7.4.5.1 General

Attribute certificate components are based on the content stated in Table 2 in 7.2.2. If an attribute certificate or one of its components cannot be verified, the holder shall be treated as having no specific role(s).

#### 7.4.5.2 Version

The `version` component of the attribute certificate shall be 1 (indicating v2).

If the certificate version does not match the expected version a security event ("error: wrong attribute certificate version.") shall be raised.

#### 7.4.5.3 Holder

The `holder` component shall convey the identity of the holder of the attribute certificate. According to ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019) different components are possible here. Implementations conforming to this document shall support the evaluation of:

- the `baseCertificateID` component. If present, it identifies a particular public-key certificate that is used to authenticate the identity of the holder when asserting privileges with this attribute certificate based on the distinguished name of the `issuer` and the `serialnumber` of the holder's public-key certificate. Following RFC 5755, the public-key certificate shall have a non-empty issuer distinguished name, which has to be present in the `directoryName` field of the `holder.baseCertificateID.issuer` component.
- the `entityName` component. If present the holder authentication is done by other means than a public-key certificate.

NOTE RFC 5755 also allows the identification of the holder of a public-key certificate based on the `subject` or `subjectAltName` to be contained in the `entityName`. The approach taken in this document is more stringent and simplifies the distinction between public-key certificate-based holder authentication and other types of holder authentication.

- Attributes for the `entityName` are defined by an operator. Note that according to IEC 62351-8, the holder may relate to a human user or a technical user (e.g., an application). An implementation shall be prepared to receive and process at least one of the following attributes in the `entityName`:
  - `otherName`: alternative holder name of any form identified by an OID and a corresponding value
  - `directoryName`: distinguished name of the holder according to ITU-T X.501 | ISO/IEC 9594-2;

Following the recommendation provided in RFC 5755, this specification requires that only one of the components shall be included in an attribute certificate. Also, in case of using the `entityName`, only one attribute for this component shall be provided. This is to avoid confusion, which component is handled as normative.

If the holder authentication is done based on a corresponding public-key certificate, the certificate shall be verified as described in 7.4.4.

If the holder could not be verified based on a corresponding public-key certificate a security event ("warning: holder of attribute certificate could not be verified based on public-key certificate.") shall be raised. In this case the holder shall be treated as having no specific role(s).

If the holder authentication is done by other means, the authentication process shall have been successfully finished before verifying the attribute certificate.

If the holder could not be verified based on an alternative, non-public-key certificate-based authentication, a security event ("warning: holder of attribute certificate could not be verified based on alternative authentication.") shall be raised. In this case the holder shall be treated as having no specific role(s).

#### 7.4.5.4 Issuer

The `issuer` component identifies the AA that issued the attribute certificate. It contains the distinguished name of the issuing AA. Following RFC 5755, it shall only contain a single distinguished name in the `directoryName` field of the `issuer` component.

According to RFC 5755, an AA may be trusted by configuration or by other means.

If the issuer does not match a configured known and trusted issuer, a security event ("error: attribute certificate issuer not trusted (not configured).") shall be raised.

In the context of this specification "other means" is provided by a close integration of the issuing AA and the issuing CA of an operator. The CA in this case may issue AA certificates. The verifier of an attribute certificate shall verify that the issuing AA certificate

- contains the `sOAIdentifier` (see also 7.4.4.10.13)
- contains the key usage `digitalSignature` (see also 7.4.4.10.6)
- was issued by a trusted CA.

If the AA cannot be verified as trusted issuer based on the verification of extensions and the issuer of the AA certificate, a security event ("error: attribute certificate issuer not trusted (by a trusted CA).") shall be raised.

#### 7.4.5.5 Signature

The `signature` contains the algorithm identifier used to validate the attribute certificate signature. For attribute certificates the same conditions hold as for public-key certificates as outlined in 7.4.4.4.

#### 7.4.5.6 Attribute

The `attributes` component contains the attributes associated with the holder.

For the power system, IEC 62351-8 defines the attribute value and type for `IECUserRoles` for carrying information related to the subject concerning the role the subject may act in as well as additional attributes providing more information about using the role, like the area of responsibility. See IEC 62351-8 Profile B for the definition of the attribute as well as the validation.

#### 7.4.5.7 attrCertValidityPeriod

The validity period of an attribute certificate depends on the organization's security policy. The validity period is determined by *not before* and *not after* values. The relying party shall check that the current date and time is between the `notBeforeTime` and `notAfterTime` values of the certificate.

- If the current time is after the `notAfterTime` value, a security event ("error: attribute certificate expired.") shall be raised.
- If the current time is before the `notBefore` value, a security event ("error: attribute certificate not yet valid.") shall be raised.

In contrast to RFC 5280, RFC 5775 for attribute certificates requires that the values for `not before` and `not after` shall always be specified as `GeneralizedTime`.

NOTE To indicate that a certificate has no well-defined expiration date, RFC 5280 uses the `GeneralizedTime` value of 99991231235959 as the `notAfter` value.

#### 7.4.5.8 Attribute certificate extensions

##### 7.4.5.8.1 General

Extensions defined for attribute certificates ACs provide methods for associating additional attributes with holders.

If the attribute certificate contains an unrecognized extension marked as critical, a security event ("error: unknown critical extension in attribute certificate.") shall be raised.

If the certificate contains unrecognized information in an extension marked as critical, a security event ("error: unrecognized information in critical extension of attribute certificate.") shall be raised.

If the certificate contains an unrecognized extension, a security event ("warning: unknown extension in attribute certificate.") shall be raised.

##### 7.4.5.8.2 Authority Key Identifier

The `authorityKeyIdentifier` may be included in an attribute certificate and is intended to aid the attribute certificate verifier in checking the signature of the attribute certificate by an Issuing AA.

If the `authorityKeyIdentifier` is contained in the attribute certificate but cannot be verified, a security event ("error: authority key identifier in attribute certificate could not be verified. ") shall be raised.

##### 7.4.5.8.3 Attribute certificate revocation handling

Attribute certificates are intended as a temporary enhancement of attributes of a holder. They may be issued with a short validity period.

If the organization's security policy allows for a short validity period not requiring a revocation, this shall be indicated by using the `noRevAvail` (no revocation available) extension. In this case the CRL distribution point and the Authority Information Access extensions shall not be set.

If the `noRevAvail` information is contained in the received attribute certificate, a security event ("notice: no revocation information intended in attribute certificate") shall be provided.

The same approach as for public-key certificates is followed for the handling of the CRL distribution point extension (see 7.4.4.10.9) and the Authority Information Access extension (see 7.4.4.10.10) but referring to the AA instead of the CA.

If the organization's security policy requires revocation information, the issuing AA conforming to this document shall provide CRL distribution point information in the `cRLDistributionPoints` extension and information about the OCSP distribution point in the `authorityInformationAccess` extension.

End entity implementations conforming to this document supporting CRLs shall be able to utilize the contained information to fetch the revocation information from a CRL distribution point.

Implementations (issuing CA and end-entity supporting CRLs) conforming to this document shall support the retrieval of CRLs using HTTP as transport protocol. The default method is HTTP GET.

Note that optional LDAP may be used in environments already applying LDAP for certificate-related operations. This may be the case when RBAC using the PULL model for Profile A or Profile B as defined in IEC 62351-8 is supported.

NOTE The CRL itself is self-contained (digitally signed) and can be transmitted independent of transport security.

If CRLs are used for certificate revocation checks by the relying party and if the CRL distribution point (CRLDP) information is not contained in the received attribute-certificate, a security event ("warning: distribution point not contained in attribute certificate") shall be provided.

End entity implementations conforming to this document supporting OCSP shall be able to utilize the contained information to fetch the revocation information from a OCSP responder using OCSP.

If the OCSP responder information is not contained in the received attribute certificate, a security event ("warning: OCSP responder information not contained in attribute certificate") shall be provided.



### 7.4.6 Certificate revocation status

An implementation claiming conformance to this document shall be capable of checking the revocation state of a received certificate. If the certificate revocation check was positive, a security event ("alarm: certificate has been revoked.") shall be raised. If available, the revocation reason may be added in the security event. Possible reason codes as defined in ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019) are:

- `unspecified (0),`
- `keyCompromise (1),`
- `cACompromise (2),`
- `affiliationChanged (3),`
- `superseded (4),`
- `cessationOfOperation (5),`

- `certificateHold (6),`
- `removeFromCRL (8),`
- `privilegeWithdrawn (9),`
- `aACompromise (10),`
- `weakAlgorithmOrKey (11)`

Provisioning of the CRL is a local matter and may be done either locally (file-based) or depending on the URI information in the CRL Distribution Point component in the certificate and may utilize HTTP or LDAP as stated in 7.4.4.10.9.

The unavailability of a CRL shall raise a security event ("warning: CRL distribution point not accessible").

The expiry of a CRL shall raise a security event ("warning: CRL expired").

The failure of validating the signature of a CRL shall raise a security event ("warning: CRL signature could not be verified").

Alternatively, an entity may use the Online Certificate Status Protocol (OCSP) to verify the revocation state of a certificate.

If OCSP is used to acquire an OCSP response for a received certificate in order to perform a certificate revocation check, the relying entity shall use the URI found in the `accessLocation` field of the `id-ad-ocsp accessMethod` of the Authority Info Access extension in the peer certificate to query the OCSP responder. Different error cases may occur:

- The inaccessibility of the OCSP responder from an entity shall raise a security event: ("warning: OCSP responder not accessible").
- A response timeout for a connection from the entity to an OCSP responder shall raise a security event: ("warning: OCSP responder connection timeout"). Note that this may relate to a normal TCP/IP time out or could be caused by a denial-of-service attack.
- If the OCSP responder does not know the certificate in the request messages, it will respond with the certificate state value "unknown". This condition shall raise a security event ("warning: Certificate not known to OCSP responder"). Note that this may indicate an unrecognized issuer that is not served by this responder.

The expiry of an OCSP response on an entity shall raise a security event ("warning: OCSP response expired").

The failure of validating the signature of an OCSP response message shall raise a security event ("warning: OCSP response signature could not be verified").

As stated in 7.4 an organization's security policy determines the final reaction to a security event. Revoked certificates shall not be accepted by entities, e.g., like during the establishment of TLS session as required in IEC 62351-3.

### 7.5 Certificate revocation

An entity's certificate shall be revoked at a minimum under the following conditions, using the reason codes specified in 9.5.3.1 of ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019)..

- Entity's private key has been compromised.
- CA has been compromised.
- Entity's affiliation has changed.
- Entity's end of life.

Additional reasons for the revocation of a certificate may be provided by the revocation policy of the organization. It is not required to revoke certificates that have been renewed or that have expired.

Provisioning of revocation information is a requirement for the PKI environment and not of a specific entity.

The public key infrastructure (PKI) shall support at a minimum the two following secure methods for revocation:

- Certificate Revocation Lists (CRL)
- Online Certificate Status Protocol (OCSP)

The PKI shall propagate revocation information, at a minimum, every 24 hours (this may vary depending on, organization's PKI policy, criticality of particular devices and number of peer connections).

Entities shall support either CRL or OCSP or both to fetch revocation information. The revocation information is required to perform certificate verification checks.

Note, that there is a trade-off regarding the selection of the revocation method (CRL or OCSP). The choice depends on the target device and on the target environment.

- CRLs can grow large and may not be able to be processed by constrained IEDs. The processing includes the initial verification of the CRL but also the storage of the information until the next CRL update. A CRL is typically fetched every 24 hours. Manufacturers may declare a maximum size of a CRL supported.
- IEDs supporting OCSP are required to process the OSCP responses (which includes verification and caching) per certificate. This may require a more frequent communication with an OCSP responder to fetch OCSP responses for received certificates. Note that the caching of OCSP responses (handled by an organization's security policy) may decrease communication with the OCSP responder, but may increase local storage needs for OCSP response caching.

NOTE System behaviour in case of unavailability of CRLs, CRL updates, or of OCSP responders is expected to be either defined in another part of IEC 62351 or as part of the organization's security policy.

### 7.6 Certificate expiration and renewal

This document imposes neither a minimum nor maximum public-key certificate life span. A certificate expiration date should be chosen depending on the certificate type and on the local security policies (see 7.3.4).

Public-key certificates may optionally include a private key usage extension, which specifies the period during which the corresponding private key may be used by its owner. This period is normally set to be shorter than the certificate validity period, ensuring that certificates remain valid for a minimum period after use by their owner. Details on the use of the private key usage extension are covered in 9.2.2.5 of ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019).

Entities shall generate a new key pair and perform a CSR in environments featuring a PKI after their public-key certificates expiration dates gets closer to the certificate validity end date. Note that the handling is typically specified by the organization's certificate policies. Entities shall renew their public-key certificates before they expire and shall log their public-key certificate renewal actions (as successful or failed events).

Entities shall allow the public-key certificate renewal policy to be configured, such as:

- Auto-renewal is supported or not
- The length of time before expiration that public-key certificate renewal shall take place.

### 7.7 Clock Synchronization and Accuracy

Clock synchronization and accuracy shall be ensured to allow for reliable verification of certificate validity information.

Time synchronization is therefore crucial, and it is strongly recommended to implement the security options for time synchronization protocols:

- If PTP (IEEE 1588v2.1) is used for time synchronization the integrated security option should be used. Note that for IEC 61850-based substation automation a PTP profile has been defined in IEC 61850-9-3/IEEEc37.248. Note also that security will be included in the next revision of that profile.
- If NTP is used for time synchronization, the application of NTS (Network Time Security, RFC 8915) should be considered.
- Alternatively, other security protocols may be used to secure time synchronization.

### 7.8 Authorization and validation lists

#### 7.8.1 General

Support of AVLs and associated protocols is optional, but if used shall meet the requirements in ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019). and ISO/IEC 9594-11:2020 | Rec. ITU-T X.510 (2020) as follows:

- Clause 11 of ISO/IEC 9594-08:2020 | Rec. ITU-T X.509 (2019) specifies AVLs.
- Clauses 8 to 12 of ISO/IEC 9594-11:2020 | Rec. ITU-T X.510 (2020) specifies a wrapper protocol that provides security for other protocols.
- Clause 13 of ISO/IEC 9594-11:2020 | Rec. ITU-T X.510 (2020) specifies the authorization and validation management protocol (AVMP), which is used for communication between an authorizer and the AVL entities its support. It utilizes the services of the wrapper protocol.
- Clause 14 of ISO/IEC 9594-11:2020 | Rec. ITU-T X.510 (2020) specifies the CA subscription protocol (CASP). This protocol is used by an authorizer to subscribe to public-key certificate status information from relevant CAs. This protocol is only used by the authorizer for constrained environments (see 5.11.3). This protocol may also be used by end entities as an alternative way of getting public-key status information. This protocol requires that CAs are updated to support the CASP protocol and associated capabilities.

#### 7.8.2 Syntax for authorization and validation list (AVL) for public-key certificates

The different components of the `TBSCertAVL` data type are described in Clause 11 of ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019). Only the specific AVL aspects relevant to power systems are described below:

- The `serialNumber` component is provided to allow more advanced functionality by allowing an authorizer to place more than one AVL in an AVL entity. It is also possible that several authorizers place one or more AVLs in the same AVL entity.
- The `constrained` component shall take the value **FALSE** to indicate that AVL entity is a non-constrained AVL entity. This value should be set until such time that CAs support public-key certificate status subscription.
- The `entries` component shall hold an element for each public-key certificate and/or entity group represented by the AVL. Each element shall be specified as follows:

The `idType` component shall take one of two alternatives:

- If the `certIdentifier` alternative is taken, it shall identify the particular public-key certificate represented by this entry. It may be done by specifying the CA issuer name together with the serial number of the public-key certificate or it may be identified by a fingerprint of the public-key certificate or just the public key. If the public-key certificate is a self-signed certificate, only a fingerprint may be used for identifying the public-key certificate.
- If the `entityGroup` alternative is taken, it shall hold the part of a distinguished name that is common for all the entities in the group. This alternative is only relevant for a non-constrained environment.

Further extensions to the AVL are:

- The `entryExtensions` component, when present, shall hold one or more extensions specific for the entry in question. The extensions specified in 7.8.3 to 7.8.6 may be included here. If a particular extension type is used as an entry extension, it shall not be included in the `avlExtensions` component.
- The `avlExtensions` component, when present, shall hold one or more extensions applicable for entries in the AVL. The extensions specified in 7.8.3 to 7.8.6 may be included here. If a particular extension type is included here, it shall not be present in the **e**`ntryExtension` component of any entry.

Note that extensions may be added to both the individual entries and the AVL as a whole. Such extensions are specified in 7.8.3 to 7.8.6

### 7.8.3 AVL scope restriction

The AVL scope is intended to limit the applicability of public-key certificates. In certain scenarios it may be desired also to include the scope into a public-key certificate as an optional extension. The actual scope constraints are defined in a separate type. This allows using the constraint also separately.

The **`scopeConstraints`** extension syntax is defined as:

```
scopeConstraints EXTENSION::= {
  SYNTAX        ScopeConstraints
  IDENTIFIED BY { avl62351Extion 1 } }

ScopeConstraints ::= SEQUENCE Of (SIZE (1..MAX)) OF ScopeConstraint

ScopeConstraint::= SEQUENCE { - contains the scope information
  aor         UTF8String (SIZE(1..64)),
  --Def. of "Area of Responsibility" of CA cert
  revision  INTEGER (0..255) OPTIONAL
  - optional revision if aor changes
}
```

The area of responsibility (**`aor`**) restricts the applicability of a public-key certificate to a certain geographical or organizational area. This document defines the field and the format for the **`aor`** as follows:

| Field name | Coding, max length (byte) | Example |
|---|---|---|
| Area of responsibility | UTF8, 64 | BAVARIA.DE |

The **`aor`** is an identifier and defines a hierarchical name space or a reference to the namespace. Note that these identifiers are typically alphanumeric. The **`aor`** would be provided per policy, e.g., from the responsible operator.

NOTE The notion of **aor** (or scope) to describe a geographical or organizational restriction has already been introduced in IEC 62351-8 and is being reused here.

### 7.8.4 AVL protocol restriction extension

The AVL protocol restriction entry extension is defined in 9.7.2 of ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019). It is used to enumerate associated protocols to be allowed to be used when communication with the entity (or entities in case of **entityGroup**). If this AVL entry extension is omitted, there are no restrictions with respect to the types of protocols to be employed.

A protocol shall be identified by the communication standard specifying that protocol. The object identifier shall follow the principle specified in A.2 and A.4 of ISO/IEC 9834-1 | Rec. ITU-T X.660.

As an example, IEC 61850-8-1 is represented by the object identifier:

{ iso(1) standard(0) iec61850(61850) series(8) part(1) }

NOTE ISO/IEC 9834-1 | Rec. ITU-T X.660 does not consider the case where the standard number consists of three items. The example shows the notation used by this part of IEC 62351.

The use of the object identifiers below, defined in Edition 1 of this document, is deprecated. For backward compatibility, implementations should be able to handle these object identifiers.

```
id-P-IEC61850-T    OBJECT IDENTIFIER::= { id-IEC62351prot 1 }
id-P-IEC61850-A    OBJECT IDENTIFIER::= { id-IEC62351prot 2 }
id-P-IEC60870-5-T  OBJECT IDENTIFIER::= { id-IEC62351prot 3 }
id-P-IEC60870-5-A  OBJECT IDENTIFIER::= { id-IEC62351prot 4 }
id-P-IEC62325-T    OBJECT IDENTIFIER::= { id-IEC62351prot 5 }
id-P-IEC62325-A    OBJECT IDENTIFIER::= { id-IEC62351prot 6 }
id-P-IEEE1518-T    OBJECT IDENTIFIER::= { id-IEC62351prot 7 }
id-P-IEEE1518-A    OBJECT IDENTIFIER::= { id-IEC62351prot 8 }
```

### 7.8.5 AVL pinning of certificate and associated identifier

This AVL **pinningId** entry extension provides an association of a distinct identifier to a public-key certificate. This identifier is most likely the IP address, but may also be another identifier. There are use cases in which a public-key certificate is only to be used on a dedicated IP address. The **pinningId** extension supports the association of a certificate with the IP address (or another identifier), if the certificate itself does not provide IP address information as part of the **subjectAltName** extension.

```
pinningId EXTENSION::= {
  SYNTAX        PinningId
  IDENTIFIED BY { avl62351Extion 2 } }

PinningId::= GeneralNames

GeneralNames::= SEQUENCE SIZE (1..MAX) OF GeneralName

GeneralName::= CHOICE {
  otherName                   [0]  OtherName,
  rfc822Name                  [1]  IA5String,
  dNSName                     [2]  IA5String,
  x400Address                 [3]  ORAddress,
  directoryName               [4]  Name,
  ediPartyName                [5]  EDIPartyName,
  uniformResourceIdentifier   [6]  IA5String,
  iPAddress                   [7]  OCTET STRING,
  registeredID                [8]  OBJECT IDENTIFIER,
  ... }
```

The **GeneralNames** data type is defined and the alternatives of **GeneralName** are described in 9.3.2.1 of ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019). The **GeneralNames** data type is copied here for easy reference.

The `iPAddress` shall be stored in the octet string in "network byte order", as specified in RFC 791. The least significant bit (LSB) of each octet is the LSB of the corresponding byte in the network address. For IP version 4, as specified in RFC 791, the octet string shall contain exactly four octets For IP version 6, as specified in RFC 2460, the octet string shall contain exactly sixteen octets.

### 7.8.6 Public-key certificate extensions related to use of AVLs

Especially if AVLs are used in a PKI context, certain public-key certificates extensions are recommended to be used. Note that if self-signed certificates are used, the inclusion of these extensions may not be enforceable in practice. The public-key certificate extensions relevant for AVLs are described in 7.4, namely:

- Authorization and validation extension in 7.4.4.10.8
- Extended key usage for issuing AVLs in 7.4.4.10.7

Note that the public-key certificate extension defined in Edition 1 of IEC 62351-9 (2017) for an AVL distribution point has been deprecated as outlined in 7.4.4.10.12. The AVL distribution point is handled by the operator and may be unknown to the issuing CA. If AVLs are used, this information may be provided through configuration.

### 7.8.7 Issuing of an AVL

Issuing an AVL is the responsibility of the operator (authorizer) of a dedicated power system. The AVL is expected to be compiled based on the engineering data for a specific system deployment. Based on the engineering data the communication relations are known as well as the protocols used between the communicating entities. The AVL may be compiled at the same time, assumed, that entity specific public-key certificates are already available. If entity specific components are not available at engineering time, the CA issuing the certificates may be configured with the engineering information and provide this information to the entities during the enrolment.

Note that an AVL needs to be updated if the communication peers in the system change. This may be the case if components of the system are removed or exchanged or newly introduced. It is assumed that at least the removal or introduction of components is accompanied by engineering.

Clause 21 of ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019) specifies procedures for management of AVLs.

### 7.8.8 Endpoint Handling of AVLs

The process of verifying a public-key certificate received during a connection establishment (e.g., during the TLS handshake) and the validation of the public-key certificate itself and the check against a locally available AVL is outlined in ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019).

## 8 Group based key management (normative)

### 8.1 GDOI requirements

An informative overview of GDOI is provided in 5.6.4.2.

The Group Domain of Interpretation (GDOI) method, RFC 6407) and RFC 8052 for "GDOI Protocol Support for IEC 62351 Security Services" shall be used for the distribution of group keys to Group Members (GM).

The GROUPKEY-PULL method is mandatory to be supported while GROUPKEY-PUSH is optional.

GDOI defines the application of GROUPKEY-PUSH to send control information to the group members. This relates for instance to the rekeying existing security association (SA) but also to the change in members of a group. To be able to acknowledge the received information from a GCKS, the GROUPKEY-PUSH acknowledgement message has been specified in RFC 8263. The capability to use the GDOI client public-key certificate credentials exchanged at connection establishment time for group member authentication is mandatory.

The KDC shall maintain the repository of the keys and associated parameters in a secure location.

The KDC shall configure the policy for renewing the session key, which shall be based on key lifetime.

Note whenever public key is mentioned (in GDOI), it is meant as part of the certificate. It has a corresponding private key, which is employed to generate signatures.

### 8.2 Internet Key Exchange Version 1 (IKEv1)

GDOI RFC 6407 provides an ISAKMP implementation where GDOI is a new ISAKMP Domain of Interpretation (DOI). RFC 6407 defines how IKEv1 (RFC 2409) is used as the phase 1 protocol for GDOI. The KDC shall use IKEv1 for its GDOI Phase 1 protocol. The KDC does not need to support all of the functionality and features of IKEv1, but at a minimum shall support the IKEv1 requirements listed in Table 3. Note that additional authentication algorithms allowing the application of ECDSA for IKEv1 and IKEv2 are defined in RFC 4754 [32].

**Table 3 – KDC IKEv1 Requirements**

| Description | Values |
|---|---|
| ISAKMP Exchanges Supported | 2 – Main Mode (ID Protection)<br>5 – Informational |
| GM ID Payload Types Supported | 9 – ID_DER_ASN1_DN (Subject ID of the GM's Identity Certificate) |
| Key Exchange Supported | Diffie-Hellman via Group Description Attribute (see below) |
| Server IP Port | UDP 848 (configurable) |
| 1 – Encryption Algorithm Attribute | 7 – AES-CBC (Key Lengths: 128/256) |
| 2 – Hash Algorithm Attribute | 4 – SHA2-256<br>5 – SHA2-384<br>6 – SHA2-512 |
| 3 – Authentication Method Attribute | 2 – DSA Signatures<br>3 – RSA Signatures (Mandatory)<br>9 – ECDSA with SHA-256 on the P-256 curve (RFC 4754) |
| 4 – Group Description Attribute | 14 – MODP-2048<br>15 – MODP-3072<br>16 – MODP-4096<br>17 – MODP 6144 (RFC 3526)<br>18 – MODP-8192 (RFC 3526) |
| 11 – Life Type (optional) | 1 – Seconds |
| 12 – Life Duration (optional) | 120:86400 Seconds (default – 120) |
| 14 – Key Length | AES-CBC (Key Lengths: 128/256) |

NOTE MODP-1024 and MODP-1538, which were stated in version IEC 62351-9:2017 have been deprecated according to recommendations from NIST and the German BSI. Implementations conforming to this specification and implement GDOI shall support the following algorithms listed in Table 3:

- Encryption: AES-CBC-128 (also mandatory for IEC 61850 TEK payload). If padding has to be performed to match the algorithm block length, PKCS#7 padding shall be used.
- Hash: SHA2-256
- Authentication: RSA-2048 signatures
- DH group

- Conformant GDOI server (KDC) implementations shall support DH groups 14-18. DH group 14, MODP-2048, is not considered sufficiently secure according to German BSI TR 02102-3 and NIST SP 800-56A and should only be used for backward compatibility cases.
- Conformant GDOI client implementations shall support DH groups 16-18 and may optionally support DH groups 14 -15.

The other algorithms stated in Table 3 may be optionally supported.

The detection of the proposal of other algorithms than the mandatory or optional algorithms listed in Table 3 shall raise a security event ("error: unspecified cryptographic algorithm proposed in IKEv1 phase 1.").

IKEv1 uses a MAC-then-encrypt approach after the DH secret has been established and the session keys are derived. The encryption is indicated in the following sub clauses by the notation HDR* to indicate, that data after the header is encrypted. Detection of padding errors during the decryption in any of the encrypted messages shall raise a security event ("error: padding error during decryption of in IKEv1 handshake.") and shall abort the GDOI handshake.

## 8.3 Phase 1 IKEv1 main mode exchange type 2

### 8.3.1 General

IKEv1 (RFC 2409) defines two methods to perform a phase 1 exchange: "main mode" and "aggressive mode". Support for IKEv1 main mode exchange is mandatory and support for IKEv1 aggressive mode exchange is prohibited. IKEv1 main mode exchange is an instantiation of the ISAKMP identify protection exchange. IKEv1 main mode exchange uses ISAKMP Exchange Type 2.

IKEv1 specifies four main mode exchanges depending on the authentication method (IKE authentication methods SA attribute, value 3) provided in the initial SA payload:

- Authentication with digital signatures
- Authentication with public key encryption
- Revised method of authentication with public key encryption
- Authentication with a pre-shared key

Support for authentication with RSA digital signatures with RSA 2048-bit keys is mandatory according to RFC 6407. The RSA digital signature is IKEv1 authentication method attribute value 3. Besides RSA, further signature algorithms such as DSA and ECDSA may be optionally supported as outlined in RFC 6407, section 5.3.6. Note that from an operational perspective it is strongly recommended to support only one signature scheme within a group or within a KDC domain to allow for unicast and multicast transfer of GDOI GROUPKEY-PUSH messages. Also supporting different signature schemes in one domain may require supporting multiple certificates per component, which increases the management overhead. Authentication with public key encryption and pre-shared keys are optional and not further specified in this document.

The main mode exchange with digital signatures is defined in Figure 19.

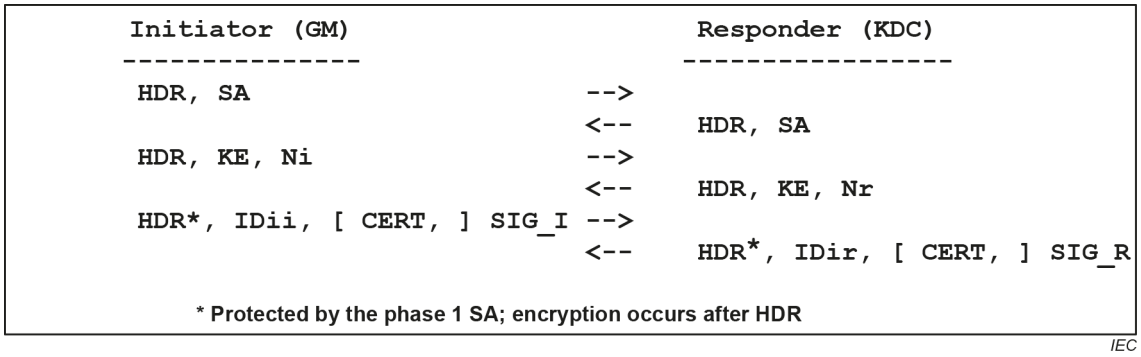



**Figure 19 – IKEv1 (RFC 2409) main mode exchange with RSA digital signatures**

As depicted in Figure 19, the group member is always the initiator of the exchange. The KDC is always the responder of the exchange. The notations (i.e., HDR, SA, KE, etc.) are taken from RFC 2409. Please refer to RFC 2409 section 3[5] for definitions of the notations.

Subclauses 8.3.2 to 8.3.5 describe each message and note where the KDC differs from or limits the IKEv1 protocol.

### 8.3.2 Certificate request payload

The GM and the KDC shall not use the certificate request payload described in ISAKMP RFC 2408 and IKEv1 RFC 2409. This is due to the requirement stated in 8.3.5.3 that the certificate payload shall always be sent in the third handshake of the phase 1 exchange. Based on that, neither the GM nor the KDC should expect to receive a certificate request payload as part of IEC 62351-9 compliant GDOI usage.

### 8.3.3 Security association exchange (1)

#### 8.3.3.1 General

The first two messages exchanged shall be used to determine the security association for the IKE SA, as shown in Figure 20. The exchange marked in red indicates the currently discussed step.

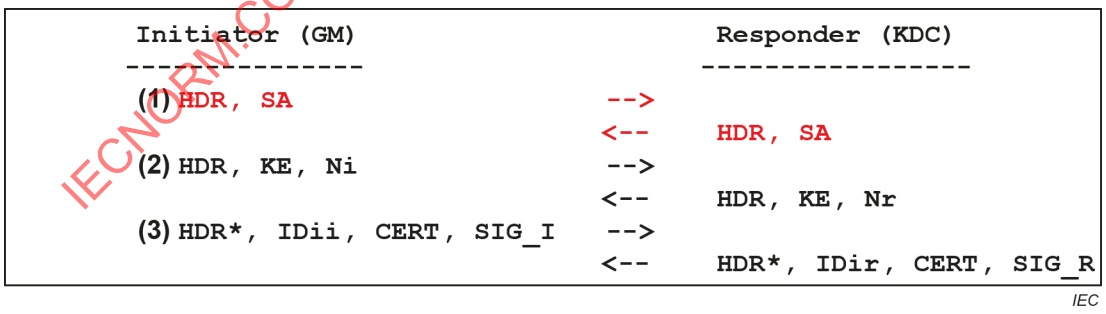



**Figure 20 – IKEv1 main mode exchange and security association messages**

The GM shall send a list of proposed transforms in order of precedence and the KDC shall choose the first compatible one.



5 http://tools.ietf.org/html/rfc2409#page-3.

The initial message sent by the GM shall contain one SA payload as defined in RFC 2409, section 5 Exchanges. An IKEv1 SA payload is defined as one or more transform payloads embedded in a proposal payload that is embedded in an SA payload. Multiple proposal payloads shall not be supported.

#### 8.3.3.2 SA payload

The SA payload (SA) shall be the same as RFC 2408, section 3.4. The KDC shall support GDOI RFC 6407, section 2.1 which requires that the phase 1 SA payload DOI field shall be set to GDOI (2). The SA payload Situation field is 4 octets and shall be set to 0. Any other values shall result in an error.

#### 8.3.3.3 Proposal payload

The proposal payload shall be the same as RFC 2408, section 3.5. The KDC expects the SPI Size to be zero, but if the SPI Size is non-zero, the contents of the SPI shall be ignored. The KDC shall support multiple transforms.

#### 8.3.3.4 Transform payload

The transform payload shall be the same as RFC 2408, section 3.6. Required SA attributes are defined in RFC 2409, section 4 and copied below:

- Encryption Algorithm, value 1
- Key Length, value 14. Required if the encryption algorithm does not specify a key length (i.e., AES_CBC).
- Hash Algorithm, value 2
- Authentication Method, value 3
- Group Description, value 4

Optional SA attributes that may be supported by the KDC are:

- Life Type, value 11
- Life Duration, value 12

The domain of each attribute is defined in Table 3. A transform payload with additional attributes shall result in the KDC rejecting that payload.

### 8.3.4 Key exchange (2)

Once a security association is negotiated, the GM and KDC shall be able to exchange Diffie-Hellman (DH) public values based on the DH group description agreed upon in the SA. The key exchange sequence is shown in Figure 21. The exchange marked in red indicates the currently discussed step.

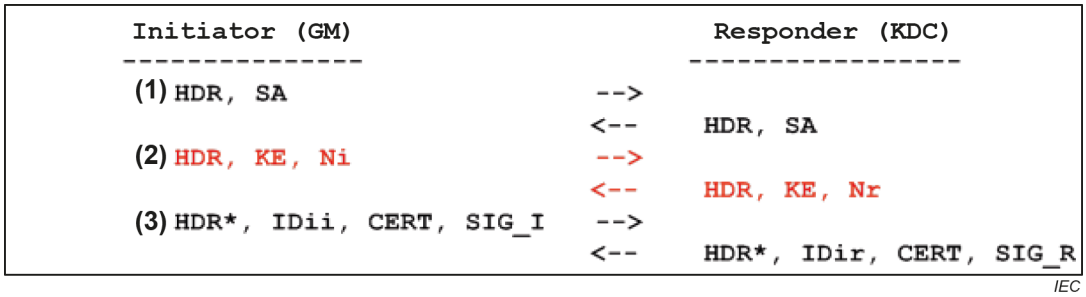



**Figure 21 – IKEv1 main mode exchange: key exchange messages**

In this key exchange sequence, the KDC shall receive an ISAKMP message with a Key Exchange (KE) payload and a nonce (Ni) payload. The KE payload contains the GM's DH public value and the Ni payload contains the GM's nonce. Based on the received GM's DH public value, the DH secret can be calculated. Likewise, the GM receives the KDC's DH public key In a Key Exchange (KE) payload and a nonce from the KDC contained in the Nr payload. The nonces must by cryptographically secure and different for each SA.

The DH secret and the received nonce information from the Ni payload is then used to derive further keys in Phase 1. The public DH parameter, the nonce, and further information from the handshake are then used to calculate the signatures based on HASH_I (initiator) and HASH_R (responder) of the third handshake of the phase 1 exchange (see 8.3.5).

As stated in the RFC 2409, section 5, the nonce shall be between 8 and 256 bytes. The KDC shall create a nonce that is at least half the size of the hash block size.

The KE and Ni/Nr payloads shall be the same as defined in RFC 2408, sections 3.8 and 3.14 respectively.

### 8.3.5 ID authentication exchange (3)

#### 8.3.5.1 General

Once the connection has been secured, the last exchange of security messages performs mutual authentication. The ID authentication sequence is shown in Figure 22. The exchange marked in red indicates the currently discussed step.

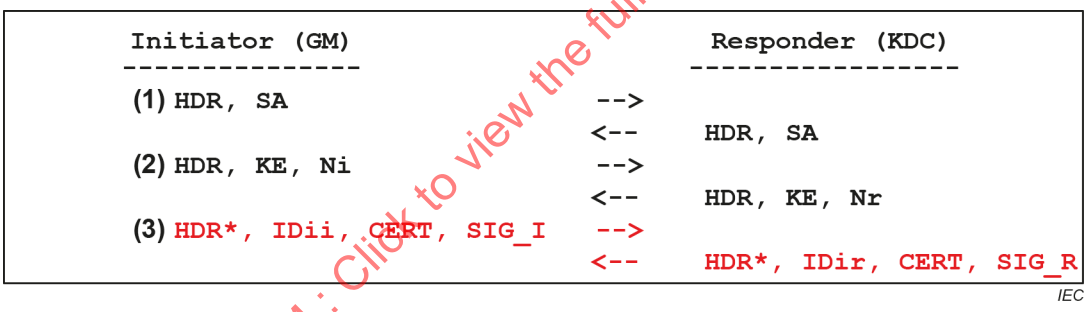



**Figure 22 – IKEv1 Main Mode Exchange: ID authentication messages**

The last message received from the GM shall contain the GM's ISAKMP Identification (Identification payload), its Public Key Certificate (Certificate payload), and the signature of HASH_I or HASH_R respectively using the private key of its X.509 public-key certificate (Signature payload).

#### 8.3.5.2 Identification payload

The ID Payload (IDii, IDir) shall be the same as defined in RFC 2408, section 3.8. The ID Types are technically DOI specific. GDOI RFC 6407 does not specify its own types but implies that the types defined for IPSec DOI are used in GDOI. These types are maintained by IANA under IKEv2 (RFC 5996 [39]). Therefore, the ID Type supported shall be ID_DER_ASN1_DN, which has a value of 9. The contents of the payload shall be the DER encoded Subject ID from the GM's X.509 public-key certificate. All other ID types shall not be supported and shall result in an error.

Note that the Subject ID is also part of the certificate payload described in 8.3.5.3. Nevertheless, it is retained here to maintain compatibility to RFC 2409.

#### 8.3.5.3 Certificate payload

The certificate payload (CERT) shall contain the DER encoded X.509 Public Key Certificate and shall be as defined in RFC 2408, section 3.9. The *X.509 Public-key Certificates – Signature* Certificate Encoding Type that has a value of 4 shall be supported. All other encoding types shall not be supported and shall result in an error.

Although RFC 2409 specifies that one or more certificate payloads may be optionally included, the KDC shall always include one and only one certificate payload in the ISAKMP message.

If the GM does provide a certificate which cannot be verified by the KDC, the KDC shall send a Phase 2 Informational Exchange containing a Delete payload, which is encrypted using the Phase 1 SA key material (see also 8.4.3).

#### 8.3.5.4 Signature payload

The signature payload (SIG_I, SIG_R) shall be unmodified from its definition in RFC 2408, section 3.12. The content shall be the signature, which is generated by using the initiators or responders private key corresponding to the public-key certificate used for sender identification, of HASH_I (initiator) or HASH_R (responder) respectively as defined in RFC 2409, section 5 and depicted in Figure 23.

```
HASH_I = prf(SKEYID, g^xi | g^xr | CKY-I | CKY-R | SAi_b | IDii_b )

HASH_R = prf(SKEYID, g^xr | g^xi | CKY-R | CKY-I | SAi_b | IDir_b )
```

IEC

**Figure 23 – IKEv1 HASH_I calculation**

Refer to RFC 2409 section 3 for definition of the notations.

It is important to note that the RSA signature is not a standard PKCS #1 formatted signature, but instead is simply encrypted using the certificate's corresponding RSA private key as described in RFC 2409, section 5.1:

> ***"Since the hash algorithm used is already known there is no need to encode its OID into the signature. In addition, there is no binding between the OIDs used for RSA signatures in PKCS #1 and those used in this document. Therefore, RSA signatures MUST be encoded as a private key encryption in PKCS #1 format and not as a signature in PKCS #1 format (which includes the OID of the hash algorithm)."***

### 8.4 Phase 1/2 ISAKMP informational exchange type 5

#### 8.4.1 General

The KDC shall use ISAKMP informational exchanges to notify its peers that an error has occurred. The KDC shall not use ISAKMP informational exchanges to provide status SAs as defined in IKEv1 RFC 2409, section 5.7.

An informational exchange may be sent during either a phase 1 or phase 2 exchange. A phase 1 informational exchange is not encrypted and has the message ID field in the ISAKMP header set to zero. A Phase 2 Informational Exchange is encrypted and has a unique message ID. A unique message ID shall be provided so that an applicable cryptographic initialization vector (IV) may be calculated.

### 8.4.2 Phase 1 informational exchange

#### 8.4.2.1 General

A phase 1 Informational exchange may be initiated while the ISAKMP connection is being established to indicate that an error in the phase 1 exchange has occurred. The phase 1 Information exchange is shown in Figure 24.


**Figure 24 – Phase 1 Informational Exchange (cf. RFC 2408, section 4.8)**

Both the GM and KDC may be the initiator of a phase 1 informational exchange. The informational exchange message shall have a single notification payload (N) as defined in RFC 2408, section 3.14.

The Delete payload defined (D) in RFC 2408 section, 3.15 shall be supported for a phase 1 Informational exchange as defined in RFC 2408, section 3.15.

The Hash payload shall not be supported for a phase 1 informational exchange. Therefore, the KDC shall not send a phase 1 informational exchange with a Hash payload and shall ignore any Hash payloads received in a phase 1 informational exchange. The phase 1 informational exchange message shall have the Message ID field in the ISAKMP header set to 0 and shall not be encrypted.

#### 8.4.2.2 Notification payload

The Notification payload is defined in RFC 2408, section 3.14. For a phase 1 informational exchange, the payload field values are defined as:

- Domain of Interpretation – Value of 2, GDOI
- Protocol-ID – Value of 0
- SPI Size – Value of 0. The I-Cookie and R-Cookie are used as the SPI.
- Notify Message Type – Any value defined in RFC 2408, section 3.14.1
- SPI – not included in payload
- Notification Data – not included in payload.

#### 8.4.2.3 Delete payload

The Delete payload is defined in RFC 2408, section 3.15. The Delete payload is used to react on potential errors during the phase 1 negotiation like on SA timeout, certificate errors, or Diffie Hellman related errors (e.g., wrong MODP, etc), no transfer sets match, etc. For a phase 1 informational exchange, the payload field values are defined as:

- Domain of Interpretation – Value of 2, GDOI
- Protocol-ID – Value of 0
- SPI Size – Value of 16
- Number of SPI – The number of SPIs contained in the Delete payload
- SPI – SPIs to be deleted. The I-Cookie and R-Cookie are used as the SPI.

### 8.4.3 Phase 2 Informational Exchange

A Phase 2 Informational Exchange may be initiated to provide Notify or Delete payloads after ISAKMP connection has been established to indicate that an error in the phase 2 exchange has occurred. The phase 2 Information exchange is shown in Figure 25.

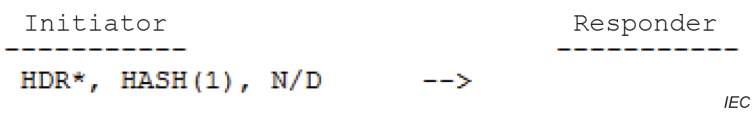



**Figure 25 – Phase 2 Informational Exchange (cf. RFC 2409, section 5.7)**

GDOI RFC 6407 mentions an Informational exchange in reference to a GM not accepting an SA policy. If the policy in the SA payload is not acceptable to the GM, “the GM should tear down the Phase 1 session after notifying the KDC with an ISAKMP Informational Exchange containing a Delete payload” (RFC 6407, section 3.3).

However, there is no mention on how the KDC or GM should handle the case where the hash is incorrect or a GM does not have access privilege to the group or the lifetime of the specific SA has expired. To cover for these cases, this document adds the requirement that a KDC shall embed a Notification Payload in the GROUPKEY-PULL exchange itself. If a failure occurs while processing a GROUPKEY-PULL message from the GM, the KDC shall return a GROUPKEY-PULL message on the same exchange with a single Notification Payload indicating the error code. The GROUPKEY-PULL exchange shall be terminated.

The KDC shall support receiving both a Notification Payload embedded in the GROUPKEY-PULL exchange and receiving an Informational Exchange with a Delete payload. If the SPI of the Notification/Delete payload matches a SA of an existing GROUPKEY-PULL exchange, it shall stop the exchange.

If this error is signalled using an Informational Exchange with a Delete payload it shall contain a HASH value. The message shall be secured relying on the established key material SKEYID_e and SKEYID_a from phase 1. The HASH value shall be calculated as defined for HASH(1) in RFC 2409, section 5.7 and depicted in Figure 26 based on SKEYID_a.

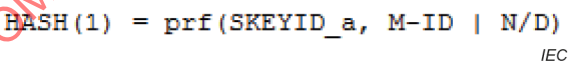



**Figure 26 – IKEv1 HASH(1) calculation**

In addition, when the Informational Exchange with a Delete payload is sent, it shall be encrypted using SKEYID_e resulting from phase 1 SA as described in RFC 2409, section 5.7.



## 8.5 Phase 2 GDOI GROUPKEY-PULL exchange type 32

### 8.5.1 General

GDOI (RFC 6407) defines the Phase 2 GROUPKEY-PULL exchange to allow Group Members a way to pull the shared security associations for a single secure multicast group. The GROUPKEY-PULL exchange is performed after the connection is secured based on the IKE SA established during the phase 1 IKEv1 main mode exchange. RFC 6407, section 3.2 defines the GROUPKEY-PULL exchange in Figure 27.

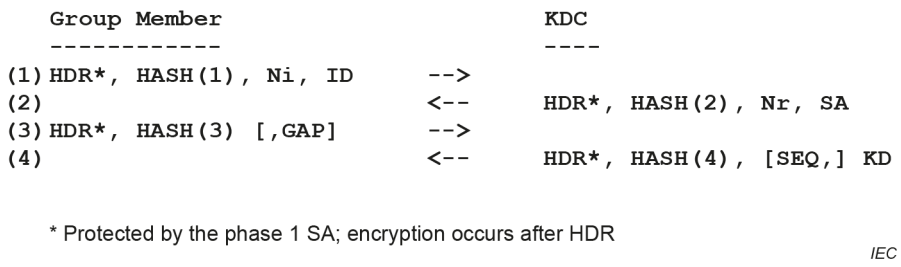



**Figure 27 – GDOI GROUPKEY-PULL as defined in RFC 6407**

Refer to RFC 2409, section 3.2 and RFC 6407, section 1.3 for notation, acronyms, and abbreviations.

A Group Member shall always initiate a GROUPKEY-PULL exchange. The KDC shall always be the responder of a GROUPKEY-PULL exchange.

When the KDC receives the initial GROUPKEY-PULL exchange message from the GM, the message is validated, and a new GROUPKEY-PULL exchange is started. The payloads are extracted and the HASH(1) is calculated and validated. An incorrect hash calculation at the KDC shall result in the exchange failing and a Phase 2 Informational Exchange shall be initiated by the KDC indicating an error type of INVALID_HASH_INFORMATION (value 23).

An incorrect hash calculation at the GM shall result in the exchange failing and a GROUPKEY-PULL response message shall be sent back to the GM with a Notification payload indicating an error type of INVALID_HASH_INFORMATION (value 23).

The secure multicast group ID shall be extracted and if the secure multicast group is not found or the GM is not an authorized group member, the exchange shall fail and a GROUPKEY-PULL response message shall be sent back to the GM with a Notification payload indicating error INVALID-ID-INFORMATION (value 18) and AUTHENTICATION_FAILURE (value 24) respectively.

Group Members shall ensure that Phase 2 handshakes for keys shall be completed prior to any other new key requests in that SA.

If a group member was not able to pull the same group key for a specific group after several retries, it shall send a Phase 2 Informational Exchange containing a Delete payload, which is encrypted using the Phase 1 SA key material (see also 8.4.3). This essentially will terminate the SA with that GM. The number of retries depends on the local security policy.



### 8.5.2 Hash computations

The hashes computed for the GROUPKEY-PULL exchange shall use the secure hash algorithm from the phase 1 SA. The hashes shall be secured with the phase 1 authentication secret, SKEYID_a, as defined in RFC 6407, section 3.2. The different hashes used in the GROUPKEY-PULL exchange shall be computed over the information specified in Figure 28.

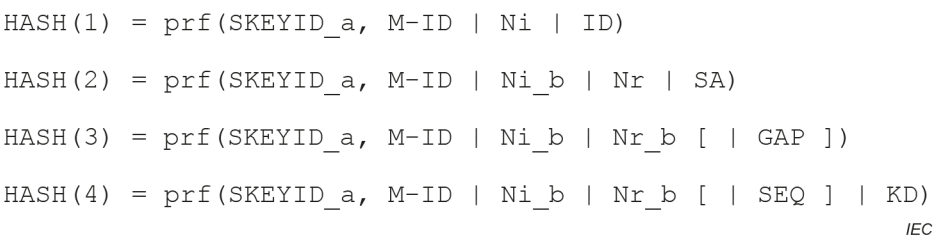



**Figure 28 – GROUPKEY-PULL hash computations**

### 8.5.3 Multi-sender and counter mode encryption algorithm

IEC 61850 SAs (see 8.5.7) shall support AES-GCM and AES-GMAC counter mode algorithms. Counter mode algorithms provide the capability for there to be multiple senders over a single SA. This is achieved by ensuring that each sender use unique initialization vectors (IV) for each packet sent. GDOI RFC 6407, section 3.5 specifies that the KDC implement RFC 6054 in order to assign a portion of the IV space to each sender in the group[6].

Support for the request of Sender IDs from a GM for counter mode-based IEC 61850 key groups is not required. If a Group Associated Policy (GAP) payload is received from a GM with a Sender-ID GAP attribute (value 3), the KDC shall fail the GROUPKEY-PULL exchange. An error type of ATTRIBUTES-NOT-SUPPORTED (value 13) shall be returned in an Informational exchange.

NOTE It is to be determined whether multiple senders on a single DATA SA will be supported in a future edition of this document.

### 8.5.4 SA KEK, SEQ, KEK/LKH key download payload support

Since the GROUPKEY-PUSH is optional, the GROUPKEY-PULL exchange is not required to support a SA KEK payload, SEQ payload, or KEK/LKH Key Packets in the Key Download payload. Therefore, within the context of a GROUPKEY-PULL, the SA KEK keys should not be used. The KEK keys shall be used only for GROUPKEY-PUSH according to RFC 6407, section 5.3.2).

Rekeying an existing SA with GROUPKEY-PUSH is described in 8.6.

The information about the assignments for the GDOI payloads, specifically the key download types is available at IANA [52]. According to this definition, the key download type for TEK is 1, while the key download type for KEK is 2.

NOTE It is to be determined when the GROUPKEY-PUSH exchange will be supported by IEC 61850 security protocols.

### 8.5.5 GROUPKEY-PULL group SA request exchange

#### 8.5.5.1 General

The initial SA Request exchange is shown in Figure 29. The exchange marked in red indicates the currently discussed step.


**Figure 29 – GROUPKEY-PULL initial SA request exchange**

The GM shall initiate a GROUPKEY-PULL exchange by sending message (1) to the KDC. The HASH (1) and Ni payloads are defined in RFC 6407, section 3. The ID Payload has been extended to support IEC 61850 secure multicast groups.



6 RFC 6054 – Using Counter Modes with Encapsulating Security Payload (ESP) and Authentication Header (AH) to Protect Group Traffic (http://tools.ietf.org/html/rfc6054).

#### 8.5.5.2 Identification payload

##### 8.5.5.2.1 General

The ID payload (ID) for the phase 2 exchange is the same ID payload used in the phase 1 exchange (see 8.3.5.2). The difference is that in the phase 1 exchange, the Identification Data contained is that of the Group Member, while for the phase 2 exchange the ID identifies the Key Group for the SAs to be pulled as shown in Figure 30. The KDC shall support IEC 61850 secure multicast groups.

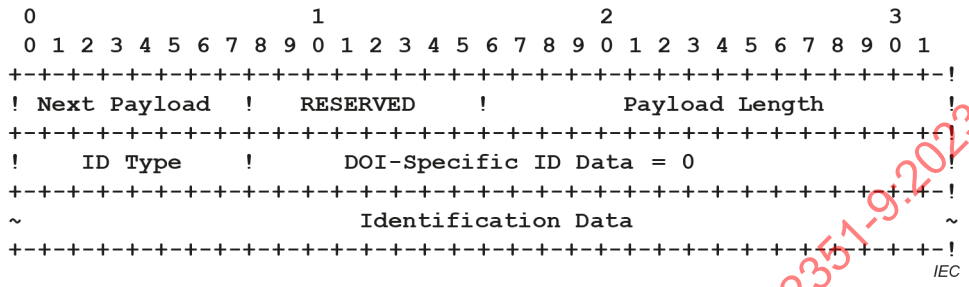



**Figure 30 – RFC 6407 Identification Payload**

The format of the identification payload consists of:

- **Next Payload** – see RFC 2408 for definition and use.
- **Reserved** – see RFC 2408 for definition and use.
- **Payload Length** – see RFC 2408 for definition and use.
- **ID Type** – An IEC 61850 secure multicast group is identified by an ID Type of ID_OID defined in RFC 8052 for “GDOI Protocol Support for IEC 62351 Security Services", section 2.1. The ID_OID ID Type has a value of 13. The Identification Data for ID_OID represents an ASN.1 Object ID (OID)[7] and its OID specific data. Both the OID and the OID data are encoded using DER encoding rules[8]. The field definitions within the Identification Data payload are defined in and depicted in Figure 31. For example, an OID could represent a GOOSE or Sampled Value protocol, also in other cases IEC 61850 also specifies a particular multicast destination address to be described in the OID Specific Payload field.
- **DOI-Specific ID Data** – Shall be set to 0.
- **Identification Data** – is ID_OID Specific and described in 8.5.5.2.2.

##### 8.5.5.2.2 Identification data for IEC 61850 objects

The GDOI Identification Payload (e.g., Phase 2 request) in a GDOI GROUPKEY-PULL exchange allows the Group Member (GM) to declare the group it would like to join. A group is defined by an ID payload as defined in GDOI RFC 6407). Figure 31 has been extracted from RFC 8052 and defines a specific ID Type value and format of the Identification Data.



---

7 The Object ID format is defined in ITU-T-X.683 – http://www.itu.int/ITU-T/studygroups/com17/languages/X.683-0207.pdf.

8 Distinguished Encoding Rules are defined in ITU-T-X.690 – http://www.itu.int/ITU-T/studygroups/com17/languages/X.690-0207.pdf.

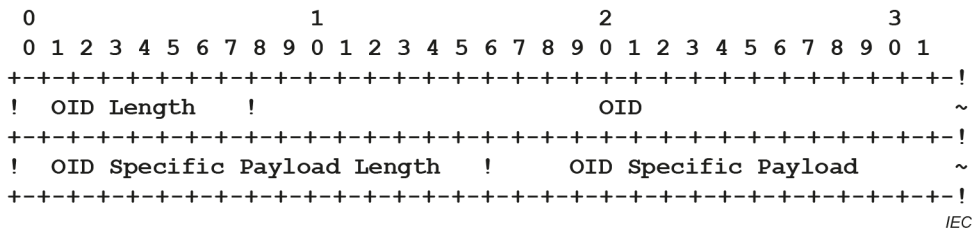



**Figure 31 – ID_OID Identification Data**

- **OID Length**: This length shall be an unsigned integer value and shall specify the number of octets of the ASN.1 encoded OID, whose value follows the length.
- **OID**: This set of octets represents an ASN.1 encoded Object Identifier. The value of the identifier defines the payload that follows. It is the use of this Object Identifier that will allow other organizations or standards to utilize this payload extension process without collision in the context of the definition. The Object Identifier values are defined as mandatory (m) or optional (o) in Table 4.
- **OID Specific Payload Length** (2 octets) – Length of the OID Specific Payload. Set to zero if the OID does not require an OID Specific Payload.
- **OID Specific Payload** (variable) – OID specific selector encoded in DER. If OID Specific Payload Length is set to zero, this field does not appear in the ID payload.

At the culmination of a GROUPKEY-PULL exchange, the key server shall have sent the group policy to all authorized group members, allowing receiving group members to participate in secure group communications.

The KDC shall support the mandatory IEC 61850 Object IDs defined in Table 4, which identify the IEC 61850 stream that the GM is requesting Key material for.

**Table 4 – IEC 61850 Object IDs: Mandatory (m) and Optional (o)**

| Object Identifier Name | Description | Value | m/o |
|---|---|---|---|
| 61850_ETHERNET_GOOSE | Specifies that the payload is requesting a key for an IEC 61850-8-1 GOOSE APDU. | 1.0.62351.9.61850.8.1.1 | m |
| 61850_UDP_ADDR_GOOSE | Specifies that the payload is requesting a key for a routable GOOSE APDU that is being sent to a particular destination IP address. | 1.0.62351.9.61850.8.1.2 | m |
| 61850_UDP _Tunnel | Specifies that the payload is requesting a key for an IEC 61850 routable Tunnel APDU that is being sent to a particular destination IP address. | 1.0.62351.9.61850.8.1.4 | o |
| 61850_ETHERNET_SV | Specifies that the payload is requesting a key for an IEC 61850-9-2 SV APDU. | 1.0.62351.9.61850.9.2.1 | m |
| 61850_UDP_ADDR_SV | Specifies that the payload is requesting a key for a routable IEC 61850 SV APDU. | 1.0.62351.9.61850.9.2.2 | m |
| 61850_IP_ISO9506 | Specifies that the payload is requesting a key for an IEC 61850-8-1 ISO 9506 endpoint. This payload definition is out-of-scope. | 1.0.62351.9.61850.8.1.4 | o |
| 61850_9_3_PTP | Specifies that the payload is requesting a key for an IEC 61850-9-3 PTP Domain | 1.0.62351.9.61850.9.3.1 | o |
| The utilized OIDs have been defined in the IEC 62351 space starting with 1.0.62351.9.xxx as opposed to the OIDs used in IEC 61850-90-5 starting with 1.2.840.10070.xxx. | | | |

#### 8.5.5.2.3 IEC 61850 OID specific payloads

The payloads for the UDP versions of GOOSE (61850_UDP_ADDR_GOOSE) and SV (61850_UDP_ADDR_SV) OIDs[9] are the same. The ASN.1 Sequence for 61850_UDP_ADDR_GOOSE and 61850_UDP_ADDR_SV OID specific data are specified in Figure 32.

```
IecUdpAddrPayload ::= SEQUENCE
{
      version     INTEGER { v1(1) },
      ipAddress   IPADDRESS,
      dsRef       VisibleString SIZE(1..128))
}
```

IEC

**Figure 32 – 61850_UDP_ADDR_GOOSE/SV ASN.1 BNF**

- `Version` is a single octet value that represents the version of the particular payload. Unless otherwise specified, the value of the VERSION shall be one (1).
- `IpAddress` is a value component that allows the use of either an IPv4 or IPv6 destination address for the entity associated with the key being requested. The IPADDRESS ASN.1 Sequence is specified in Figure 33.
- The `dsRef` value component allows for the specification of an IEC 61850 DataSet Reference (**DSRef**), as specified in IEC 61850-7-2.

```
IPADDRESS ::= SEQUENCE
{
   typeOfAddress ENUMERATED { IPv4(0), IPv6(1) },
   address  CHOICE {
      ip      OCTET STRING (SIZE(4 | 16)),
      dns     VisibleString (SIZE(1..65536))
      }
}
```

IEC

**Figure 33 – IPADDRESS ASN.1 BNF**

Figure 34 is an example of the 61850 OID Address Payload for either 61850_UDP_ADDR_GOOSE or 61850_UDP_ADDR_SV OIDs.



9 The payload for the IP versions of GOOSE and SV are the same.

```
groupaddr IecUdpAddrPayload ::=
{
   version v1,
   ipAddress
   {
        typeOfAddress IPv4,
        address dns: "www.iec.org"
   }
   dsRef "@somedataref"
}

DER Encoding:

30230201 0130100A 01001A0B 7777772E
6965632E 6F72671A 0C40736F 6D656461
74617265 66
```

IEC

**Figure 34 – Example IecUdpAddrPayload ASN.1 Data with DER Encoding**

The ASN.1 Sequence for 61850_UDP_TUNNEL OID specific data is specified in Figure 35.

```
IecUdpTunnelPayload ::= SEQUENCE
{
    version      INTEGER { v1(1) },
    ipAddress    IPADDRESS
}
```

IEC

**Figure 35 – 61850_UDP_TUNNEL Payload ASN.1 BNF**

The `dsRef` field is not present in this payload because multiple Ethernet multicast frames (e.g., GOOSE or SV) may be sent in one Tunnel SPDU. Therefore, the destination IP address itself differentiates the data sets.

The OID payload for the Ethernet versions of GOOSE (61850_ETHERNET_GOOSE) and SV (61850_ETHERNET_SV) are the same. The ASN.1 Sequence for 61850_ETHERNET_GOOSE/SV OID specific data is specified in Figure 36.

```
IecEthernetAddrPayload ::= SEQUENCE
{
    version   INTEGER { v1(1) },
    dstMAC    OCTET STRING (SIZE(6)),
    dsRef     VisibleString (SIZE(1..256))
}
```

IEC

**Figure 36 – 61850_ETHERNET_GOOSE/SV Payload ASN.1 BNF**

- Version is a single octet value that represents the version of the particular payload. Unless otherwise specified, the value of the VERSION shall be one (1).
- The `dstMAC` is a value that consists of six (6) octets. The value shall be in Ethernet transmission order.

- The `dsRef` value component allows the specification of an IEC 61850 DataSet Reference (DSRef), as specified in IEC 61850-7-2.

#### 8.5.5.2.4 IEC 61850_9_3 Specific Payloads

The payloads for the IEC 61850-9-3 (i.e., power profile of PTP) shall specify the PTP time domain (`ptpDomain`) and an identifier for that domain besides the domain ID. This identifier shall be a mutually agreed upon value between the GM and KDC so that the same domain number may be reused in other parts of the infrastructure with different keys and policies.

```
Iec61850PTPPayload::= SEQUENCE
{
  ptpDomain       INTEGER  - defined by the allowed range per IEEE 1588
  ptpSubDomain    [0] IMPLICIT INTEGER  OPTIONAL - defined by IEEE 1588
  keyMngtDomain   VisibleString (Size(1..256)
}
```

The `keyMngtDomain` value is used to allow the same `ptpDomain` and `ptpSubDomain` identifiers to be deployed in two non-related areas of the utility enterprise and the ability to deliver different keys and policies to those entities. This value is a local configuration issue and shall not be NULL or empty. The unique identification for the keys being requested in the GROUPKEY-PULL is a unique triplet of values consisting of the values of `ptpDomain`, `ptpSubDomain`, and `keyMngtDomain`.

It is expected that all PTP instances for a particular PTP domain or PTP subdomain use the same key management protocol.

### 8.5.6 SA TEK payload

The SA TEK payload shall contain security attributes for a single set of policies associated with a group TEK. The type of policy to be used with the TEK is described by a Protocol-ID field included in the SA TEK. As shown in Figure 37 reproduced from RFC 6407, each Protocol-ID describes a particular TEK Protocol-Specific Payload definition.

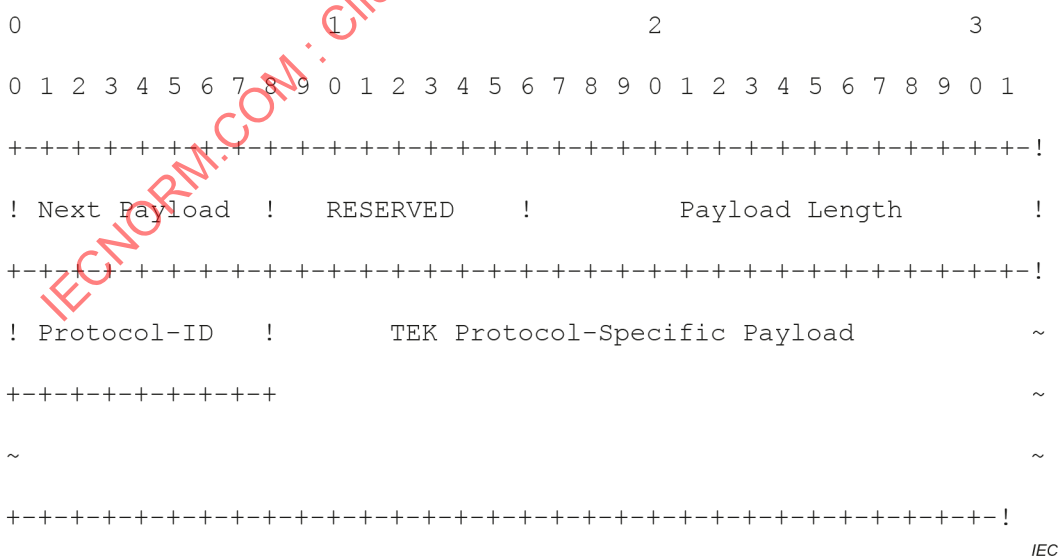



**Figure 37 – RFC 6407 SA TEK Payload**

The KDC shall comply with the IEC-61850 SA TEK payload format defined in RFC 8052 section 2.2. The IEC-61850 SA TEK payload defines two optional SA Data Attributes, the Activation Time Delay (SA_ATD) and the Key Delivery Assurance (SA_KDA) attributes. The values for these attributes are defined in RFC 8052 section 4.0, IANA Considerations.

The Protocol ID name of GDOI_PROTO_IEC_61850 is defined in RFC 8052, section 2.2 and has the value of 3.

### 8.5.7 IEC 61850 SA TEK payload

GDOI_PROTO_IEC_61850 SA TEK shall include an OID and (optionally) an OID Specific Payload that together define the selectors for the network traffic. The selector fields shall be followed by security policy fields indicating how the specified traffic is to be protected. As shown in Figure 38 reproduced from RFC 8052 each 61850 TEK Payload describes a particular TEK Protocol-Specific Payload definition.

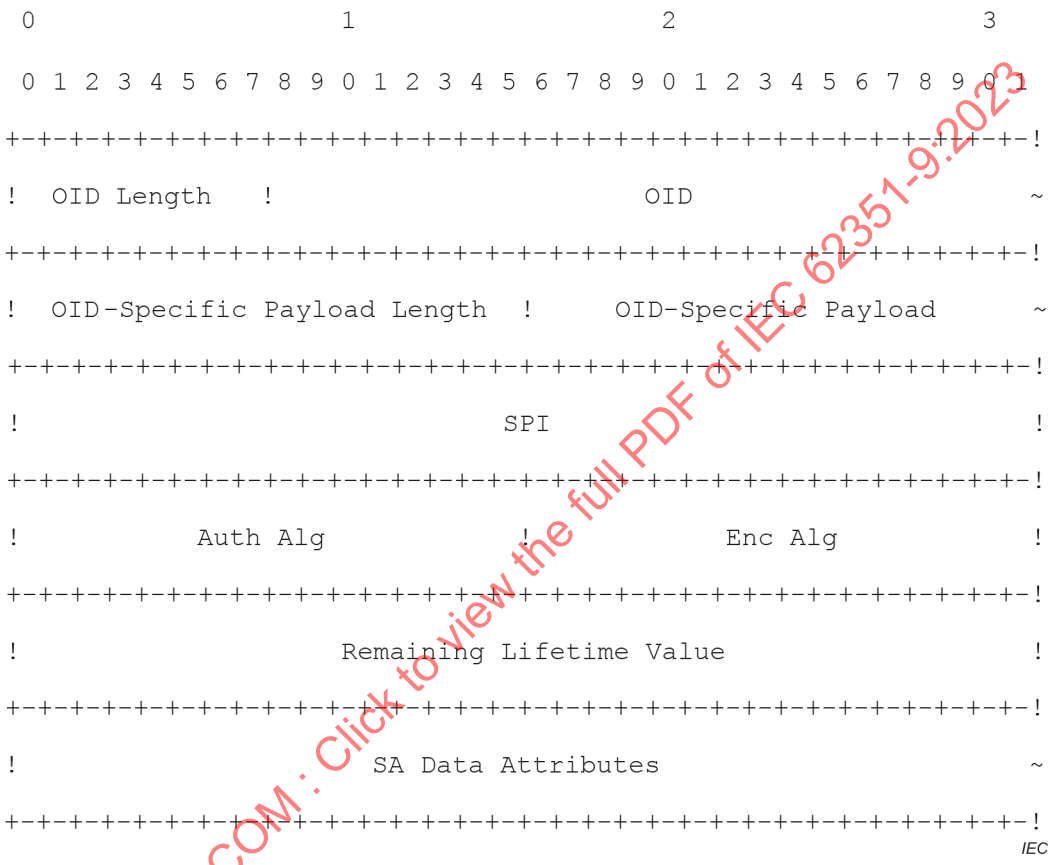



**Figure 38 – IEC-61850 SA TEK Payload**

The GDOI_PROTO_IEC_61850 SA TEK Payload fields are defined in RFC 8052 as follows:

a) OID Length (1 octet) – Length of the OID field.

b) OID (variable) – An ASN.1 object Identifier encoded using DER. OIDs defined in IEC 61850 declare the type of IEC 61850 message to be protected, as defined in Table 4.

c) OID Specific Payload Length (2 octets) – Length of the OID Specific Payload. This field is set to zero if the policy does not include an OID Specific Payload.

d) OID Specific Payload (variable) – The traffic selector (e.g., multicast address) specific to the OID encoded using DER. Some OID policy settings do not require the use of an OID Specific Payload, in which case this field is not included in the TEK and the OID Specific Payload Length is set to zero.

e) SPI (4 octets) – Identifier for the Current Key. This field represents a SPI.

f) Auth Alg (2 octets) – Authentication Algorithm ID. Valid values are defined in RFC 8052, section 2.2.2. HMAC-SHA256-128 is mandatory. In addition, this document also requires support for HMAC-SHA256, AES-GMAC-128, and AES-GMAC-256.

g) Enc Alg (2 octets) – Confidentiality Algorithm ID. Valid values are defined in RFC 8052, section 2.2.3. AES-CBC-128 is mandatory.

h) Remaining Lifetime value (4 octets) – The number of seconds remaining before this TEK expires. A value of zero (0) shall indicate that the TEK does not have an expiry time. Maximum lifetime shall be correlated with the CRL refresh time to ensure that group members with expired certificates or revoked certificates are handled according to the organization's security policy.

i) SA Data Attributes (variable length) – Contains zero or more attributes associated with this SA. RFC 8052, section 2.2.4 defines attributes.

There are some restrictions on how the Authentication Algorithm and the Confidentiality Algorithm may be combined. The restrictions are summarized below:

a) If an AEAD algorithm like AES-GCM is applied (providing both confidentiality and authentication), the confidentiality algorithm shall specify the AEAD algorithm, i.e., AES-GCM-128 (value 4) or AES-GCM-256 (value 5). The Authentication Algorithm field shall be set to NONE (value 1).

b) If a non-AEAD algorithm is specified for confidentiality, i.e., AES-CBC-128 (value 2) or AES-CBC-256 (value3), an Authentication Algorithm shall be specified. RFC 8052 defines supported message authentication algorithms, the mandatory to support is HMAC-SHA256-128 (value 2). Alternatively, HMAC-SHA256-128 (value 3), AES-GMAC-128 (value 4), or AES-GMAC-256 (value 5) may be used. Use of the AES-GMAC variants is not foreseen here, as it essentially employs an AEAD algorithm in GMAC mode.

c) As confidentiality is optional, the Confidentiality Algorithm field may be set to NONE (value 1). The Authentication Algorithm shall be specified based on the supported algorithms defined in RFC 8052, i.e., HMAC-SHA256-128 (value 2), HMAC-SHA256-128 (value 3), AES-GMAC-128 (value 4) or AES-GMAC-128 (value 5).

### 8.5.8 SA TEK payload for IEC 61850-9-3

The SA-TEK payload for IEC 61850-9-3 (i.e., power profile of PTP, IEEE 1588:2019) shall provide the required parameter to secure the PTP exchanges. The integrated security option of PTP is specified in 16.14 of IEEE 1588:2019. This clause specifies an AuthenticationTLV, used to provide integrity protection on a per message base. To utilize this AuthenticationTLV, certain parameters are to be provided by the key management. IEEE 1588:2019 does not define a key management by its own and expects a PTP profile to select an appropriate key management for the considered domain. With GDOI one potential option is already outlined Annex P of IEEE 1588:2019. As GDOI is already applied in the power system domain to distribute security parameter and security policy to secure GOOSE and SV communication, it may also be used to distribute this information for application in PTP.

Note that IEEE 1588:2019 defines support for immediate security processing (the group key is already available at each PTP instance) or delayed security processing (the group key to verify the integrity of a PTP message is released at a later point in time, leading to a delayed validation). GDOI targets the immediate security processing as required in power systems.

IEEE 1588:2019, clause 16.14.2 requires the following parameter to be available to provide PTP message integrity protection. Note that selected values are intended to be part of the AUTHETICATION TLV defined in IEEE 1588:2019, 16.14.2.

- Security Parameter Pointer (SPP): Identifies the security association (SA). This shall be the SA TEK SPI.
- IntegrityAlgTyp: Identifies integrity algorithm type used to compute the size of the ICV. The algorithm type shall be specified in the Auth Alg field of the SA TEK.
- icvLength: Indicates the length of the calculated ICV. The length is algorithmically determined as required by the Auth Alg in the SA TEK.

- Key: A symmetric key to be used in conjunction with the selected algorithm. This value shall be provided in the KD payload as described in 8.5.11 for GROUPKEY-PULL and the Key information delivered in the SA TEK of GROUPKEY-PUSH. Note that 8.5.12 additionally defines the TEK download handling
- keyLength: Indicates the length of the disclosed key (optional parameter, only if delayed security processing is used). The key length shall be determined through the used *Key Alg* of the SA TEK GDOI Payload.
- Indication for use of the optional field `sequenceNo` and the desired length of the `sequenceNo` field. According to IEEE1588:2019 this field shall be "0". Therefore, this value is not provided in the SA TEK. It is expected to be a local configuration issue.
- Indication of the `sequenceID` window for anti-replay (based on the `sequenceID` in the PTP message header). As IEEE 1588:2019 does not define a specific handling for the anti-replay window and rather refers to monitoring the advancement of the sequence numbers, this `sequenceID` window will not be provided by GDOI.
- Immediate Security: A Boolean indicating whether the SA is for immediate security or delayed security. Using GDOI provides support of immediate security. This value is therefore not provided in the SA TEK. It is expected to be a local configuration issue.
- Indication for use of an optional field RES in the Authentication TLV and the desired length of the RES field. According to IEEE1588:2019 this field shall be "0". Therefore, this value is not provided in the SA TEK. It is expected to be a local configuration issue.

These parameters are mapped to the IEC-61850 SA TEK Payload as defined in 8.5.7. This mapping may directly support IEC 61850-9-3 describing the Power Profile of PTP. The following maps the IEEE 1588:2019 security parameter field stated above to the GDOI_PROTO_IEC_61850 SA TEK Payload fields:

a) OID Length (1 octet) – Length of the OID field.

b) OID (variable) – An ASN.1 object Identifier encoded using DER. OIDs defined in IEC 61850 declare the type of IEC 61850 message to be protected, as defined in Table 4.

c) OID Specific Payload Length (2 octets) – Length of the OID Specific Payload. This field is set to zero if the policy does not include an OID Specific Payload.

d) OID Specific Payload (variable) – The traffic selector (e.g., multicast address) specific to the OID encoded using DER. Some OID policy settings do not require the use of an OID Specific Payload, in which case this field is not included in the TEK and the OID Specific Payload Length is set to zero.

e) SPI (4 octets) – Identifier for the Current Key. Note that the SPP in IEEE 1588:2019 is only 1 octet, while the SPI has a length of 4 octets. This requires that the PTP instances only uses a part of the SPI value as SPP to match the required length. A PTP instance shall use the low order values of the SPI (SPI [0..7]) as SPP to be included into the Authentication TLV of PTP.

f) Auth Alg (2 octets) – Authentication Algorithm ID. This field contains the `IntegrityAlgTyp`. IEEE 1588:2019 requires support of HMAC-SHA256-128, which is also defined in RFC 8052, Section 2.2.2.

g) Enc Alg (2 octets) – Confidentiality Algorithm ID. As IEEE1588:2019 does not support confidentiality protection for PTP messages. Therefore, this value shall be set to "0"

h) Remaining Lifetime value (4 octets) – The number of seconds remaining before this TEK expires. A value of zero (0) shall indicate that the TEK does not have an expiry time. Maximum lifetime shall be correlated with the CRL refresh time to ensure that group members with expired certificates or revoked certificates are handled according to the organization's security policy. This value is not explicitly stated in IEEE 1588:2019, clause 16.14 but required by the key management to ensure that the group key may be updated regularly.

i) SA Data Attributes (variable length).

### 8.5.9 SPI discussion

#### 8.5.9.1 General

There are two different SPI values that are utilized and defined for different purposes within GDOI. These are SPI values that are contained in the SA TEK and SA KEK. Descriptions of both are in 8.5.9.2 to 8.5.9.3.

#### 8.5.9.2 SA TEK SPI

RFC 6407 requires that characteristics of a SA TEK SPI shall be defined. A SPI in a GDOI_PROTO_IEC_61850 SA TEK is represented as a Key Identifier (KeyID). The SPI size is 4 octets. The SPI is unilaterally chosen by the KDC using any method chosen by the implementation. However, an implementation needs to take care not to duplicate a SPI value that is currently in use for a particular group.

#### 8.5.9.3 SA KEK SPI

RFC 6407, section 5.3 defines that the SA KEK is 16 octets and is utilized to populate the Initiator-Cookie and Responder-Cookie of a GROUPKEY-PUSH Header.RFC 6407, section 3.2 (GROUPKEY-PULL) defines the value of the SPI to be determined by the GCKS. It is this value that provides correlation of keys and policy deliveries between the GROUPKEY-PULL and GROUPKEY-PUSH. Figure 39 shows the relationship.

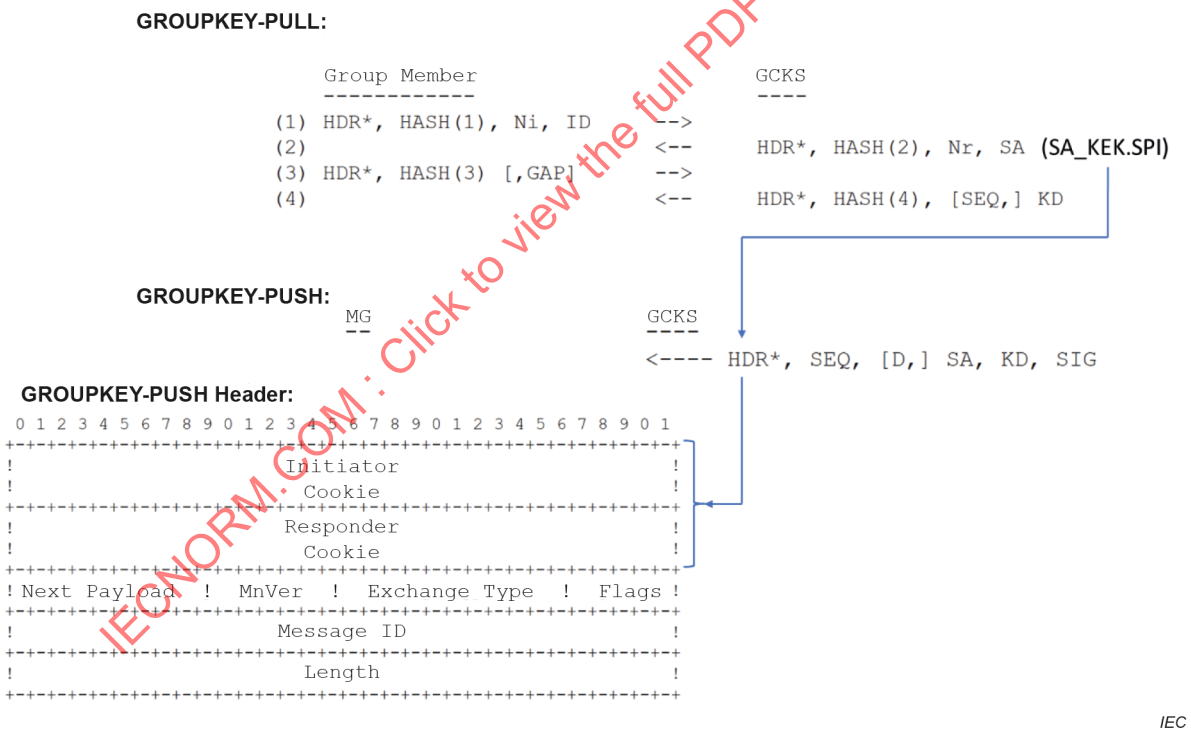



**Figure 39 – Correlation of SPI Value**

This correlation allows for GMs to determine if a GROUPKEY-PUSH is of interest. If the SPI matches an SA KEK SPI that it received during a GROUPKEY-PULL, the GROUPKEY-PUSH should be processed otherwise, its processing is a local issue.

KDC (GCKS) implementations claiming conformance to IEC 62351-9 shall provide SA KEK SPI values that are globally unique to each unique key group, as defined by 8.5.5.2.3 and 8.5.5.2.4

### 8.5.10 SA data attributes

#### 8.5.10.1 General

The following attributes shall be present in the SA TEK for IEC 61850 SAs. The attributes shall follow the format described in RFC 8052, Appendix C.

#### 8.5.10.2 Activation time delay (SA_ATD) SA data attribute (value 1)

A KDC shall distribute an SA TEK in advance of when it is expected to be used. This is communicated to group members using the SA Activation Time Delay (SA_ATD) attribute. When a GM receives an SA TEK with this attribute, it shall wait for the number of seconds contained within the attribute before installing it for either transmission or receiving. The KDC shall include the SA_ATD SA data attribute, even if the value is 0. The SA_ATD attribute is assigned a value of 1.

#### 8.5.10.3 Key delivery assurance (SA_KDA) SA data attribute (value 2)

Group policy may include notifying a multicast source ("Publisher") of an indication of whether multicast receivers ("Subscribers") have previously received the SA TEK. This notification allows a Publisher to set a policy as to whether to activate the new SA TEK or not based on the percentage of Subscribers that are able to receive packets protected by the SA TEK. The attribute value is a number between 0 and 100 (inclusive). Support for KDA is optional. If KDA is not supported, the KDC shall set the SA_KDA value to 100.

The SA_KDA attribute is assigned a value of 2.

### 8.5.11 GROUPKEY-PULL group key download exchange

After the SA policies are sent to the GM, the KDC shall send the key material for each SA. The exchange is described in Figure 40.


**Figure 40 – GROUPKEY-PULL Key Download Exchange**

Before the key material is sent to the GM, the GM shall first respond to the SA exchange by sending message (3) with the Hash payload containing the HASH(3) computation as shown here:

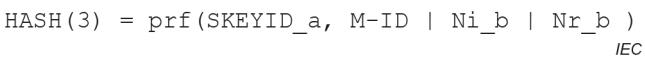



**Figure 41 – GROUPKEY-PULL group key download hash computations**

If the KDC cannot validate the hash calculation, a Phase 2 Informational Exchange shall be initiated with an error indicating INVALID_HASH_INFORMATION (value 23).

The KDC shall respond with the last message (4) in the exchange. This includes a Hash payload containing HASH (4) and a Key Download payload. The Key Download (KD) payload shall be formatted as defined in RFC 8052,section 2.3. The KDC shall not alter or specify any restrictions to the format of KD payload defined.

The information about the assignments for the GDOI payloads, specifically the key download types (as defined in RFC 6407) is available at IANA [52]. According to this definition, the key download type for TEK is 1, while the key download type for KEK is 2.

The renewal of the session key is normally triggered before the ‘Remaining Lifetime Value’ of the current key has expired. "Remaining Lifetime Value" (aka lifetime) is one of the parameters returned in the SA TEK payload and it signifies the number of seconds remaining before the associated SA expires.

This process is shown in Figure 42 as the following sequence:

- The KDC creates a session key for a group of devices along with its ‘Key Identifier (KeyID)’[10], a Remaining Lifetime Value and SA Time Activation Delay (SA_ATD) parameter. It maintains that session key for the group throughout that session:
  - ‘KeyID’: is the unique identifier for the session key sent by the KDC in the GDOI GROUPKEY-PULL or GROUPKEY-PUSH message payload
  - Remaining Lifetime Value: is the number of seconds remaining before the SA it is associated with expires
  - ‘SA Time Activation Delay (SA_ATD)': This parameter specifies when the key is expected to be used because the KDC will sometimes distribute a SA TEK in advance.
- The KDC sends two SAs and associated keys, one (K0) that is current with SA_ATD zero, so it may be used right away, and a second one (K1) with a SA_ATD non-zero, to be used later on. KDC synchronizes the two SAs, so that the time delay SA_ATD for the second key is less than the lifetime of the first key.
- When the GM obtains the two session keys K0 and K1, it starts using K0 right away and starts (two) timers, one for K0 remaining lifetime and a SA_ATD timer for when K1 will be activated
- GM switches to the new key K1 when the SA_ATD time expires.
- When K0 Remaining Lifetime Value expires, the GM sends a key renewal request to the KDC to obtain the next session key K2. Note that the KDC should always keep at least two SAs, a current active one, and a next with its ATD set as non-zero. If the trigger for creating a new SA is based on the expiration of the current one, then the KDC should have no more than two SAs at any time. However, if the trigger to create a new SA is based on the ATD expiration of the next SA, there will be situations where the KDC may need to hold three SAs for a group: a current, a next and a new one. The KDC may send all these SAs during the pull, so a GM should be prepared to receive them. However, if the GM sends the pull request at the expiration of the current key, and not at the expiration of ATD for the next key, it always receives only two SAs and keys.



[10] GDOI refers to this as a SPI.

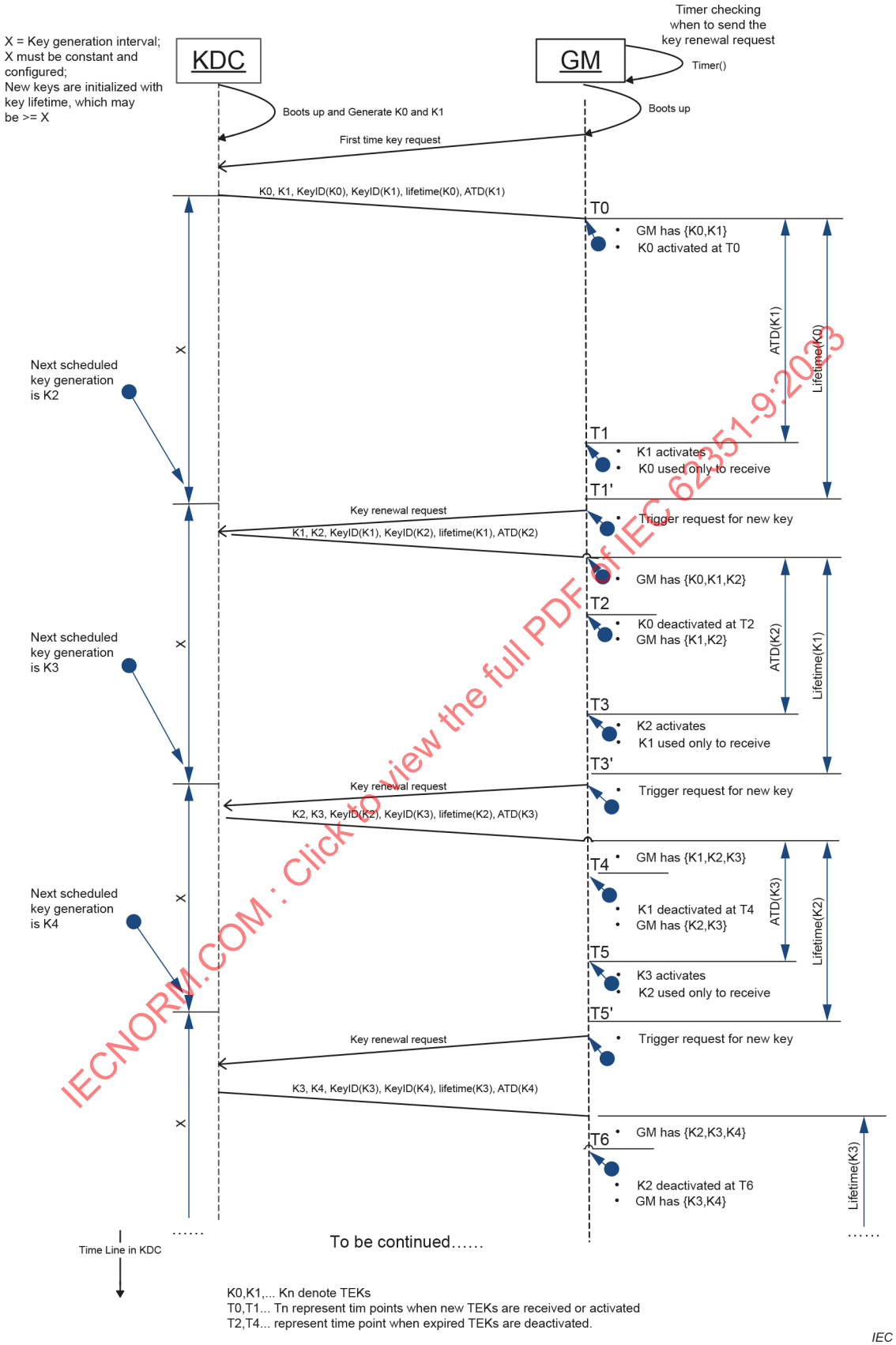





**Figure 42 – Key renewal triggered by the entities**

If the negotiation of the DH algorithms and key size fails, the GROUP-PULL-PHASE1-KEY-NEGOTIATION event shall be raised.

If an SA is used that has expired, the event GROUP_PULL_EXPIRED_SA shall be raised.

If decryption fails as part of the GROUPKEY-PULL Phase 2 exchange(s), the event GROUP_PULL_PHASE2_DECRYPTION shall be raised.

If a request for a key of an unknown GROUP is performed, the GROUP-PULL-PHASE2-GROUP-NONEXISTENT event shall be raised.

#### 8.5.12 TEK Key Download Handling

IEC 62351-9 defines that two (2) Key Download (KD) payloads are to be provided via GROUPKEY-PULL and GROUPKEY-PUSH. These KDs have different activation times (SA_ATD) and remaining lifetime values (see Figure 42). These values relate to one and only one KEY_ID and thus determine the time at which the associated key is to be used and when it expires.

### 8.6 Phase 2 GROUPKEY-PUSH exchange type 33

#### 8.6.1 General

GDOI (RFC 6407) defines the Phase 2 GROUPKEY-PUSH exchange to allow a KDC to provide rekey information to GMs using a push message (see Figure 43). It is an ISAKMP protocol, where cryptographic policy and keying material ("Rekey SA") has been distributed by the KDC as part of the group policy in the GROUPKEY-PULL exchange.

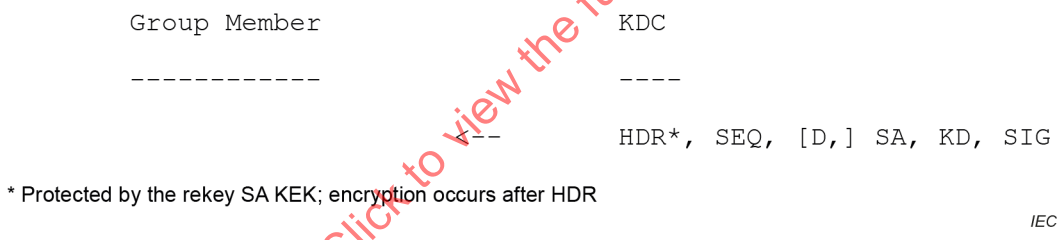



**Figure 43 – GROUPKEY-PUSH message (from RFC 6407)**

The signature payload is created using the corresponding private key to the certificate indicated in the phase 1 exchange (see 8.3.5.3).

A KDC may use the GROUPKEY-PUSH to send control information to the group to update SAs, which may also result in excluding members from the group. To ensure that the updated information has been received by the GMs, RFC 8263 defines the GROUPKEY-PUSH acknowledgement message, which may be used to acknowledge the key delivery towards the KDC (see Figure 44).

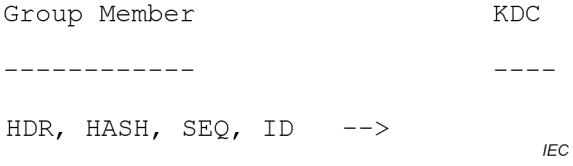



**Figure 44 – GROUPKEY-PUSH ACK message (from RFC 8263)**

### 8.6.2 GROUPKEY-PUSH Message

When sending a GROUPKEY-PUSH message from the KDC as shown in Figure 43, a SIG payload shall be included. As per RFC 6407, this SIG payload shall provide the signature value calculated over the complete GROUPKEY-PUSH message. This also includes the ISAKMP header HDR.

The KDC shall include the KEK_ACK_REQUESTED attribute (defined in RFC 8263) in the SA KEK payload only for GROUPKEY-PUSH messages to avoid the generation of acknowledgement messages for GROUPKEY-PULL messages.

If a GROUPKEY-PUSH is attempted to be sent to an unreachable address, the GROUP-PUSH-UNREACHABLE-DESTINATION event shall be raised.

### 8.6.3 GROUPKEY-PUSH acknowledgement message

RFC 8263 specifies the ability of a KDC to request that the GM return an acknowledgement of receipt of its rekey message using the KEK_ACK_REQUESTED attribute associated with the KEK and specifies the acknowledgement method.

As seen in Figure 44, the acknowledgement message contains a HASH value. The hash is calculated according to RFC 8263 as following.

```
HASH = prf(ack_key, SEQ | ID)
```
IEC

**Figure 45 – GROUPKEY-PUSH ACK hash computations**

The utilized `prf` is determined by the corresponding SA-KEY carrying the KEK_ACK_REQUESTED attribute.

Note that (according to RFC 6407, section 5.3.5) this is either explicitly defined in the SA KEK through the SIG_HASH_ALGORITHM directly or implicitly by the selected SIG_ALGORITHM for the algorithms SIG_ALG_ECDSA-256, SIG_ALG_ECDSA-384, or SIG_ALG_ECDSA-521.

The `ack_key` applied is derived from the `base_key` according to RFC 8263 as following

```
ack_key = prf(base_key, "GROUPKEY-PUSH ACK" | SPI | L)
```
IEC

**Figure 46 – GROUPKEY-PUSH ack_key computations**

The `base_key` shall be the key received from the KDC in the corresponding GROUPKEY-PUSH message. This provides assurance to the KDC that the new key may be used accordingly. The utilized function for the key derivation shall be the same as signalled in the SA KEK. According to RFC 8263

- SPI: The I-Cookie and R-Cookie are used as the SPI. The cookies contain the 16 bytes of the SPI of the SA KEK provided during the PULL-Exchange. This will allow matching of GROUPKEY-PUSH messages of interest.
- L: length field matching the number of bits in the ack_key. L shall match the length of the base_key. The value L is represented as two octets in network byte order

In the context of this document the SA KEK attributes are selected to be consistent throughout the GDOI key management messages. Therefore, the SA KEK attribute are restricted to the following combinations of values:

- SIG_HASH_ALGORITHM=3 (corresponds to SIG_HASH_SHA256)

- KEK_ACK_REQUESTED = 1 (corresponds to REKEY_ACK_KEK_SHA256)
- Length "L" used for ACK SIG calculation should be 512 (0x02,0x00)

or

- SIG_HASH_ALGORITHM=5 (corresponds to SIG_HASH_SHA512)
- KEK_ACK_REQUESTED = 3 (corresponds to REKEY_ACK_KEK_SHA512)
- Length "L" used for ACK SIG calculation should be 1024 (0x04,0x00)

RFC 8263 specifies that the GM has to send the acknowledgement message to the source port of the corresponding GROUPKEY-PUSH request of the KDC. As ephemeral ports need to be avoided to cope with for NAT and Firewalls on the path, this specification strongly recommends utilizing the port number 848 on the KDC when sending the GROUPKEY-PUSH message. This port number is registered at the IANA [51] for GDOI.

In consequence of 8.6.2 GMs shall only send GROUPKEY-PUSH ACK messages for received GROUPKEY-PUSH messages containing a SA KEK with the KEK_ACK_REQUESTED attribute.

A GKCS that does not achieve the levels to provide the KDA to the publisher, shall raise the KDA-FAILURE event.

## 8.7 Operational considerations

### 8.7.1 General

This subclause addresses operational considerations for handling group key management.

### 8.7.2 Group Security Policy

A group security policy should contain information about specific situations in which either the KDC is not available or a specific key, identified by the KeyID, is not available at a GM.

- If a KDC is not reachable by the GMs and the time for a key rotation is approaching, the current TEK shall be used until the KDC is available again. The GM signals the prolonged usage of the TEK key by continued use of the corresponding KeyID in the application protocol. In addition, the GM shall issue a security event ("warning: connection to KDC not available, prolonged usage of TEK"). GMs shall be able to be configured to accept messages secured with an expired group key (TEK) for a limited time interval. The initial value shall be less than 1 hour. During that time, the GM shall try to regain connectivity to the KDC. If the GM was not able to contact the KDC in that period, it may renew and start at an hour again. In any case, a security event shall be used to report this situation.
- If a GM detects it does not possess the TEK for a specific KeyID signalled in the application protocol, it shall immediately initiate a GROUPKEY-PULL request to the KDC. The GM shall also issue a security event ("warning: current TEK not available, GROUPKEY-PULL initiated").

### 8.7.3 Group dynamicity

#### 8.7.3.1 General

Groups may be stable over a certain period of time. But changes in the group membership are likely due to new setups or device replacements or identification of compromised devices or others. To manage dynamic groups, it is necessary to support functions like:

- Add: adding a new member to a group
- Delete: deleting a member of the group without updating the existing group key
- Revoke: revoking a group member leading to an immediate change of the group key and a deletion of the KEK with the revoked group member.

A precondition for becoming a group member is the availability of credentials (X.509 public-key certificates) and trust anchors (common Root CA certificate) on the KDC and GDOI client to be able to establish a security association. A further precondition is the configured KDC address for a (new) group member (GM). The association of a client to a group is based on an organization's security policy.

These group functions directly relate to the utilized distribution methods (PULL and PUSH) to handle the security parameter corresponding to the group. Approaches for group operations are described in the following subclauses.

#### 8.7.3.2 Adding group members

Adding a group member to an existing group may be done in different ways:

- PULL: GM is configured for group participation and uses a PULL request (GROUPKEY-PULL) to query the security parameters (KEK, TEK) for the specific group from the KDC. The KDC delivers the information upon successful verification of the requests including the authentication. GM is able to participate in group communication (either as publisher or subscriber).
- PUSH: If the new GM is configured at the KDC, the KDC may PUSH the current TEK to the new GM. Since the GM does not possess the KEK to decrypt the TEK, it will perform a PULL to fetch the complete set of security association parameter (including the KEK).

As the data exchanged before the new GM is added has no specific confidentiality requirements, a key update during the addition of a new GM is not required.

#### 8.7.3.3 Deleting group members

Deleting a group member from one group does not require immediate removal of the GM from that group and thus does not require an immediate key update. This may be the case when an IED is moved from one bay to another and still retains the trust relation to the KDC. On the KDC this would result in a reconfiguration by adding the GM to a different group.

Deleting a GM from the old group will be effective with the next key update, which may be done in different ways:

- PULL: Before the next key period, each GM will fetch the new TEK using a GROUPKEY-PULL request. The KDC will not deliver the updated TEK to the removed GM. The removed GM on the other hand is not expected to ask for the new TEK.
- PUSH: Key updates may be initiated from the KDC as unicast (phase 2) GROUPKEY-PUSH informational exchange message carrying a 'delete' payload as outlined in 8.6.2. The TEK protocol ID (GDOI_PROTO_IEC_61850) is carried in the Protocol-ID field of the message. The receiving GM shall then delete the TEK associated with the specific protocol but retain the KEK with the KDC. Using a multicast GROUPKEY-PUSH with a 'delete' payload to all group members will result in the deletion of the group association at the group member and shall be omitted.

  Note that as PUSH support is optional, an aligned handling of deletion of group members in mixed groups of GMs supporting PULL and PUSH is necessary to avoid non-availability situations. GMs not supporting PUSH will continue to utilize the current TEK until the next PULL action.

#### 8.7.3.4 Revoking group members

Revoking of a GM does require immediate removal from the group and thus requires an immediate key update. This may be the case when an IED is known to be compromised and therefore no longer considered trusted. In this case, not only the TEK is deleted, but also the KEK shall be changed. This process is comparable to the revocation of a certificate. Note that certificate revocation may be triggered in parallel. The revocation of group members is done as following:

- KDC uses a Phase 2 GROUPKEY-PUSH message with a new TEK encrypted with a new KEK to all group members. This will require all GM to initiate a GROUPKEY-PULL as they do not possess the new KEK to decrypt the TEK. The KDC shall reject the revoked group member.
- All GMs will initiate a GROUPKEY-PULL of the security parameter for the assigned group as described in 8.7.3.2. The revoked group member shall be rejected at the KDC. Note that this requires that the GM only delete the SA information for a specific group, but not the group information, so that the group members will be able to query for new security parameters.

### 8.7.4 Handling of Key Delivery Assurance (informative)

Key Delivery Assurance (KDA) provides assurance that the security parameters have been received and may be applied. This is useful in situations in which the key switching is done based on the availability of the new key at all GMs. It should not be used if the key scheduling is based on time windows. Switching of keys based on key reception by group members as described in 8.5.10.3 is optional.

An information about the delivery and reception of key and policy information is provided in GDOI in two directions. A GM signals the reception of the SA parameters as acknowledgement to the KDC depending on the key distribution approach (PULL/PUSH). In addition, the KDC may provide the information about the distribution of SA information to all group participants to a multicast source (publisher) of a group (see also 8.5.10.3):

- KDA provided from GM to KDC to acknowledge receiving (updated) SA parameter (TEK)
  - PULL: the GM acknowledges the reception of the TEK and the connected policy with the third message in the GOUPKEY-PULL exchange according to RFC 6407, section 3.2.
  - PUSH: if requested from KDC during the GROUPKEY-PUSH, the GM sends the Key Acknowledgement to the KDC.
- KDA provided from KDC to GM (group publisher) to signal that the key (TEK) has been received by all group members. This has to be distinguished for the two approaches for key delivery.
  - PULL: In the PULL case, the KDA could only be provided to the last GM fetching the updated security parameter. There is currently no signalling to the other GMs except by using a GROUPKEY-PUSH with an informational payload.
  - PUSH: The KDC may provide information about the reception of SA information at the other GMs to a multicast source in a group using a GROUPKEY-PUSH with an informational payload. The behaviour of KDA in a group, whose policy is GROUPKEY-PULL only is indeterminate and should be avoided.

## 9 Protocol Implementation Conformance Statement (PICS)

### 9.1 General

Static conformance requirements specify what shall be implemented, what may be implemented and what shall not be implemented for an implementation claiming conformance to this document.

### 9.2 Notation

The following notations are used for specifying conformance requirements:

m: Mandatory support. The item shall be implemented.

o: Optional support. The item may be but need not be implemented.

c: Conditional support. The item shall be implemented as specified by the condition.

x: Excluded. The item shall not be supported.

### 9.3 Conformance to general key management requirements

Table 5 shows PICS for general key management.

**Table 5 – PICS for general key management**

| Item | Description | Client/Server | Reference |
|---|---|---|---|
| G-1 | Required cryptographic materials | m | 6.3 |
| G-2 | Random Number Generation | m1 | 6.4 |
| G-3 | Recognition for Object Identifiers Branches for AVL<br>– AVL extensions: **avl62351Extiona**<br>– AVL entry extensions: **avl62351EntryExt**<br>– Protocol Identifiers **id-62351prot**<br>Support of AVL extensions<br>– AVL scope restriction: **scopeConstraints**<br>– AVL protocol restrictions<br>Support of AVL entry extensions<br>– AVL pinning of certificates: **pinningId** | c | 6.5.2<br><br><br><br><br><br>7.8.3<br>7.8.4<br><br>7.8.5 |
| G-4 | Mechanism for announcing security events according to IEC 62351-14 | o | 6.2 |
| m1: RNGs are mandatory to support in entities. If an entity has not enough entropy is should be provided with a seed out-of-band in a secure way.<br>c: if AVL are supported, the stated AVL extensions and AVL entry extensions may be supported. | | | |

### 9.4 Conformance to requirements for asymmetric key management

Table 6 shows PICS for asymmetric key management.

**Table 6 – PICS for asymmetric key management**

| Item | Description | End entity | PKI | Reference |
|---|---|---|---|---|
| A-1 | Public-Key certificates components according to Table 1 | m | m | 7.2.1 |
| A-2 | Attribute certificates components according to Table 2 | o | m | 7.2.2 |
| A-3 | Private and public key generation and installation | m | m | 7.3.1 |
| A-4 | Cryptographic key protection | m | m | 7.3.2 |
| A-5 | Use of existing security key management infrastructure | o | m | 7.3.3 |
| A-6 | Entity registration for identity establishment | - | m | 7.3.5 |
| A-7 | Entity configuration | m | - | 7.3.6 |
| A-8 | Entity enrolment | o1 | m1 | 7.3.7 |

| Item | Description | End entity | PKI | Reference |
|---|---|---|---|---|
| A-9 | Trust anchor information update | o | o | 7.3.8 |
| A-10 | Public-key certificate verification | m | m | 7.4.3, 7.4.4 |
| A-11 | Attribute certificate verification | o | m | 7.4.3, 7.4.5 |
| A-12 | Certificate revocation | c | m | 7.5 |
| A-13 | Certificate expiration and renewal | m | m | 7.6 |
| A-14 | Clock synchronization and accuracy | m | m | 7.7 |
| A-15 | Secure time synchronization | o | o | 7.7 |
| A-16 | Support of authorization validation lists | o | o | 7.8 |
| o1: key management for an entity may be manual/EST/SCEP<br>m1: The PKI must support automated enrolment with SCEP and EST.<br>c: the end entity shall support at least one revocation mechanism out of CRL or OCSP | | | | |

## 9.5 Requirements for group-based key management

Table 7 shows PICS for group-based key management.

**Table 7 – PICS for group-based key management (valid for KDC and Client)**

| Item | Description | m/o/c | Reference |
|---|---|---|---|
| S-1 | Group based key management | c | 8 |
| S-2 | GDOI requirements | c | 8.1 |
| S-3 | Internet Key Exchange Version 1 (IKEv1) | c | 8.2 |
| S-4 | Phase 1 IKEv1 main mode exchange type 2 | c | 8.3 |
| S-5 | Phase 1/2 informational messages | c | 8.4 |
| S-6 | Phase 2 GDOI PULL exchange | c | 8.5 |
| S-7 | Phase 2 GDOI PUSH exchange | o | 8.6 |
| c: when GOOSE or SV security is supported | | | |

## 9.6 Supported GDOI Payload OIDs

The PICS shown in Table 8 are for the case that GDOI is supported in order to provide the key material and security policy for GOOSE, SV, or PTP integrated security.

**Table 8 – PICS for supported OIDs for the identification payload**

| | | m/o/c | | |
|---|---|---|---|---|
| Item | Description | Client | KDC | Reference |
| P-1 | 61850_ETHERNET_GOOSE | o | m | 8.5.5.2.2 |
| P-2 | 61850_UDP_ADDR_GOOSE | o | m | 8.5.5.2.2 |
| P-3 | 61850_UDP _Tunnel | o | o | 8.5.5.2.2 |
| P-4 | 61850_ETHERNET_SV | o | m | 8.5.5.2.2 |
| P-5 | 61850_UDP_ADDR_SV | o | m | 8.5.5.2.2 |
| P-6 | 61850_IP_ISO9506 | o | o | 8.5.5.2.2 |
| P-7 | 61850_9_3_PTP | o | o | 8.5.5.2.2 |
| | | | | |

# Annex A
(informative)

## Relations to other parts of IEC 62351 and other IEC documents

Figure A.1 illustrates the connections between this document and other parts of the IEC 62351 series.

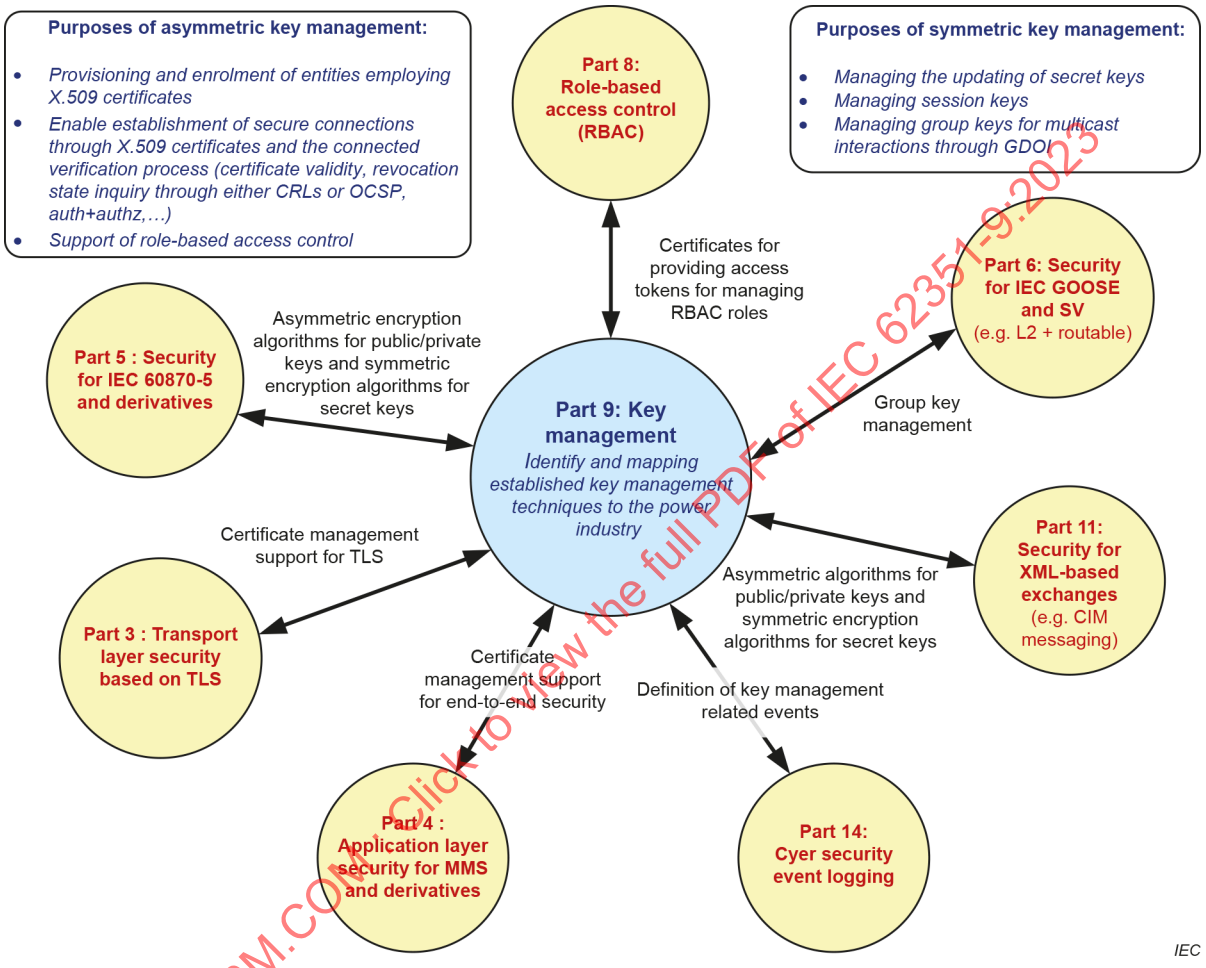



**Figure A.1 – IEC 62351-9 relationship to other parts of IEC 62351**



Asymmetric keys are used in most parts, primarily for authentication and the secure transport or negotiation of symmetric keys:

- IEC 62351-3
- IEC 62351-4
- IEC 62351-5
- IEC 62351-6
- IEC 62351-8
- IEC 62351-11

Symmetric key distribution is needed in most parts of IEC 62351. The focus is on those parts of IEC 62351 which use symmetrical keys at the application layer. This relates mainly to IEC TS 62351-5, IEC 62351-6 and to the group based key management approach in IEC 61850-90-5. For the other parts, using TLS, symmetric key handling is encapsulated within TLS itself and it may be rather a policy issue (cipher suites).

- IEC 62351-3 and IEC 62351-4, as part of the TLS handshake.
- IEC 62351-5 for the protection of IEC 60870-5 and derivate protocols.
- IEC 62351-6 for the protection of GOOSE and Sampled Values (SV) as well as R-GOOSE and R-SV.
- IEC 62351-8 for Profile C.

IEC 62351-10 provides recommendations and guidelines while IEC 62351-9 is directly addressing the requirements for key handling.

IEC 62351-14 provides security event logging, which can be used also for key management. The security events for this document are defined throughout the document and mapped to IEC 62351-14 in Annex D.

# Annex B
(informative)

# Cryptographic algorithms and mechanisms

## B.1 Trust and trust anchor

One important part of securing digital communication is the need for trust. Entities (systems or devices) should only accept data (communicate) with entities that they may authenticate and trust (see 4.2.3). Public-key certificates provide the basis to establish such trust by asserting the association of a public-key certificate with a unique entity. Trust for a particular public-key certificate is established by validating a so-called certification path that has its root in a trust anchor that is trusted by the entity doing the validation, also called the relying party. A certification path is a chain of public-key certificates starting with the public-key certificate signed by the trust anchor ending with the public-key certificate to be validated. The public-key at the of the certification path is an end-entity-public-key certificate. The other public-key certificates, if any, are CA certificates. The concepts of relying party, trust anchor, and certification paths are further described in clauses 6, 7.5, and 7.7 of ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019).

In some cases, trust may be established directly between a trust anchor and the relying party. However, a chain of trust may be established if direct access between these parties is not feasible, which is often the case for power system operations. If a relying party receives a public-key certificate from an entity with which it has a trust relationship, this public-key certificate may be validated without going through the complete certification path, instead relying on the chain of trust. In this case, the public-key certificate may not need to be issued and signed by the trust anchor but may be signed by a certification authority that has established a chain of trust back to the trust anchor.

For power system implementations, certification authorities might be the relevant organizational units within a company, the company itself, a governmental entity, or an accepted third party.

Public-key digital certificates have a finite period of validity, after which they expire. Trust may also be revoked in the event of a compromise or a change in use of the relying party. Management of public-key certificates is closely related to cryptographic key management and is therefore covered in this document.

See also 5.7 and 5.8.

## B.2 Cryptographic algorithms

### B.2.1 Introduction

Secure cryptographic algorithms are important components of a security policy. They therefore play a major role within PKI and related technologies, such as transport layer security (TLS) as defined by IETF RFC 8446. A detailed description of cryptographic algorithms is beyond the scope of this document. The intention is to provide sufficient information to give a high-level understanding of the concepts with references to more detailed specifications.

Protocols not using ASN.1 for syntax specification typically use character strings or similar to identify cryptographic algorithms. Such character strings are typically registered by Internet Assigned Numbers Authority (IANA) to ensure uniqueness. As examples:

a) TLS uses multiple cryptographic algorithms during an instance of communication called a cipher suite. A cipher suite is identified by a two octets identifier registered by IANA. Note that recommendations of cipher suites for power systems are handled in IEC 62351-3.

b) In other cases, like Internet Key Exchange version 2 (IKEv2), a cryptographic algorithm is identified by both a character string and an integer.

When the abstract syntax notation one (ASN.1) is used for specification of communication protocols, a cryptographic algorithm is identified by an object identifier and, if relevant, by parameters particular for the type of algorithm. ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019) uses the following ASN.1 information object class for identifying cryptographic algorithms in protocol specifications using the ASN.1 notation:

```
ALGORITHM::= CLASS {
  &Type          OPTIONAL,
  &DynParms      OPTIONAL,
  &id            OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
  [PARMS         &Type]
  [DYN-PARMS     &DynParms]
  IDENTIFIED BY &id }
```

where the `&Type` field is used for specifying parameters, if any, providing additional algorithm specification and the `&DynParms` field specifies parameters relevant when deploying the cryptographic algorithm.

Cryptographic algorithms are defined in many different specifications, typically in IETF RFCs and in National Institute of Standards and Technology (NIST) specifications. For easy reference and for easy import into ASN.1 modules, Annex B of ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019) provides copies of some relevant cryptographic algorithm definitions.

The security properties of a cryptographic algorithm depend on:

a) The security parameters of the algorithm. This is discussed in B.2.2;

b) The processing speed of computers, which determines the feasibility of breaking an algorithm by brute force, i.e., trying all possibilities. It is asserted that there may be adversaries with access to abundant computing power.

c) The cryptographic algorithm in use. An algorithm may have features that allow skilled mathematicians to find ways to break it that do not require a complete brute force attack.

An algorithm may be implemented in such a way that it does not provide the level of security it should theoretically offer. As an example, if a poor random number generator is used for generating cryptographic keys, some values are more likely than others. This may be used by an adversary to reduce the effort in revealing the key.



### B.2.2 Security strength

Security strength is a number associated with the amount of work (that is, the number of operations of some sort) required to break a cryptographic algorithm. The security strength is expressed in terms of "number of bits" and is a discrete value from the set {80, 112, 128, 192, 256}.

NIST SP 800-57 Part 1 revision 5 [57] and German BSI TR 02102-1 [58] have detailed consideration on security strength.

## B.3 Public-key algorithms

### B.3.1 General

In asymmetric cryptography, an entity is assigned two mathematically related keys, a private key, that the user is assumed to protect from disclosure and a public-key that may be more freely distributed. Although the two keys are interrelated, it is considered infeasible to determine the private key from the corresponding public key.

If the public key is broken in a way that discloses the private key or the private key is known to be otherwise compromised, the data it is supposed to protect is now unprotected. For example, it is possible to produce false digital signatures.

NIST FIPS PUB 186-5 [64] is a guide on digital signatures issues by NIST. In addition to specifying digital signature algorithms, it also includes public-key algorithm specifications.

### B.3.2 The RSA public-key algorithm

#### B.3.2.1 General

RSA (Rivest–Shamir–Adleman) is one of the first public-key cryptosystems. The acronym RSA is the initial letters of the surnames of Ron Rivest, Adi Shamir, and Leonard Adleman. The RSA public-key algorithm is currently the most frequently used public-key algorithm.

The RSA public-key algorithm may be utilized with different key length (e.g., 2048-bit, 3072-bit, 4096-bit). In contrast to some other public-key algorithms, the RSA public-key algorithm also allows encryption of data. Data encrypted by the public key may be decrypted by the corresponding private key and vice versa. The encryption/decryption is process heavy and should only be used for the following two purposes:

a) Digital signature generation and verification.

b) Transport of symmetric key.

A private key used as digital signature key is recommended not to be used for key transport.

#### B.3.2.2 Key generation

An RSA public key consists of a modulus $n$ and a public-key exponent $e$, i.e., the public key is $(n,e)$. Modulus $n$ is the product of two positive large prime integers $p$ and $q$, i.e., $n = p \times q$. The public-key exponent $e$ is limited to be within a specific range determined by $n$, $p$ and $q$. Conventionally the prime value $2^{16} - 1 = 65537$ is used.

An RSA private key consists of the same modulus $n$ and a private-key exponent $d$, i.e., the private key is $(n, d)$, where the private-key exponent $d$ also fulfils certain mathematical requirement determined by $e$, $p$ and $q$ in such a way that if $p$ and $q$ is known, $d$ can be calculated.

#### B.3.2.3 Security considerations

Both modules $n$ and the public-key exponent are made publicly available and are therefore not secret. The private-key exponent $d$ has to be kept secret. However, $d$ may be calculated if the two factors $p$ and $q$ are known. The security of the RSA private key is dependent on the difficulty of finding the two factors when their multiplication $n$ is known. This is known as the factorization problem. It has been shown that it is infeasible even for a powerful computer to find the two factors if the public-key length is sufficiently large.

NOTE The appearance of quantum computers in the horizon may change this. See B.9.

There are different recommendations regarding the minimum key length to be used with the RSA algorithm. Commonly used references comprise NIST SP 800-57 Part 1 [57] and the German BSI TR 02102-1 [58] recommending acceptable key length. According to German BSI TR 02102-1 a key size of 2048 bits is considered sufficient at least till 2023. The use of 2048 bit key length should be handled with care according to an organization's security policy. It is strongly recommended to switch to a minimum key length of 3072 bits.

### B.3.3 The DSA public-key algorithm

The digital signature algorithm (DSA) algorithm was originally defined in FIPS PUB 186 [63]. It was developed by the United States National Security Agency (NSA). However, in FIPS 186-5 (currently a draft) [64], NIST disallows this algorithm for generation of digital signatures and only allows it for verifying digital signatures.

### B.3.4 The ECDSA public-key algorithm

#### B.3.4.1 General

Elliptic curve digital algorithm (ECDSA) is a set of cryptographically secure public-key algorithm based on the elliptic curve cryptography (ECC). The ECC cryptography is superior to the RSA cryptosystem, as ECC uses smaller keys and signatures than RSA for the same level of security and allows fast key generation, fast key agreement and fast signature handling.

ECDSA is a specification for a set of algorithms based on the general elliptic curve equation, also called the Weierstrass curve:

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

This curve is used over prime field p, where p is a large odd prime. This will result in:

$$y^2 \equiv x^3 + ax + b \text{ (mod p)}$$

This formular result in a group of curve points with integer coordinates. The concept of cyclic groups as discussed in B.8.2 may also be applied to curve points. It can be shown that a subgroup of the defined points forms a cyclic group. It can also be shown that the number of points on the curve divided by the number of points of points 'n' in such a subgroup is always an integer called the cofactor 'h'. If the cofactor is '1', all the points on the curve forms a cyclic group. Carefully selected curves have a cofactor of '1'.

A specific curve is specified as a set of parameters, called domain parameters. The curve parameters 'a', 'b', and 'p' in above equations are three such domain parameters. Other parameters are the number of point 'n' in the subgroup and the cofactor h. The point on the curve used as the basis for defining a subgroup is called the generator point 'G' is also a domain parameter. This domain parameter is carefully selected by the curve designer..

Coordinates (x,y) fulfilling above equation for certain values of a and b, and where x and y are integers equal or greater than 0 and smaller than p, form a group of coordinates that are relevant for the specification of ECDSA algorithms.

A set of domain parameters for a curve is assigned an object identifier, which then identifies a particular ECDSA algorithm.

NIST FIPS 186-5 (draft) [64] provides supplementary information to the ECDSA public-key algorithms.

ECDSA is according to IETF RFC 3279 [81] formally specified as:

```
ecPublicKey ALGORITHM::= {
  PARMS         OBJECT IDENTIFIER
  IDENTIFIED BY id-ecPublicKey }

id-ecPublicKey OBJECT IDENTIFIER::= { iso(1) member-body(2) us(840) ansi-
x962(10045) keyType(2) ecPublicKey(1) }
```

All ECDSA curves are identified by this information object, while the **PARMS** field identifies the individual curves by their assigned object identifiers.

### B.3.4.2 Defines curves

The following Weierstrass elliptic curves are recognized by this document:

a) **secp256r1** (by NIST called P-256) is an elliptic curve over a 256-bits prime field. It is assumed to have a 128 bits security level.

   Recommended values for the prime and other parameters are given in SEC 2, version 2.0.

b) **brainPoolP256r1** is also an elliptic curve over a 256-bits prime field. It is also assumed to have a 128 bits security level.

   Recommended values for the prime and other parameters are given in IETF RFC 5639.

According to IETF RFC 5480 [82], the following object identifier is allocated to **secp256r1**:

```
secp256r1 OBJECT IDENTIFIER::= { iso(1) member-body(2) us(840)
          ansi-x9-62(10045) curve(3) prime(1) 7 }
```

According to IETF RFC 5639 [37], the following object identifier is allocated to **brainpoolP256r1**:

```
brainpoolP256r1 OBJECT IDENTIFIER::= { iso(1) identified-organization(3)
   teletrust(36) algorithm(3) signature-algorithm(3) ecSign(2)
   ecStdCurvesAndGeneration(8) ellipticCurve(1) versionOne(1)
   brainpoolP256r1(7) }
```

### B.3.4.3 Key generation

For both curves discussed above, a private key is generated as a 256 bits random bit string.

The corresponding public key is the coordinates of a point on the elliptic curve, i.e., the public key is made up of two components, an x-value, and a y-value. It is calculated taking a point on the curve, called the generator, $G$ and multiplying that point with the private key using the special multiplication rules for elliptic curves. $G$ is fixed for a particular curve and is one of the curve parameters.

The public key may then be expressed as:

$$P = k \circ G$$

where $P$ is the public key, $k$ is the private key and $\circ$ is the special multiplication sign.

A point on the curve may be given in an uncompressed form, i.e., a concatenation of the x-coordinate and the y-coordinate. A point may also be given in a compressed form where only the x-coordinate is given, and the y-coordinate may then be calculated from the curve equation where y can take both a positive value $y_1$ and a negative value $-y_1$. As a result of the modular arithmetic $y_2 \equiv (-y_1 + p)$ to get a value in the range $\{0, \ldots, p-1\}$ resulting in two values $y_1$ and $y_2$.for the y-coordinate. By necessity, if one value is even, the other on is odd and visa verse. If $P = k \circ G$ results in an even y-value, the x-coordinate is prefixed a 0x02 octet and if it results in odd y-value, the x-value is prefixed a 0x03 octet..

If the uncompressed form is used, the result is prefixed a 0x04 octet.

### B.3.4.4 Security issues

As indicated above, the public key is calculated as:

$$P = k \circ G$$

Calculating the public key $P$ when the private key $k$ is known is a fast operation. Knowing the public key $P$ and trying to find the private key $k$ to fulfil above equation has no efficient solution (for carefully selected finite field and elliptic curve). This is known as the elliptic curve discrete logarithm problem (ECDLP).

NOTE The appearance of quantum computers in the horizon may change this. See B.9.

### B.3.5 The EdDSA public-key algorithms

#### B.3.5.1 General

Edwards-curve Digital Signature Algorithm (EdDSA) is also a cryptographically secure public-key algorithm. It is quite different from ECDSA. While det curves defined for ECDSA are all variation of the Weierstrass curve mentioned in B.3.4.1.

EdDSA is based on the general twisted Edwards curve:

$$ax^2 + y^2 = 1 + dx^2y^2$$

Only two curves are defined using this general equation. The mathematics behind type of curve is different from the one used for the Weierstrass curve result different techniques for generating public keys and digital signatures.

The twisted Edwards curve is a variation of (birationally equivalent with) the general Montgomery curve.

$$By^2 = x^3 + Ax^2 + x \; mod \; p$$

The Montgomery curve is used for two key agreement (Diffie Hellman) algorithms labelled X25519 and X448.

The two curves used for EdDSA are called Edwards15519 and Edwards448.

The two curves used for key agreement are called Curve15519 and Curve448.

In contrast io other types of digital signature algorithm, the two defined curves have as a parameter an associated hash algorithm, SHA512 is specified for the Curve25519/Edwards25519 curves and SHAKE256 is specified for Curve448/Edwards448 curves.

EdDSA has some advantages over ECDSA:

a) It has better performance.

b) Does not require a unique random number for each invocation.

c) it is more resilient to side-channel attacks.

d) Small public-keys and signatures.

NIST FIPS 186-5 (draft) [64] provides supplementary information about the EdDSA public-key algorithms.

#### B.3.5.2 Defined curves

The following curves are relevant for this document.

a) Ed25519: It based on an underlying prime field with $p = 2^{255} - 19$. It has a private key size of 256 bits. It is assumed to have a 128 bits security level.

b) Ed448: It based on an underlying prime field with $p = 2^{448} - 2^{224} - 1$. It has a private key size of 256 bits. It is assumed to have a 224 bits security level.

IETF RFC 7748 [43] specifies the parameters for the above curves.

According to IETF RFC 8410 [79], the following object identifiers are allocated for Ed25519 and Ed448:

```
id-Ed25519 OBJECT IDENTIFIER::= { iso(1) identified-organization(3) thawte(101) 112
}


id-Ed448 OBJECT IDENTIFIER::= { iso(1) identified-organization(3) thawte(101) 113 }
```

#### B.3.5.3 Key generation

IETF RFC 8032 [77] specifies in sections 5.1.5 and 5.2.5 key generation for the two curves.

For the Ed25519 curves a private key is generated as a 256 bits random bit string, while a 456 bits random string is generated for the Ed448 curve.

Generation of the public key is quite different from ECDSA. For both curves it involves a hashing of the private key. This means that a hashing algorithm is part of the domain components for the two curves. For Ed25519, SHA512 is used and for Ed448, SHAKE256 is used.

As discussed in B.3.4.3 for ECDSA, the public key is calculated as a curve point multiplied by the private key and where the private key may in principle be derived by an adversary by solving the ECDLP. For the EdDSA the public key is generated from the hash of the private which could be expressed as:

$$P = 'HASH'(k) \circ G$$

However, as the hash of the private key generates a digest having a size twice of the private key, only the first half of the digest is used for the calculation above.

#### B.3.5.4 Security issues

As seen from B.3.5.3, solving the ECDLP (see B.3.4.4) does not yield the private key, but the half hash of the private key. However, it can be shown, that this is enough for an adversary to generate a signature that will be accepted by the verifier.

Details on security issues are given in the security consideration section of IETF RFC 8032 [77].

#### B.3.5.5 Signature generation and verification

Details on signature generation and verification are given IETF RFC 8032 [77].

### B.3.6 Digital signature algorithms

#### B.3.6.1 General

A private key and a corresponding public key are mathematically related in the sense that certain actions performed by the private key, may be un-done by the public key, and in some cases also visa verse. So, a digital signature may be generated using the private key and may be verified by the public key.

A digital signature involves a hash algorithm. Some public-key algorithms allow a choice among multiple hash algorithms. This is the case for RSA and ECDSA public-key algorithms. Such combinations then need their own identities for reference when used as digital signature algorithms.

Some public-key algorithms have a specific hash algorithm as part of their parameters, which is also used for digital signature generation and verification. For such public-key algorithms there is no need for a different identification, when used as a digital signature algorithm.

#### B.3.6.2 RSA digital signature algorithms

##### B.3.6.2.1 General

The RSA public-key algorithm may together with selected hash algorithms make-up a set of digital signature algorithms.

There are two schemes for generation and verifying digital signatures:

a) RSA PKCS#1 v1.5;

b) RSA-PSS

Both schemes use the same basic technic. The data to be signed using an RSA digital signature algorithm is hashed by the sender using a secure hashing algorithm and the digest is then encrypted using the private key to generate the signature. The recipient decrypts the received encrypted digest using the corresponding public key and generate its own digest over the received data. If the two digests are identical, the digital signature is verified.

The encryption of the generated digest is quite simple. If *d* is the private-key exponent, *n* is the modules (see B.3.2.2) and $m$ is the digest of the message to be encrypted after being converted to an integer, then the encrypted digest $c$ is given by:

$$c = m^d \ (mod\ n)$$

The decryption is similar simple. If e is the public-key exponent, then the decrypted m is given by:

$$m = c^e \ (mod\ n)$$

##### B.3.6.2.2 The RSA PKCS#1 v1.5 scheme

There are several RSA PKCS#1 digital signature algorithms as listed in Annex A of IETF RFC 8017. The relevant digital signature algorithms for this document are:

a) **`sha256WithRSAEncryptionAlgorithm`**.

b) **`sha512WithRSAEncryptionAlgorithm`**.

These two digital signature algorithms have the following formal specifications:

```
sha256WithRSAEncryption ALGORITHM::= {
  PARMS NULL
  IDENTIFIED BY { pkcs-1 sha256WithRSAEncryption(11) };

sha512WithRSAEncryption ALGORITHM::= {
  PARMS NULL
  IDENTIFIED BY { pkcs-1 sha512WithRSAEncryption(13) };

pkcs-1 OBJECT IDENTIFIER::= { iso(1) member-body(2) us(840)
  rsadsi(113549) pkcs(1) }
```

#### B.3.6.2.3 The RSA PSS scheme

The RSA-PSS digital signature algorithm is formally specified by.

```
rSASSA-PSS ALGORITHM::= {
  Parms      RSASSA-PSS-params
  IDENTIFIED BY {pkcs-1 10} }

RSASSA-PSS-params::= SEQUENCE {
  hashAlgorithm     [0] AlgorithmIdentifier DEFAULT sha-1,
  maskGenAlgorithms [1] AlgorithmIdentifier DEFAULT mgf1SHA1,
  saltLength        [2] INTEGER             DEFAULT 20,
  trailerField      [3] TrailerField        DEFAULT trailerFieldBC,
  ... }
```

There is only one algorithm specified as it has the hash algorithm used as one the parameters.

There are some other parameters that indicates that the algorithm is somewhat more complicated than discussed in B.3.6.1. The details are outside the scope of this part of IEC 62351. See IETF RFC 8017 for details.

### B.3.6.3 ECDSA digital signature algorithm

#### B.3.6.3.1 General

Like for the RSA public-key algorithm, the ECDSA public-key algorithm may together with selected hash algorithms make-up a family of digital signature algorithms.

For the same level of security, the size of the digital signature is smaller for ECDSA than for RSA.

For signature generation the data to be signed is hashed. As ECDSA does not support encryption/decryption, the private key is used in another way to generate the signature. The verification follows the same rules using the public key as described in NIST FIPS 186-5 [64].

#### B.3.6.3.2 Formal specification of ECDSA signature algorithms

There are several ECDSA digital signature algorithms as listed in IETF RFC 5758 [80]. The relevant digital signature algorithms for this document are:

a) **`ecdsa-with-SHA256-Algorithm`**.

b) **`ecdsa-with-SHA512-Algorithm`**.

These two algorithms have the following formal specifications:

```
ecdsa-with-SHA256-Algorithm ALGORITHM::= { - IETF RFC 5758
  IDENTIFIED BY { iso(1) member-body(2) us(840) ansi-x962(10045)
    signatures(4) ecdsa-with-SHA2(3) 2 }}

ecdsa-with-SHA512-Algorithm ALGORITHM::= { - IETF RFC 5758
  IDENTIFIED BY { iso(1) member-body(2) us(840) ansi-x962(10045)
    signatures(4) ecdsa-with-SHA2(3) 4 }}
```

#### B.3.6.3.3 Digital signature generation and verification

The message to be signed is hashed and then converted to an integer as a first step for both digital signature generation and verification.

The digital signature is generated based on:

a) The private key.

b) The generator points of group of curve points (elements) on the elliptic curve.

c) The number of elements in the group.

d) A secure random number generated by secure random number generator. Alternatively, a secret number may be generated based to message to be signed and the private key referred to as deterministic ECDSA.

The digital signature is verified based on:

a) The digital signature.

b) The public key.

c) The number of elements in the group.

## B.4 Symmetric key algorithms

### B.4.1 Stream ciphers vs. block ciphers

Symmetric keys are used for two major types of ciphers, namely block cipher and stream cipher.

Block ciphers process blocks of data at a time with a fixed block size. If the length of the data to be encrypted is not a multiple of the block size, the data is padded, i.e., before encryption, dummy octets are added to make a multiple of the cipher's block size. These octets are then stripped off during the decryption phase.

Stream ciphers process data in one small unit at a time (typically an octet or even a bit). This allows for ciphers to process an arbitrary amount of data without padding.

Only block ciphers are considered in B.4.3 to B.4.4.

### B.4.2 Advance encryption standard

AES is a block cipher with a 128-bit block size (16 octet block size). The data to be encrypted is therefore split-up in 128-bit blocks, and the blocks are encrypted/decrypted one-by-one. Three key sizes are defined: a 128-bit size, a 192-bit size and a 256-bit size.

NIST FIPS 197 is the basic specification for how an AES-block is encrypted/decrypted.

How the blocks are combined for a complete encrypted/encryption of a message is specified separately for different AES modes. Below are discussed two modes relevant for this part of IEC 62351.

The encryption/decryption of a block goes though some steps, which then again are repeated in several rounds depending on the symmetric key size.

### B.4.3 Advanced encryption standard – cipher block chaining (AES-CBC)

The advance encryption standard – cipher block chaining (AES-CBC) symmetric-key algorithms are widely used. Three are key size variants of AES-CBC symmetric-key algorithms (128 bits, 192 bits and 256 bits). Only the 128 bits variant and the 256 bits variant are relevant for this document.

In cipher block chaining, the first block of plain text is modified by 'XOR'ing it with a random generated bit string of the same size as the block size, called the initialization vector. The result is encrypted using the symmetric key, which results in a block of cipher text. This cipher text block is then used to modify ('XOR'ing) the second block of plain text and the result is encrypted. This second resulting block of cipher text is then used to modify the third block of plain text, etc, until all blocks have been encrypted with the concatenated cipher text blocks as the final result.

When a message is not a multiple of the block size, the message is padded. PKCS#7 as defined in IETF RFC 2315 specifies a the most used padding technique. The basic idea is that the last octet of the padding tells the length of the patting allowing the recipient to remove the padding after decryption. To avoid ambiguity, even when a message is an exact multiple of the block size, it is padded,

The initialization vector is generated using a random number generator and a new value is generated for each invocation of the algorithm. However, the initialization vector is not a secret value. It is also used for decryption, and it is transferred to the recipient in the clear.

Prior to ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019) the **ALGORITM** information object class listed in B.2.1 did not have the **DYN_PARMS** field, but this field is now required for the AES algorithms according to ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019). This required redefinition of the algorithms and allocation of new object identifiers. The new definitions are defined by ISO/IEC 9595-11 | Rec. ITU-T X.510.

ISO/IEC 9594-8:2020 | Rec. ITU-T X.509 (2019) has allocated the following object identifier arc for cryptographic algorithms:

```
id-algo OBJECT IDENTIFIER::= {joint-iso-itu-t(2) ds(5) algo(44)
```

The formal definitions of the two AES-CBC algorithms are:

```
aes-cbc128 ALGORITHM::= {
  DYN-PARMS     AES-InitializationVector
  IDENTIFIED BY {id-algo aes(2) 1 } }

aes-cbc256 ALGORITHM::= {
  DYN-PARMS     AES-InitializationVector
  IDENTIFIED BY {id-algo aes(2) 3 } }
AES-InitializationVector::= OCTET STRING (SIZE (16))
```

### B.4.4 Advanced encryption standard – counter mode (AES-CTR)

The advanced encryption standard – counter mode (AES CTR) uses a counter mode technique where a 128 bits counter block consists of two fields, a nonce field concatenated with a counter field.

An initial counter value is used for the first block of plain text. The resulting counter block is AES encrypted, and the result is XORed with the first plain text block to create the first cipher text block. The following blocks of plain text are treated in the same way, except that the counter is increased with a fixed value for each block processed. The same nonce is used for the entire process. However, it is important that a new nonce is generated for each message to be encrypted.

As in B.4.3, it is necessary to use new definitions defined by ISO/IEC 9595-11 | Rec. ITU-T X.510.

The formal definitions for the two AES-CTR algorithms are:

```
aes-ctr128 ALGORITHM::= {
  DYN-PARMS     CtrNonce
  IDENTIFIED BY {id-algo aes(2) 13 } }

aes-ctr256 ALGORITHM::= {
  DYN-PARMS     CtrNonce
  IDENTIFIED BY {id-algo aes(2) 15 } }

CtrNonce::= OCTET STRING (SIZE (8))
```

The initial counter block structure chosen here is the second approach specified in section B.2 of NIST SP 800-38A.

The size of a counter block is 16 octets. The left most (most significant) 8 octets hold the nonce, while the remaining 8 octets are used as counter field. The counter field of the first block of a message shall have the hex value 0x0000 0000 0000 0001. The counter shall be incremented by '1' for each subsequent block of the message.

## B.5 Hash algorithms

A hash algorithm is the specification on how to take an arbitrary message (bit string) and produce a fixed length hash value called a digest. A good hash algorithm is designed to satisfy the following properties:

a) it is a one-way algorithm, meaning it is computationally infeasible to reverse the algorithm to find the message that maps to a specific digest;

b) it is infeasible to modify a message without changing the digest;

c) it is collision resistant, meaning it is computationally infeasible to find any two distinct messages that map to the same digest;

d) it provides for even a small change in the input bit string causing an unpredictable change of the output digest.

The application of a hash algorithm is useful for different purposes, including:

a) integrity protection;

b) part of the digital signatures process;

c) for generating fingerprints;

d) generation of a keyed-hash message authentication code (HMAC);

e) key derivation functions, i.e., a function that derives one or more symmetric keys from a shared secret value, such as a password;

f) random number generation.

NIST has in FIPS PUB 180-4 [66] published specifications for so-call secure hash algorithms (SHAs) to replace previous hash algorithms not considered sufficiently safe. This specification defines SHA-1, SHA-224, SHA-256, SHA-512, SHA-512/224 and SHA-512/256. SHA-1 is now considered insecure and therefore is not further considered. To ensure interworking, this document only considers SHA-256 and SHA-512.

SHA-256 applied to a message of less than $2^{64}$ bits results in digest of 256 bits (32 octets) and is assumed to have a security level of 128 bits.

SHA-512 applied to a message of less than $2^{128}$ bits results in digest of 512 bits (64 octets) and is assumed to have a security level of 256 bits.

When generating a digest, SHA-256 operates on block size of 64 octets, while SHA-512 uses a block size of 128 octets.

## B.6 Integrity check value (ICV) algorithms

### B.6.1 General

An integrity check value (ICV), also called a message authentication code (MAC), is generated or verified according to an ICV algorithm that takes as input the data to be protected and a symmetric key. It may only be used if two communicating entities share the same symmetric key. An ICV is added to the message it is intended to protect before transmission. An ICV has some similarities with a digital signature but is faster to compute.

The sender of data generates the ICV according to an ICV algorithm. The recipient generates an ICV using the same algorithm. If the two values are equal, the integrity and authenticity of the sender is assured by the recipient by some certainty.

An ICV algorithm has an underlying hash algorithm, and therefore the ICV algorithm assures integrity of the data to been protected, i.e., the recipient of protected data may assume with some certainty that if the ICV is verified, then the data has not been changed since the ICV was added by the sender. Also, if ICV is verified, the recipient may also assume with some certainty that ICV has been generated using the same symmetric key as the one used for verification. The recipient therefore has some assurance of the authentication of the sender, assuming that the shared symmetric key is generated independently of keys used for other pair-wise communications.

When encryption and ICV generation are separate processes, then:

a) Different symmetric keys should be used for the two processes.

b) It is recommended first to encrypt and then to generate the ICV over the encrypted data.

### B.6.2 Keyed-hash message authentication code (HMAC) algorithm

The HMAC algorithm is an ICV algorithm involving a cryptographic hash function and a secret symmetric key. It is defined in FIPS PUB 198-1 [67].

HMAC may be used in combination with any cryptographic hash function, but here only the following are considered:

a) **`hmacWithSHA256`**, which is based on SHA-256 hashing algorithm.

b) **`hmacWithSHA512`**, which is based on SHA-512 hashing algorithm.

It is recommended to use a shared symmetric key size having the same length as the block length of the used hash algorithm, i.e., 64 octets (512 bits) for **`hmacWithSHA256`** and 128 octets (1024 bits) for **`hmacWithSHA512`**.

IETF RFC 4231 [85] provides the formal definitions of these HMAC algorithms:

```
digestAlgorithm OBJECT IDENTIFIER::= {iso(1) member-body(2) us(840)
   rsadsi(113549) 2 }

hmacWithSHA256 ALGORITHM::= {
  PARMS         NULL
  IDENTIFIED BY { digestAlgorithm 9 } }

hmacWithSHA512 ALGORITHM::= {
  PARMS         NULL
  IDENTIFIED BY { digestAlgorithm 11 } }
```

### B.6.3 Advance Encryption Standard (AES) – Galois message authentication code (GMAC) algorithm

GMAC is based on the Galois/Counter Mode (GCM) of operation as specified in B.7.2. It uses the GCM specification for generating the tag (ICV) but does no encryption of plain text.

There are three defined AES-GMAC corresponding to the three key sizes defined for AES: `aes128-GMAC`, `aes192-GMAC` and `aes256-GMAC`. Only `aes128-GMAC` and `aes256-GMAC` are recognized by this document.

The formal definitions of the AES-GMAC algorithms are given in IETF RFC 9044 [86] for use in CMS. However, these algorithms have dynamic parameters and are therefore redefined by ISO/IEC 9594-11 | Rec. ITU-T X.510.

```
aes128-GMAC ALGORITHM::= {
  PARMS         MACLength
  DYN-PARMS     GMAC-nonce
  IDENTIFIED BY { id-algo aes(2) 29 } }

aes256-GMAC ALGORITHM::= {
  PARMS         MACLength
  DYN-PARMS     GMAC-nonce
  IDENTIFIED BY { id-algo aes(2) 31 } }

MACLength::= INTEGER (12 | 13 | 14 | 15 | 16)
GMAC-nonce::= OCTET STRING (SIZE (12))
```

The ASN.1 value `id-algo` is defined in B.4.3.

## B.7 Authenticated encryption with associated data (AEAD) algorithms

### B.7.1 General

The AEAD algorithms introduced in the following subclauses are all variants of AES ciphers.

The input to an AEAD algorithm includes four items:

a) symmetric key;

b) data that will be both authenticated and encrypted;

c) associated data that will be authenticated but not be encrypted;

d) nonce.

It is crucial for the security of AEAD algorithms that a new nonce is generated for each invocation of an AEAD algorithm.

The output of an encryption process is a concatenation of cipher text having the same length as the plain text to be encrypted and the tag. The output of a decryption process is the plain text and an indication about whether tag verified correctly or not.

### B.7.2 Advanced encryption standard (AES) – Galois/Counter Mode (GCM)

Advance encryption standard with Galois/counter mode (AES-GCM) is an efficient authenticated encryption with associated data (AEAD) algorithm. Especially when implemented in hardware it can operate with high speed and low latency.

The encryption part is like the AES-CTR algorithm. It uses a 16 octets counter block consisting of an initialization vector (IV) field concatenated with a counter field.

The IV field should consist of 12 octets (96 bits), for optimal performance and higher level of operability. The counter field for the first block of a message has the value 0x0000 0001 and is incremented.

If the same IV and symmetric key combination is used more than once, the implementation is vulnerable to attack. NIST SP 800-38D in section 8 states: "The probability that the authenticated encryption function ever will be invoked with the same IV and the same key on two (or more) distinct sets of input data shall be no greater than $2^{-32}$". The risk may be minimized by a good random number generator (RNG) and/or periodic symmetric key replacements.

There are three defined AES-GCM algorithms corresponding to the three key sizes defined for AES: **aes128-GCM**, **aes192-GCM** and **aes256-GCM**. Only **aes128-GCM** and **aes256-GCM** are recognized by this document.

The formal definitions of the AES-GCM algorithms are given in IETF RFC 5084 for use in CMS. However, these algorithms have dynamic parameters and are therefore redefined by ISO/IEC 9594-11 | Rec. ITU-T X.510.

```
aes128-GCM ALGORITHM::= {
  PARMS          GCM-ICVlen
  DYN-PARMS      GCM-nonce
  IDENTIFIED BY { id-algo aes(2) 17 } }

aes256-GCM ALGORITHM::= {
  PARMS          GCM-ICVlen
  DYN-PARMS      GCM-nonce
  IDENTIFIED BY { id-algo aes(2) 19 } }

GCM-ICVlen::= INTEGER (16)
GCM-nonce::= OCTET STRING (SIZE (12))
```

The ASN.1 value **id-algo** is defined in B.4.3.

For more information about GCM see NIST SP 800-38D [68].

### B.7.3 Advanced encryption standard (AES) – Counter with CBC-MAC (CCM)

CCM stands for counter with CBC-MAC and where CBC-MAC stands for cipher block chaining message authentication code.

CCM uses a counter mode symmetric key algorithm for encryption and decryption (see B.4.4). It uses a CBC-MAC algorithm for generation and verification of the authentication tag.

The CBC-MAC algorithm specifies a ICV generation and verification technique not using a hash, but encryption much in the same way as cipher block chaining, except it does not require an initialization vector and the last cipher block is used as tag and not concatenation of all cipher blocks.

There are three defined AES-CCM algorithms corresponding to the three key sizes defined for AES: **aes128-CCM**, **aes192-CCM** and **aes256-CCM**. Only **aes128-CCM** and **aes256-CCM** are recognized by this document.

The formal definitions of the AES-CCM algorithms are given in IETF RFC 5084 for use in CMS. However, these algorithms have dynamic parameters and are therefore redefined by ISO/IEC 9594-11 | Rec. ITU-T X.510.

```
aes128-CCM ALGORITHM::= {
  PARMS          CCM-ICVlen
  DYN-PARMS      CCM-nonce
  IDENTIFIED BY { id-algo aes(2) 25 } }

aes256-CCM ALGORITHM::= {
  PARMS          CCM-ICVlen
  DYN-PARMS      CCM-nonce
  IDENTIFIED BY { id-algo aes(2) 27 } }

CCM-ICVlen::= INTEGER (16)
CCM-nonce::= OCTET STRING (SIZE (12))
```

The ASN.1 value **id-algo** is defined in B.4.3.

For more information about CCM see NIST SP 800-38C [69].

## B.8 Diffie-Hellman key agreement

### B.8.1 General

The Diffie-Hellman key agreement method allows two parties that have to exchange so-called Diffie-Hellman public keys to jointly arrive at a shared secret in a way that does not allow an eavesdropper to learn about this shared secret. This shared secret may then be used to derive one or more symmetric keys (see B.9). NIST SP 800-56A, Revision 3 [72] gives detailed specification of the method.

The Diffie-Hellman algorithm also uses a variant of the public-key algorithms. A key pair may either be ephemeral, i.e., they are typically generated as needed and then only used once, or a key pair may be static, where a CA has certified the key pair and provided the public-key in a public-key certificates. There are then three possible modes of operation:

a) static-static, where both parties are in possession of static key-pairs
b) ephemeral-ephemeral: where both parties generate ephemeral key-pairs as needed
c) ephemeral-static, where one side generates a key pair as needed and where the other side is in possession of a static key pair.

### B.8.2 Introduction to cyclic groups

To understand the how the Diffie-Hellman key exchange relates the discrete logarithm problem, a short introduction to the group theory is necessary.

A group is defined as a set of elements together with a group operator. Only groups consisting of integers from 1 to p-1, where p is a prime is considered here. This may be written as:

$$\mathbb{Z}_p^* = \{1, 2, \ldots, p-1\}$$

As group operator a special way of multiplication is used:

$$a \circ b \equiv a \times b \bmod p$$

where ∘ is the group operator and × is the classic multiplication sign.

A cyclic group is a group where one or more elements are so-called generators. A generator g has the characteristics that the set of $\{g, g^2, \ldots, g^{p-1}\}$ $(mod\ p)$ has the same set of elements as $\{1, 2, \ldots, p-1\}$, just in another order.

The Diffie-Hellman key agreement is defined over a cyclic group.

Alice pics an arbitrary element $x_a$ within the group as her Diffie-Hellman (DH) private key and calculates her DH public key as $y_a = g^{x_a}$. Bob also pics an arbitrary element $x_b$ within the group as his DH private key and calculates his DH public key as $y_b = g^{x_b}$.

### B.8.3 Diffie-Hellman method over finite field

This method is specified in IETF RFC 2631 and uses discrete logarithm cryptography. In the method, two partners, here called Alice and Bob, want to exchange messages. Alice generates or is in possession of a private key $x_a$ associated with a public key, given by $y_a = g^{x_a}\ mod\ p$, where g, called the generator, is one of the agreed-upon parameters and where the prime p is another agreed-upon parameter. Likewise, Bob generates or is in possession of a private key $x_b$ associated with another public key $y_b = g^{x_b}\ mod\ p$.

The public keys are then exchanged by the two parties. Alice calculates:

$$ZZ = y_b^{x_a}\ mod\ p = (g^{x_b}\ mod\ p)^{x_a}\ mod\ p = g^{x_b x_a}$$

while Bob calculates:

$$ZZ = y_a^{x_b}\ mod\ p = (g^{x_a}\ mod\ p)^{x_b}\ mod\ p = g^{x_a x_b}$$

where ZZ is the shared secret.

### B.8.4 The discrete logarithm problem

As seen, there is a relationship between the DH private key and the corresponding DH public key. When knowing the values of $y_a$ and $g$ for $g^{x_a} = y_a$ for Alice could an adversary find the value $x_a$,i.e., Alice's DH private key by solving the equation $y_b = g^{x_b}$ or $log_g y_a = x_a$. If that would be possible, the adversary would be able to calculate the shared secret. This is known as the discrete logarithm problem.

There is no known efficient way to solve this problem, but the brute force way could be to calculate $g^2, g^3, \ldots$, until the result is $y_a$. With a large prime, e.g., 2048 bit prime, this is infeasible even for a fast computer.

NOTE The appearance of quantum computers in the horizon may change this. See B.9.

IETF RFC 3526 defines a list of cyclic groups with different length of the prime p and with a defined generator g. Each group is assigned a group number. By agreeing on a group, the parameters p and g are established between communication partners.

### B.8.5 Elliptic curve Diffie-Hellman key agreement

Elliptic curve Diffie-Hellman (ECDH) key exchange is based on the associative rules for elliptic curve multiplication.

In B.3.4.3 it is shown that the ECC public key is the generator point *G* multiplied by the private key *a*. Over an insecure channel the public keys $(a_x \cdot G)$ and $(a_y \cdot G)$ are exchanged and where $a_x$.is Alice's private key and $a_y$is Bob's private key.

Alice will then calculate $(a_y \cdot G) \cdot a_x$ and Bob will calculate $(a_x \cdot G) \cdot a_y$. According to the associative rules mention above, the calculated values are identical and therefore represent the shared secret.

Another way of stating it is that Alice's public key multiplied by Bob's private key is equal to Bob's public key multiplied by the Alice's private key, which again is equal to the shared secret.

### B.8.6 Key establishment algorithms

#### B.8.6.1 General

ISO/IEC 9594-11:2020 | Rec. ITU-T X.510 (2020) defines key establishment algorithms for combinations of the Diffie-Hellman methods and key derivation functions. Future, possibly quantum safe key establishment methods may also be expressed as key establishment algorithms. The purpose is to allow for negotiation and migration techniques similar to other types of cryptographic algorithms.

#### B.8.6.2 Diffie-Hellman group 14 algorithm with HKDF-256

The following is a specification for key establishment algorithm based on the Diffie-Hellman key agreement method and the key derivation method specified in B.9.

```
dhModpGr14Hkdf256Algo ALGORITHM::= {
  PARMS          Group14
  DYN-PARMS      Payload14
  IDENTIFIED BY id-algo-dhModpGr14Hkdf256Algo }

Group14::= INTEGER (14)

Payload14::= SEQUENCE {
  dhPublicKey OCTET STRING (SIZE (256)),
  nonce       OCTET STRING (SIZE (32)) OPTIONAL,
  ... }
```

The **PARMS** token specifies that the fixed parameters shall be those parameters specified for group number 14 for the 2048-bit MODP as specified in IETF RFC 3526.

The **DYN-PARMS** token specifies that the dynamic parameters shall be those specified by the **PayLoad14** data type.

The **PayLoad14** data type has the following components:

a) The **dhPublicKey** component shall specify the length in octets of the Diffie-Hellman public key to be used by the sender. For the used group, the size is always 256 octets.

b) The **nonce** component shall specify the length in octets of a random value to be used by the shared keys derivation specified in B.9. It shall be absent if a nonce may be generated in another method. Otherwise, it shall be present.

#### B.8.6.3 Diffie-Hellman group 23 algorithm with HKDF-256

The following is a specification for key establishment algorithm based on the Diffie-Hellman key agreement method and the key derivation method specified in B.9.

```
dhModpGr23Hkdf256Algo ALGORITHM::= {
  PARMS          Group23
  DYN-PARMS      Payload23
  IDENTIFIED BY id-algo-dhModpGr23Hkdf256Algo }

Group23::= INTEGER (23)

Payload23::= SEQUENCE {
  dhPublicKey OCTET STRING (SIZE (65)),
```

```
  nonce       OCTET STRING (SIZE (32)),
  ... }
```

The **PARMS** token specifies that the fixed parameters shall be those parameters specified for group number 23, for the ECDH curve **secp256r1** as specified in IETF RFC 5114.

The **DYN-PARMS** token that the dynamic parameters shall be those specified by the **PayLoad23** data type.

The **PayLoad23** data type has the same components as the **PayLoad14** data type except that the **dhPublicKey** shall have length of 65 octets.

#### B.8.6.4 Diffie-Hellman group 28 algorithm with HKDF-256

The following is a specification for key establishment algorithm based on the Diffie-Hellman key agreement method and the key derivation method specified in B.9.

```
dhModpGr28Hkdf256Algo ALGORITHM::= {
  PARMS          Group28
  DYN-PARMS      Payload28
  IDENTIFIED BY id-algo-dhModpGr28Hkdf256Algo }

Group28::= INTEGER (28)

Payload28::= SEQUENCE {
  dhPublicKey OCTET STRING (SIZE (65)),
  nonce       OCTET STRING (SIZE (32)),
  ... }
```

The **PARMS** token specifies that the fixed parameters shall be those parameters specified for group number 28, for the ECDH curve **brainpoolP256r1** as specified in IETF RFC 6932.

The **DYN-PARMS** field specifies that the dynamic parameters shall be those specified by the **PayLoad28** data type.

The **PayLoad28** data type has the same components as the **PayLoad14** data type except that the **dhPublicKey** shall have length of 65 octets.

## B.9 Key derivation

### B.9.1.1 General

When the outcome of a key agreement method is a shared secret, this shared secret is expanded by use of a key derivation function to provide sufficient material for the required symmetric keys.

### B.9.1.2 HMAC-based extract-and-expand key derivation function

The hash-based key derivation function (HKDF) is specified in IETF RFC 5869. It requires the use of the keyed-hash message authentication code (HMAC) algorithm as specified in IETF RFC 2104.

HKDF follows the "extract-then-expand" paradigm, where the KDF logically consists of two modules: the first stage takes the input keying material (IKM) and "extracts" from it a fixed-length pseudorandom key (PRK), and then the second stage "expands" this key into several additional output pseudorandom keys (OKM).

The HKDF-extract may be expressed as follows

**PRK = HMAC-Hash(salt, IKM)** where:

a) **salt** is a non-secret random value – it takes the value of the **nonce** component of the key establishment algorithm instance in question that require the use of HKDF.

b) **IKM** stands for input keying material – it is the shared secret resulting from the DH key agreement.

c) The OKM is defined as:

**OKM** = HKDF-expand (PRK, info, L), where:

**PRK** is the PRK generated by the HKDF-expand above,

info is optional context and application specific information and may be a zero-length string (as of RFC 5869),

**L** is the combined length of the keys to be generated.

## B.10 Migration of cryptographic algorithms

Migration of cryptographic algorithms across entities has proved difficult, as not all entities can or will migrate simultaneously, causing interworking problems. Rec. ITU-T X.509 | ISO/IEC 9594-8 provides specifications allowing migration of cryptographic algorithms for public-key certificates, attribute certificates, certificate revocation lists (CRLs) and authorization and validation lists (AVLs). Rec. ITU-T X.510 | ISO/IEC 9594-11 provides recommendations for how to migrate cryptographic algorithms used by protocol specifications.

The general technique during a migration period is to include two algorithms of a particular type, referred to a native algorithm, which is the algorithm to be migrated from, and an alternative algorithm to which migration is wanted. The alternative algorithm is supplied in such a way that it will be ignored by an entity that has not migrated yet. After the migration period, the alternative algorithm becomes the new native algorithm. If relevant, an alternative digital signature or ICV may also be included following the same principles.

NOTE Some cryptographic migrations may not be able to be accomplished without a system upgrade.

## B.11 Post-quantum computing cryptography

As the power system components applying key management typically remain in the field for a longer period, crypto-agility for the underlying cryptographic algorithms needs to be considered to address the recent developments in cryptography, e.g., specifically in the field of post-quantum cryptography. This holds for the utilized certificate format, but also for the security protocols used in the certificate management.

The problem with currently popular asymmetric algorithms is that their security relies on one of three hard mathematical problems:

a) the integer factorization problem, as discussed in B.3.2.3.

b) the discrete logarithm problem, as discussed in B.8.4.

c) the elliptic-curve discrete logarithm problem, as discussed in B.3.4.4.

Quantum computers are known to be very efficient in solving these mathematical problems, which makes asymmetric vulnerable to quantum computers.

Symmetric cryptographic algorithms are less vulnerable to quantum computer. The recommendation is to double the key size for symmetric. For hash algorithms is suggested to multiply the digest size by 2,5.

The Alexandra Institute has issued a white paper on Post-Quantum Cryptography [78].

NIST has a process in going to analyse and select quantum-safe cryptographic algorithms to be standardized. NISTIR 8309, *Status Report on the Second Round of NIST Post-Quantum Cryptographic Standardization Process* gives good picture of what we may expect [83].

NIST has also issued a paper on "Getting Ready for Post-Quantum Cryptography" [84].

As a first step, protocol specifications should provide migration capabilities as discussed in B.9 to be ready for post-quantum algorithms,

## B.12 Random Number Generation (RNG)

### B.12.1 Random number generation types

Random number generators (RNG) are a precondition for randomness and key generation in a secure manner. The stated security strength of hashing and symmetric key algorithms assumes the use of a somewhat-perfect random number generator.

Different devices may use different algorithms for generating random numbers. An implementer may choose an appropriate random number generation mechanism for their devices and applications.

What follows is a simplistic overview of commonly implemented random number generation methods. An in-depth discussion of random number generation is found in the US National Institute of Standards and Technology Special Publication 800-90A Revision 1 [61] as well as in ISO/IEC 18031 [87] and ISO/IEC 20543 [88].

One way to produce random numbers is by generating random bits using a process where the bit-wise output (Entropy) comes from a physical mechanism that is unpredictable. These are known as non-deterministic random bit generators (NRBGs). Often the physical mechanism is limited in the speed that it can produce random binary digits. Therefore, there is a strategy quickly to compute bits of pseudo-random data from the Entropy using an expansion algorithm; this class of RBGs is known as Deterministic Random Bit Generators (DRBGs).

### B.12.2 Deterministic random bit generators

A DRBG mechanism uses a physical mechanism to provide an entropy input. This is called the seed. Then a DRBG algorithm produces a very rapid sequence of bits from the seed. A DRBG outputs pseudorandom bits, rather than random bits, with often a large increase in speed. The entropy input to the DRBG must provide an assurance of randomness. The DRBG will be unpredictable, up to the strength of the seed, as long as the seed is kept secret. The DRBG algorithm must be a good one or the strength of the seed will not matter.

The security of a RBG that uses a DRBG mechanism is an implementation issue. Implementers must provide a strong enough entropy source and a good DRBG algorithm.

A good DRBG provides backtracking resistance. In other words, it should not be possible to guess previous random outputs without knowledge of the seed. A good DRBG algorithm also provides unbiased output and prediction resistance. It should not be possible to predict the next pseudo-random number without knowledge of the seed. Finally, a good DRBG uses a conditioned entropy source. Entropy conditioning includes continuous tests of randomness and non-repetition, which will catch entropy source malfunctions such as 11111, 00000, or 010101, etc.

For implementation, a DRBG algorithm is chosen and then coupled with a conditioned entropy input. A seed size and seed life are chosen to provide appropriate resistance to prediction attacks. Because the seed life is relative to the amount of pseudorandom data generated from it, the speed of a DRBG is ultimately limited by the re-seed rate the entropy source can support.

Many host devices will not have a good source of entropy input all the time. Therefore, the DRBG must be instantiated at a time when the DRBG actually does have access to some reliable source of entropy input. The source of entropy input perhaps is only occasionally available. A DRBG may be implemented in a way that it can accumulate additional entropy from the application as additional input. A DRBG implementation should accept additional input whenever it can. An application that may have access to some entropy, such as timestamps or nonce from other devices, should provide these values to the DRBG as additional inputs.

### B.12.3 Non-deterministic random number generation

This relates to basically all parts of IEC 62351 using cryptographic keys (3,4,5,6,8).

Non-Deterministic Random Number Generators and entropy sources are the only cryptographic primitive, which is not subject to standardization.

ISO/IEC 18031 or NIST SP 800-90 provide a hypothetical noisy diode example. However, it is not easy to build a good diode-based random number generator. One of the problems of that approach is the low quality of inexpensive A/D converters, another the ease with which powerline hum and other electromagnetic noise make it into the circuitry, and another is that certain diodes might go bad over time.

There are several off-the-shelf hardware-based random bit generators sold for computerized gambling and cryptographic use. Their implementations vary from noisy diode and unlocked oscillator design to a commercially available quantum optical design.

Hardware random number generators should be constantly monitored for proper operation. RFC 4086 and FIPS Pub 140-2 include tests that may be used for this.

Many noise sources exist. Since their output is largely predictable, these sources must be avoided for cryptographic use. Examples of what to avoid:

- Pink Noise generation circuits. While inexpensive, they produce pseudo-random digital noise so are not suitable for crypto use, unfortunately.
- Microphones and radios: They are an external input that may be available to manipulation by an attacker.
- The 1955 RAND table and other similar published sources of random numbers. These are well-known sources and have no forward nor backward security in a cryptographic setting.

### B.12.4 Entropy sources

Entropy is a measure of disorder in some small process or device that is a closed system but can be measured from outside. The measurement process produces a string of random bits. Each bit of a bit string with full entropy is unpredictable, has a uniform distribution and is independent of every other bit. To make an entropy source useful for Random Number Generation the process starts with a noise source, such as thermal noise; a digitization process; an assessment process; an optional conditioning process and health tests. Input noise from a network interface or a power line is usually not acceptable because it is not reliable. Side channel attacks must be considered. For example, a temperature extremes attack to quiet a thermal noise source would make it less random and reduce the real randomness of random numbers it produces, perhaps substantially. To account for possible attacks, a source of entropy should be measured by its worst-case scenario or Minimum Entropy it provides.

# Annex C
(informative)

# Certificate enrolment and renewal flowcharts

## C.1 Certificate Enrolment

See Figure C.1 for a potential flow for certificate enrolment.

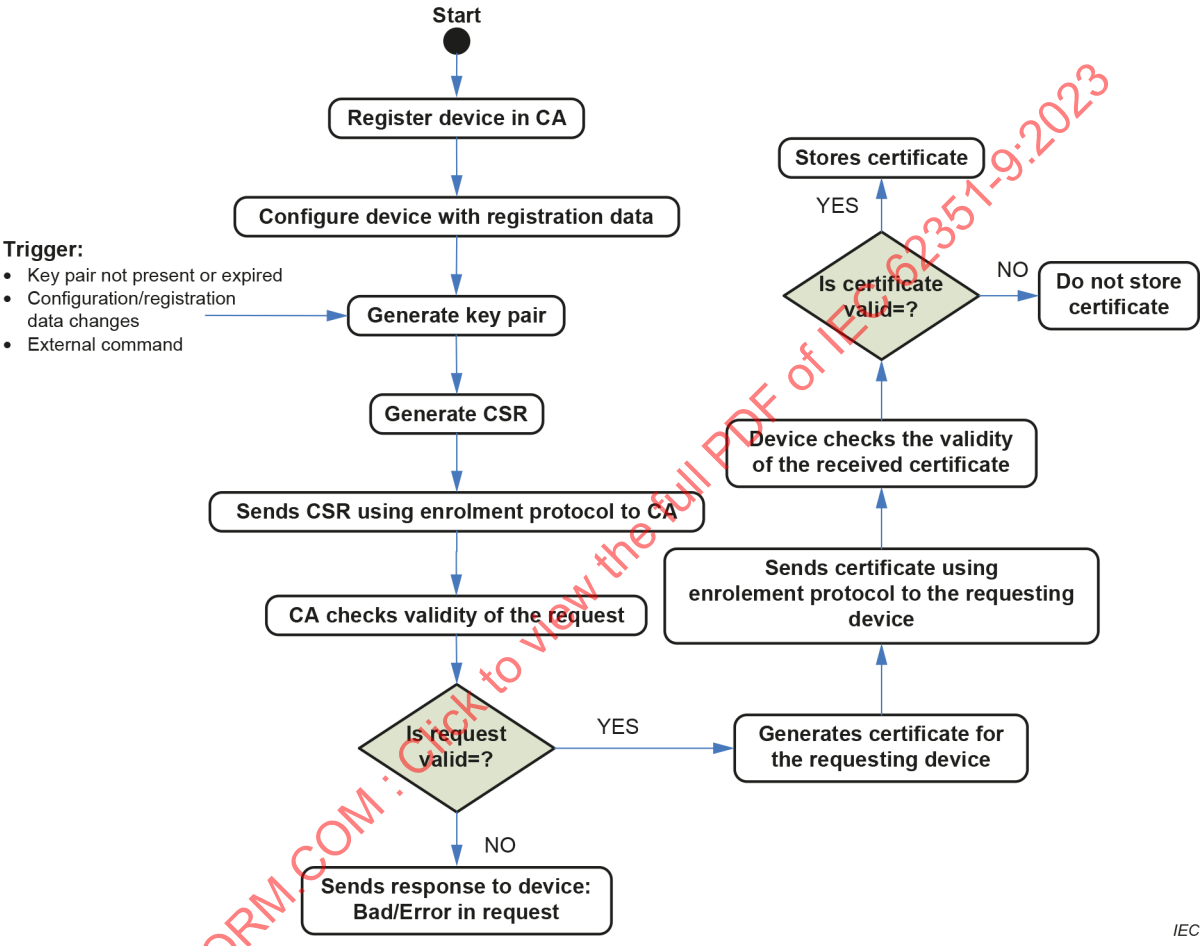



**Figure C.1 – Certificate Enrolment (general)**

## C.2 Certificate Renewal

See Figure C.2 for a state machine for certificate renewal:

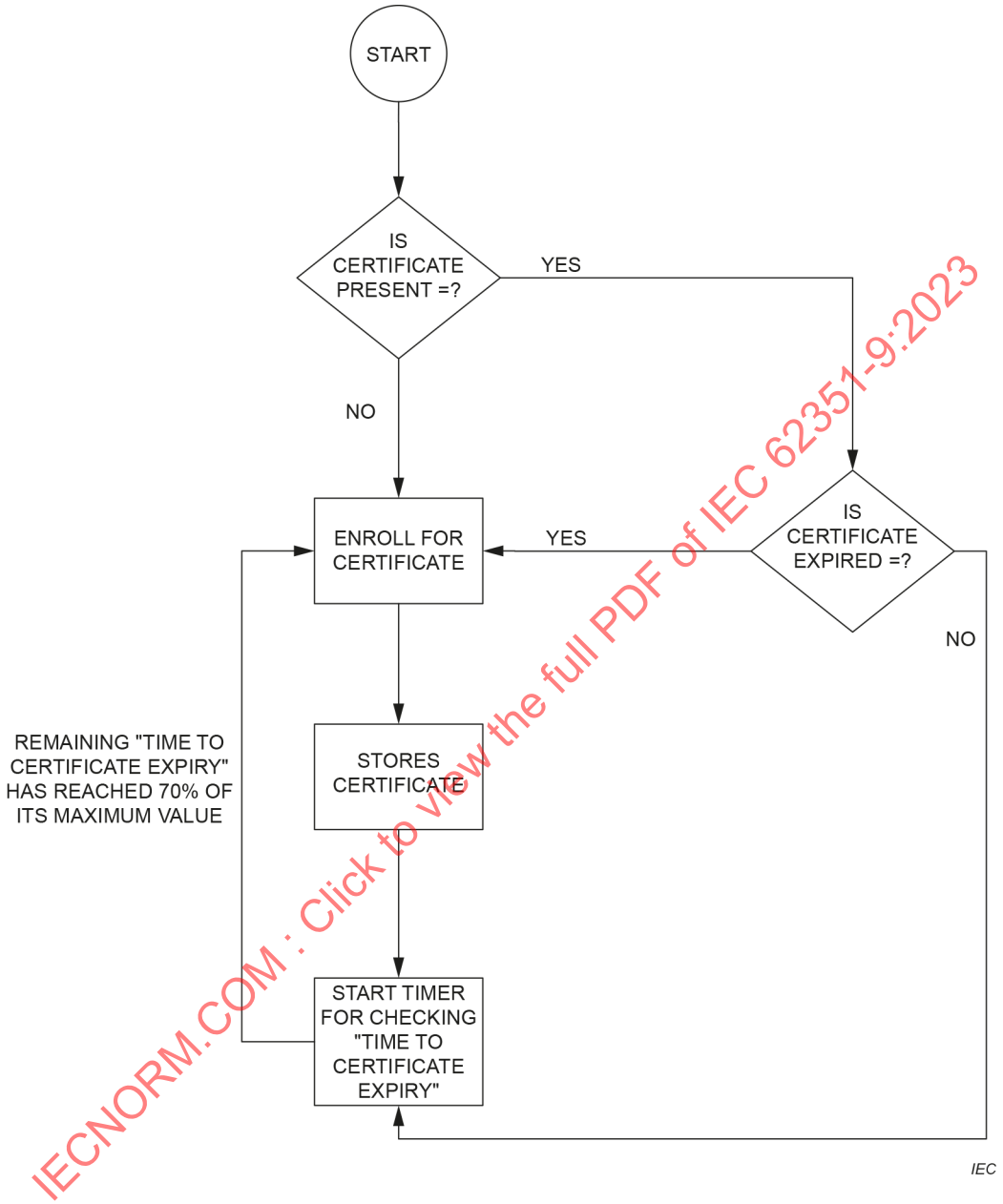





**Figure C.2 – Certificate Renewal State Machine**

# Annex D
(informative)

# Security Event mapping to IEC 62351-14

## D.1 General

This annex contains the mapping of the security events defined in this document to the format specified in IEC 62351-14.

The information about the security events and potential detailed information to be contained in the ExtraInfo may only be provided by the entity based on the availability of this information through the underlying platform or utilized components.

The IEC Version of all security events shall be set to "1". For the security events, the following grouping is used:

- Value "1" relates to public-key certificate transport related events, including enrolment protocol errors
- Value "2" public-key certificate verification specific security events
- Value "3" relates to attribute certificate verification specific security events
- Value "4" relates to certificate revocation status events (valid for public-key certificates and attribute certificates
- Value "5" relates to GDOI specific security events

## D.2 Security event log records for credential transport and enrolment

Table D.1 contains events related to credential transport and enrolment.

**Table D.1 – Security event logs for credential transport and certificate enrolment mapped to IEC 62351-14**

| Security Event | Clause/ Subclause (IEC 62351-9) | MNEMONIC | Severity | Event identifier | Text | ExtraInfo |
|---|---|---|---|---|---|---|
| Key Transport: PKCS #12 format error | 7.3.2 | CRED_PKCS12_FORMAT | Warning | IEC 62351-9:1.1 | PKCS #12 format mismatch. | |
| Key Transport: PKCS #12 format error | 7.3.2 | CRED_PKCS8_FORMAT | Warning | IEC 62351-9:1.2 | PKCS #8 format mismatch. | |
| Enrolment with SCEP successfully performed | 7.3.7.2 | CRED_ENR_SCEP_SUCC | Notice | IEC 62351-9:1.3 | Enrolment using SCEP successfully performed. | Issued public-key certificate |
| Enrolment with SCEP aborted due to failure | 7.3.7.2 | CRED_ENR_SCEP_FAIL | Error | IEC 62351-9:1.4 | Enrolment using SCEP aborted due to SCEP failure. | |
| Enrolment with EST successfully performed | 7.3.7.2 | CRED_ENR_EST_SUCC | Notice | IEC 62351-9:1.5 | Enrolment using EST successfully performed. | Issued public-key certificate |
| Enrolment with EST aborted due to failure | 7.3.7.2 | CRED_ENR_EST_FAIL | Error | IEC 62351-9:1.6 | Enrolment using EST aborted due to SCEP failure. | |

## D.3 Security event log records for public-key certificate verification

Table D.2 contains events related to public-key certificate verification.

**Table D.2 – Security event logs defined for public-key certificate verification mapped to IEC 62351-14**

| Security Event | Clause/ Subclause (IEC 62351-9) | MNEMONIC | Severity | Event identifier | Text | ExtraInfo |
|---|---|---|---|---|---|---|
| Certificate encoding error | 7.4.2 | CERT_V_FORMAT | Warning | IEC 62351-9:2.1 | Certificate format mismatch. Failed verification. | |
| Public Certificate signature verification failed | 7.4.3 | CERT_V_PK_SIG_WRONG | Alarm | IEC 62351-9:2.2 | Public-key certificate signature could not be verified. | |
| Version of X.509 certificate not matching expected version | 7.4.4.2 | CERT_V_PK_VERSION | Error | IEC 62351-9:2.3 | Wrong public-key certificate version. | |
| Issuer does not match known and trusted issuer. | 7.4.4.3 | CERT_V_PC_ISSUER | Error | IEC 62351-9:2.4 | Public-key certificate issuer not trusted. | |
| Signature algorithm not supported | 7.4.4.4 | CERT_PK_SIG_ALG | Error | IEC 62351-9:2.5 | Unsupported signature algorithm. | |
| Public-key certificate validity: expired | 7.4.4.5 | CERT_V_PK_EXPIRED | Error | IEC 62351-9:2.6 | Public-key certificate expired. | |
| Public-key certificate validity: not valid, yet | 7.4.4.5 | CERT_V_PK_EARLY | Error | IEC 62351-9:2.7 | Public-key certificate not valid yet. | |
| Subject not included in public-key certificate | 7.4.4.6 | CERT_V_PK_SUBJECT | Warning | IEC 62351-9:2.8 | Subject not included in public-key certificate. | |
| Algorithm in public-key certificate not supported | 7.4.4.7 | CERT_V_PK_ALG_MISMATCH | Error | IEC 62351-9:2.9 | Native public key algorithm in public-key certificate not supported. | |
| Path validation error as trust anchor is not supported | 7.4.4.8 | CERT_V_PK_TA | Error | IEC 62351-9:2.10 | Trust anchor not supported. | |
| Certification path validation error | 7.4.4.8 | CERT_V_PK_PATH | Error | IEC 62351-9:2.11 | Certification path could not be verified. | |
| Critical extension unknown | 7.4.4.10.1 | CERT_V_PKCE_CEXT | Error | IEC 62351-9:2.12 | Unknown critical extension in public-key certificate. | |
| Critical extension value unknown | 7.4.4.10.1 | CERT_V_PKCE_EXT_VALUE | Error | IEC 62351-9:2.13 | Unknown information in critical extension. | |

| Security Event | Clause/ Subclause (IEC 62351-9) | MNEMONIC | Severity | Event identifier | Text | ExtraInfo |
|---|---|---|---|---|---|---|
| Authority key identifier not included in public-key certificate | 7.4.4.10.2 | CERT_V_PK_AKID | Error | IEC 62351-9:2.14 | Authority key identifier not included in public-key certificate. | |
| Subject key identifier not included in public-key certificate | 7.4.4.10.3 | CERT_V_PKCE_SKID | Error | IEC 62351-9:2.15 | Subject key identifier not included in public-key certificate. | |
| Subject alternative Name not included in TLS server certificates | 7.4.4.10.4 | CERT_V_PKCE_SAN | Error | IEC 62351-9:2.16 | Subject alternative Name not included in TLS server certificates. | |
| BC not included in CA certificate | 7.4.4.10.5 | CERT_V_PKCE_BC | Error | IEC 62351-9:2.17 | Basic constraints not included in CA certificate. | |
| Violation of path length constraints in CA certificate | 7.4.4.10.5 | CERT_V_PKCE_PL | Error | IEC 62351-9:2.18 | Path len constraints in CA certificate violated. | |
| Key usage for digital signatures not included in TLS certificate | 7.4.4.10.6 | CERT_KU_DIGSIG | Error | IEC 62351-9:2.19 | Key usage for digital signatures not included in TLS certificate. | |
| Key usage for key encipherment not included when TLS cipher used with key encryption is used | 7.4.4.10.6 | CERT_PKCE_KU | Error | IEC 62351-9:2.20 | Key usage for key encipherment not included when TLS cipher used with key encryption is used. | |
| Key usage not included for signing certificates. | 7.4.4.10.6 | CERT_PKCE_KU_KCS | Error | IEC 62351-9:2.21 | Key usage not included for signing certificates. | |
| Key usage not included for signing CRLs in CRL signature certificate | 7.4.4.10.6 | CERT_PKCE_KU_CRLS | Error | IEC 62351-9:2.22 | Key usage not included for signing CRLs in CRL signature certificate. | |
| Extended Key Usage for server authentication not included in TLS server certificate. | 7.4.4.10.7 | CERT_PKCE_EKU_TLS_SA | Error | IEC 62351-9:2.23 | Extended Key Usage for server authentication not included in TLS server certificate. | |
| Extended Key Usage for client authentication not included in TLS client certificate. | 7.4.4.10.7 | CERT_PKCE_EKU_TLS_CA | Error | IEC 62351-9:2.24 | Extended Key Usage for client authentication not included in TLS client certificate. | |

| Security Event | Clause/ Subclause (IEC 62351-9) | MNEMONIC | Severity | Event identifier | Text | ExtraInfo |
|---|---|---|---|---|---|---|
| Extended Key Usage not included for OCSP signing in OCSP responder certificate | 7.4.4.10.7 | CERT_PKCE_EKU_OCSPS | Error | IEC 62351-9:2.25 | Extended Key Usage not included for OCSP signing in OCSP responder certificate. | OCSP responder certificate |
| CRL distribution point not contained in public-key certificate | 7.4.4.10.9 | CERT_V_PKCE_NCDP | Warning | IEC 62351-9:2.26 | CRL distribution point not contained in public-key certificate. | |
| OCSP responder information not contained in public-key certificate | 7.4.4.10.10 | CERT_V_PKCE_NOCSP | Warning | IEC 62351-9:2.27 | OCSP responder information not contained in public-key certificate. | |

## D.4 Security event log records for attribute certificate verification

Table D.3 contains events related to attribute certificate verification

**Table D.3 – Security event logs defined for attribute certificate verification mapped to IEC 62351-14**

| Security Event | Clause/ Subclause (IEC 62351-9) | MNEMONIC | Severity | Event identifier | Text | ExtraInfo |
|---|---|---|---|---|---|---|
| Certificate encoding error | 7.4.2 | CERT_V_AC_FORMAT | Warning | IEC 62351-9:3.1 | Certificate format mismatch. Failed verification. | |
| Attribute certificate signature verification failed | 7.4.3 | CERT_V_AC_SIG_WRONG | Alarm | IEC 62351-9:3.2 | Public-key certificate signature could not be verified. | |
| Attribute certificate version information mismatch | 7.4.5.2 | CERT_V_AC_VERSION | Error | IEC 62351-9:3.3 | Wrong attribute certificate version. | |
| Holder of attribute certificate could not be verified based on public-key certificate | 7.4.5.3 | CERT_V_AC_HOLDER_PK | Warning | IEC 62351-9:3.4 | Holder of attribute certificate could not be verified based on public-key certificate | |
| Holder of attribute certificate could not be verified based on alternative authentication. | 7.4.5.3 | CERT_V_AC_HOLDER_NPK | Warning | IEC 62351-9:3.5 | Holder of attribute certificate could not be verified based on alternative authentication. | |

| Security Event | Clause/ Subclause (IEC 62351-9) | MNEMONIC | Severity | Event identifier | Text | ExtraInfo |
|---|---|---|---|---|---|---|
| Configured attribute certificate issuer not trusted | 7.4.5.4 | CERT_V_AC_ISSUER_CONFIG | Error | IEC 62351-9:3.6 | Attribute certificate issuer not trusted (not configured). | |
| Attribute certificate issuer not trusted based on a trusted CA | 7.4.5.4 | CERT_V_AC_ISSUER_TCA | Error | IEC 62351-9:3.7 | Attribute certificate issuer not trusted (by a trusted CA). | |
| Attribute certificate expired | 7.4.5.7 | CERT_V-AC_EXPIRED | Error | IEC 62351-9:3.8 | Attribute certificate expired. | |
| Attribute certificate not yet valid | 7.4.5.7 | CERT_V_AC_EARLY | Error | IEC 62351-9:3.9 | Attribute certificate not yet valid. | |
| unknown critical extension in attribute certificate | 7.4.5.8.1 | CERT_V_ACE_EXT | Error | IEC 62351-9:3.10 | unknown critical extension in attribute certificate. | |
| unrecognized information in critical extension of attribute certificate | 7.4.5.8.1 | CERT_V_ACE_EXT_VALUE | Error | IEC 62351-9:3.11 | unrecognized information in critical extension of attribute certificate. | |
| unknown extension in attribute certificate | 7.4.5.8.1 | CERT_V_ACCE_EXT | Warning | IEC 62351-9:3.12 | unknown extension in attribute certificate. | |
| authority key identifier in attribute certificate could not be verified | 7.4.5.8.2 | CERT_V_ACE_AKID | Error | IEC 62351-9:3.13 | authority key identifier in attribute certificate could not be verified. | |
| no revocation information intended in attribute certificate | 7.4.5.8.3 | CERT_V_ACE_NRII | Notice | IEC 62351-9:3.14 | no revocation information intended in attribute certificate. | |
| CRL distribution point not contained in attribute certificate | 7.4.5.8.3 | CERT_V_ACE_NCDP CRL | Warning | IEC 62351-9:3.15 | CRL distribution point not contained in attribute certificate. | |
| OCSP responder information not contained in attribute certificate | 7.4.5.8.3 | CERT_V_ACE_NOCSP | Warning | IEC 62351-9:3.16 | OCSP responder information not contained in attribute certificate. | |

## D.5 Security event log records for certificate revocation status

Table D.4 contains events related to certificate revocation status

**Table D.4 – Security event logs defined for certificate revocation status mapped to IEC 62351-14**

| Security Event | Clause/ Subclause (IEC 62351-9) | MNEMONIC | Severity | Event identifier | Text | ExtraInfo |
|---|---|---|---|---|---|---|
| Certificate has been revoked | 7.4.6 | CERT_V_REVOKED | Alarm | IEC 62351-9:4.1 | Certificate has been revoked. | Revocation reason |
| CRL distribution point not accessible | 7.4.6 | CERT_V_R_NR_CRL | Warning | IEC 62351-9:4.2 | CRL distribution point not accessible. | CRLDP URL |
| CRL expired | 7.4.6 | CERT_V_CRL_EXP | Warning | IEC 62351-9:4.3 | CRL expired. | |
| CRL signature verification failed | 7.4.6 | CERT_V_CRL_SIG_FAIL | Warning | IEC 62351-9:4.4 | CRL signature verification failed. | |
| OCSP responder not accessible | 7.4.6 | CERT_V_R_NR_OCSP | Warning | IEC 62351-9:4.5 | OCSP responder not accessible. | OCSP responder URL |
| OCSP responder connection timeout | 7.4.6 | CERT_V_R_TO_OCSP | Warning | IEC 62351-9:4.6 | OCSP responder connection timeout. | |
| Certificate not known to OCSP responder | 7.4.6 | CERT_V_R_CU_OCSP | Warning | IEC 62351-9:4.7 | Certificate not known to OCSP responder. | |
| OCSP response expired | 7.4.6 | CERT_V_R_OCSP_EXP | Warning | IEC 62351-9:4.8 | OCSP response expired. | |
| OCSP response message signature verification failed | 7.4.6 | CERT_V_OCSP_SIG_FAIL | Warning | IEC 62351-9:4.9 | OCSP response signature verification failed. | |

## D.6 Security event log records for group-based key management with GDOI

Table D.5 contains events related to GDOI.

**Table D.5 – Security event logs for GDOI mapped to IEC 62351-14**

| Security Event | Clause/ Subclause (IEC 62351-9) | MNEMONIC | Severity | Event identifier | Text | ExtraInfo |
|---|---|---|---|---|---|---|
| GDOI Cryptographic algorithms not supported | 8.2 | IKEv1_CALG_ MISMATCH | Error | IEC 62351-9:5.1 | Proposal of unsupported cryptographic algorithms | |
| GDOI Padding Error | 8.2 | IKEv1_PADDIN G | Error | IEC 62351-9:5.2 | Padding error during decryption of IKEv1 message | |
| GDOI-PULL Use of Expired SA | 8.5.11 | GROUP_PULL _EXPIRED_SA | Warning | IEC 62351-9:5.3 | Use of Expired SA Requested. | |
| GDOI-PULL Phase 2 decryption issue | 8.5.11 | GROUP_PULL _PHASE2_DEC RYPTION | Warning | IEC 62351-9:5.4 | Unable to Decrypt Phase 2 message. | |
| GROUPKEY-PULL Key for unknown group requested | 8.5.11 | GROUP_PULL _PHASE2_GR OUP_NONEXI STENT | Warning | IEC 62351-9:5.5 | Key for unknown group requested. | |
| GROUPKEY-PULL unable to agree to PHASE 1 keys | 8.5.11 | GROUP_PULL _PHASE1_KEY _NEGOTIATIO N | Notice | IEC 62351-9:5.6 | Unable to negotiate GDOI PHASE 1. | |
| GROUPKEY-PUSH Unreachable Destination | 8.6.2 | GROUP_PUSH _UNREACHAB LE_DESTINATI ON | Error | IEC 62351-9:5.7 | Attempted to send to unreachable group member. | |
| KDA failure | 8.6.3 | KDA_FAILURE | Notice | IEC 62351-9:5.8 | KDA cannot be provided to publisher. | |

# Bibliography

[1] RFC 2631, *Diffie Hellman Key Exchange*, June 1999. https://tools.ietf.org/html/rfc2631

[2] Ecommerce PKI Glossary, http://www.ecommercepki.com/cps/glossary.htm

[3] IEC TS 62351-1, *Power systems management and associated information exchange – Data and communications security – Part 1: Communication network and system security – Introduction to security issues*, January 2007

[4] IEEE 1588, *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, Version 2.1, 11/2019

[5] IEEE 1686:2022, *IEEE Standard for Substation Intelligent Electronic Devices Cyber Security Capabilities*, 2013 (currently under revision)

[6] ISO/IEC 11770-1:2009, *Information technology – Security techniques – Key management Part 1: Framework*, December 2009

[7] ISO/IEC 11770-2:2008, *Information technology – Security techniques – Key management – Part 2: Mechanisms using symmetric techniques*

[8] ISO/IEC 11770-3:2008, *Information technology – Security techniques – Key management – Part 3: Mechanisms Using Asymmetric Techniques*

[9] ISO/IEC 19790:2012, *Information technology – Security techniques – Security requirements for cryptographic modules*

[10] ISO/IEC 8802-3-2000, *Telecommunications and information exchange between systems, October 2006*

[11] NIST SP 800-130, *A Framework for Designing Cryptographic Key Management Systems*, Elaine Barker, Miles Smid, Dennis Branstad, Santosh Chokhani, April 2012

[12] NIST SP 800-133, *Recommendation for Cryptographic Key Generation*, Elaine Barker and Allen Roginsky, July 2012

[13] NIST SP 800-57, *Recommendation for Key Management – Part 1: General (Revision 5)*, May 2020

[14] NIST SP 800-90 A, *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*, Elaine Barker and John Kelsey, January 2012

[15] NIST SP 800-90 B, *Recommendation for the Entropy Sources Used for Random Bit Generation*, Elaine Barker and John Kelsey, August 2012

[16] NIST SP 800-90 C, *Recommendation for Random Bit Generator (RBG) Constructions*, Elaine Barker and John Kelsey, August 2012

[17] NIST SP 800-97, *Establishing Wireless Robust Security Networks*, Sheila Frankel, Bernard Eydt, Les Owens, Karen Scarfone, February 2007

[18] IETF RFC 2986:2000, *PKCS #10: Certification Request Syntax Specification Version 1.7*

[19] IETF RFC 2985:2000, *PKCS #9: Selected Object Classes and Attribute Types Version 2.0,* https://tools.ietf.org/html/rfc2985

[20] IETF RFC 4211:2005, *Internet X.509 Public Key Infrastructure – Certificate Request Message Format (CRMF),* https://tools.ietf.org/html/rfc4211

[21] PKCS #11, *Cryptographic Token Interface Standard – Version 2.4*, RSA Laboratories, June 2003

[22] R. Kissel, Glossary of Key Information Security Terms, NIST IR 7298, April 2006. http://csrc.nist.gov/publications/nistir/NISTIR-7298_Glossary_Key_Infor_Security_Terms.pdf

[23] RFC 2407, *Internet Security Association and Key Management Protocol (ISAKMP)*, November 1998, http://www.ietf.org/rfc/rfc3647.txt

[24] RFC 2818, *HTTP Over TLS*, E. Rescorla, May 2000 http://www.ietf.org/rfc/rfc2818.txt

[25] RFC 3447, *Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1*, February 2003 https://www.rfc-editor.org/pdfrfc/rfc3447.txt.pdf

[26] RFC 3647, *Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework,* S. Chokhani, W. Ford, R. Sabett, C. Merrill, S. Wu, November 2003, https://tools.ietf.org/html/rfc3647

[27] RFC 3740, *The Multicast Group Security Architecture*, T. Hardjono, B. Weis, March 2004, https://tools.ietf.org/html/rfc3740

[28] RFC 4082, *Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction*, June 2005, https://tools.ietf.org/html/rfc4082

[29] RFC 4120, *The Kerberos Network Authentication Service (V5)*, July 2005, https://tools.ietf.org/html/rfc4120

[30] RFC 4210, *Internet X.509 Public Key Infrastructure Certificate Management Protocols*, September 2005, https://tools.ietf.org/html/rfc4210

[31] RFC 4476, *Attribute Certificate (AC) Policies Extension,* May 2006, https://tools.ietf.org/html/rfc4476

[32] RFC 4754, *IKE and IKEv2 Authentication Using the Elliptic Curve Digital Signature Algorithm (ECDSA)*, January 2007

[33] RFC 4949, *Internet Security Glossary, Version 2*, August 2007. https://tools.ietf.org/html/rfc4949

[34] RFC 5055, *Server-Based Certificate Validation Protocol (SCVP)*, December 2007 https://tools.ietf.org/html/rfc5055

RFC 5246, *The Transport Layer Security (TLS) Protocol Version 1.2*, August 2008 https://tools.ietf.org/html/rfc5246

RFC 5272, *Certificate Management over CMS (CMC),* June 2008, https://tools.ietf.org/html/rfc5272

[35] RFC 5273, *Certificate Management over CMS (CMC): Transport Protocols*, June 2008, https://tools.ietf.org/html/rfc5273

[36] RFC 5280, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*, May 2008, https://tools.ietf.org/html/rfc5280

[37] RFC 5639, *Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation*, March 2010, https://tools.ietf.org/html/rfc5639

[38] RFC 5905, *Network Time Protocol Version 4: Protocol and Algorithms Specification*, June 2010, https://tools.ietf.org/html/rfc5905

[39] RFC 5996, *Internet Key Exchange Protocol Version 2 (IKEv2),* September 2010, https://tools.ietf.org/html/rfc5996

[40] RFC 5998, *An Extension for EAP-Only Authentication in IKEv2*, Sep0tember 2010, https://tools.ietf.org/html/rfc5998

[41] RFC 6066, *Transport Layer Security (TLS) Extensions: Extension Definitions, D. Eastlake 3rd*, January 2011, https://tools.ietf.org/html/rfc6066

[42] RFC 6712, *Internet X.509 Public Key Infrastructure – HTTP Transfer for the Certificate Management Protocol (CMP)*, September 2012, https://tools.ietf.org/html/rfc6712

[43] RFC 7748, *Elliptic Curves for Security*, January 2016 https://tools.ietf.org/html/rfc7748

[44] RFC 8366, *A Voucher Artifact for Bootstrapping Protocols*, May 2018, https://tools.ietf.org/html/rfc8366

[45] RFC 8446, *The Transport Layer Security (TLS) Protocol Version 1.3*, August 20018 https://tools.ietf.org/html/rfc8446

[46] RFC 8520, *Manufacturer Usage Description Specification*, March 2019, https://tools.ietf.org/html/rfc8520

[47] RFC 8995, *Bootstrapping Remote Secure Key Infrastructures (BRSKI)*, May 2021, https://tools.ietf.org/html/rfc8995

RFC 8951, *Clarification of Enrollment over Secure Transport (EST): transfer encodings and ASN.1*, November 2020, tools.ietf.org/html/rfc8951

[48] IETF Draft, *Lightweight CMP Profile*, https://datatracker.ietf.org/doc/draft-ietf-lamps-lightweight-cmp-profile/

[49] BSI/AIS31, *A proposal for: Functionality classes for random number generators*, September 2011 https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Zertifierung/Interpretation/AIS31_Functionality_classes_for_random_number_generators.pdf

[50] FIPS PUB 201-1, *Personal Identity Verification*, March 2006

[51] IANA Service Name and Transport Protocol Port Number Registry, https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

[52] IANA Assignments for GDOI Payloads, https://www.iana.org/assignments/gdoi-payloads/gdoi-payloads.xhtml

[53] FIPS PUB 140-2, *Security requirements for cryptographic modules,* May 2001 http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

[54] IEC 62351-12, *Power systems management and associated information exchange – Data and communications security – Part 12: Resilience and security recommendations for power systems with distributed energy resources (DER) cyber-physical systems*

[55] IEC 62351-1, *Power systems management and associated information exchange – Data and communications security – Part 1: Communication network and system security – Introduction to security issues*

[56] IEC 62351-8, *Power systems management and associated information exchange – Data and communications security – Part 8: Role-based access control*

[57] NIST Special Publication 800-57 Part 1 Rev. 5, *Recommendation for Key Management*, 05/2020, https://doi.org/10.6028/NIST.SP.800-57pt1r5

[58] TR-02102-1, *BSI Technical Guideline: Cryptographic Mechanisms: Recommendations and Key Lengths*, 02/2022, https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-1.pdf

[59] RFC 8915, *Network Time Security for the Network Time Protocol*, September 2020, tools.ietf.org/html/rfc8915/

[60] *Standards for Efficient Cryptography,* http://www.secg.org/sec2-v2.pdf

[61] NIST SP 800-90A, Revision 1, *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*

[62] RFC 4086, *Randomness Requirements for Security*, June 2005, https://tools.ietf.org/html/rfc4086

[63] FIPS PUB 186, *Digital Signature Standard (DSS)*, 1994

[64] FIPS PUB 186-5 (Draft), *Digital Signature Standard (DSS)*, 2019, https://doi.org/10.6028/NIST.FIPS.186-5-draft

[65] FIPS PUB 186-4, *Digital Signature Standard (DSS)*, 2013, https://doi.org/10.6028/NIST.FIPS.186-4

[66] FIPS PUB 180-4, *Secure Hash Standard (SHS)*, 2015, https://doi.org/10.6028/NIST.FIPS.180-4

[67] FIPS PUB 198-1, *The Keyed-Hash Message Authentication code (HMAC)*, 2008, https://doi.org/10.6028/NIST.FIPS.198-1

[68] NIST SP 800-38D, *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*, 2007, https://doi.org/10.6028/NIST.SP.800-38D

[69] NIST SP 800-38C, *Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality*, 2007, https://doi.org/10.6028/NIST.SP.800-38C

[70] FIPS 140-3, *Security Requirements for Cryptographic Modules*, 2019, https://doi.org/10.6028/NIST.FIPS.140-3

[71] ISO/IEC 19790, *Information technology – Security techniques – Security requirements for cryptographic modules*, 2012

[72] NIST SP 800-56A Rev.3, *Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography*, 2018, https://doi.org/10.6028/NIST.SP.800-56Ar3

[73] IEC 62443-3-3, *Industrial communication networks – Network and system security – Part 3-3: System security requirements and security levels*, 2013

[74] IEC 62443-4-2, *Security for industrial automation and control systems – Part 4-2: Technical security requirements for IACS components*, 2019

[75] IEEE c37.240, *IEEE Standard Cybersecurity Requirements for Substation Automation, Protection, and Control Systems*, 2015 (currently under revicsion)

[76] RFC 8572, *Secure Zero Touch Provisioning (SZTP)*, https://datatracker.ietf.org/doc/html/rfc8572

[77] RFC 8032, *Edwards-Curve Digital Signature Algorithm (EdDSA)*, January 2017, https://tools.ietf.org/html/rfc8032

[78] Alexandra Institute, *Post-Quantum Cryptography*, Alexandra-Instituttet-Whitepaper-Post-Quantum-Cryptography.pdf

[79] RFC 8410, *Algorithm Identifiers for Ed25519, Ed448, X25519, and X448 for Use in the Internet X.509 Public Key Infrastructure*, August 2018, https://datatracker.ietf.org/doc/html/rfc8410

[80] RFC 5758, *Internet X.509 Public Key Infrastructure: Additional Algorithms and Identifiers for DSA and ECDSA*, January 2010, https://datatracker.ietf.org/doc/html/rfc5758

[81] RFC 3279, *Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*, April 2002, https://datatracker.ietf.org/doc/html/rfc3279

[82] RFC 5480, *Elliptic Curve Cryptography Subject Public Key Information*, March 2009, https://datatracker.ietf.org/doc/html/rfc5480

[83] NISTIR 8309, *Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process*, Julye 2020, https://doi.org/10.6028/NIST.IR.8309

[84] NIST White Paper, *Getting Ready for Post-Quantum Cryptography: Exploring Challenges Associated with Adopting and Using Post-Quantum Cryptographic Algorithms*, April 2021, https://doi.org/10.6028/NIST.CSWP.04282021

[85] RFC 4231, *Identifiers and Test Vectors for HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512*, December 2005, https://datatracker.ietf.org/doc/html/rfc4231

[86] RFC 9044, *Using the AES-GMAC Algorithm with the Cryptographic Message Syntax (CMS)*, June 2021, https://datatracker.ietf.org/doc/html/rfc9044

[87] ISO/IEC 18031, *Information technology – Security techniques – Random bit generation*, 2011

[88] ISO/IEC 20543, *Information technology – Security techniques – Test and analysis methods for random bit generators within ISO/IEC 19790 and ISO/IEC 15408*, 2019

[89] NIST SP 800-131A Rev.2, *Transitioning the Use of Cryptographic Algorithms and Key Lengths*, March 2019, https://doi.org/10.6028/NIST.SP.800-131Ar2

[90] Draft ISO/IEC 9594-12 | Rec. ITU-T X.508, *Information technology – Open systems interconnection – Part 12: The Directory: Key management and public-key infrastructure establishment and maintenance*

# SOMMAIRE

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

_____

# GESTION DES SYSTÈMES DE PUISSANCE ET ÉCHANGES D'INFORMATIONS ASSOCIÉS – SÉCURITÉ DES COMMUNICATIONS ET DES DONNÉES –

## Partie 9: Gestion de clé de cybersécurité des équipements de système de puissance

## AVANT-PROPOS

1) La Commission Électrotechnique Internationale (IEC) est une organisation mondiale de normalisation composée de l'ensemble des comités électrotechniques nationaux (Comités nationaux de l'IEC). L'IEC a pour objet de favoriser la coopération internationale pour toutes les questions de normalisation dans les domaines de l'électricité et de l'électronique. À cet effet, l'IEC – entre autres activités – publie des Normes internationales, des Spécifications techniques, des Rapports techniques, des Spécifications accessibles au public (PAS) et des Guides (ci-après dénommés "Publication(s) de l'IEC"). Leur élaboration est confiée à des comités d'études, aux travaux desquels tout Comité national intéressé par le sujet traité peut participer. Les organisations internationales, gouvernementales et non gouvernementales, en liaison avec l'IEC, participent également aux travaux. L'IEC collabore étroitement avec l'Organisation Internationale de Normalisation (ISO), selon des conditions fixées par accord entre les deux organisations.
2) Les décisions ou accords officiels de l'IEC concernant les questions techniques représentent, dans la mesure du possible, un accord international sur les sujets étudiés, étant donné que les Comités nationaux de l'IEC intéressés sont représentés dans chaque comité d'études.
3) Les Publications de l'IEC se présentent sous la forme de recommandations internationales et sont agréées comme telles par les Comités nationaux de l'IEC. Tous les efforts raisonnables sont entrepris afin que l'IEC s'assure de l'exactitude du contenu technique de ses publications; l'IEC ne peut pas être tenue responsable de l'éventuelle mauvaise utilisation ou interprétation qui en est faite par un quelconque utilisateur final.
4) Dans le but d'encourager l'uniformité internationale, les Comités nationaux de l'IEC s'engagent, dans toute la mesure possible, à appliquer de façon transparente les Publications de l'IEC dans leurs publications nationales et régionales. Toutes divergences entre toutes Publications de l'IEC et toutes publications nationales ou régionales correspondantes doivent être indiquées en termes clairs dans ces dernières.
5) L'IEC elle-même ne fournit aucune attestation de conformité. Des organismes de certification indépendants fournissent des services d'évaluation de conformité et, dans certains secteurs, accèdent aux marques de conformité de l'IEC. L'IEC n'est responsable d'aucun des services effectués par les organismes de certification indépendants.
6) Tous les utilisateurs doivent s'assurer qu'ils sont en possession de la dernière édition de cette publication.
7) Aucune responsabilité ne doit être imputée à l'IEC, à ses administrateurs, employés, auxiliaires ou mandataires, y compris ses experts particuliers et les membres de ses comités d'études et des Comités nationaux de l'IEC, pour tout préjudice causé en cas de dommages corporels et matériels, ou de tout autre dommage de quelque nature que ce soit, directe ou indirecte, ou pour supporter les coûts (y compris les frais de justice) et les dépenses découlant de la publication ou de l'utilisation de cette Publication de l'IEC ou de toute autre Publication de l'IEC, ou au crédit qui lui est accordé.
8) L'attention est attirée sur les références normatives citées dans cette publication. L'utilisation de publications référencées est obligatoire pour une application correcte de la présente publication.
9) L'attention est attirée sur le fait que certains des éléments du présent document de l'IEC peuvent faire l'objet de droits de brevet. L'IEC ne saurait être tenue pour responsable de ne pas avoir identifié de tels droits de brevets.

L'IEC 62351-9 a été établie par le WG 15, Sécurité des communications et des données, du comité d'études 57 de l'IEC, Gestion des systèmes de puissance et échanges d'informations associés. Il s'agit d'une Norme internationale.

Cette deuxième édition annule et remplace la première édition parue en 2017. Cette édition constitue une révision technique.

Cette édition inclut les modifications techniques majeures suivantes par rapport à l'édition précédente:

a) des composants de certificats et leur vérification ont été ajoutés;
b) le GDOI a été mis à jour pour inclure les résultats des essais d'interopérabilité;
c) des aspects liés au fonctionnement du GDOI ont été ajoutés;
d) la prise en charge du GDOI pour PTP (IEEE 1588) a été ajoutée comme spécifié par le profil de puissance de l'IEC/IEEE 61850-9-3;
e) l'enregistrement des événements de cybersécurité a été ajouté, ainsi que la mise en correspondance avec l'IEC 62351-14;
f) l'Annex B qui fournit des informations sur les algorithmes et mécanismes cryptographiques utilisés a été ajoutée.

Le texte de cette Norme internationale est issu des documents suivants:

| Projet | Rapport de vote |
|---|---|
| 57/2579/FDIS | 57/2594/RVD |

Le rapport de vote indiqué dans le tableau ci-dessus donne toute information sur le vote ayant abouti à son approbation.

La langue employée pour l'élaboration de cette Norme internationale est l'anglais.

Ce document a été rédigé selon les Directives ISO/IEC, Partie 2, il a été développé selon les Directives ISO/IEC, Partie 1 et les Directives ISO/IEC, Supplément IEC, disponibles sous www.iec.ch/members_experts/refdocs. Les principaux types de documents développés par l'IEC sont décrits plus en détail sous www.iec.ch/publications.

NOTE Les caractères d'imprimerie suivants sont employés:

Notation de syntaxe abstraite numéro un (ASN.1): en type `courier new` et **`bold courier new`**.

Le comité a décidé que le contenu de ce document ne sera pas modifié avant la date de stabilité indiquée sur le site web de l'IEC sous webstore.iec.ch dans les données relatives au document recherché. À cette date, le document sera:

- reconduit,
- supprimé,
- remplacé par une édition révisée, ou
- amendé.



**IMPORTANT – Le logo "colour inside" qui se trouve sur la page de couverture de ce document indique qu'elle contient des couleurs qui sont considérées comme utiles à une bonne compréhension de son contenu. Les utilisateurs devraient, par conséquent, imprimer cette publication en utilisant une imprimante couleur.**

# GESTION DES SYSTÈMES DE PUISSANCE ET ÉCHANGES D'INFORMATIONS ASSOCIÉS – SÉCURITÉ DES COMMUNICATIONS ET DES DONNÉES –

## Partie 9: Gestion de clé de cybersécurité des équipements de système de puissance

## 1 Domaine d'application

La présente partie de l'IEC 62351 spécifie la gestion des clés cryptographiques, principalement axée sur la gestion des clés à long terme, qui sont le plus souvent des paires de clés asymétriques, telles que des certificats de clés publiques et les clés privées correspondantes. Comme les certificats constituent la base, le présent document établit une fondation pour de nombreux services de l'IEC 62351 (voir également Annex A). La gestion des clés symétriques est également prise en compte, mais uniquement en ce qui concerne les clés de session pour les communications de groupe, telles qu'elles sont appliquées dans l'IEC 62351-6. L'objectif du présent document est de définir les exigences et les technologies permettant d'assurer l'interopérabilité de la gestion des clés en spécifiant ou en limitant les options de gestion de clés à utiliser.

Le présent document présume qu'une organisation (ou un groupe d'organisations) a défini une politique de sécurité pour sélectionner le type de clés et d'algorithmes cryptographiques qui seront utilisés, qui peuvent être à aligner sur d'autres normes ou exigences réglementaires. Le présent document spécifie donc uniquement les techniques de gestion de ces infrastructures de clé et de cryptographie sélectionnées. Le présent document présume que le lecteur a des notions de base en cryptographie et sur les principes de gestion des clés.

Les exigences relatives à la gestion des paires de clés (de session) symétriques dans le contexte des protocoles de communication sont spécifiées dans les parties de l'IEC 62351 qui utilisent ou spécifient une communication par paire, telles que:

- l'IEC 62351-3 pour TLS en profilant les options TLS;
- l'IEC 62351-4 pour la sécurité de bout en bout de la couche application;
- l'IEC 62351-5 pour le mécanisme de sécurité de la couche application pour l'IEC 60870-5-101/104 et l'IEEE 1815 (DNP3).

Les exigences relatives à la gestion des clés de groupe symétriques dans le contexte des protocoles de communication des systèmes de puissance sont spécifiées dans l'IEC 62351-6 pour l'utilisation de sécurité de groupe pour protéger les communications GOOSE et SV. L'IEC 62351-9 utilise GDOI comme protocole de gestion de clés par groupe déjà spécifié par l'IETF (Internet Engineering Task Force) pour gérer le paramètre de sécurité de groupe et améliore ce protocole pour transporter le paramètre de sécurité pour les communications GOOSE, SV et PTP.

Le présent document définit également les événements de sécurité pour des conditions spécifiques susceptibles d'identifier des problèmes pouvant exiger un traitement des erreurs. Cependant, les actions de l'organisation en réponse à ces conditions d'erreur ne relèvent pas du domaine d'application du présent document et sont censées être définies par la politique de sécurité des organisations.

À l'avenir, lorsque la cryptographie à clé publique sera mise en danger par l'évolution des ordinateurs quantiques, le présent document examinera également la cryptographie post-quantique dans une certaine mesure. Il est à noter qu'à l'heure actuelle, aucune mesure spécifique n'est prévue.

## 2 Références normatives

Les documents suivants sont cités dans le texte de sorte qu'ils constituent, pour tout ou partie de leur contenu, des exigences du présent document. Pour les références datées, seule l'édition citée s'applique. Pour les références non datées, la dernière édition du document de référence s'applique (y compris les éventuels amendements).

IEC TS 62351-2, *Power systems management and associated information exchange – Data and communications security – Part 2: Glossary of terms* (disponible en anglais seulement)

IEC 62351-3:— [1], *Gestion des systèmes de puissance et échanges d'informations associés – Sécurité des communications et des données – Partie 3: Sécurité des réseaux et des systèmes de communication – Profils comprenant TCP/IP*

IEC 62351-4, *Gestion des systèmes de puissance et échanges d'informations associés – Sécurité des communications et des données – Partie 4: Profils comprenant le MMS et ses dérivés*

IEC 62351-5, *Gestion des systèmes de puissance et échanges d'informations associés – Sécurité des communications et des données – Partie 5: Aspects de sécurité pour l'IEC 60870- 5 et ses dérivés*

IEC 62351-6, *Gestion des systèmes de puissance et échanges d'informations associés – Sécurité des communications et des données – Partie 6: Sécurité pour l'IEC 61850*

IEC 62351-14:— [2], *Power systems management and associated information exchange – Data and communications security – Part 14: Cyber security event logging* (disponible en anglais seulement)

ISO/IEC 9594-8:2020, Rec. UIT-T X.509 (2019), *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire – Partie 8: Cadre général des certificats de clé publique et d'attribut*

ISO/IEC 9594-11:2020, Rec. UIT-T X.510 (2020), *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire – Partie 11: Spécifications de protocole pour un fonctionnement sécurisé*

ISO/IEC 9834-1:2012, Rec. UIT-T X.660 (2011), *Technologies de l'information – Procédures opérationnelles des autorités d'enregistrement des identificateurs d'objet – Partie 1: Procédures générales et arcs sommitaux de l'arborescence des identificateurs d'objet internationale*

IETF RFC 5272, *Certificate Management over CMS (CMC)* (disponible en anglais seulement)

IETF RFC 5755, *An Internet Attribute Certificate Profile for Authorization* (disponible en anglais seulement)

IETF RFC 5934, *Trust Anchor Management Protocol (TAMP)* (disponible en anglais seulement)

IETF RFC 6407, *The Group Domain of Interpretation* (disponible en anglais seulement)



---

[1] En cours d'élaboration. Stade au moment de la publication: IEC/RFDIS 62351-3:2023.

[2] En cours d'élaboration. Stade au moment de la publication: IEC/ACDV 62351-14:2023.

IETF RFC 6960, *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP* (disponible en anglais seulement)

IETF RFC 7030, *Enrolment over Secure Transport* (disponible en anglais seulement)

IETF RFC 8052, *Group Domain of Interpretation (GDOI) Protocol Support for IEC 62351 Security* (disponible en anglais seulement)

IETF RFC 8263, *Group Domain of Interpretation (GDOI) GROUPKEY-PUSH Acknowledgement Message* (disponible en anglais seulement)

IETF RFC 8894, *Simple Certificate Enrolment Protocol* (disponible en anglais seulement)

# 3 Termes, définitions et abréviations

## 3.1 Termes et définitions

Pour les besoins du présent document, les termes et définitions suivants s'appliquent.

L'ISO et l'IEC tiennent à jour des bases de données terminologiques destinées à être utilisées en normalisation, consultables aux adresses suivantes:

- IEC Electropedia: disponible à l'adresse http://www.electropedia.org/
- ISO Online browsing platform: disponible à l'adresse http://www.iso.org/obp

**3.1.1**
**clés asymétriques**
deux clés associées, l'une publique et l'autre privée, utilisées pour exécuter des opérations complémentaires comme le chiffrement et le déchiffrement ou la génération et la vérification de signature

**3.1.2**
**liste d'autorisation et de validation**
AVL
liste signée contenant des informations destinées à une entité d'AVL concernant les entités potentielles de communication et les restrictions possibles des communications qu'impliquent ces entités

[SOURCE: ISO/IEC 9594-8:2020, 3.5.9]

**3.1.3**
**entité de liste d'autorisation et de validation**
entité d'AVL
entité qui, lorsqu'elle agit en tant que partie utilisatrice, dépend de l'AVL délivrée par un agent d'autorisation désigné

[SOURCE: ISO/IEC 9594-8:2020, 3.5.10]

**3.1.4**
**agent d'autorisation**
entité à laquelle une ou plusieurs entités fonctionnant comme entités d'AVL ont accordé leur confiance afin de créer, maintenir et signer des listes d'autorisation et de validation

[SOURCE: ISO/IEC 9594-8:2020, 3.5.11]

**3.1.5**
**chemin de certification**
liste ordonnée d'un ou de plusieurs certificats de clé publique, commençant par un certificat de clé publique signé par l'ancre de confiance et se terminant par le certificat de clé publique d'entité finale à valider

Note 1 à l'article: Tous les certificats de clé publique intermédiaires, le cas échéant, sont des certificats de CA dans lesquels le sujet du certificat de clé publique précédent est l'émetteur du certificat de clé publique suivant.

[SOURCE: ISO/IEC 9594-8:2020, 3.5.21]

**3.1.6**
**demande de certification**
demande émise lorsqu'un nouveau certificat de clé publique ou un renouvellement de certificat de clé publique est exigé

[SOURCE: IETF RFC 2986]

**3.1.7**
**fonction de contrôleur**
intersection du droit de propriété, du contrôle physique et du contrôle logique sur un dispositif ou un système, dans laquelle la nature des ententes contractuelles entre la propriété et le contrôle du dispositif ou du système n'est pas importante dans ce contexte

**3.1.8**
**liaison cryptographique**
utilisation d'une ou de plusieurs techniques cryptographiques par un système de gestion de clé cryptographique afin d'établir une association de confiance entre une clé et les éléments de métadonnées sélectionnés

[SOURCE: NIST SP 800-130]

**3.1.9**
**jeu de données**
ensemble de données

**3.1.10**
**dispositif**
composant mettant en œuvre des fonctionnalités spécifiques qui peuvent agir en tant que client (par exemple client TLS) ou serveur (par exemple centre de distribution de clé pour GDOI)

**3.1.11**
**signature numérique**
résultat d'une transformation cryptographique des données qui, si elle est correctement mise en œuvre, offre un mécanisme pour vérifier l'authentification d'origine, l'intégrité des données et la non-répudiation du signataire

[SOURCE: FIPS 186]

**3.1.12**
**entité finale (PKI)**
entité à laquelle a été attribué un certificat de clé publique d'entité finale, où la clé privée ne peut pas être utilisée pour signer d'autres certificats de clé publique, mais qui peut être utilisée pour la signature à d'autres fins

**3.1.13**
**entité**
terme générique couvrant les utilisateurs humains, les systèmes d'automatisation, les applications logicielles, les nœuds de communication, les dispositifs de terrain et d'autres types d'actifs

**3.1.14**
**empreinte digitale**
résultat d'un hachage ("empreinte de clé") utilisé pour authentifier une clé publique ou d'autres données

[SOURCE: IETF RFC 4949]

**3.1.15**
**contrôleur de groupe/serveur de clés**
GCKS
dispositif qui définit la politique de groupe et distribue les clés pour cette politique

[SOURCE: IETF RFC 3740]

**3.1.16**
**domaine d'interprétation de groupe**
GDOI
domaine qui gère les associations de sécurité de groupe, qui sont utilisées par IPsec et éventuellement d'autres protocoles de sécurité des données

Note 1 à l'article: Ces associations de sécurité protègent une ou plusieurs clés de chiffrement de clé (KEK), clés de chiffrement du trafic (TEK) ou les données partagées par les membres du groupe. Le GDOI utilise la notion de contrôleur de groupe qui prend en charge l'établissement des associations de sécurité entre les membres d'un groupe.

[SOURCE: IETF RFC 6407]

**3.1.17**
**membre du groupe**
GM
membre autorisé d'un groupe sûr, qui envoie et/ou reçoit des paquets IP en relation avec le groupe

**3.1.18**
**fonction de hachage**
fonction (mathématique) qui mappe des données de taille arbitraire à des données de taille fixe appelées "résumé"

**3.1.19**
**code d'authentification de message par hachage**
HMAC
code cryptographique utilisé pour l'authentification avec des clés symétriques et pour l'intégrité des données

[SOURCE: IETF RFC 2104]

**3.1.20**
**centre de distribution de clés**
KDC
centre qui, dans un contexte décrit dans l'IEC 62351-9, offre un service de réseau qui fournit des clés de session (symétriques) provisoires à un ensemble préalablement défini d'homologues, après authentification

Note 1 à l'article: Également appelé "contrôleur de groupe/serveur de clés (GCKS)" (voir GDOI).

**3.1.21**
**système de gestion de clé**
système de gestion (par exemple génération, distribution, stockage, sauvegarde, archivage, récupération, utilisation, révocation et destruction) des clés cryptographiques et de leurs métadonnées

**3.1.22**
**code d'authentification de message**
MAC
somme de contrôle cryptographique réalisée sur les données et qui utilise une clé symétrique pour détecter les modifications accidentelles et volontaires des données

[SOURCE: SP 800-63; FIPS 201]

**3.1.23**
**identificateur d'objet**
suite ordonnée de valeurs entières primaires allant de la racine de l'arborescence des identificateurs d'objet internationaux jusqu'à un nœud, qui identifie ledit nœud sans ambiguïté

[SOURCE: ISO/IEC 9834-1:2012, 3.5.11]

**3.1.24**
**protocole d'état de certificat en ligne**
OCSP
protocole qui permet aux applications de déterminer l'état (de révocation) d'un certificat identifié

Note 1 à l'article: L'OCSP peut être utilisé pour satisfaire à certaines des exigences opérationnelles visant à fournir des informations de révocation plus pertinentes qu'avec les listes de révocation de certificat (CRL). Il peut être utilisé pour obtenir des informations d'état supplémentaires. Un client OCSP envoie une demande d'état à un répondeur OCSP et suspend l'acceptation du certificat en question tant que le répondeur n'a pas donné de réponse.

[SOURCE: IETF RFC 6960]

**3.1.25**
**opérateur**
type particulier d'utilisateur qui est généralement responsable du fonctionnement correct du matériel commandé

**3.1.26**
**clé prépartagée**
PSK
secret partagé à l'avance entre les deux entités (des applications logicielles ou des dispositifs, par exemple) de manière à être en mesure de s'authentifier ou d'établir une connexion sécurisée

**3.1.27**
**clé privée**
(dans un cryptosystème à clé publique) clé d'une paire de clés d'entité qui est uniquement reconnue par cette entité

[SOURCE: ISO/IEC 9594-8:2020, 3.5.50]

**3.1.28**
**certificat de clé publique**
clé publique d'une entité, accompagnée d'autres informations, rendue infalsifiable par signature numérique avec la clé privée de la CA qui l'a délivrée

Note 1 à l'article: Un certificat de clé publique est souvent appelé "certificat X.509" ou "certificat numérique". Cependant, de tels termes sont ambigus, car ils peuvent également désigner des certificats d'attribut qui sont également définis par l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019).

[SOURCE: ISO/IEC 9594-8:2020, 3.5.58, modifié, ajout de la Note 1 à l'article]

**3.1.29**
**normes de chiffrement à clé publique**
PKCS
spécifications publiées par les Laboratoires RSA en coopération avec les développeurs de systèmes sécurisés du monde entier afin d'accélérer le déploiement de la cryptographie à clé publique

[SOURCE: www.rsa.com]

**3.1.30**
**génération de nombres aléatoires**
RNG
processus permettant de générer une série imprévisible de nombres

Note 1 à l'article: Chaque valeur individuelle est dite aléatoire si chacune d'elles présente une probabilité égale d'être sélectionnée dans la population totale.

[SOURCE: NIST SP 800-57]

**3.1.31**
**autorité d'enregistrement**
RA
aspects liés aux responsabilités d'une autorité de certification quant à l'identification et à l'authentification du sujet d'un certificat de clé publique à émettre par ladite autorité de certification

Note 1 à l'article: Une autorité d'enregistrement peut être une entité distincte ou faire partie intégrante de l'autorité de certification.

Note 2 à l'article: Le domaine d'application de cette définition diffère de celui stipulé dans la Rec. UIT-T X.660 | l'ISO/IEC 9834-1.

[SOURCE: ISO/IEC 9594-8:2020, 3.5.61]

**3.1.32**
**partie utilisatrice**
entité qui se fie aux données d'un certificat de clé publique pour prendre des décisions

[SOURCE: ISO/IEC 9594-8:2020, 3.5.62]

**3.1.33**
**association de sécurité**
SA
relation établie entre au moins deux entités, leur permettant de protéger les données qu'elles échangent

[SOURCE: NIST IR 7298 Rév. 1]

**3.1.34**
**force de la sécurité**
aptitude des technologies de sécurité à empêcher un assaillant potentiel de contourner ou corrompre la sécurité

Note 1 à l'article: Elle est souvent mesurée en bits de sécurité.

[SOURCE: NIST SP 800-130]

**3.1.35**
**clé de session**
dans le contexte du chiffrement symétrique, clé provisoire ou utilisée sur une courte durée

[SOURCE: IETF RFC 2828]

**3.1.36**
**clé symétrique**
clé cryptographique utilisée pour réaliser des opérations cryptographiques et leurs opérations inverses (chiffrer et déchiffrer, créer et vérifier un code d'authentification de message, par exemple)

[SOURCE: NIST SP 800-63]

**3.1.37**
**confiance**
ferme conviction de la fiabilité et de la véracité des informations ou de la capacité et de la disposition d'une entité à agir de manière appropriée, dans un contexte spécifié

**3.1.38**
**ancre de confiance**
entité de confiance à laquelle une partie utilisatrice a accordé sa confiance et qu'il a utilisée pour valider des certificats de clé publique dans les chemins de certification

[SOURCE: ISO/IEC 9594-8:2020, 3.5.72]

**3.1.39**
**informations d'ancre de confiance**
au moins le nom distinctif de l'ancre de confiance, la clé publique associée, l'identificateur d'algorithme, les paramètres de clé publique (le cas échéant) et toutes les contraintes relatives à son utilisation (y compris une période de validité)

Note 1 à l'article: Les informations d'ancre de confiance peuvent être fournies sous forme de certificat de CA autosigné ou de certificat de CA normal (c'est-à-dire un certificat croisé).

[SOURCE: ISO/IEC 9594-8:2020, 3.5.73]

**3.1.40**
**magasin d'ancres de confiance**
ensemble des informations d'ancre de confiance au niveau d'une partie utilisatrice pour une ou plusieurs ancres de confiance

[SOURCE: ISO/IEC 9594-8:2020, 3.5.74]

### 3.2 Abréviations et acronymes

Des abréviations et acronymes supplémentaires sont inclus dans l'IEC TS 62351-2.

AA Attribute Authority (autorité d'attribut)

AEAD Authenticated Encryption with Associated Data (chiffrement authentifié avec données associées)

ASN.1 Abstract Syntax Notation One (notation de syntaxe abstraite nº 1)

AVL Authorization and Validation List (liste d'autorisation et de validation)

AVMP Authorization and Validation Management Protocol (protocole de gestion des autorisations et des validations)

BRSKI Bootstrapping of Remote Secure Key Infrastructures (protocole BRSKI)

CA Certification Authority (autorité de certification)

CASP Certification Authority Subscription Protocol (protocole CASP)

CIA Confidentiality, Integrity and Availability (confidentialité, intégrité et disponibilité)

CMC Certificate Management over CMS (gestion de certificats sur CMS)

CMS Cryptographic Message Syntax (syntaxe de message cryptographique)

CPS Certificate Practice Statement (déclaration d'utilisation du certificat)

CSR Certificate Signing Request (demande de signature de certificat, également utilisée dans le contexte d'une demande de certification)

DER Distinguished Encoding Rules (règles de codage distinctives)

DOI Domain Of Interest (domaine d'intérêt)

DSA Digital Signature Algorithm (algorithme de signature numérique)

ECDSA Elliptic Curve Digital Signature Algorithm (algorithme de signature numérique à courbe elliptique)

EdDSA Edwards-curve Digital Signature Algorithm (algorithme de signature numérique de la courbe d'Edwards)

EST Enrollment over Secure Transport (protocole EST)

FQDN Fully Qualified Domain Name (nom de domaine pleinement qualifié)

GCKS Group Controller/Key Server (contrôleur de groupe/serveur de clés)

GAP Group Associated Policy (politique associée au groupe)

GCM Galois Counter Mode (mode Galois/compteur)

GDOI Group Domain Of Interpretation (domaine d'interprétation de groupe)

GKDC Group KDC (KDC de groupe)

GM Group Member (membre du groupe)

GMAC Galois Message Authentication Code (code d'authentification de message avec le mode Galois)

GOOSE Generic Object-Oriented Substation Event (événement générique de sous-station orientée objet)

HMAC Hash-based Message Authentication Code (code d'authentification de message par hachage)

HTTP HyperText Transfer Protocol (protocole HTTP)

ICV Integrity Check Value (valeur de contrôle d'intégrité)

IDevID Initial Device IDentifier (IEEE 802.1AR) (identificateur de dispositif initial)

IED Intelligent Electronic Device (dispositif électronique intelligent)

ID IDentity (identité)

IKE Internet Key Exchange (échange de clés Internet)

| | |
|---|---|
| ISAKMP | Internet Security Association And Key Management Protocol (protocole ISAKMP) |
| KD | Key Download (téléchargement de clé) |
| KDC | Key Distribution Centre (centre de distribution de clés, également appelé GKDC) |
| KDF | Key Derivation Function (fonction de dérivation de la clé) |
| KEK | Key Encryption Key (clé de chiffrement de clé) |
| LDevID | Local Device IDentifier (IEEE 802.1AR) (identificateur de dispositif local) |
| MUD | Manufacturer Usage Description (RFC 8520) (description du dispositif du fabricant) |
| OCSP | Online Certificate Status Protocol (protocole OCSP) |
| OTP | One Time Password (mot de passe à usage unique) |
| PDU | Protocol Data Unit (unité de données du protocole) |
| PEM | Privacy-enhanced Electronic Mail (courrier électronique à confidentialité améliorée) |
| PKCS | Public-Key Cryptography Standard (normes de chiffrement à clé publique) |
| PKI | Public-Key Infrastructure (infrastructure de clé publique) |
| RA | Registration Authority (autorité d'enregistrement) |
| RNG | Random Number Generation (génération de nombres aléatoires) |
| RSA | Rivest Shamir Adleman (cryptosystème à clé publique de Rivest-Shamir-Adleman) |
| SA | Security Association (association de sécurité) |
| SCEP | Simple Certificate Enrollment Protocol (protocole SCEP) |
| SPI | Security Parameter Index (indice du paramètre de sécurité) |
| SV | Sampled Values (valeurs échantillonnées) |
| TAMP | Trust Anchor Management Protocol (protocole TAMP) |
| TEK | Traffic Encryption Key (clé de chiffrement du trafic) |

# 4 Concepts de sécurité applicables aux systèmes de puissance

## 4.1 Généralités

Le but du présent article est d'introduire des concepts cryptographiques applicables dans les systèmes de puissance. Il fournit une vue d'ensemble des objectifs de sécurité dans les systèmes de puissance, des algorithmes cryptographiques qui peuvent être utilisés pour atteindre ces objectifs de sécurité, et de la façon dont ils peuvent être appliqués dans différents environnements de déploiement. En outre, le présent article, conjointement avec l'Annex B, fournit des informations générales sur les algorithmes cryptographiques utilisés, ainsi que les conditions potentielles aux limites dans différents environnements. La partie normative du présent document spécifie leur application. Les événements de sécurité sont également traités de manière générale.



## 4.2 Objectifs de sécurité

### 4.2.1 Confidentialité

La confidentialité est nécessaire pour empêcher l'accès aux données sensibles, en particulier lorsqu'elles sont en transit.

Dans les mises en œuvre de systèmes de puissance, l'objectif cryptographique de confidentialité des données est en général atteint en chiffrant le message par cryptographie à clé symétrique. Le message peut être chiffré individuellement au niveau de l'application et/ou du canal de communication sous-jacent. Comme indiqué précédemment, la confidentialité de ce message dépend de l'expéditeur et du destinataire qui gardent le secret de la clé symétrique. De plus, la cryptographie asymétrique est souvent utilisée pour échanger ou négocier des clés de session symétriques qui sont valables pendant une durée spécifiée, en réduisant ainsi le risque lié à la conservation d'un secret de clé de session spécifique.

Pour limiter les dommages si une clé symétrique est compromise, il est recommandé que la clé symétrique utilisée sur une connexion ou une association soit unique. Cela peut être obtenu en utilisant une négociation de clé éphémère comme décrit à l'Article B.8. Pour la TLS, cela peut être effectué par le choix de suites chiffrées appropriées (comme indiqué dans l'IEC 62351-3).

### 4.2.2 Intégrité des données

L'intégrité des données concerne la détection de modifications non autorisées des données pendant le transit ou au repos. Elle permet au vérificateur de la valeur de contrôle d'intégrité de détecter toute violation d'intégrité des données associées.

Deux méthodes sont utilisées pour détecter la violation de l'intégrité des données. Toutes deux utilisent des fonctions de hachage cryptographique pour détecter toute modification des données reçues. Comme indiqué à l'Article B.5, une fonction de hachage entraîne la production d'un résumé de longueur fixe sur certaines données d'entrée. Ce résumé varie de manière imprévisible, même pour un léger changement dans les données à hacher. Les deux méthodes utilisées pour détecter la violation d'intégrité sont:

a) l'utilisation des signatures numériques (voir B.3.6);

b) l'utilisation des valeurs de contrôle d'intégrité (ICV) (voir Article B.6).

Le chiffrement authentifié est une autre variante qui assure la confidentialité et l'intégrité des données en une seule opération. Il est décrit à l'Article B.7.

### 4.2.3 Authentification

Lorsqu'une entité reçoit des données d'une autre entité, il est essentiel que l'expéditeur de ces données soit authentifié de façon sûre. Le présent document identifie les deux mêmes méthodes d'authentification que celles mentionnées pour l'intégrité des données en 4.2.2:

a) l'utilisation de la signature numérique, où une signature numérique vérifiée prouve avec une certaine certitude que l'expéditeur est en possession de la clé privée correspondant à la clé publique utilisée pour vérifier la signature numérique. Elle est examinée plus en détail B.3.6;

b) l'utilisation de la valeur de contrôle d'intégrité (ICV), où une ICV vérifiée prouve avec une certaine certitude que l'expéditeur est en possession de la même clé symétrique que le destinataire. La clé symétrique utilisée pour générer une ICV étant établie dans le cadre du processus d'authentification, elle est donc liée à cette authentification. L'ICV est examinée plus en détail à l'Article B.6.

### 4.2.4 Non-répudiation

La non-répudiation se rapporte à la capacité de rattacher une action (par exemple une commande ou un message) à une entité émettrice de manière irréfutable. Elle peut associer un expéditeur à l'action d'envoi d'un message spécifique, ou un récepteur à l'action de réception d'un message spécifique.

La série ISO/IEC 13888 spécifie différents niveaux de non-répudiation, ainsi que deux manières distinctes d'établir la non-répudiation:

a) l'ISO/IEC 13888-2 spécifie des procédures utilisant la cryptographie à clé symétrique. Ces techniques exigent l'intervention d'un tiers de confiance;

b) l'ISO/IEC 13888-3 spécifie des procédures d'utilisation la cryptographie à clé asymétrique.

### 4.3 Algorithmes cryptographiques et concepts associés

Le domaine d'application du présent document est la gestion des clés pour l'application aux systèmes de puissance, incluant la gestion basée sur la PKI du matériel de clé asymétrique, ainsi que la gestion des clés de groupe pour les clés symétriques. La gestion des clés s'applique à toutes les parties de la série IEC 62351 qui utilisent un matériel de clé cryptographique pour protéger les informations échangées dans les protocoles d'automatisation des systèmes de puissance.

L'Annex B fournit des explications détaillées sur les différents types d'algorithmes cryptographiques et les concepts associés, tels que la confiance, le niveau de sécurité des ancres de confiance, la génération de nombres aléatoires et la migration d'un algorithme cryptographique à un autre (généralement plus forts). Ce dernier concept devient plus important pour faire face aux progrès de l'informatique quantique qui met spécifiquement en danger les algorithmes cryptographiques asymétriques.

En plus d'une introduction générale aux algorithmes cryptographiques, les types d'algorithmes suivants sont examinés dans une certaine mesure:

- algorithmes asymétriques qui comprennent les algorithmes RSA, ECDSA et EdDSA, incluant la description générale, la génération de clés et les aspects liés à la sécurité des différents types d'algorithmes asymétriques;
- algorithmes à clé symétrique dans différents modes opérationnels. Seuls les algorithmes basés sur la norme de chiffrement avancé (AES) sont examinés. Les modes opérationnels traités sont l'enchaînement de blocs de chiffrement (AES-CBC) et un mode basé sur un mécanisme de compteur (AES-CTR). Ils sont tous deux définis pour les tailles de clés de 128 bits et 256 bits;
- algorithmes de hachage. Les algorithmes concernés sont des algorithmes dits de hachage de sécurité (SHA) et seuls SHA-256 et SHA-512 sont examinés. La description comprend une liste des domaines où des algorithmes de hachage sont utilisés;
- algorithmes de création de valeurs de contrôle d'intégrité (ICV). Deux types d'algorithmes ICV sont pertinents pour le présent document. Le premier type est constitué des algorithmes de code d'authentification de message par hachage de clé (HMAC) qui spécifient l'utilisation d'une clé symétrique et d'un algorithme de hachage. Deux algorithmes HMAC sont inclus, l'un basé sur SHA-256 et l'autre sur SHA-512. L'autre type d'algorithme ICV est la norme de chiffrement avancé – code d'authentification de message avec le mode Galois (AES-GMAC);
- les algorithmes de chiffrement authentifié avec données associées (AEAD) comme AES-GCM ou AES-CCM sont des algorithmes cryptographiques qui, en plus de spécifier le chiffrement/déchiffrement, fournissent également des capacités d'intégrité pour les données chiffrées, ainsi que pour certaines données associées. Deux types d'algorithmes AEAD sont inclus. Ils sont tous deux basés sur l'AES pour le chiffrement, mais ils utilisent différentes techniques pour générer la partie ICV. Ils sont tous deux prévus pour les tailles de clés AES de 128 bits et 256 bits;
- accord de clé Diffie-Hellman pour calculer une clé partagée basée sur des clés privées non partagées. La génération de clés symétriques à la volée est essentielle pour la gestion des clés et la méthode Diffie-Hellman est une technique très utilisée à cette fin. Deux types de schémas Diffie-Hellman sont proposés en utilisant deux différents types de mathématiques (champs de nombres premiers et courbes elliptiques). Ils sont tous deux définis de manière à prendre en charge les migrations vers des algorithmes plus sûrs.

Ces algorithmes et concepts étant appliqués dans des systèmes de puissance, l'Annex B donne une vue d'ensemble, ainsi que des informations générales et des renvois vers des informations complémentaires. Cette annexe est destinée à servir d'introduction afin de mieux comprendre l'application de ces algorithmes dans les paragraphes normatifs suivants du présent document.

# 5 Techniques d'établissement et de gestion des clés

## 5.1 Généralités

Le présent article offre une vue d'ensemble des techniques de gestion du matériel de clés symétriques et asymétriques, ainsi que des concepts et infrastructures nécessaires associés. Pour le matériel de clé symétrique, l'accent est mis sur la gestion des clés de groupe. Le présent article fournit les informations de base qui seront utilisées dans la description normative des articles suivants.

## 5.2 Cycle de vie de la gestion de clés

### 5.2.1 Gestion de clés pendant le cycle de vie d'un dispositif

La gestion de clés fait partie intégrante du cycle de vie des dispositifs, suivant le schéma général représenté à la Figure 1.

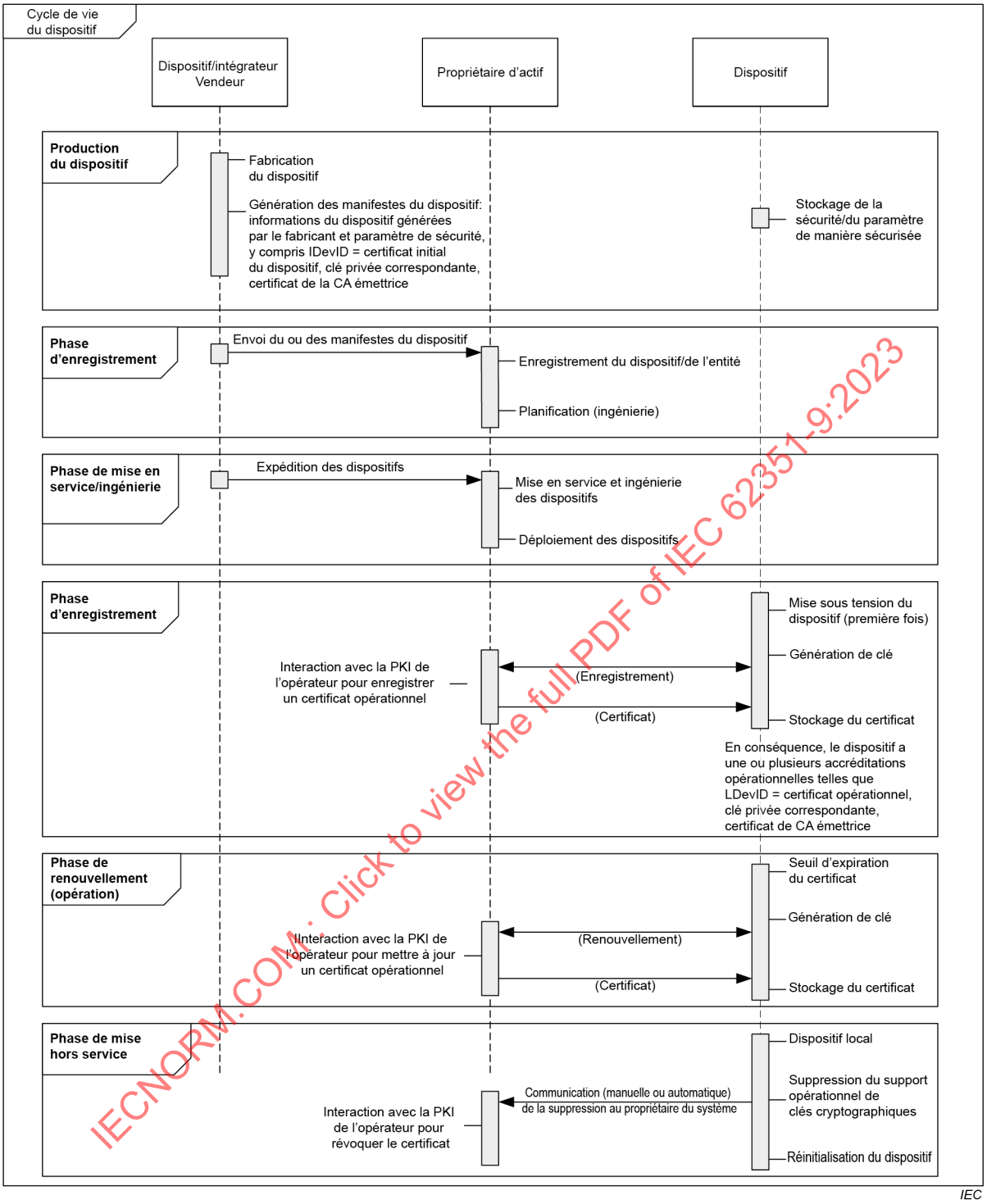





**Figure 1 – Vue d'ensemble de la gestion de clés pendant le cycle de vie d'une entité**

Les fabricants peuvent prendre en charge le cycle de vie de la gestion de clés d'une entité en fournissant un manifeste de sécurité des informations d'identité (identificateur unique du fabricant, mot de passe à usage unique, certificat provenant d'une CA ou certificat d'entité du fabricant, par exemple) pour chacun de leurs dispositifs et systèmes. L'IEEE 802.1AR définit cet ensemble initial d'accréditations sous la forme d'un identificateur de dispositif initial (IDevID) constitué du certificat initial, de la clé privée correspondante, ainsi que de la chaîne de certificats jusqu'à l'ancre de confiance. À chaque changement de propriété, voire d'état (par exemple entreposage par rapport à déploiement), il convient d'enregistrer les nouveaux certificats, en créant ainsi une chaîne digne de confiance de l'identité de produit actuellement utilisée. Lorsque ces entités sont finalement conçues et déployées, leurs certificats de fabricant

(si disponibles) peuvent être utilisés pour enregistrer l'entité dans les opérations, ce qui conduit à un ou plusieurs certificats opérationnels (identificateur de dispositif local, LDevID) qui leur permettent de s'authentifier et de commencer leurs échanges d'informations. Afin d'éviter des erreurs de vérification des certificats opérationnels, il convient de renouveler ces certificats peu de temps avant leur expiration.

Il convient également d'envisager la mise hors service des dispositifs, accompagnée de la suppression de toute accréditation cryptographique opérationnelle pour s'assurer que ces informations ne peuvent pas être utilisées à mauvais escient une fois que le dispositif a été mis hors service. Il est à noter que l'IDevID (le cas échéant) n'est généralement pas supprimé.

Les fabricants sont encouragés à simplifier le processus d'effacement en toute sécurité d'un dispositif de toutes accréditations opérationnelles (certificats, base de données d'ancres de confiance, etc.) et de restauration de l'ensemble initial d'accréditations par défaut ou, si ce processus est nécessairement complexe, à le documenter au moins clairement. Cela contribuerait à la fois à une bonne hygiène de fin de service et, le cas échéant, à la certitude que les références d'essais en atelier ou sur banc d'essai ont été supprimées avant le déploiement. Les propriétaires de systèmes participent au processus de mise hors service dans le cadre du cycle de vie (voir 5.2.2).

### 5.2.2 Cycle de vie d'une clé cryptographique

Les clés cryptographiques en général, et les certificats en particulier, suivent un cycle de vie. Ils sont créés, distribués, installés, appliqués, mis à jour et détruits de manière à satisfaire aux exigences en matière de politique de sécurité de la gestion de clés. Par exemple, le cycle de vie classique des clés se trouvant dans des systèmes PKI est représenté à la Figure 2. La gestion du cycle de vie général des clés est présentée plus en détail dans l'ISO/IEC 11770 [6] [3] et dans le document NIST SP 800-130 [11]. Le cycle de vie d'un certificat, qui est généralement géré, est présenté ici à un niveau abstrait. Pour plus de détails sur la gestion des certificats, voir 5.8.



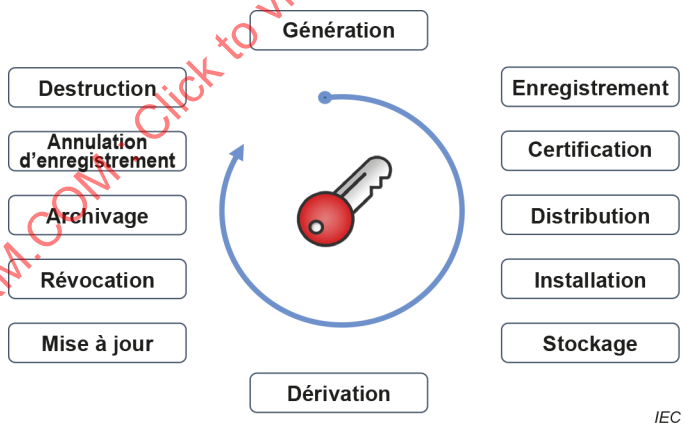



**Figure 2 – Cycle de vie d'une clé cryptographique**

La liste suivante décrit brièvement chacune des étapes possibles du cycle de vie d'une clé cryptographique. Toutes les étapes sont exigées pour les clés asymétriques, l'enregistrement, la certification, l'annulation d'enregistrement et la révocation ne l'étant pas pour les clés symétriques:

- Génération: les clés cryptographiques peuvent être créées par l'entité elle-même si elle dispose des capacités appropriées. En variante, les clés peuvent être créées en externe et installées sur l'entité cible. Souvent, cette dernière approche est utilisée si l'entité ne peut pas prendre en charge l'un des composants critiques pour générer les clés, à savoir le

[3] Les chiffres entre crochets renvoient à la Bibliographie.

générateur de nombres aléatoires (RNG) qui doit assurer le caractère aléatoire à un niveau élevé (voir B.12). Il peut également être nécessaire de créer des clés en externe si elles doivent être archivées (par exemple pour les clés de chiffrement utilisées pour les données stockées). Si les certificats sont sur le point d'expirer, il est nécessaire que les entités fassent une demande de renouvellement de certificat;

- Enregistrement: si des certificats et un processus de PKI sont utilisés, l'enregistrement par l'intermédiaire d'une autorité d'enregistrement (RA) établit "l'identité" d'une entité. Ceci est réalisé en générant la paire de clés soit localement à l'entité, soit de manière centralisée. L'entité fournit une demande de certification à la RA, qui est ensuite traitée conjointement avec l'autorité de certification (CA);
- Certification: à la suite de l'enregistrement d'une entité par une RA, la CA délivre un certificat signé numériquement à l'entité qui connecte ensuite sa clé publique à la clé privée détenue par l'entité. Selon la génération de clé, cette opération peut faire partie intégrante de la génération de clé dans un centre de confiance ou être effectuée à l'aide des informations envoyées dans une demande de certification;
- Distribution: la distribution de clés est le processus de transmission sécurisée des clés et de leurs informations de gestion aux entités autorisées. Les clés privées (secrètes), bien qu'idéalement générées dans les dispositifs finaux et n'exigeant donc aucune distribution, doivent être distribuées de manière sécurisée si elles ont été générées en externe. Les clés publiques, à l'intérieur des certificats de clé publique, peuvent être distribuées de manière non sécurisée étant donné que les dispositifs finaux peuvent s'assurer de leur authenticité en vérifiant directement la signature du certificat;
- Installation: une clé peut être installée dans une entité lors des processus de fabrication et/ou d'ingénierie (clé prépartagée) ou peut l'être ultérieurement par des procédures en ligne. Ce dernier processus peut exiger de communiquer avec un serveur de sécurité hors bande à l'aide d'un canal de communication distinct, ou dans la bande dans le cadre d'une communication de service;
- Stockage: il est recommandé de stocker les clés privées/symétriques dans l'entité de manière sécurisée. Il convient que le stockage sécurisé des clés soit conforme à la FIPS 140-3 [70] ou à l'ISO/IEC 19790 [71];
- Dérivation: les clés cryptographiques utilisées comme des clés de session ou autres clés à court terme peuvent être dérivées des clés privées ou symétriques stockées dans l'entité;
- Mise à jour: il est nécessaire de mettre régulièrement à jour les clés cryptographiques. Les clés cryptographiques ont une durée de vie prévue (par exemple les certificats de clés publiques délivrés aux utilisateurs peuvent avoir une durée de vie de 2 ans), alors que les certificats de clés publiques délivrés aux serveurs peuvent être limités à 1 an ou moins, les clés prépartagées à quelques semaines ou quelques mois et les clés de session à quelques heures et/ou quelques milliers de paquets. Des recommandations concernant la durée de vie des clés cryptographiques peuvent être consultées dans le document NIST SP 800-57, Partie 1 [73] et dans la norme allemande BSI TR 02102-1 [58];
- Révocation: un certificat peut être révoqué lorsque son utilisation n'est plus autorisée. Ce cas de figure peut se produire si un dispositif est retiré du service ou que son rôle change, ou bien si une clé privée est compromise ou suspectée de l'être;
- Archivage: il convient d'archiver les clés symétriques nécessaires au déchiffrement des données stockées à long terme dans un stockage confidentiel sécurisé afin de pouvoir les récupérer (pour permettre d'accéder ultérieurement aux données chiffrées stockées). De plus, pour prendre en charge la future vérification de signature, il convient également d'archiver les certificats de clé publique, même si le stockage confidentiel sécurisé n'est pas nécessaire;
- Annulation d'enregistrement: cette procédure est généralement réalisée par l'autorité d'enregistrement pour supprimer l'association d'une entité à une clé dédiée. Elle est souvent utilisée pendant la phase de destruction de clé;
- Destruction: lorsqu'une clé cryptographique est détruite, elle ne doit être ni récupérée ni utilisée. Cette opération intervient à la fin du cycle de vie de la clé cryptographique.

### 5.3 Utilisation des clés cryptographiques

Les clés cryptographiques sont utilisées à différentes fins et à différentes phases du cycle de vie du produit. Elles sont appliquées comme suit:

- certificats de clé publique et clés privées correspondantes: utilisés pour identifier, authentifier et autoriser des entités. En outre, ils sont souvent employés dans la négociation ou l'établissement de clés de session;
- clés prépartagées (par exemple pour la communication en temps réel): utilisées comme un secret partagé entre des entités pour sécuriser l'échange de données (intégrité, confidentialité) entre elles. Cela peut être utile dans les environnements qui ne prennent pas en charge une PKI. De plus, ce schéma peut être utilisé lors de l'enregistrement de PKI, en permettant à une entité de s'authentifier auprès d'une autorité de certification (CA), lors du traitement d'une demande de certification, par exemple. *Il est à noter que la sécurité de la clé prépartagée est liée à sa complexité.* Les règles de complexité des mots de passe sont généralement régies par les stratégies de sécurité de l'organisation;
- clés de session (par paire ou basées sur le groupe): utilisées pour le chiffrement efficace ou les contrôles d'intégrité des messages de communication;
- paramètres de session (algorithmes cryptographiques dédiés): utilisés pour prendre en charge les sessions (clés, durée de vie, etc.);
- jetons d'accès cryptographiques: souvent utilisés pour transférer/fournir une autorisation/un accès à des ressources aux entités;
- autres accréditations pertinentes des entités ou organisations.

### 5.4 Politique de sécurité du système de gestion de clés

Il est nécessaire d'assurer la protection des clés cryptographiques. Par conséquent, il convient que chaque organisation (ou groupe d'organisations qui prévoient d'échanger des données) développe une politique de sécurité pour son système de gestion de clés, afin d'établir et de spécifier toutes les exigences en matière de protection de la confidentialité, de l'intégrité, de la disponibilité et de l'authentification de la source de toutes les clés cryptographiques et leurs métadonnées associées utilisées par l'organisation. Il convient que ces exigences de protection couvrent l'ensemble du cycle de vie d'une clé, y compris lorsqu'elle est initialement fournie, opérationnelle, stockée et transportée. Il convient également d'inclure dans la politique de sécurité de gestion de clés, la sélection de tous les mécanismes cryptographiques et les protocoles à utiliser dans tous les systèmes de l'organisation.

Les systèmes de gestion de clés ont également besoin d'un soutien administratif de la sécurité afin de s'assurer que les politiques de sécurité sont suivies et que les procédures sont maintenues. La spécification de ce soutien ne relève pas du domaine d'application du présent document, mais elle peut être consultée dans d'autres documents tels que l'ISO/IEC 11770 [6], la NIST SP 800-130 [11] ou l'IEC 62443-4-2 [74].

### 5.5 Principes de conception de la gestion de clés pour les opérations des systèmes de puissance

Les clés cryptographiques sont utilisées pour sécuriser les communications entre différentes entités de système telles que les utilisateurs, les systèmes, les applications logicielles et les nœuds de communication, et le nombre potentiellement élevé d'équipements et de dispositifs souvent installés sur des sites éloignés et non dignes de confiance, et de plus en plus détenus/exploités par différentes entreprises.

Il convient de gérer ces clés cryptographiques de manière à pouvoir les fournir efficacement et en toute sécurité aux entités qui exigent des échanges de données sécurisés. Il est nécessaire que cette gestion de clés tienne compte de nombreux problèmes allant des capacités des entités jusqu'à la protection contre les attaques des propres processus de gestion de clés, en passant par les différents types d'emplacements de ces entités, le délai de fourniture et de révocation des clés, les changements de propriété des entités et les différentes juridictions réglementaires possibles.

Par exemple, la puissance de calcul et la capacité de mémoire de bon nombre de petits dispositifs sont limitées, la bande passante disponible des réseaux de communication pouvant l'être également. Par conséquent, certaines techniques de gestion de clés utilisées dans les environnements traditionnels des systèmes technologiques de l'information d'entreprise avec de puissants systèmes et des communications à bande passante large ne sont pas adaptées à l'automatisation des systèmes de puissance et aux environnements de communication.

Par ailleurs, dans les groupes d'organisations qui prévoient d'échanger des données opérationnelles, certaines organisations peuvent ne pas avoir l'expertise ou le personnel nécessaire pour gérer les clés cryptographiques sans l'aide d'un tiers. Il convient donc d'intégrer ces réalités dans la conception des processus de gestion de clés.

Pour faire face à ces contraintes, le présent document spécifie différentes techniques de gestion de clés qui peuvent être utilisées pour des exigences et des contraintes différentes. Il précise de manière spécifique la manière de gérer les clés pour chacune des fonctions cryptographiques spécifiées dans les autres parties de l'IEC 62351. Les recommandations relatives aux principes de conception de la gestion de clés sont issues des sources suivantes:

- référence [6], ISO/IEC 11770-1:2010 Technologies de l'information – Techniques de sécurité – Gestion de clés – Partie 1: Cadre général;
- référence [8], ISO/IEC 11770-3:2008 Technologies de l'information – Techniques de sécurité – Gestion de clés – Partie 3: Mécanismes utilisant des techniques asymétriques;
- référence [13], NIST 800-57, Partie 1.

## 5.6 Établissement de clés symétriques

### 5.6.1 Vue d'ensemble

Les clés cryptographiques symétriques sont exigées pour deux raisons:

a) elles sont utilisées pour le chiffrement et le déchiffrement des données pendant le transfert; et

b) elles sont utilisées pour la génération et la vérification d'ICV.

Pour des raisons de sécurité, les clés d'une communication par paires sont censées être différentes entre deux paires d'entités de communication. Les clés symétriques entre deux entités peuvent être établies de différentes manières:

a) par des moyens définis dans d'autres parties de l'IEC 62351, plus précisément dans:
   - l'IEC 62351-3 pour TLS en profilant les options TLS. Ici, pendant le protocole de transfert TLS entre deux entités, les techniques basées sur des algorithmes à clé publique sont utilisées pour établir une paire de clés;
   - l'IEC 62351-4 pour le profil de sécurité de bout en bout de la couche application. Comme pour TLS, une paire de clés est établie entre deux entités de communication sur la base des algorithmes à clé publique;
   - l'IEC 62351-5 pour le mécanisme de sécurité de la couche application pour l'IEC 60870-5-101/104 et l'IEEE 1815 (DNP3) s'appuie également sur l'application d'algorithmes à clé publique;

b) en utilisant un accord de clé pour l'établissement des paires de clés, comme décrit dans la méthode d'accord de clé Diffie-Hellman (5.6.2);

c) en utilisant la méthode de la fonction de dérivation de la clé (KDF) (5.6.3), par exemple basée sur une clé symétrique existante;

d) en utilisant la distribution de clés, par exemple en utilisant le chiffrement à clé publique utilisé dans certaines suites chiffrées TLS.

Les clés symétriques établies entre deux entités selon a) à d) ne sont souvent pas directement appliquées pour protéger la communication, mais sont plutôt utilisées pour dériver des clés spécifiques au service de sécurité. Dans l'exemple de l'IEC 62351-4, des clés symétriques distinctes sont disponibles pour la protection de l'intégrité et également pour la protection de la confidentialité. De plus, ces clés dépendent de la direction, ce qui entraîne l'existence de plusieurs clés entre deux entités de communication. Ces clés ont également un cycle de vie qui exige que la clé symétrique initiale soit établie de manière sécurisée et qu'elle soit gérée pendant toute la durée de la connexion de communication. Comme indiqué dans la liste de points ci-dessus, l'établissement de clés symétriques est généralement soutenu par d'autres méthodes afin de limiter les coûts de gestion des clés symétriques pures entre les entités de communication.

Une méthode pouvant être envisagée pour les entités connectées en réseau, qui reposent sur la communication par multidiffusion, est la gestion de clés par groupe. En particulier, les clés de groupe peuvent être utilisées dans les systèmes d'automatisation des systèmes de puissance. Sont utilisés à cet effet des protocoles de gestion de clés par groupe, qui comprennent un composant dédié à la gestion des clés et des politiques associées aux membres authentifiés d'un groupe. La gestion des clés de groupe est décrite en 5.6.4.

### 5.6.2 Méthode d'accord de clé Diffie-Hellman

La méthode d'accord de clé Diffie-Hellman permet à deux parties qui ont à échanger des clés publiques dites clés Diffie-Hellman, de parvenir conjointement avec un secret partagé d'une manière qui ne permet pas à un espion d'en savoir plus sur ce secret partagé.

Des détails sur l'accord de clé Diffie-Hellman sont donnés à l'Article B.8 pour la méthode Diffie-Hellman classique et celle basée sur les courbes elliptiques.

### 5.6.3 Méthode de la fonction de dérivation de la clé (KDF)

Une fonction de dérivation de la clé (KDF) est un algorithme cryptographique qui déduit une ou plusieurs clés symétriques d'une valeur secrète telle qu'une autre clé symétrique, un mot de passe ou une phrase de passe, en utilisant une fonction pseudo-aléatoire.

La fonction KDF est détaillée à l'Article B.9.

### 5.6.4 Gestion des clés de groupe

#### 5.6.4.1 Finalité des clés de groupe

Pour les interactions multidiffusion entre entités formant un groupe qui a des exigences de performance strictes, l'application d'une clé de groupe est plus efficace que d'avoir des relations d'égal à égal séparées dans ce groupe. De plus, dans de telles circonstances, la gestion de clés est susceptible d'être effectuée en dehors du contexte du protocole réel appliquant les clés. En règle générale, la gestion de clés de groupe utilise une combinaison de cryptographie asymétrique et symétrique. La partie asymétrique est utilisée pour identifier et authentifier les entités, tandis que la partie symétrique protège la clé réelle et le transport de la politique.

Pour configurer une clé basée sur le groupe, un système ou un dispositif est généralement désigné contrôleur de groupe, qui authentifie à son tour d'autres entités par l'intermédiaire de leurs certificats ou de leurs clés prépartagées. Une fois les entités authentifiées, le contrôleur de groupe distribue à chacune d'elles la clé de groupe réelle. Ainsi, le contrôleur de groupe rassemble les fonctionnalités d'un centre de distribution de clés (KDC) (voir Figure 3).

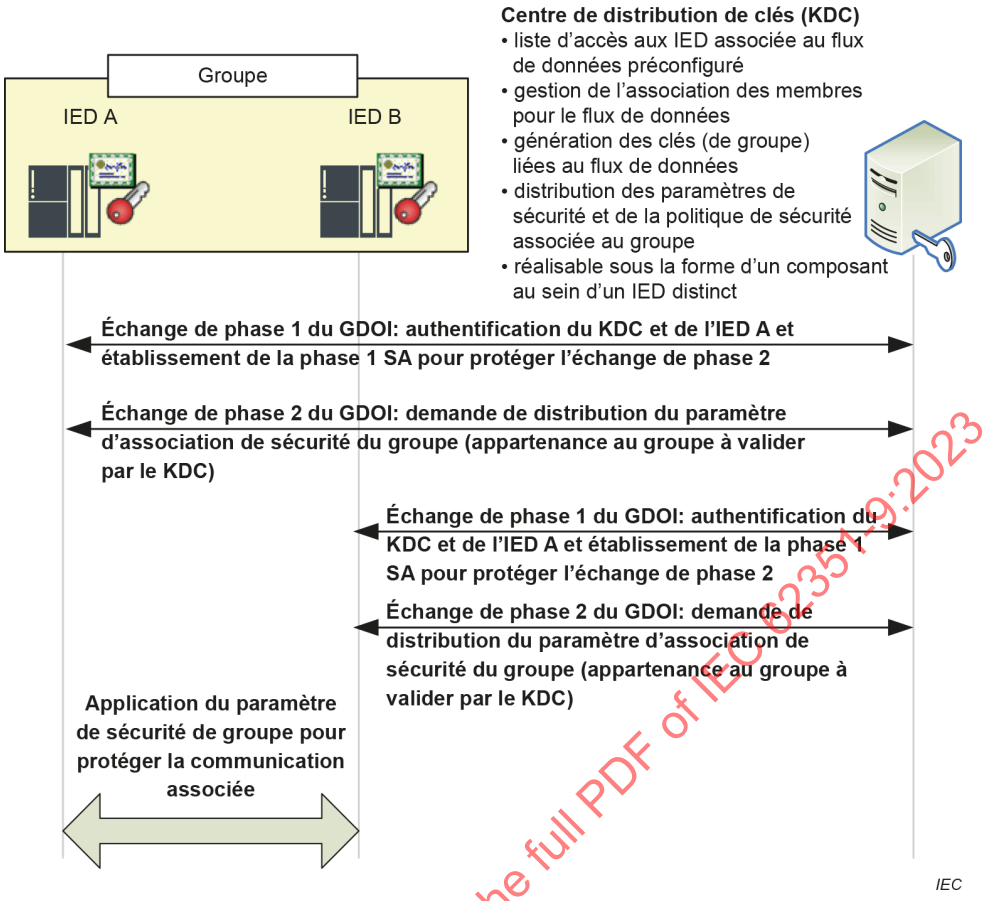



**Figure 3 – Vue d'ensemble de la gestion des clés de groupe dans l'exemple de GDOI**

La Figure 3 représente l'abonnement d'un dispositif électronique intelligent (IED) pour les paramètres de sécurité d'un flux de données. Le flux de données est associé à un groupe de communication. Il est à noter que le contrôleur de groupe peut être une entité distincte ou peut être déployé à l'intérieur d'un IED. Il faut également noter que la version représentée utilise une authentification par certificat employant des signatures numériques.

Il existe un certain nombre de protocoles différents pour la distribution des clés de groupe. Cependant, le domaine d'interprétation de groupe (GDOI) a été choisi comme le protocole le plus approprié pour l'automatisation des systèmes de puissance et 5.6.4.2 en fournit une description approfondie.



### 5.6.4.2 Domaine d'interprétation de groupe (GDOI)

#### 5.6.4.2.1 Généralités

Le domaine d'interprétation de groupe (GDOI) défini dans la RFC 6407 prend en charge la distribution de clés de groupe symétriques (c'est-à-dire les clés de chiffrement du trafic, TEK) à toutes les entités préconfigurées ou sinon enregistrées (en général, des dispositifs). Cette méthode exige la présence d'un centre de distribution de clés (KDC) chargé de distribuer les ensembles de clés de session symétriques aux entités enregistrées selon le processus GDOI. Ce processus utilise des communications point à point entre le KDC et chaque membre du groupe (GM) pour distribuer les clés de groupe symétriques et la politique de sécurité associée. La clé de groupe elle-même est ensuite appliquée conformément à la politique de sécurité transmise afin de protéger les communications ultérieures au sein du groupe.

Il est à noter qu'une défaillance du KDC interrompt la communication du groupe. Il est donc impératif d'assurer la redondance du KDC. Toutefois, la méthode permettant d'assurer la redondance du KDC ne relève pas du domaine d'application du présent document.

Le GDOI est spécifié se dérouler en deux phases qui sont décrites plus en détail dans les paragraphes suivants.

#### 5.6.4.2.2 Phase 1 du GDOI: Phase 1 d'échange de clés Internet (IKE)

L'association de sécurité de la phase 1 du GDOI permet une authentification et une autorisation mutuelles. Le résultat est utilisé par les participants au protocole pour exécuter un échange GDOI sécurisé de phase 2. La RFC relative au GDOI incorpore (c'est-à-dire qu'elle utilise, mais ne redéfinit pas) la définition de l'association de sécurité IKEv1 de phase 1 du DOI Internet [RFC 2407], [RFC 2409], comme représenté à la Figure 4. Le transfert des clés symétriques entre le KDC et les entités est lancé après l'authentification mutuelle.

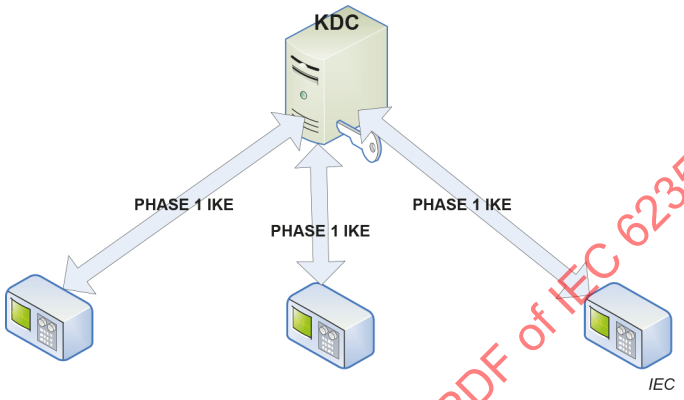



**Figure 4 – Phase 1 IKE du GDOI – Authentification et sécurisation du canal de communication**

NOTE Bien que les RFC 2407, RFC 2408 et RFC 2409 aient été rendues obsolètes par la RFC 4306 (et ensuite par la RFC 5996), elles sont utilisées par le GDOI car les définitions de protocole restent pertinentes pour les protocoles ISAKMP autres qu'IKEv2.

#### 5.6.4.2.3 Phase 2 du GDOI: Distribution de clés symétriques du GDOI

##### 5.6.4.2.3.1 Généralités

Dans le processus GDOI (RFC 6407), deux méthodes sont définies pour transférer les clés du KDC dans les entités. La première est la méthode PULL (les clés sont extraites du KDC par les entités), la seconde étant la méthode PUSH (les clés sont poussées dans les entités par le KDC). L'échange de phase 2 du GDOI est protégé par la SA établie lors de la phase 1 du GDOI.

NOTE La méthode PUSH exige que les membres du groupe soient directement accessibles à partir du KDC. Cette exigence est à satisfaire lorsque la conception de l'architecture du système sous forme de pare-feu ou de dispositifs NAT peut gêner cette communication.

Un échange GROUPKEY-PULL ou GROUPKEY-PUSH peut être utilisé pour distribuer des clés symétriques aux entités. La prise en charge de l'échange GROUPKEY-PULL dans le GDOI est exigée, car il s'agit de la seule méthode qui fonctionne conjointement avec l'authentification et l'autorisation de la phase 1. Chacune des deux méthodes présente des avantages. GROUPKEY-PULL assure le contrôle de l'acheminement du trafic, alors que GROUPKEY-PUSH permet de révoquer ou de rejeter en temps opportun une entité pour un meilleur rendement.

Le GDOI a été étendu pour prendre en charge les services de sécurité IEC 62351 dans la RFC 8052. Ces services sont décrits en 5.6.4.2.4.

##### 5.6.4.2.3.2 Échange de protocole d'enregistrement GROUPKEY-PULL

La méthode PULL du GDOI est représentée à la Figure 5.

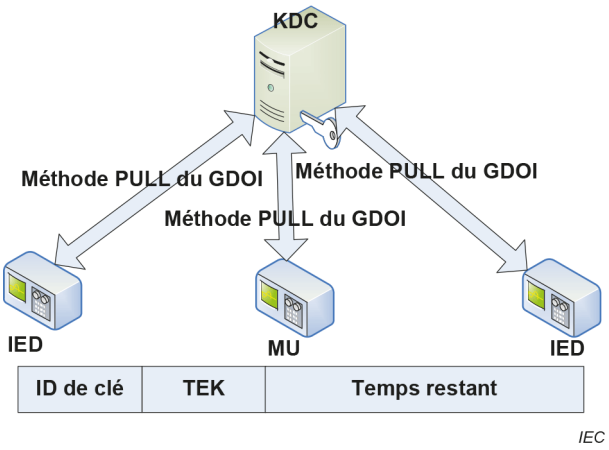



**Figure 5 – Phase 2 de la méthode PULL du GDOI**

Il existe deux phases de communication dans la méthode GROUPKEY-PULL du GDOI:

- phase 1 du GDOI: établissement de la connexion et authentification des deux entités participantes: un membre du groupe et le KDC, comme décrit en 5.6.4.2.2;
- phase 2 du GDOI: détermination des politiques utilisées pour interroger le groupe et télécharger les clés: cette opération est réalisée par un membre du groupe (GM) qui adresse une demande de charge utile Identification du GDOI au KDC. Le KDC répond à cette demande avec les politiques (par exemple algorithmes de chiffrement et de signature) prises en charge. Les politiques sont renvoyées à l'aide de la SA du GDOI, qui renvoie une charge utile SA TEK. Si le GM ne prend pas en charge les politiques, il convient d'annuler les communications. Si le GM peut les prendre en charge, il délivre un accusé de réception auquel le KDC répond par la charge utile Key Download (KD).

#### 5.6.4.2.3.3 Échange de protocole de renouvellement de clé GROUPKEY-PUSH

La politique du KDC peut prévoir d'utiliser la méthode PUSH pour le renouvellement de clé, auquel cas un datagramme initié ("poussé") par le KDC est délivré à tous les membres du groupe à l'aide d'une adresse IP multidiffusion ou est envoyé à un membre particulier du groupe par monodiffusion IP. La méthode GROUPKEY-PUSH est utile pour rejeter les membres du groupe qui peuvent avoir été compromis et dont les certificats peuvent avoir été révoqués. Avec cette méthode, le KDC peut renouveler la clé de tous les membres autorisés du groupe, tout en excluant le membre du groupe qui est rejeté.

En cas d'utilisation de la méthode GROUPKEY-PUSH, la RFC 8263 définit la fourniture d'un message d'accusé de réception provenant des membres du groupe pour informer le KDC de la réception des informations poussées. La demande de fourniture du message d'accusé de réception est indiquée par le KDC dans la politique associée envoyée aux membres du groupe.

#### 5.6.4.2.4 Prise en charge du protocole GDOI pour les services de sécurité IEC 62351

La RFC 8052 relative à la prise en charge du protocole GDOI pour les services de sécurité IEC 62351 a été développée pour favoriser l'utilisation du GDOI dans la famille de normes IEC 61850 relatives à l'automatisation des systèmes électriques. Elle décrit les méthodes à utiliser pour distribuer des transformations de sécurité qui sont utilisées pour la protection des messages pour certains protocoles de sécurité liés à l'IEC 61850: la protection des messages est définie dans l'IEC 62351-6 [IEC-62351-6], l'IEC 61850-8-1 [IEC-61850-8-1] et l'IEC 61850-9-2 [IEC-61850-9-2]. En règle générale, les messages IEC 61850 protégés contiennent la sortie d'un code d'authentification de message (MAC) et peuvent être chiffrés à l'aide d'un chiffrement symétrique tel que la norme de chiffrement avancé (AES).

## 5.7 Confiance basée sur des infrastructures de clé publiques (PKI) et des infrastructures de gestion de privilèges (PMI)

### 5.7.1 Généralités

Le présent article donne un aperçu de l'infrastructure PKI utilisée pour gérer les certificats tout au long du cycle de vie des entités du système de puissance (voir Figure 6). A noter qu'une entité peut être un dispositif, un système mais aussi un utilisateur humain dans un système de puissance.

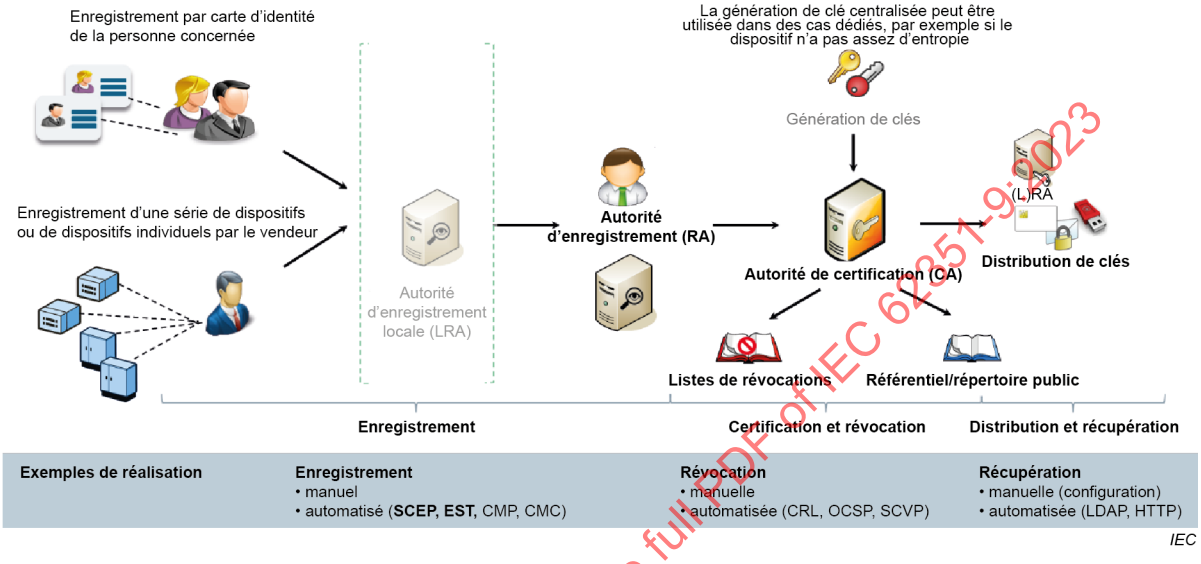



**Figure 6 – Vue d'ensemble de l'infrastructure PKI et exemples de réalisation**

Les paragraphes 5.7.2 à 5.7.6 décrivent la fonctionnalité générale de l'autorité d'enregistrement et de l'autorité de certification, ainsi que les certificats utilisés comme des objets gérés.

### 5.7.2 Autorités d'enregistrement (RA)

Dans les systèmes de PKI, il est exigé que les entités prouvent en premier lieu leur identité par l'intermédiaire d'une autorité d'enregistrement (RA). Pour les êtres humains, les RA peuvent déterminer une identité individuelle au moyen des certificats de naissance, des passeports ou d'autres documents officiels. Pour les systèmes et les dispositifs, les RA peuvent déterminer l'identité d'une entité au moyen d'un certificat numérique délivré par le fabricant ou à un mot de passe à usage unique (OTP).

La RA peut être soit une entité distincte, soit elle peut faire partie de l'autorité de certification (CA).

### 5.7.3 Autorité de certification (CA)

Une fois son identité établie, un système ou un dispositif a besoin d'être associé à un certificat par une autorité de certification (CA). La CSR contenant la clé publique de la paire de clés générée est soit générée par l'entité elle-même, soit par le fabricant si l'entité n'est pas capable de générer des clés, par exemple en raison d'une entropie manquante. La CA vérifie la CSR et délivre un certificat numérique en fonction des données que contient la CSR. Avec les certificats, les CA établissent une liaison cryptographique de confiance entre la clé publique de l'entité et les données des composants des certificats de clé publique (c'est-à-dire la signature numérique de la CA calculée sur la combinaison de la clé publique et des métadonnées). Ce certificat de clé publique lie donc l'identité de l'entité à sa clé publique, de manière à générer une combinaison clé publique/clé privée de confiance que l'entité peut utiliser pour échanger des informations avec d'autres entités. La confiance dans la clé de l'entité repose donc sur la confiance dans la validité de la clé de la CA (voir 5.7.4).

Les CA peuvent être utilisées par l'organisation elle-même, en permettant d'établir un groupe de communication fermé et contrôlé par l'organisation, ou par un tiers (par exemple fournisseur de services, opérateur de système ou gestionnaire de réseau) publiquement accepté et ayant de ce fait une sphère de confiance plus étendue. Les CA de tiers exigent une méthode sécurisée pour accepter l'identité d'une nouvelle entité. Cette méthode peut aller de la validation de l'identité d'une personne ou d'une entité commerciale, jusqu'à des chaînes de sécurité de certificats de clé publique liant des certificats de clé publique déjà validés à de nouveaux certificats de clé publique.

### 5.7.4 Certificats de clé publique

#### 5.7.4.1 Généralités

Un certificat de clé publique est un document numérique qui crée une liaison cryptographique entre l'identité d'une entité à une clé publique de cette entité. La clé publique a une clé privée correspondante. Comme indiqué en 5.7.3, cette liaison est vérifiée par une signature numérique par la CE émettrice. Outre la clé publique et l'identité du détenteur d'un certificat de clé publique, le certificat de clé publique contient les informations vérifiées concernant la période de validité et l'identité de l'émetteur. Un certificat de clé publique peut inclure des extensions donnant des informations supplémentaires. Une extension est identifiée par un identificateur d'objet attribué par l'organisation qui la définit. Un certificat de clé publique peut être délivré à une CA (il s'agit alors d'un certificat de CA) ou à une entité finale (il s'agit alors d'un certificat de clé publique d'entité finale).

Dans un environnement PKI, les certificats de clé publique peuvent être signés numériquement par la CA de l'organisation à laquelle l'entité appartient. Dans certains cas, plusieurs certificats de clé publique sont créés au fur et à mesure qu'une entité avance dans la chaîne d'approvisionnement entre le fabricant et le distributeur, l'acheteur, l'installateur, etc. Les certificats racines de la CA émettrice deviennent les ancres de confiance, le cas échéant. En règle générale, il convient que les organisations installent uniquement les certificats de CA provenant des CA dans lesquelles elles ont confiance, y compris les certificats de clé publique délivrés par leur propre CA.

Des certificats de clé publique peuvent être accordés à des entités telles que des dispositifs, des utilisateurs humains ou des applications logicielles.

Les certificats de clé publique sont définis par un ensemble de base et ses extensions. Les extensions sont identifiées par des identificateurs d'objet.

#### 5.7.4.2 Certificats de clé publique éphémères

Il peut s'avérer inutile de révoquer les certificats éphémères (certificats de clé publique ou certificats d'attribut), car la probabilité de compromission est relativement faible. Les certificats de clé publique éphémères ne sont probablement pas très utiles dans l'industrie électrique, car ils peuvent impliquer des coûts supplémentaires pour la délivrance et la distribution de nouveaux certificats de clé publique. Ils peuvent toutefois être utilisés dans certaines circonstances particulières.

#### 5.7.4.3 Renouvellement des certificats de clé publique

Lorsque les certificats sont sur le point d'expirer, il est nécessaire que les entités fassent une demande de renouvellement à l'aide d'un processus similaire à l'enregistrement, mais plus simple en ce sens que la mise à jour du certificat peut déjà utiliser les informations disponibles sur ce certificat.

En règle générale, les protocoles d'enregistrement existants permettent le réenregistrement en utilisant le certificat précédemment délivré. Le processus spécifique mis en œuvre dépend de la politique de sécurité de l'opérateur.

#### 5.7.4.4 Processus alternatif pour les clés asymétriques générées hors de l'entité

Les paires de clés asymétriques peuvent être générées en externe si l'entité manque de capacités de génération de nombres aléatoires ou de la puissance de calcul nécessaire. Dans ce cas, le processus de génération peut être délégué à un composant compatible avec l'infrastructure PKI comme un outil PKI. Il est nécessaire de distribuer les clés hors bande dans le cadre d'une procédure de confiance. La clé privée peut être protégée avec une clé de transport, comme dans PEM, PKCS #8 ou en général PKCS #12, qui peut contenir d'autres objets, comme des certificats de CA ou des certificats racines. La Figure 7 représente une option possible de génération centralisée de clés et de certificats. Le matériel de clé peut être distribué en local à l'aide d'un outil PKI.

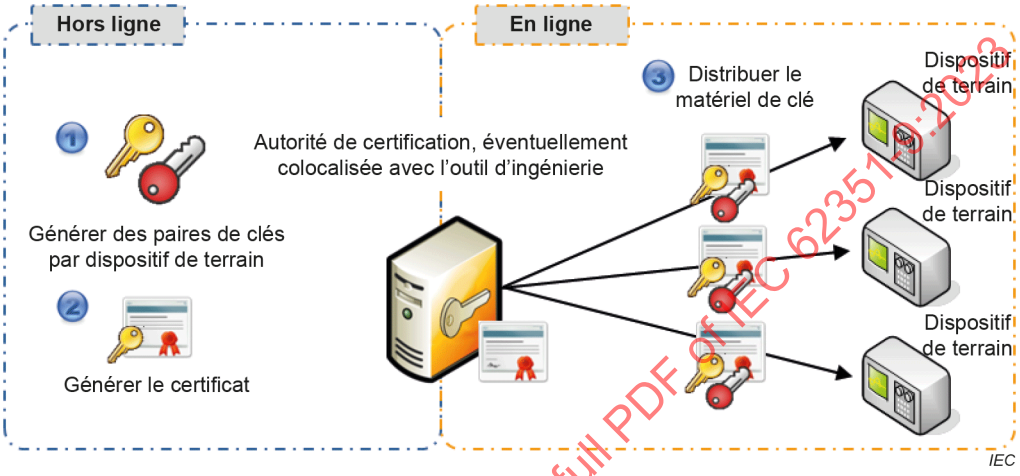



**Figure 7 – Génération centralisée de certificats**

Si une accréditation établie par le fabricant est déjà disponible dans l'entité de terrain et que l'infrastructure lui fait confiance (CA émettrice dans le nouveau domaine de déploiement), elle peut être utilisée pour identifier l'entité de terrain et sécuriser l'opération de distribution de clé. Cela exige que l'accréditation soit déjà connue de la CA.

### 5.7.5 Certificats d'attribut

Les certificats d'attribut constituent un moyen efficace de séparer la gestion de l'identité de la gestion des autorisations associées à une identité. Une séparation complète peut être obtenue en gérant les certificats de clé publique avec une PKI et les certificats d'attribut avec une "infrastructure de gestion de privilèges" (qui utilise la même technologie de CA qu'une PKI).

Comme représenté à la Figure 8, les certificats d'attributs peuvent être utilisés pour étendre les informations dans un certificat de clé publique. Ils permettent, par exemple, l'amélioration provisoire des permissions du détenteur du certificat de clé publique au moyen d'informations d'accès spécifiques basées sur les rôles. Cette approche a été exploitée dans l'IEC TS 62351-8.

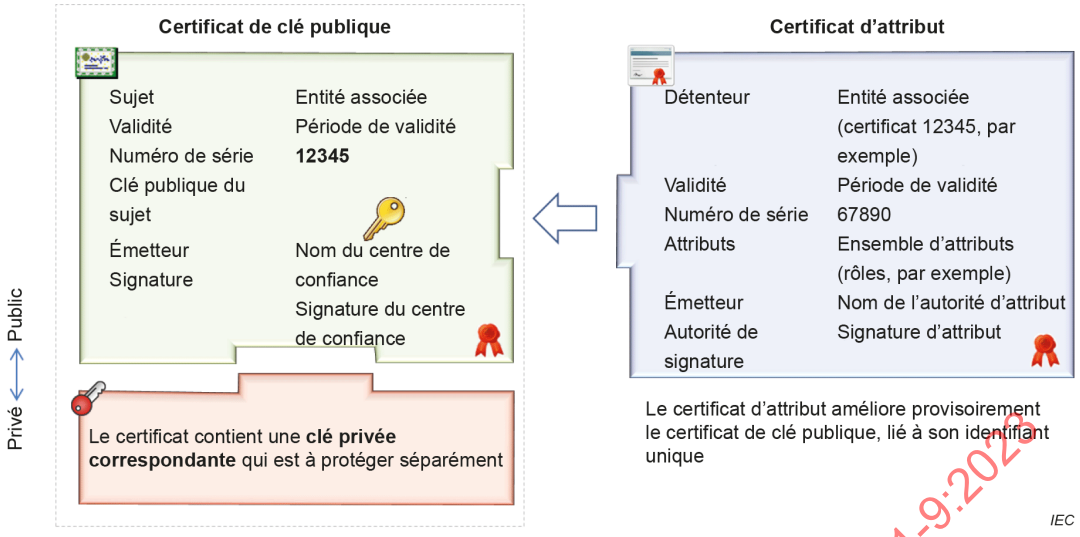



**Figure 8 – Relation entre les certificats de clé publique et les certificats d'attribut**

Des certificats d'attribut peuvent être accordés à des entités telles que des dispositifs, des utilisateurs humains ou des applications logicielles.

Les certificats d'attribut sont définis par un ensemble de base et ses extensions. Les extensions sont identifiées par des identificateurs d'objet.

Les certificats d'attributs peuvent être considérés comme une autre approche pour obtenir une extension éphémère ou temporelle d'un certificat de clé publique (voir également 5.7.4.2). Des certificats d'attribut éphémères peuvent être appliqués pour étendre temporairement les permissions d'une entité telle que définie dans l'IEC 62351-8, par exemple pour les RBAC en général et dans les cas d'urgence. Il convient qu'un certificat d'attribut contienne l'extension d'absence d'informations de révocation définie dans l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019).

### 5.7.6 Extensions de certificat de clé publique et de certificat d'attribut

Les certificats de clé publique et les certificats d'attribut permettent d'inclure des extensions de manière facultative. Chaque extension donne des informations supplémentaires.

Un type d'extension est constitué d'un identificateur d'objet (voir 7.6) qui identifie le type d'extension et spécifie la syntaxe des informations associées. En outre, il comprend une valeur booléenne qui spécifie si une extension particulière est indiquée comme étant critique ou non critique. Si une extension est indiquée comme étant critique, elle ne peut pas être ignorée et le certificat de clé publique ou d'attribut est considéré comme non valide si la partie utilisatrice ne prend pas en charge le type d'extension. Si l'extension est indiquée comme étant non critique, la partie utilisatrice peut ignorer l'extension si le type d'extension n'est pas pris en charge. Pour de plus amples informations, voir 7.3 de l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019).

L'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019) spécifie plusieurs extensions générales applicables. L'IEC 62351-8 spécifie l'extension `IECUserRoles`, qui a été définie pour les systèmes de puissance afin de fournir des moyens de contrôle d'accès basé sur les rôles.

## 5.8 Gestion des certificats de clé publique

### 5.8.1 Processus de gestion de certificats

Le processus de gestion de certificats peut être répété lors de la création d'un certificat, de chaque changement de propriétaire ou de l'utilisation de l'entité, et lorsque le certificat est sur le point d'expirer.

Au-delà des mécanismes et protocoles utilisés pour la gestion de certificats, qui sont détaillés dans les paragraphes suivants, deux propriétés importantes sont à respecter dans le processus de gestion de certificats par rapport au demandeur:

- preuve d'identité, pour s'assurer que la bonne entité est prise en compte dans la gestion de certificats. Cette preuve peut être obtenue en utilisant différents moyens, tels que des informations relatives au dispositif, ainsi qu'un code d'activation (OTP) ou un certificat déjà disponible et une clé privée correspondante. Cette dernière permet spécifiquement d'élever le degré d'automatisation du processus;
- preuve de possession de la clé privée correspondante, pour s'assurer que le fournisseur de la clé publique possède également la clé privée. Cette preuve est généralement donnée en signant numériquement les informations de demande (par exemple une demande de certification PKCS #10) qui peuvent être vérifiées par l'autorité centrale d'enregistrement.

Pour demander un certificat, une preuve d'identité et une preuve de possession sont exigées pour s'assurer qu'un certificat de clé publique est émis de manière autorisée pour l'entité prévue.

### 5.8.2 Création d'un certificat initial

Tout comme une personne a besoin d'un certificat de naissance comme preuve de son identité et que sa présence physique est exigée dans un bureau d'enregistrement pour son identification personnelle, il est nécessaire qu'un dispositif ou un système détienne la preuve de son identité, en général lorsqu'il est encore chez le fabricant et lorsque le nouveau propriétaire le reçoit. Cette preuve est obtenue en l'enregistrant auprès d'une RA et en émettant un certificat par la CA du fabricant. Ce certificat peut être vérifié par rapport à l'ancre de confiance de ce fabricant, qui nécessite donc d'être accessible au public. L'enregistrement peut être manuel pour les entités individuelles ou traité par lots si les entités sont nombreuses. Le processus d'enregistrement agit alors comme la source de confiance de la chaîne d'approvisionnement ou de la provenance de l'entité.

Dans le contexte de l'IEEE 802.1AR, l'accréditation initiale est appelée IDevID (identificateur de dispositif initial). Il s'agit d'un triplet qui comprend le certificat de dispositif initial, la clé privée correspondante et la chaîne de certificats jusqu'à l'ancre de confiance de l'émetteur (ici le fabricant).

### 5.8.3 Intégration d'une entité

L'introduction d'une entité dans un réseau exige l'instauration d'une confiance mutuelle entre l'entité et le domaine opérationnel, ce qui implique généralement que le dispositif possède l'ancre de confiance du domaine opérationnel, ainsi que son propre certificat et la clé privée correspondante pour authentifier d'autres composants du même domaine. Ici, un triplet similaire à l'accréditation de dispositif initiale est formé en ayant une accréditation opérationnelle. L'IEEE 802.1AR désigne cette accréditation opérationnelle LDevID (identificateur de dispositif local), constitué du certificat de dispositif local, de la clé privée correspondante et de la chaîne de certificats jusqu'à un certificat racine de l'émetteur local. Il est à noter qu'un dispositif peut posséder plusieurs LDevID à différentes fins.

L'intégration peut être un processus entièrement manuel en fournissant les informations de livraison du vendeur au domaine en vue d'une vérification ultérieure du côté de l'infrastructure. Une partie de ce processus est la configuration des informations de confiance du domaine dans le dispositif. Cette étape fournit la condition limite nécessaire pour l'enregistrement à l'étape suivante. Cette dernière partie, établissant la confiance de l'entité dans le domaine opérationnel, peut également être automatisée, ce qui donne lieu à une intégration sans contact du point de vue du dispositif. Pour prendre en charge cette automatisation, les objets d'information et les protocoles d'interaction définis par l'IETF dans le contexte d'une mise en réseau autonome peuvent être exploités. Il est à noter que les exemples suivants ne constituent pas une liste exhaustive des approches possibles.

La RFC 8366 [44] spécifie un artifice justificatif qui peut être utilisé pour acheminer des informations (le certificat de domaine du nouveau propriétaire) du fabricant vers un dispositif dans un conteneur protégé (signé). Cette signature peut être validée par le dispositif en exigeant que ce dernier possède le certificat racine du fabricant. L'hypothèse ici sous-jacente est que les dispositifs sont expédiés avec un IDevID.

La RFC 8572 [76] définit un mécanisme permettant de fournir en toute sécurité un dispositif de mise en réseau lorsqu'il démarre dans un état de sortie d'usine par défaut (Provisionnement sûr à zéro touche, SZTP). Il peut être utilisé dans des réseaux publics ou privés et s'appuie sur le justificatif défini dans la RFC 8366 pour faciliter l'établissement de la confiance, ainsi que l'enregistrement des certificats opérationnels pour le domaine. Après finalisation de l'amorçage, le dispositif est en mesure d'établir des connexions sécurisées avec d'autres systèmes.

La RFC 8995 [47] s'appuie sur le processus de justification défini et définit une architecture et un protocole (BRSKI) pour établir la confiance mutuelle en fournissant un nouveau dispositif avec les informations de certificat de domaine local et en impliquant le fabricant émettant un justificatif incluant ce certificat de domaine. Ce justificatif peut être vérifié par le dispositif, car il possède l'ancre de confiance du fabricant. Une fois que le dispositif possède le certificat de domaine, il peut s'enregistrer dans le domaine cible et recevoir un certificat de LDevID (identificateur de dispositif local) pour le nouveau domaine. Le LDevID est également un triplet qui comprend le certificat de dispositif local, la clé privée correspondante et la chaîne de certificats jusqu'à une ancre de confiance de l'émetteur (ici l'opérateur du domaine cible). En utilisant les informations de démarrage du LDevID au niveau du réseau, le dispositif peut recevoir d'autres LDevID spécifiques à l'application pour établir des connexions de communication sécurisées avec d'autres dispositifs.

En outre, la RFC 8520 [46] (description du dispositif du fabricant - MUD) définit une autre approche de prise en charge de l'automatisation lors de l'installation de nouveaux dispositifs, en spécifiant des objets d'information à utiliser par un fabricant pour fournir des informations sur le dispositif lui-même, ainsi que sur le comportement/les attentes en matière de communication du dispositif. Ce dernier peut être utilisé pour adapter l'infrastructure aux besoins de communication conformément à la politique de sécurité locale. Ces informations peuvent être utilisées, par exemple, pour affiner les listes de contrôle d'accès dans l'infrastructure ou les ports de communication par rapport aux services externes. En conséquence, la MUD permet la segmentation du réseau en utilisant des définitions de contrôle d'accès fines reposant sur la finalité et les capacités du dispositif.



### 5.8.4 Enregistrement d'une entité

L'enregistrement est le processus par lequel l'entité reçoit un certificat signé de la part de la CA dans l'environnement opérationnel. Sur la base des accréditations de sécurité existantes de l'entité, différents scénarios d'enregistrement peuvent être distingués:

- mot de passe à usage unique sans certificat;
- avec certificat délivré par une CA (il peut s'agir d'un IDevID émis par une CA du fabricant);
- avec certificat autosigné (autorisé) connu;
- avec certificat expiré ou révoqué (il peut s'agir d'un IDevID émis par une CA du fabricant ou d'un LDevID provenant d'une CA locale);
- avec un certificat de CA expiré ou révoqué.

Il existe différents protocoles d'enregistrement et de gestion de certificats offrant différents ensembles de fonctionnalités. Quatre d'entre eux sont brièvement décrits en 5.8.6. Parmi ceux-ci, deux sont axés sur les applications utilisées dans le domaine des systèmes de puissance: SCEP (voir 5.8.6.1) et EST (voir 5.8.6.2). Les deux protocoles traitent de la fonctionnalité principale considérée comme nécessaire pour gérer les certificats des dispositifs d'un système de puissance, tout en conservant la simplicité, et les deux fournissent un sous-ensemble de la fonctionnalité fournie par les protocoles plutôt riches en caractéristiques que sont CMP (voir 5.8.6.3) et CMC (voir 5.8.6.4). Ces deux derniers protocoles d'enregistrement fournissent un meilleur support dans les scénarios hors ligne et en soutien du traitement de la révocation.

Un exemple d'enregistrement d'une entité à l'aide du protocole SCEP est représenté à la Figure 9. Pour le SCEP, le code d'activation est un OTP (mot de passe à usage unique) comme représenté à la Figure 9 ou un certificat existant. Pour l'EST, le code d'activation peut être un nom d'utilisateur et un mot de passe ou une paire de clés existantes [par exemple un certificat délivré (imprimé) par le fabricant avec une clé privée correspondante et l'émission d'informations de CA, IDevID].

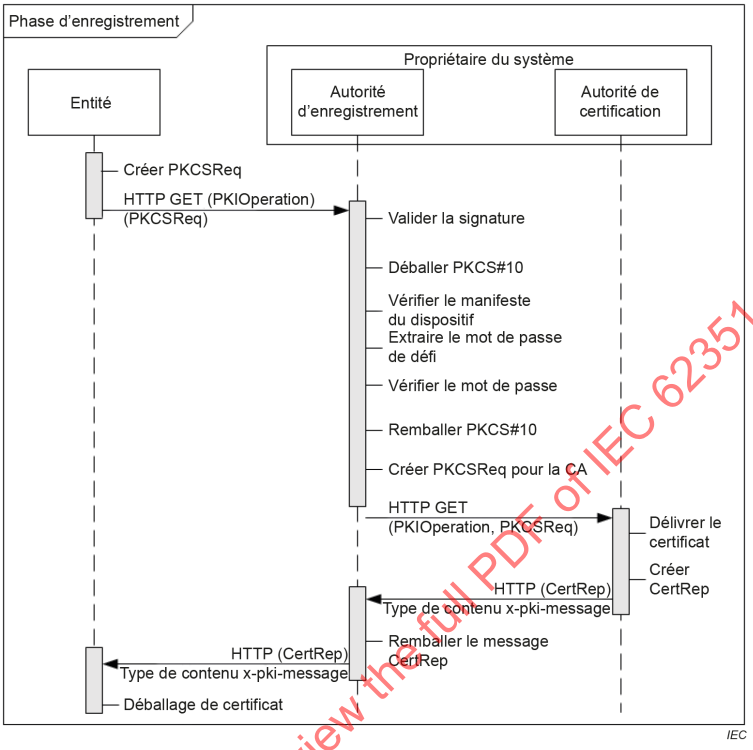



**Figure 9 – Exemple d'enregistrement de l'entité SCEP et processus de demande de signature de certificat (CSR)**

Un exemple d'enregistrement d'une entité à l'aide du protocole EST est représenté à la Figure 10.

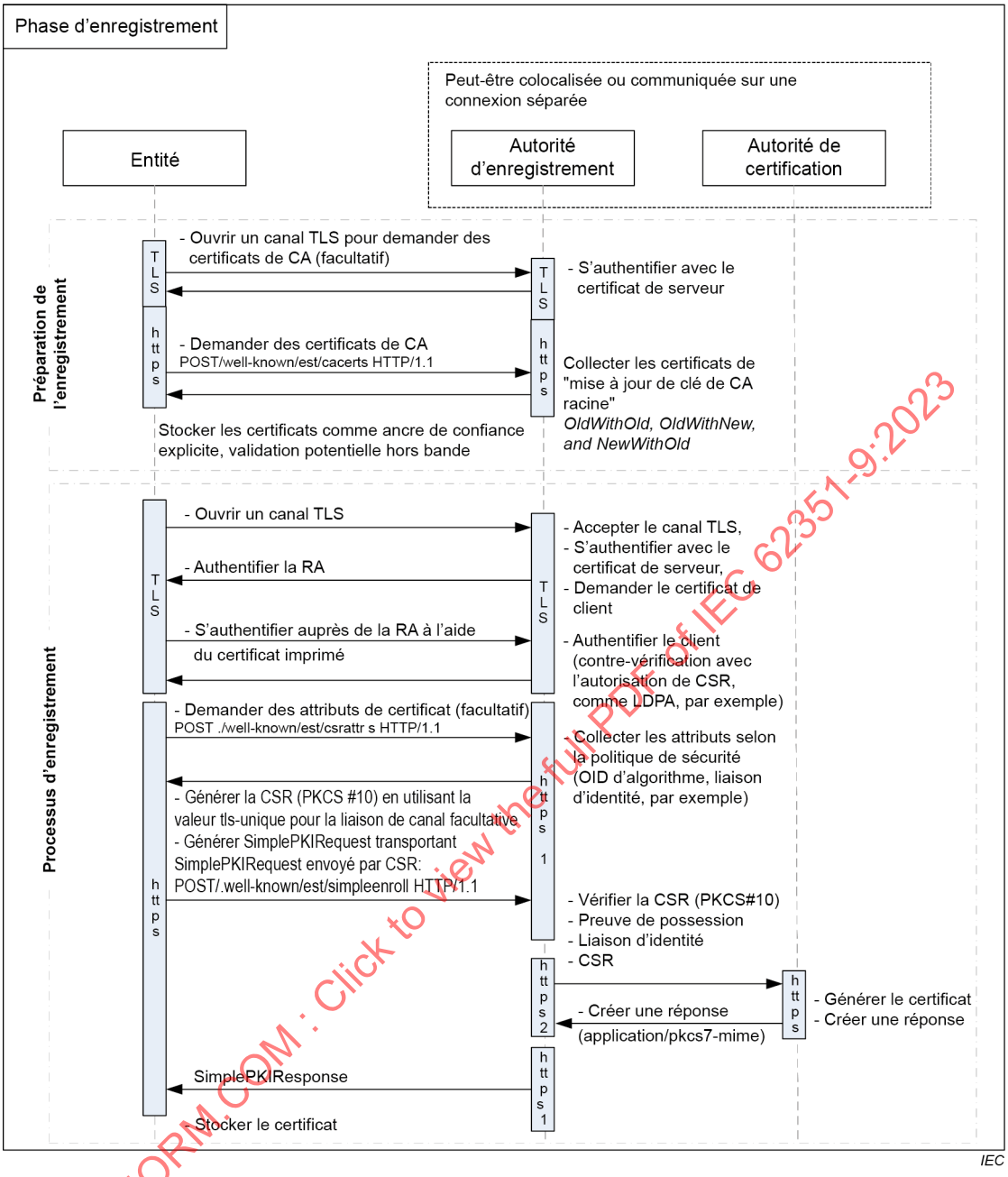





**Figure 10 – Exemple d'enregistrement de l'entité EST et processus de demande de signature de certificat (CSR)**

Notes concernant Figure 9 et Figure 10:

- les dispositifs doivent être configurés avant l'enregistrement. Outre leur configuration normale, comme leur ou leurs adresses IP, les données d'enregistrement de PKI doivent également leur être indiquées, ainsi que l'adresse IP de la RA/CA, les certificats d'ancre de confiance (certificat de CA/racine), etc. Comme indiqué en 5.8.3, ce processus peut être manuel ou automatique. Le provisionnement automatique peut être pris en charge par des exemples de protocoles indiqués en 5.8.3;
- l'entité vérifie l'authenticité des homologues de communication d'après le certificat racine de la CA. Ce certificat racine est transmis au dispositif pendant la phase de configuration du dispositif (voir 7.3.6) ou dans le cadre d'une intégration automatique. La CA signe les certificats délivrés à l'aide de la clé privée de la CA. Les entités s'assurent de l'authenticité du certificat reçu en vérifiant la signature présentée dans les certificats à l'aide de la clé publique de la CA.

### 5.8.5 Traitement d'une demande de signature de certificat (CSR)

#### 5.8.5.1 Processus d'enregistrement

Le processus d'enregistrement implique la signature d'un certificat par une CA afin de vérifier l'identité d'une entité, de vérifier que l'entité possède la clé privée associée à la clé publique et de délivrer un certificat à cet effet. Cette signature de certificat exige la délivrance par l'entité (le dispositif) d'une demande de certification à la RA/CA. Le traitement de la CSR, en utilisant un matériel de clé généré par le dispositif, comprend les étapes suivantes, représentées à la Figure 11.

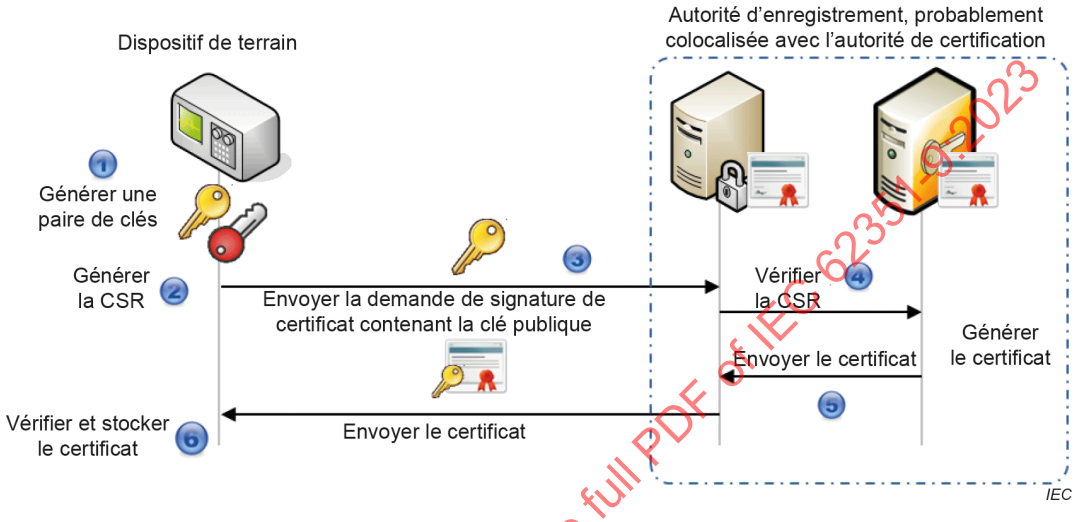



**Figure 11 – Traitement d'une CSR**

1) L'entité génère une paire de clés publique et privée.
2) L'entité génère un *CertificationRequestInfo* en utilisant ici le format de spécification PKCS #10 [18]. Cette demande contient un "nom distinctif de sujet", la clé publique issue de la paire de clés publique/privée qui vient d'être générée [voir ISO/IEC 9594-8:2020 | Rec. UIT-T X.509 (2019)], et éventuellement un ensemble d'attributs. Le *CertificationRequestInfo* est signé avec la clé privée de l'entité. Le *CertificationRequestInfo*, un identificateur d'algorithme de signature, et la signature de l'entité composent une structure *CertificationRequest*. Voir [18].
3) L'entité envoie le message *CertificationRequest* à la RA/CA pour l'autorisation et la certification dans le cadre d'un protocole d'enregistrement (voir 5.8.6).
4) La RA valide la demande en vérifiant la signature sur la demande entrante.
5) Si la demande est valide, la RA authentifie l'entité demandeuse et, si elle est valide et autorisée, contacte la CA, qui crée un certificat de clé publique à partir du nom distinctif et de la clé publique, du nom de l'émetteur et du numéro de série choisi par l'autorité de certification, de la période de validité, des extensions facultatives et de l'algorithme de signature numérique donnant les informations pour la signature des données figurant dans la liste. La CA envoie ensuite le certificat à l'entité par l'intermédiaire de la RA.
6) L'entité vérifie la signature sur le certificat reçu de la CA pour s'assurer qu'il a été réellement délivré et signé par la CA de confiance. Il est à noter qu'il est présumé ici que les certificats de la CA de confiance sont distribués et installés dans l'entité pendant la phase de mise en service/intégration/configuration. L'entité vérifie la signature sur le certificat reçu de la CA opérationnelle à l'aide de ces certificats de CA de confiance. Si la signature de la CA est valide, l'entité stocke le certificat au format préférentiel de la mise en œuvre et l'extension de fichier appropriée lorsque le stockage est basé sur des fichiers (.der, .pem, .cer, .crt, etc.).

Le processus CSR peut exiger une intervention humaine qui va le déclencher, l'activer ou l'approuver. Ce support manuel peut s'avérer nécessaire en même temps que la demande CSR ou peut avoir lieu préalablement, en fonction de la procédure d'authentification de l'entité.

Si une entité est authentifiée avec une accréditation délivrée par le vendeur (IDevID) ou avec un mot de passe à usage unique, ces informations peuvent être fournies à l'avance à la RA.

Les deux paragraphes suivants décrivent les formats de demandes de certification PKCS #10 et CRMF.

#### 5.8.5.2 Demande de certification PKCS #10

La syntaxe de demande de certification, telle que définie par l'IETF RFC 2986 [18], est utilisée par différents protocoles de gestion de certificats de clé publique pour transmettre une demande de certification d'un client (entité finale) à une CA ou à une entité qui s'interface avec la CA, par exemple un serveur RA ou EST (voir RFC 7030).

L'IETF RFC 2986 ne spécifie aucun format de réponse ni aucun processus d'exécution d'une demande de certification, mais elle renvoie à une spécification de référence.

La demande de certification est spécifiée sous la forme PKCS #10 dans la RFC 2986 [18]. Une demande de certification est envoyée par une entité à l'autorité d'enregistrement (RA), qui peut être colocalisée avec la CA ou être physiquement séparée.

La Figure 12 représente la structure d'une demande de certification. Elle comporte les composants suivants:

a) le composant `Version` indique la version de la demande de certification. La version spécifiée par l'IETF RFC 2986 est la version 1 et est indiquée par une valeur '0';

b) le composant Subject spécifie le nom à utiliser dans le composant Subject du certificat de clé publique demandé;

c) le composant subjectPublicKeyInfo spécifie les informations à inclure dans le composant `subjectPublicKeyInfo` du certificat de clé publique demandé;

d) le composant Attributes contient un ensemble d'attributs de répertoire qui fournissent des informations supplémentaires pour la génération des certificats de clé publique demandés.

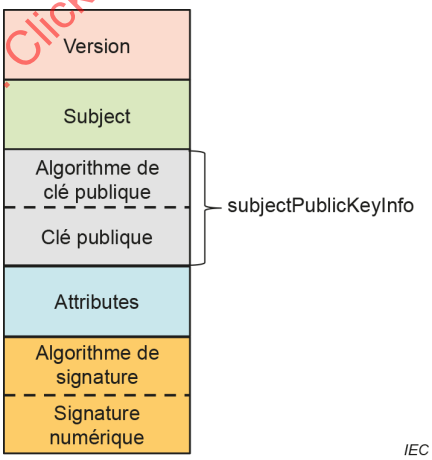



**Figure 12 – Format d'une demande de certification**

La demande de certification est signée numériquement en utilisant la clé privée correspondant à la clé publique fournie dans la demande de certification. Si le destinataire est en mesure de valider la signature à l'aide de la clé publique fournie dans la demande de certification, il prouve que l'émetteur est en possession de la clé privée correspondant à la clé publique présentée.

L'IETF RFC 2985 [19] définit deux types d'attributs dont l'intégration dans une demande de certification est considérée comme pertinente:

a) un attribut du type d'attribut `challengePassword` peut être inclus et, s'il est présent, le même attribut doit être utilisé pour une demande de révocation;

b) un attribut du type d'attribut `extensionRequest` peut être inclus et doit alors contenir une séquence d'extensions de certificat de clé publique à inclure dans les certificats de clé publique.

#### 5.8.5.3 Format de message de demande de certificat (CRMF)

Le format de message de demande de certificat (CRMF) est spécifié dans l'IETF RFC 4211 [20] et représenté à la Figure 13.

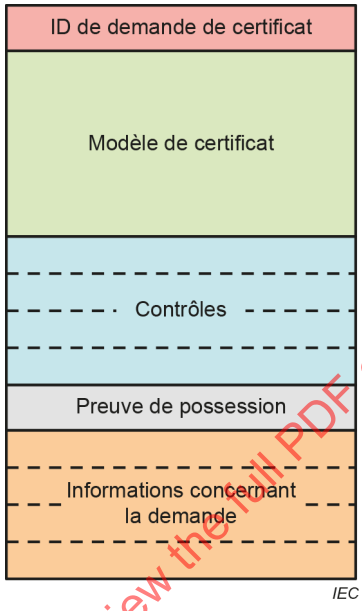



**Figure 13 – Format de message de demande de certificat**

Comme la demande de certification décrite en 5.8.5.2, le CRMF ne constitue pas un protocole, mais il s'agit d'une spécification de type de données à inclure dans les types de protocoles de référence.

Le CRMF est détaillé dans l'IETF RFC 4211. Les différentes parties du CRMF s'organisent comme suit:

a) l'ID de demande de certificat peut être utilisé pour l'appariement des demandes et des réponses;

b) le modèle de certificat permet au demandeur de spécifier ou d'omettre du contenu pour chaque composant du certificat de clé publique demandé. L'omission de spécifications pour certains composants peut être quelque peu perturbante, bien que l'ASN.1 permette leur présence. Un seul composant (informations de clé publique) est présent, mais il est toujours indiqué comme facultatif. Pour de plus amples détails, voir IETF RFC 4211;

c) les contrôles sont des attributs qui ne font pas partie du certificat de clé publique, mais qui fournissent des informations sur le contexte dans lequel délivrer la clé publique. L'IETF RFC 4211 définit un certain nombre de contrôles, tels que des informations d'authentification supplémentaires sur le sujet du certificat de clé publique, des suggestions concernant la publication du certificat de clé publique, des suggestions sur la manière dont il convient d'archiver la clé privée, des informations sur un éventuel ancien certificat de clé publique et une clé de chiffrement à utiliser par la CA pour chiffrer la réponse;

d) la preuve de possession, lorsqu'elle est présente, fournit une spécification détaillée de la méthode à suivre pour prouver la possession de la clé privée, selon que la clé est destinée au chiffrement de signatures numériques (RSA) ou à un accord sur une clé;

e) les informations concernant la demande.

La syntaxe et la sémantique du format de message de demande de certificat (CRMF) sont utilisées pour transmettre une demande de certificat à une CA, éventuellement par l'intermédiaire d'une autorité d'enregistrement (RA), pour la production de certificats de clé publique. La demande comprend généralement une clé publique et les informations d'enregistrement associées.

### 5.8.6 Protocoles d'enregistrement

#### 5.8.6.1 Protocole simple d'enregistrement de certificat (SCEP)

Le protocole simple d'enregistrement de certificat (SCEP) a été développé pour simplifier l'enregistrement d'un grand nombre de dispositifs et pour rendre l'émission des certificats numériques évolutive. La fonctionnalité du SCEP se limite à l'enregistrement et au renouvellement des certificats, ainsi qu'à la distribution des certificats de CA et des CRL. Les entités peuvent utiliser le protocole SCEP pour demander leur certificat numérique par voie électronique en utilisant la CMS et PKCS #10 sur HTTP. Le matériel de clé n'est généré que du côté client.

L'authentification côté client de la demande d'enregistrement peut être effectuée en utilisant des certificats autosignés ou des certificats délivrés par une CA. Pour les certificats autosignés, il est nécessaire d'utiliser un secret supplémentaire (mot de passe de défi) pour permettre une autorisation de la demande de certification. Pour les certificats délivrés par une CA, cela peut être directement lié au certificat délivré par la CA, s'il peut être validé du côté RA. Le SCEP lie la preuve d'identité et la preuve de possession à l'objet de demande de certification.

Il est à noter que le SCEP utilise le chiffrement et les signatures numériques pour protéger les messages échangés. Les messages de demande et de réponse échangés sont basés sur les objets de messagerie SCEP (reposant sur CMS). Le chiffrement des messages dépend de l'algorithme cryptographique asymétrique utilisé de la façon suivante:

- pour les certificats RSA (côté entité ou côté CA), les données sont chiffrées à l'aide de la clé publique du destinataire;
- pour les certificats ECDSA (côté entité ou côté CA), les données sont chiffrées sur la base du mot de passe de défi. Il est à noter que cette conception a une influence pendant l'opération, car elle exige également de distribuer le mot de passe de défi avant qu'une mise à jour de certificat puisse avoir lieu.

Le SCEP est défini par l'IETF et est disponible en tant que RFC 8894 informative. Une partie de la révision du document initial portait sur la transition de PKCS #7 à CMS, ce qui permet une couverture plus étendue des algorithmes cryptographiques. Elle peut donc être considérée comme un profil de CMC.

#### 5.8.6.2 Protocole EST

Le protocole EST est défini dans la RFC 7030 et repose sur la manipulation de PKI simple dans la CMC. Il définit certaines des fonctionnalités de la CMC comme étant facultatives, donnant lieu à un protocole moins complexe. Il peut être considéré comme un profil simplifié de CMC. La fonctionnalité prise en charge par l'EST comprend la distribution de certificats de CA et de CRL, l'extraction d'attributs de demande de certification pour demander des informations supplémentaires ou des conditions limites avant de générer une CSR, l'enregistrement et le renouvellement de certificat. Il permet la génération de clés côté client ou côté serveur. Dans l'EST, seule l'interaction demande/réponse PKI simple est obligatoire, alors que la prise en charge de l'ensemble de la procédure PKI est facultative. L'EST utilise TLS en tant que canal sécurisé et assure l'authentification du canal TLS pour l'identification et l'autorisation du demandeur en associant la CSR à la session TLS actuelle. Outre la preuve de l'identité, la partie CMC fournit une preuve de possession de la clé privée correspondant à la clé publique dans la CSR.

En plus des fonctions d'enregistrement et de gestion de certificats, l'EST permet de gérer les ancres de confiance sur les dispositifs à partir de la CA émettrice, sur la base d'une ancre de confiance déjà installée.

L'EST peut se terminer au niveau de la RA ou directement au niveau de la CA. La fin de l'EST au niveau de la RA ne coupe pas la communication entre la RA et la CA. La fin de l'EST au niveau de la RA exige que la RA comporte une extension spécifique dans le certificat utilisé pour s'authentifier auprès de la CA afin de s'assurer que l'autorisation de la demande a été effectuée par l'entité appropriée. Il est également possible d'utiliser d'autres protocoles d'enregistrement entre la RA et la CA.

NOTE Il convient que les implémenteurs prennent connaissance de l'IETF RFC 8951 qui clarifie les codages de transfert et ASN.1.

#### 5.8.6.3 Protocole de gestion de certificats (CMP) PKI Internet X.509

Le protocole de gestion de certificats (CMP) est un protocole Internet permettant d'obtenir des certificats de clé publique dans un environnement PKI. Il définit un protocole d'interaction entre un client et les composants de la PKI. En plus de l'extraction de CRL, du traitement des demandes de certification, de l'option d'identification/autorisation du demandeur et de la preuve de possession de la clé privée associée, le CMP offre également des fonctions supplémentaires telles que la certification croisée et la révocation de certificat. Le CMP prend en charge la génération du matériel de clé côté client et côté serveur. Il lie la preuve d'identité et la preuve de possession à l'objet de demande de certification, ce qui le rend indépendant du transport. Il utilise le format de message CRMF pour les messages de PKI. Le transport des demandes PKCS #10 est également pris en charge.

Le protocole CMP est défini dans la RFC 4210 [30]. Le transport de CMP sur HTTP est défini dans un document distinct, la RFC 6712 [42].

Étant donné que le CMP est très riche en caractéristiques, un profil léger est en cours de normalisation [48] afin de traiter les cas d'utilisation industrielle en limitant la fonctionnalité aux échanges nécessaires.

#### 5.8.6.4 Gestion de certificats sur CMS (CMC)

La gestion de certificats par syntaxe de message cryptographique (CMC) s'appuie sur la syntaxe de message cryptographique (CMS, définie dans la RFC 5652), PKCS #10 (RFC 2986) et CRMF pour prendre en charge la gestion des certificats. De même que le CMP, la CMC prend en charge l'extraction de CRL, le traitement des demandes de certification, l'option d'identification/autorisation du demandeur, ainsi que la preuve de possession de la clé privée associée. La CMC fournit également des fonctionnalités supplémentaires telles que la certification croisée et la révocation de certificats. Cependant, bien qu'elle définisse un protocole de transfert demande/réponse PKI simple et complet, elle exige que les deux soient mis en œuvre. Le CMC prend en charge la génération du matériel de clé côté client et côté serveur.

Elle lie la preuve d'identité et la preuve de possession à l'objet de demande de certification lors de l'utilisation d'une demande PKI complète. Dans le cas de la demande PKI simple, une preuve d'identité doit être fournie par d'autres moyens.

La CMC est définie dans la RFC 5272. Le transport de la CMC elle-même est défini dans un document distinct, la RFC 5273 [35].

### 5.8.7 Protocole de gestion des ancres de confiance (TAMP)

Les protocoles d'enregistrement décrits dans les paragraphes précédents concernent spécifiquement le traitement des certificats d'entité finale, mais ils fournissent également un support partiel pour la gestion des certificats racines sous-jacents sous forme d'ancres de confiance. Le protocole de gestion des ancres de confiance (TAMP) s'appuie sur la CMS.

TAMP est spécifié dans la RFC 5934 et assure une gestion indépendante du transport des ancres de confiance (TA) dans un magasin d'ancres de confiance de dispositif.

## 5.9 Révocation de certificats de clé publique

### 5.9.1 Listes de révocation de certificats (CRL)

Une CRL est une liste de numéros de série de certificats qui ont été révoqués, accompagnés d'un horodatage indiquant à quel moment ils l'ont été, ainsi que d'une signature numérique provenant de la CA émettrice. La raison de la révocation peut également être indiquée, car elle est définie comme une extension d'une CRL. Il convient de considérer que les certificats révoqués ne sont plus valides et que les entités ne s'y fient plus.

Il convient de mettre à jour les CRL à chaque fois qu'un certificat est révoqué. Il convient de les tenir à disposition de toutes les entités concernées en temps voulu. Une précision adéquate de la synchronisation temporelle entre les entités et le système qui crée les CRL est nécessaire pour assurer l'exactitude des informations temporelles dans les CRL et permettre aux entités de disposer d'informations exactes sur les certificats révoqués. La prise en charge de la synchronisation et de l'exactitude d'horloge est définie dans différents protocoles, tels que:

- le protocole NTPv4 (IETF RFC 5905);
- le protocole PTP (PTP, IEEE 1588 v2.1).

Pour ces deux protocoles, des améliorations de sécurité ont été définies (voir aussi 7.7).

Un exemple de CRL est représenté à la Figure 14.

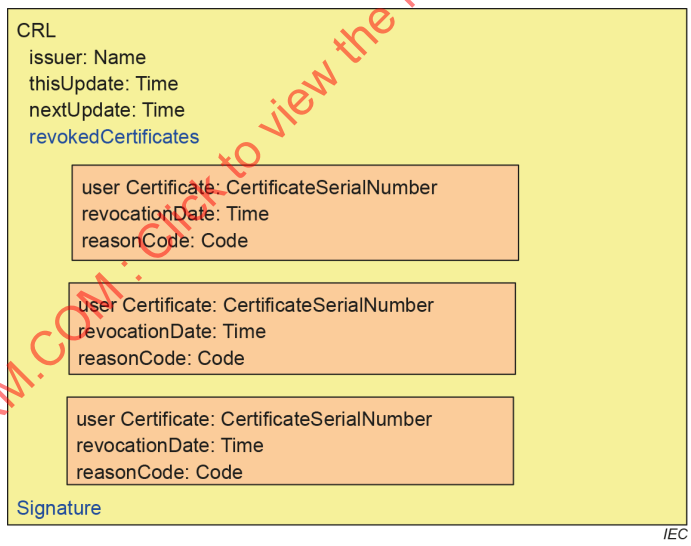





**Figure 14 – Liste de révocation de certificats**

Les CRL cumulant les certificats révoqués au fil du temps, elles peuvent s'agrandir et devenir très volumineuses. Leur taille importante peut alors être problématique pour les systèmes intégrés dont la capacité de mémoire peut s'avérer insuffisante pour les stocker. Les CRL peuvent donc être segmentées afin de réduire le plus possible leur taille pour une entité particulière. D'autres méthodes de contrôle de la révocation peuvent être encore plus efficaces, telles que décrites en 5.9.2.

Il convient de révoquer les certificats pour les raisons suivantes et d'utiliser les codes justificatifs définis au paragraphe 9.5.3.1 de l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019):

- la clé privée est soupçonnée d'être compromise;

- la clé privée de la CA associée à un certificat de CA est soupçonnée d'être compromise;
- l'affiliation de l'entité a changé;
- le certificat de clé publique a été remplacé;
- l'entité a cessé son activité;
- le certificat a été annulé;
- le privilège a été retiré;
- la clé privée d'une autorité d'attribut est soupçonnée d'être compromise.

### 5.9.2 Protocole d'état de certificat en ligne (OCSP)

Le protocole d'état de certificat en ligne (OCSP), tel que défini par la RFC 6960, est une alternative à l'extraction de CRL lors de la vérification de l'état de révocation d'un certificat de clé publique.

Si une partie utilisatrice est concernée par le caractère de validité en cours d'un certificat de clé publique particulier, elle peut envoyer une demande d'état de révocation OCSP au serveur OCSP (ou à la CA) responsable du certificat de l'entité. Cette demande OCSP contient la version du protocole, la demande de service, l'identificateur de certificat de l'entité et les extensions. Pour éviter les attaques par répétition, un "nonce" (valeur à usage unique) est obligatoire pour distinguer cette demande d'état des demandes précédentes. Le répondeur OCSP valide ensuite le certificat et renvoie une réponse "good" (correct), "revoked" (révoqué) ou "unknown" (inconnu), en utilisant sa propre signature numérique pour authentifier la réponse.

Cette séquence OCSP est représentée à la Figure 15.

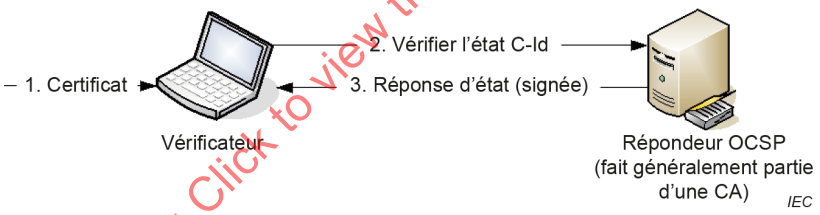



**Figure 15 – Vue d'ensemble du protocole d'état de certificat en ligne (OCSP)**

En règle générale, une connectivité en ligne est exigée entre l'entité et le répondeur OCSP pour assurer la transmission de la demande OCSP et de la réponse OCSP. Toutefois, une connectivité continue peut poser problème pour certaines configurations d'entités sur le terrain. De plus, l'effort de calcul nécessaire au traitement de la réponse OCSP et le délai de communication peuvent ne pas être possibles pour certaines entités. Il convient également que les réponses OCSP présentent une courte période de validité, car les répondeurs OCSP ne délivrent pas de mises à jour d'état non sollicitées, même si un certificat a été révoqué peu de temps après une vérification d'état.

La mise en mémoire cache OCSP est une autre option permettant au récepteur d'une réponse OCSP de mettre en mémoire cache la réponse pendant un certain temps. Cela permet d'éviter les demandes OCSP répétées pendant une certaine période. Il permet également à l'expéditeur d'insérer la réponse OCSP dans son certificat, en évitant l'interaction du répondeur avec le serveur OCSP. L'IEC 62351-3 utilise cette option.

Par conséquent, selon la configuration du système et les capacités de l'entité, une combinaison hybride de CRL et d'OCSP peut être utilisée lorsqu'une entité disposant normalement d'une connectivité agit comme un répondeur OCSP proxy. Cette entité proxy extrait une CRL sur une période spécifique (toutes les heures ou sur 24 heures, par exemple). L'entité proxy (par exemple contrôleur de station) fait alors office de répondeur OCSP pour d'autres entités qui ne

disposent normalement pas de connexion à l'OCSP. Cette approche est représentée à la Figure 16.

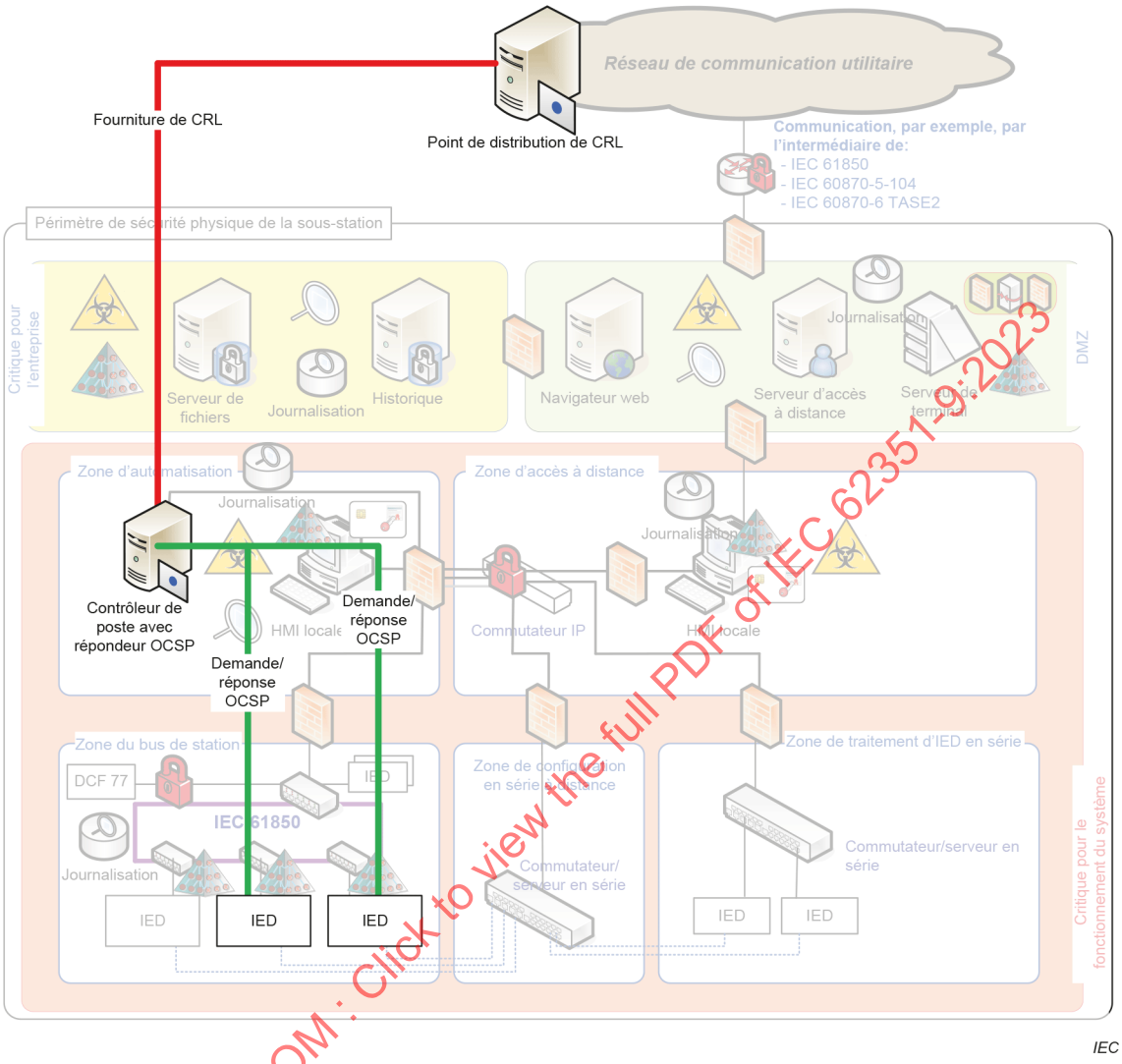



**Figure 16 – Schéma utilisant une combinaison de CRL et de processus OCSP**

Cette approche présente l'avantage évident qu'elle permet l'utilisation du processus OCSP dans des réseaux locaux ayant des difficultés à établir une connectivité permanente aux données CRL centrales. De plus, cela réduit le trafic de données avec le système principal, qui peut être nécessaire pour les connexions à bande passante étroite.

Il convient que les entités aient accès à une horloge en temps réel de confiance afin de vérifier et d'assurer l'exactitude de l'horodatage OCSP. Si, dans un délai configurable (par exemple 15 secondes), aucune réponse n'arrive, il convient que les entités déclenchent une alarme de défaillance OCSP et présument que le certificat n'a pas été révoqué (en particulier si la disponibilité est un élément essentiel).

Pour permettre cette approche, il est nécessaire que l'entité OCSP proxy (par exemple contrôleur de station) soit en possession d'un certificat et de la clé privée correspondante pour agir comme un répondeur OCSP. Le répondeur OCSP serait ainsi configuré comme un nœud de répondeur de confiance avec un certificat signé de la CA.

Les demandes OCSP peuvent également être enchaînées entre des répondeurs OCSP homologues afin de rechercher et d'interroger la CA appropriée quant au certificat d'entité en

cours de vérification, les répondeurs validant chaque réponse en fonction de la CA racine utilisant leurs propres demandes OCSP.

Les demandes OCSP peuvent être formulées de manière proactive ou réactive, comme représenté à la Figure 17. Dans le cas proactif, l'entité demande à l'avance le statut de son certificat au répondeur OCSP et envoie la réponse OCSP accompagnée de son certificat au nœud homologue qui valide l'entité. Cette procédure est appelée "agrafage OCSP" et fait supporter la charge de la communication OCSP par l'entité proactive. Dans le cas réactif, le nœud récepteur formule la demande OCSP de vérification de l'état de révocation du certificat présenté par l'entité homologue. L'option réactive est la plus souvent utilisée, mais l'option proactive est prise en charge par des protocoles tels que TLS 1.2 (RFC 5246), à l'aide d'une extension définie dans la RFC 6066 [41], permettant au serveur TLS 1.2 de transmettre une réponse OCSP dans le cadre d'un message `ServerHello`. Cela concerne le cas du serveur proactif représenté à la Figure 17.

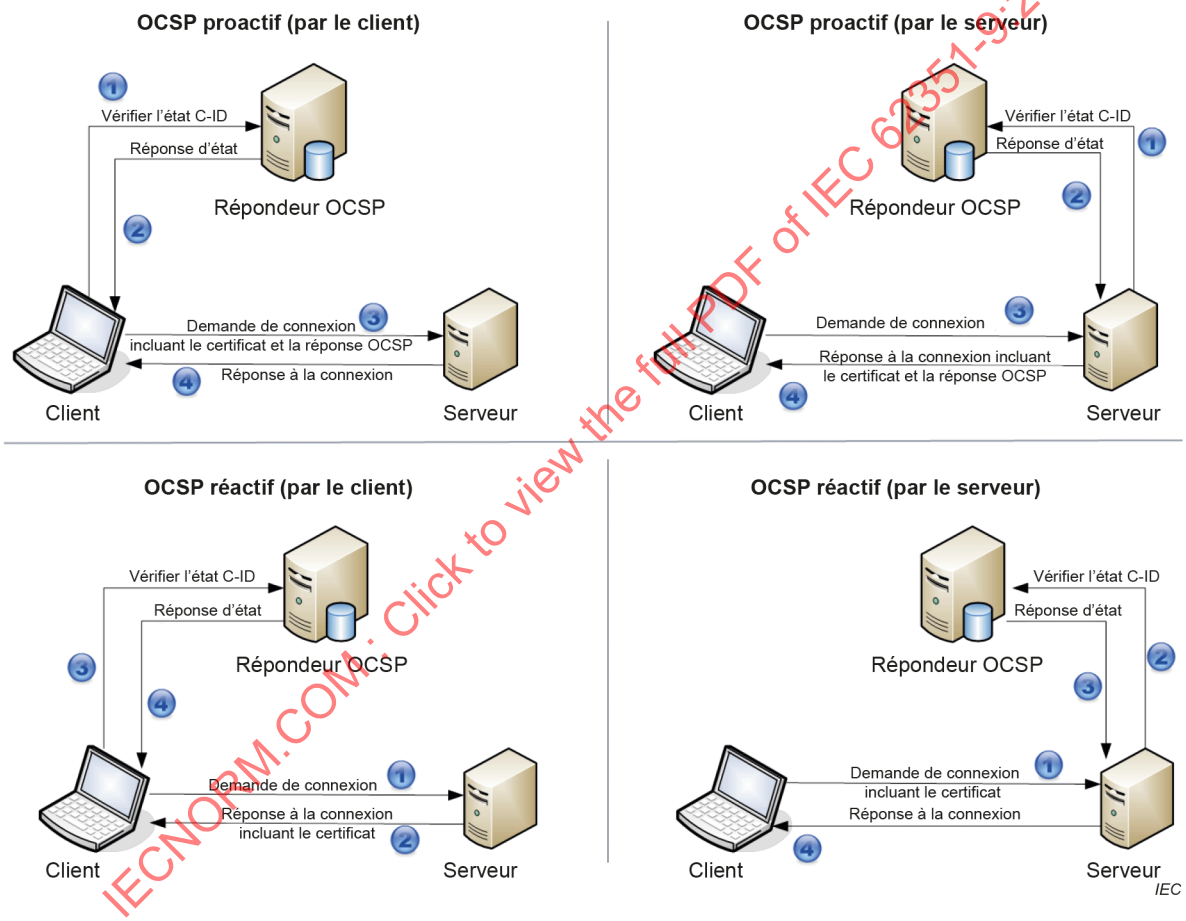





**Figure 17 – Flux d'appels du protocole d'état de certificat en ligne (OCSP)**

Il est à noter que TLS 1.3 (RFC 8446, [45]) permet également l'agrafage OCSP pour le certificat TLS du côté client.

Si la révocation de certificat ou le contrôle de validité du certificat n'aboutit pas, l'entité est dans la plupart des cas censée continuer à fonctionner après l'envoi d'une alarme OCSP pour avertir les opérateurs d'une possible communication non authentifiée. Cela empêche les attaques par déni de service dues à la mise en indisponibilité des répondeurs OCSP.

### 5.9.3 Protocole de validation de certificat fondée sur le serveur (SCVP)

Le contrôle de la validité d'un certificat peut devenir complexe, en particulier si le chemin de certification est plutôt long. Dans ce cas, le protocole de validation de certificat fondée sur le

serveur (SCVP) permet à une entité de déléguer la construction et la validation du chemin de certification à un nœud "principal". Ce protocole est défini dans la RFC 5055 ([34]). Le SCVP peut être utilisé conjointement avec la CRL, ainsi qu'avec l'OCSP (comme représenté à la Figure 18).

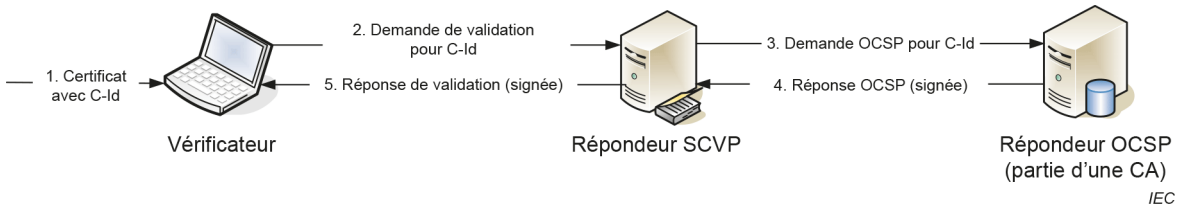



**Figure 18 – Vue d'ensemble du protocole SCVP utilisant le système principal OCSP**

De plus, il est recommandé de garder les hiérarchies de CA aussi planes et compactes que possible, de manière à réduire les pertes de performances induites lors de la validation des chaînes de certificat.

### 5.9.4 Récupération de la révocation de certificat d'une entité finale

Un certificat d'entité finale peut être révoqué pour différentes raisons, comme indiqué en 5.9.1. Pour se remettre de cette situation, la poursuite du traitement dépend de la raison réelle de la révocation. L'évaluation de cette raison et de la réaction appropriée sont généralement définies par la politique de sécurité d'une organisation (plan de reprise) et sont également reflétées dans la CPS de l'émetteur de certificat. Par conséquent, la discussion suivante vise à donner une vue d'ensemble des moyens potentiels d'obtenir un nouveau certificat pour le dispositif afin de le maintenir opérationnel.

Si la révocation est due à une clé privée compromise dans les dispositifs ou à toute autre cause impliquant un contact physique avec le dispositif par un attaquant, il est fortement recommandé qu'une procédure locale de dispositif soit mise en œuvre par un technicien de service pour modifier/actualiser le certificat et la clé privée correspondante. Il convient d'y inclure également la vérification de l'intégrité (physique et logique) du dispositif et de ses réglages opérationnels, afin de s'assurer qu'il ne peut pas être utilisé à mauvais escient comme initiateur d'une attaque et aussi pour s'assurer que le dispositif peut fonctionner en toute sécurité.

En général, la récupération après la révocation d'un certificat peut impliquer des moyens administratifs permettant le renouvellement à distance d'un certificat. Exemples:

- le dispositif possède un certificat dédié utilisé pour la gestion des accréditations de sécurité. Ce certificat peut être un certificat opérationnel (LDevID) ou un certificat fourni par le fabricant (IDevID). S'il n'est pas révoqué, ce certificat peut être utilisé pour enregistrer un nouveau certificat. Le dispositif peut vérifier de manière autonome l'état de révocation de son certificat et déclencher immédiatement l'enregistrement d'un nouveau certificat. La décision d'autoriser un nouvel enregistrement peut être soutenue par une mise en œuvre spécifique du dispositif. Par exemple, le dispositif peut prendre en charge un élément sécurisé ou un matériel de sécurité dédié pour stocker et/ou gérer l'IDevID de manière sûre, tandis que les accréditations opérationnelles (LDevID) peuvent être stockées dans un autre emplacement. Il incombe ensuite à la PKI opérationnelle de décider du renouvellement du certificat;
- le dispositif peut être déclenché pour renouveler le certificat par voie administrative. Il peut s'agir d'une connexion séparée utilisant des protocoles tels que Web (https), SNMP, ssh ou autre, pour laquelle l'accréditation a été fournie hors bande ou pendant l'intégration initiale. Il peut également être possible d'appuyer sur un bouton dédié sur les dispositifs afin de relancer l'amorçage.

Comme indiqué ci-dessus, il est fortement recommandé d'évaluer la raison de la révocation afin de pouvoir planifier les étapes suivantes en conséquence.

### 5.10 Confiance découlant des certificats (autosignés) délivrés hors PKI

Les certificats autosignés jouent un rôle important dans un environnement PKI. Les hiérarchies de confiance ou les chaînes de confiance dans les PKI sont enracinées par des "certificats racines" (voir 5.7.4). Les certificats racines de PKI peuvent être des certificats autosignés appelés "ancres de confiance". Ils sont créés par des CA de confiance permettant de distribuer cette confiance.

Dans certaines circonstances, pour établir la confiance, la mise en œuvre de certificats autosignés délivrés hors PKI peut être plus aisée que celle de certificats signés de PKI plus complexes. Cela peut notamment être le cas dans les déploiements plus petits ne comprenant qu'un nombre limité d'entités. Les certificats autosignés délivrés hors PKI sont des paires de clés publiques/privées générées en interne par une entité, la clé privée étant utilisée pour signer la propre clé publique de l'entité avec des informations supplémentaires telles que le nom du sujet, la durée de validité, etc. Pour pouvoir les utiliser comme des certificats de PKI (par exemple pour authentifier les interactions dans les connexions TLS), il est exigé que les certificats de clé publique autosignés soient conformes à la structure du certificat de clé publique décrite dans l'ISO/IEC 9594-8.

Ces certificats autosignés ne peuvent généralement pas être acceptés comme étant dignes de confiance. Ainsi, pour permettre à un groupe limité d'entités d'accepter les certificats autosignés pour l'authentification, il convient de ne les utiliser qu'avec des listes d'autorisation et de validation (voir 5.11).

Le niveau de complexité de l'utilisation de certificats autosignés spécifiques à une entité et de listes d'autorisation et de validation peut être comparé à celui de l'utilisation de clés prépartagées. Les certificats autosignés sont délivrés par entité, alors que les clés prépartagées le sont par connexion par paires. Par conséquent, le nombre de certificats autosignés dépend du nombre de points d'extrémité, le nombre de clés prépartagées dépendant du nombre de connexions entre ces points d'extrémité.

Comme il est nécessaire que les certificats autosignés soient des certificats de clé publique, leur application peut ouvrir un processus de migration vers des mises en œuvre de certificats PKI de clé publique, de sorte que les certificats autosignés puissent être aisément remplacés par des certificats PKI dès que ces derniers sont disponibles.

### 5.11 Listes d'autorisation et de validation

#### 5.11.1 Généralités

Les listes d'autorisation et de validation (AVL) fournissent des informations sur les entités de communication potentielles et les restrictions possibles des communications avec ces entités dans un environnement particulier. Elles sont spécifiées à l'Article 11 de l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019).

Une AVL est utilisée par une entité lorsque cette entité agit comme une partie utilisatrice. Cette entité est appelée "entité d'AVL". Une AVL est placée dans une entité appelée "agent d'autorisation" qui est responsable du contenu de l'AVL, c'est-à-dire que l'entité d'AVL fait confiance à l'agent d'autorisation pour fournir des informations valides. Une AVL est signée par l'agent d'autorisation émetteur.

La communication entre l'agent d'autorisation et l'entité d'AVL est à protéger comme toute autre communication en ce qui concerne l'intégrité, l'authenticité et la confidentialité possible. Afin de simplifier la validation des AVL, il est recommandé qu'un agent d'autorisation soit étroitement lié aux entités d'AVL qu'il sert, par exemple en étant au sein de la même organisation de façon à pouvoir établir une relation de confiance entre l'agent d'autorisation et les entités d'AVL qu'il sert (voir B.1).

Les AVL peuvent être utilisées dans des environnements non contraints ou contraints, tel que décrit en 5.11.2 et 5.11.3.

### 5.11.2 AVL dans les environnements non contraints

Un environnement non contraint est un environnement dans lequel les entités d'AVL sont en mesure d'effectuer la traditionnelle validation des certificats reçus.

Les AVL sont utilisées pour restreindre les relations de communication au sein d'un ensemble d'entités spécifié. Elles peuvent imposer des restrictions à ces entités au-delà de la longueur de chemin, de la politique et de la restriction de nom, comme détaillé en 11.3 de l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019).

Les entités avec lesquelles la communication est possible sont identifiées dans l'AVL par référence à leurs certificats de clé publique ou à la structure de noms de l'organisation à laquelle l'entité appartient.

### 5.11.3 AVL dans les environnements contraints

Une entité peut être contrainte de diverses manières:

- elle fonctionne sur batterie et nécessite une limitation des traitements afin d'économiser de la batterie;
- elle a de faibles capacités de traitement;
- son canal de communication a une bande passante limitée;
- elle a une capacité de stockage limitée;
- elle peut avoir des contraintes temporelles strictes devant répondre très rapidement à des incidents, sans laisser de temps à la communication externe pour valider les certificats de clé publique reçus.

Il est de la responsabilité de l'agent d'autorisation d'AVL de confirmer que tous les certificats de clé publique représentés dans l'AVL sont valides et peuvent être dignes de confiance au moment de l'émission d'une AVL. Il est également de la responsabilité de l'agent d'autorisation d'être à jour concernant la validité des certificats de clé publique dans leurs AVL. L'agent d'autorisation décide également s'il convient d'utiliser ou non un certificat de clé publique expiré ou révoqué lorsque son retrait est susceptible d'affecter une entité critique. Une entité d'AVL part du principe que l'utilisation d'un certificat de clé publique figurant dans la liste est valide.

Ce mode de fonctionnement exige une communication entre l'agent d'autorisation et les CA qui ont émis les certificats de clé publique figurant dans l'AVL.



## 6 Gestion des clés (normative)

### 6.1 Généralités

La gestion des clés s'applique au traitement d'un matériel de clé asymétrique et/ou symétrique. Pour la manipulation du matériel de clé asymétrique, en particulier la gestion des certificats et des clés privées correspondantes tout au long du cycle de vie du dispositif et du matériel de clé, l'Article 7 doit s'appliquer. Pour la gestion des clés symétriques, en particulier pour la gestion des clés de groupe, l'Article 8 doit s'appliquer.

### 6.2 Traitement des événements de sécurité

Tout au long du présent document, des événements de sécurité sont définis. Ces événements de sécurité sont destinés à prendre en charge le traitement des erreurs et donc à accroître la résilience du système. Il convient que les mises en œuvre fournissent un mécanisme pour annoncer les événements de sécurité.

Les informations relatives aux événements de sécurité et les informations détaillées potentielles ne peuvent être fournies par l'entité que sur la base de la disponibilité de ces informations par l'intermédiaire de la plate-forme sous-jacente ou des composants utilisés.

Il est fortement recommandé que les événements de sécurité définis tout au long du présent document soient mis à la disposition de l'infrastructure opérationnelle par des événements de cybersécurité spécifiés dans l'IEC 62351-14 et/ou par des objets de surveillance spécifiés dans l'IEC 62351-7. L'Annex D fournit une mise en correspondance des événements définis dans le présent document et la notion de l'IEC 62351-14.

Il est à noter que les avis, les avertissements, les erreurs et les alarmes sont utilisés pour indiquer la sévérité d'un événement du point de vue de la sécurité. La notion suivante de l'IEC 62351-14 est utilisée:

– un *avis* fait référence à une activité liée à la cybersécurité pendant l'utilisation ou la maintenance de routine d'une entité. Il ne porte ni sur une atteinte à la cybersécurité, ni sur une attaque, ni sur un écart par rapport à la condition de fonctionnement normal d'une entité;
– un *avertissement* est un écart par rapport à la condition de fonctionnement normal d'une entité, mais n'est pas nécessairement une cyberattaque;
– une *erreur* décrit une condition imprévue, qui peut indiquer une activité non autorisée. Elle peut ne pas exiger d'action immédiate;
– une *alarme* est une indication d'un problème sérieux qui peut indiquer une activité non autorisée. Il est recommandé de mener immédiatement une action.

Dans tous les cas, la politique de sécurité d'une organisation est censée déterminer le traitement final des événements en fonction de l'environnement opérationnel. Par exemple, l'évaluation de l'une des alarmes peut atteindre le niveau d'un incident.

### 6.3 Matériel cryptographique exigé

Les normes faisant référence au présent document doivent spécifier les matériels cryptographiques exigés qui doivent être présents pour établir des communications sécurisées, conformément au formulaire de déclaration de conformité d'instance de protocole (PICS) présenté à l'Article 9.

### 6.4 Génération de nombres aléatoires

La génération de valeurs aléatoires liées à la gestion de clés doit être conforme à l'ISO/IEC 18031 [87]. Les générateurs de paires de clés doivent être chargés de fournir des générateurs de nombres aléatoires (RNG) statistiquement adaptés et de les utiliser de manière appropriée.

NOTE Des recommandations peuvent être consultées dans le document NIST SP 800-90A [61], ainsi que dans l'IETF RFC 4086 [62].

Les clés doivent être générées en externe pour les entités qui ne sont pas en mesure de les générer en interne. Les CA ou d'autres systèmes autorisés peuvent proposer cette fonction de génération de clé externe. Les clés doivent ensuite être installées dans ces entités de manière sécurisée. Voir aussi Article B.12.

### 6.5 Identificateurs d'objet

#### 6.5.1 Concept d'identificateur d'objet

Un identificateur d'objet (OID, développé par l'UIT-T et l'ISO/IEC) est un mécanisme d'identification permettant d'associer tout type d'objet à un nom persistant. Il est généralement structuré de manière hiérarchique (sous forme d'arbre à OID). Voir aussi ISO/IEC 9834-1 | Rec. UIT-T X.660.

Les OID sont attribués par différentes normes internationales, y compris l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019), et peuvent être enregistrés dans le cadre d'un usage privé. Les valeurs (racines) de l'espace de noms OID de l'IEC 62351 sont 1.0.62351 et 1.2.840.10070. L'IEC TS 62351-8 utilise cette dernière approche pour définir les jetons d'accès (autorisations). Il convient que le présent document et toutes les nouvelles parties à venir de l'IEC 62351 utilisent le premier OID (1.0.62351).

### 6.5.2 Utilisation des identificateurs d'objet par le présent document

Le principe d'attribution des identificateurs d'objet (OID) est spécifié dans l'ISO/IEC 9834-1 | la Rec. UIT-T X.660. L'attribution des OID pour le présent document est définie comme suit:

```
id-IEC62351-9 OBJECT IDENTIFIER::= { 1 0 62351 9 }
```

La branche suivante est attribuée pour la définition des OID pour des extensions AVL:

```
avl62351Extion OBJECT IDENTIFIER::= { id-IEC62351-9 1 }
```

La branche suivante est attribuée pour la définition des OID pour des extensions AVL d'entrée:

```
avl62351EntryExt OBJECT IDENTIFIER::= { id-IEC62351-9 2 }
```

La branche suivante est attribuée pour la définition des OID pour des identificateurs de protocole:

```
id-62351prot OBJECT IDENTIFIER::= { id-IEC62351-9 3 }
```

## 7 Gestion de clés asymétriques (normative)

### 7.1 Généralités

Le présent article décrit les exigences normatives pour le traitement d'un matériel de clé asymétrique. Le matériel de clé asymétrique examiné dans le contexte de la présente spécification est constitué des trois éléments d'un certificat: la clé publique, la clé privée et les informations relatives aux autorités de certification émettrices (ancre de confiance, chaîne de certificats). Il est à noter que dans le contexte de l'IEEE 802.1AR, il s'agit de DevID (Device Identity). Le présent article définit la génération de paires de clés asymétriques sur l'entité finale, leur certification par l'intermédiaire d'une PKI, ainsi que la gestion du cycle de vie à travers la PKI.

### 7.2 Composants des certificats

#### 7.2.1 Composants des certificats de clé publique

Les certificats de clé publique utilisés pour la gestion des clés asymétriques doivent être des certificats de clé publique tels que définis dans l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019). La plupart des extensions générales sont spécifiées par l'ISO/IEC 9594-8 | la Rec. UIT-T X.509. Les extensions plus spécifiques sont spécifiées par d'autres documents. À titre d'exemple, l'IEC 62351-8 spécifie une extension pour la prise en charge des rôles d'utilisateur pour le contrôle d'accès.

Le Tableau 1 comprend les composants obligatoires et facultatifs des certificats de clé publique qui doivent pouvoir être reconnus et traités par les entités du système de puissance:

**Tableau 1 – Composants des certificats de clé publique**

| Nom du composant | Prise en charge | Contenu |
|---|---|---|
| Version | O | Version prise en charge du certificat (voir 7.4.4.2) |

| Nom du composant | Prise en charge | Contenu |
|---|---|---|
| Serial Number | O | Valeur entière, définie par la CA émettrice. Elle doit être unique pour chaque certificat délivré par une CA donnée |
| Signature | O | Algorithme de signature utilisé pour générer le certificat, déterminé par un OID (voir 7.4.4.4) |
| Issuer | O | Nom distinctif de la CA émettrice (voir 7.4.4.3) |
| Validity | O | Durée de validité du certificat (voir 7.4.4.5) |
| Subject | O | Identifie l'entité associée à la clé publique (voir 7.4.4.6). |
| Subject Public Key Info | O | Identificateur d'algorithme de la clé publique et clé publique elle-même, voir 7.4.4.7 |
| **Extensions** | | |
| Authority Key Identifier | O | Identificateur de clé de la CA émettrice: détermine la clé utilisée pour vérifier la signature du certificat (voir 7.4.4.10.2) |
| Subject Key Identifier | C1 | Identificateur de la clé publique des entités (voir 7.4.4.10.3) |
| Key Usage | O (c) | Identifie l'utilisation prévue du certificat de clé publique (voir 7.4.4.10.6) |
| Extended Key Usage | C | Identifie des finalités supplémentaires pour l'utilisation du certificat de clé publique (la sélection est conditionnelle, selon le cas d'utilisation) (voir 7.4.4.10.7) |
| Certificate Policy | F | Pour un certificat d'entité finale, il indique la politique en vertu de laquelle le certificat a été émis et les buts dans lesquels le certificat peut être utilisé par un OID [36]. |
| Subject Alternative Name | C | Contient un ou plusieurs noms alternatifs pour le sujet de certificat Voir 7.4.4.10.4. |
| Basic Constraints | C (c) | À définir pour émettre des certificats de CA sinon CA = faux (ou vide/absent); le composant supplémentaire `pathLenConstraint` peut être utilisé pour limiter le chemin de certification, selon la politique de sécurité de l'organisation (voir 7.4.4.10.5) |
| CRL Distribution Point | C2 | Emplacement de CRL pour http ou/et LDAP (voir 7.4.4.10.9) |
| Authority Information Access | C2 | Indique la méthode d'accès aux informations et services de l'émetteur du certificat: si des informations de révocation sont fournies par l'intermédiaire de l'OCSP, il contient l'emplacement du répondeur OCSP (`id-ad-ocsp`) (voir 7.4.4.10.10) |
| Role-based Access control | C3 | Active le contrôle d'accès basé sur les rôles conformément au profil A de l'IEC 62351-8 Voir 7.4.4.10.11. |
| Authorization Validation | F (c) | S'il est utilisé, il doit être mis à "c" (critique) pour s'assurer que la partie utilisatrice vérifie une AVL avant d'accepter le certificat d'entité finale (voir aussi 7.4.4.10.8). |
| SOA identifier | C4 | Source d'autorité dans l'émission du certificat d'AA Voir 7.4.4.10.13. |

C1: Doit être défini dans les certificats de CA

C2: La définition de cette extension relève de la responsabilité de la CA émettrice. Cela permet à une CA d'émettre également des certificats à court terme, qui n'exigent pas de contrôle de la révocation. Il convient que la période de validité du certificat à court terme soit soigneusement évaluée et choisie par l'opérateur, en particulier si les informations relatives au CRLDP ou au répondeur OCSP ne sont pas incluses dans le certificat délivré. En l'absence de ces informations, la révocation d'un certificat de clé publique ne peut pas être reconnue par la partie utilisatrice. Comme l'infrastructure doit prendre en charge la CRL et l'OCSP, les deux extensions peuvent être fournies pour permettre à la partie utilisatrice de vérifier l'état de la révocation.

C3: Doit être défini dans le certificat d'entité finale lorsqu'il prend en charge le profil A de l'IEC 62351-8.

C4: Doit être défini dans l'émission de certificats d'AA pour permettre l'identification d'une AA par la CA émettrice.

La notation suivante est utilisée dans le tableau:

O Exigé: le composant doit être inclus et traité.

F Facultatif: le composant peut être inclus.

C Utilisation conditionnelle (voir remarques en bas du Tableau 1).

(c) Cette extension est critique. Si une mise en œuvre reconnaît qu'une extension "critique" est présente, mais qu'elle ne peut pas l'interpréter, elle doit rejeter le certificat.

Le contenu du certificat est contrôlé pendant la vérification des certificats. Le traitement des erreurs dépend des problèmes identifiés au cours de la vérification. La vérification des certificats est décrite en 7.4.

### 7.2.2 Composants des certificats d'attributs

Si des certificats d'attributs sont utilisés, ils doivent être tels que définis dans l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019). Les extensions du certificat peuvent être définies dans d'autres documents.

Le Tableau 2 comprend les composants obligatoires et facultatifs des certificats d'attributs qu'il convient que les entités du système de puissance utilisant des certificats d'attributs soient en mesure de reconnaître et de traiter.

**Tableau 2 – Composants des certificats d'attributs**

| Nom du composant | Prise en charge | Contenu |
|---|---|---|
| Version | O | Version prise en charge du certificat (voir 7.4.5.2) |
| Holder | O | Contient une référence au détenteur d'un certificat de clé publique ou d'un autre nom d'entité (voir 7.4.5.3) |
| Issuer | O | Contient le nom distinctif de l'AA émettrice (voir 7.4.5.4) |
| Signature | O | Algorithme de signature utilisé pour générer le certificat, déterminé par un OID (voir 7.4.5.5) |
| Serial Number | O | Valeur entière, définie par la AA émettrice. Elle doit être unique pour chaque certificat délivré par une AA donnée |
| Attributs | O | Voir 7.4.5.6. Si un contrôle d'accès basé sur les rôles conformément au profil B de l'IEC 62351-8 est utilisé, l'attribut défini doit être utilisé pour les certificats d'attributs. |
| attrCertValidityPeriod | O | Contient la période de validité du certificat d'attribut (voir 7.4.5.7) |
| **Extensions** | | |
| Authority Key Identifier | F | Identificateur de clé de l'AA émettrice Voir 7.4.5.8.2. |
| CRL Distribution Point | C1 | Même extension que celle utilisée dans les certificats de clé publique. Dans le contexte des certificats d'attributs, elle est utilisée pour fournir l'emplacement de l'ACRL (voir 7.4.5.8.3) |
| Authority Information Access | C1 | Indique la méthode d'accès aux informations et services de l'émetteur du certificat d'attribut (voir 7.4.5.8.3) |
| No Revocation Available (noRevAvail) | C1 | Indique que les informations concernant l'état de révocation ne sont pas fournies pour ce certificat d'attribut (voir 7.4.5.8.3) |
| C1 Si `noRevAvail` est fourni, CRLDP/AIA ne doit pas l'être. Si CRLDP ou AIA est fourni, `noRevAvail` ne doit pas l'être pour éviter les incohérences. La définition de ces extensions relève de la responsabilité de l'AA émettrice. Cela permet à une AA d'émettre également des certificats d'attributs avec une période de validité très courte, qui n'exigent pas de contrôle de la révocation. | | |
| La notation suivante est utilisée dans le tableau:<br>O Exigé: le composant doit être inclus et traité.<br>F Facultatif: le composant peut être inclus.<br>C Utilisation conditionnelle (voir remarques en bas du Tableau 2). | | |



Le contenu du certificat est contrôlé pendant la vérification des certificats. Le traitement des erreurs doit dépendre des problèmes identifiés au cours de la vérification. La vérification des certificats est décrite en 7.4.

Il est à noter que la prise en charge fonctionnelle des certificats d'attributs dans l'IEC 62351 n'est spécifiée que pour RBAC. Les autres utilisations ne relèvent pas du domaine d'application du présent document.

### 7.3 Génération et installation des certificats

#### 7.3.1 Génération et installation des clés privées et publiques

Les entités exécutant des fonctions cryptographiques asymétriques doivent détenir au moins une paire de clés asymétriques. Une entité doit soit générer sa propre paire de clés cryptographiques asymétriques et contacter une infrastructure PKI pour la certification, soit recevoir une paire de clés cryptographiques générées en externe, y compris le certificat d'une CA émettrice. La clé privée de la paire de clés doit être stockée de manière sécurisée. Il convient que la distribution de la paire de clés cryptographiques générée de manière centralisée soit effectuée dans un emplacement protégé.

Les entités doivent générer ou recevoir de nouvelles paires de clés dans les conditions suivantes:

- aucune paire de clés n'est présente au démarrage (si le certificat de clé publique associé est absent ou a expiré);
- modification de la fonction de contrôleur de l'entité (changement de propriété, d'autorité de contrôle et/ou reconfiguration de l'entité);
- par commande d'une entité autorisée (par exemple demande de renouvellement de certificat);
- la clé privée de l'entité a été compromise;
- l'adresse IP (ou FQDN) de l'entité a changé.

Il est fortement recommandé que la génération de clés côté client soit prise en charge par les dispositifs du système de puissance, afin d'éviter le transport de la clé privée à l'extérieur du dispositif.

Le format utilisé pour la fourniture de la clé est décrit en 7.3.2.

#### 7.3.2 Protection des clés cryptographiques

Les clés privées et les clés symétriques à long terme (y compris les clés prépartagées) de chaque entité doivent être protégées contre les modifications, l'extraction et le clonage non autorisés.

Pendant le transport, la clé privée doit être protégée contre les interceptions et la falsification en étant chiffrée à l'aide d'une clé de transport telle que définie dans le PEM (RFC 7468), PKCS #8 (RFC 7468) et PKCS #12 (RFC 7292). Toutes les entités doivent prendre en charge le traitement de PKCS #12. Il convient que le conteneur PKCS #12 inclue également tous les certificats de (sous-)CA émettrices.

L'incapacité à traiter les objets PKCS #12 en raison de problèmes de formatage doit générer un événement de sécurité ("avertissement: format PKCS #12 incompatible"). Il convient que les mises en œuvre fournissent un mécanisme pour annoncer les avertissements de sécurité.

Si les objets PKCS #8 sont pris en charge, mais qu'ils ne peuvent pas être traités en raison de problèmes de formatage, un événement de sécurité ("avertissement: du format PKCS #8 incompatible") doit être déclenché. Il convient que les mises en œuvre fournissent un mécanisme pour annoncer les avertissements de sécurité.

Dans la mesure des possibilités techniques, toutes les tentatives de compromission d'une clé doivent être détectables. Ce type d'événement doit être consigné dans un journal et doit déclencher une alarme.

NOTE Des recommandations sur la protection des clés cryptographiques peuvent être consultées dans le document NIST SP 800-57, Partie 1 [57] et dans le document FIPS 140-2 [53]. En outre, l'IEEE 1686 [5], l'IEEE c37.240 [75] et l'IEC 62443-4-2 [74] spécifient des exigences relatives à la protection des informations sensibles.

### 7.3.3 Utilisation de l'infrastructure existante de gestion des clés de sécurité

L'utilisation d'une infrastructure existante de gestion des clés publiques de sécurité (PKI) doit être autorisée tant que les CA émettrices fournissent une chaîne complète de certificats de confiance (CA intermédiaires) jusqu'au certificat de CA racine et prennent en charge les protocoles de gestion/enregistrement de certificats exigés comme indiqué en 7.3.7 pour s'interfacer avec des dispositifs/entités.

Il est fortement recommandé de n'installer que les certificats de CA racines qui sont nécessaires dans l'environnement opérationnel et pour la tâche accomplie par le dispositif. Le choix du ou des certificats de CA racines pris en charge fait généralement partie de la politique de sécurité d'une organisation.

### 7.3.4 Politique de certification

Il est vivement recommandé de définir une politique de certification (voir RFC 3647 ([26]) pour obtenir des recommandations).

### 7.3.5 Enregistrement d'entité pour l'établissement d'identité

Toutes les entités à mettre en service dans une organisation doivent être enregistrées dans au moins une autorité d'enregistrement (RA), qui peut être colocalisée avec l'autorité de certification (CA) approuvée par l'organisation. Cette RA doit être en mesure de vérifier l'identité de l'entité sur une demande de signature de certificat (CSR). L'enregistrement peut être manuel (par exemple pour un petit nombre d'entités) ou automatique en exécutant des scripts ou une liste positive configurée qui sont générés à partir des données techniques. L'enregistrement peut avoir lieu hors bande, hors site ou sur site. La CSR peut être exportée et importée manuellement ou transportée par l'intermédiaire d'un protocole d'enregistrement comme décrit en 7.3.7.

Les données d'enregistrement doivent inclure au moins l'un des éléments suivants:

- le sujet de l'entité, qui identifie l'entité et le nom unique qui apparaît dans le certificat de l'entité ou un autre identificateur unique du sujet, par exemple `serialNumber`. Il est à noter que le certificat peut inclure le numéro de série physique de l'appareil comme `X520SerialNumber` avec la balise OID `id-at-serialNumber`. Voir Tableau 1;
- si le SCEP est utilisé pour l'enregistrement de certificat, un code d'activation unique à une seule utilisation (ou OTP), qui permet à l'entité de s'authentifier elle-même auprès de la RA, lors de l'exécution d'une demande de certification (CSR), doit être fourni comme données d'enregistrement. Note: La manière dont les OTP sont créées et distribuées aux entités qui s'enregistrent ne relève pas du domaine d'application du présent document;
- un numéro de série de certificat et un émetteur de certificat de clé publique intégré au niveau du fabricant, qui permet à l'entité d'être déjà authentifiée à l'aide de ce certificat de clé publique;
- une empreinte digitale du certificat de clé publique utilisé pour authentifier l'entité qui s'enregistre;
- une CA émettrice de confiance, qui permet l'inscription avec des certificats arbitraires provenant de cette CA émettrice.

### 7.3.6 Configuration d'entité

En plus des paramètres de certificat de base définis dans l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019), les données de configuration d'entité doivent inclure les éléments suivants:

- le ou les certificats de CA de la ou des organisations auxquelles l'entité doit faire confiance et avec lesquelles elle doit communiquer (voir 5.6.4). Une mise en œuvre revendiquant la conformité au présent document doit prendre en charge au moins 5 ancres de confiance

liées à l'autorité de certification. Le nombre réel dépend du cas d'utilisation cible. Il est à noter que cette exigence est alignée sur l'IEC 62351-3:— [4];

- l'adresse IP de la RA de l'organisation ou un nom de domaine (tel que IEC62351.LocalCA) qui doit être admis pour exécuter des opérations de PKI, comme le traitement des CSR;
- une partie du certificat de l'entité est le sujet contenant le sujet du dispositif (par exemple nom commun (CN) ou autres valeurs de sujet), qui identifie l'entité de manière unique. Ce nom d'entité figure dans le certificat de l'entité. Voir également 7.2.

L'entité peut interroger son nom DNS en utilisant son adresse IP. Une entité peut énumérer plusieurs `dnsName` et les adresses IP correspondantes. Les adresses IP peuvent être utilisées dans des environnements sans service DNS.

Le paramètre de délai d'expiration de CSR est un problème de mise en œuvre locale, et le dispositif se réenregistre s'il n'obtient pas de certificat. Il est attendu qu'il soit traité par la politique de sécurité de l'opérateur.

Les données d'enregistrement doivent être installées et configurées individuellement dans chaque entité afin de s'assurer que la RA peut authentifier l'entité lorsqu'elle exécute une CSR.

Un code d'activation (mot de passe à usage unique) ou un certificat fourni par le fabricant doit être pris en charge pour l'enregistrement dans un réseau d'infrastructure PKI d'opérateurs, ce qui permet à l'entité de s'authentifier elle-même auprès de la CA. La méthode d'authentification utilisée pendant l'enregistrement dépend des capacités du système (PKI déployée).

Il est à noter que la configuration d'entité peut être effectuée manuellement à l'aide d'un outil de configuration ou d'ingénierie. Elle peut également être prise en charge par des protocoles d'intégration comme indiqué pour les paramètres de sécurité en 5.8.3.

### 7.3.7 Enregistrement d'entité

#### 7.3.7.1 Généralités

Une fois que les entités ont été configurées avec les données d'enregistrement exigées et qu'elles ont généré leur propre paire de clés asymétriques, elles doivent lancer une procédure de CSR afin d'être opérationnelles, selon les politiques de sécurité de certificat de l'organisation. Une connexion en ligne à la RA/CA de l'organisation est exigée pour l'enregistrement de certificat, sauf en cas d'enregistrement hors bande.

Seules les entités enregistrées doivent être autorisées au niveau de la RA/CA (voir 7.3.5).

Pour l'enregistrement manuel, les entités doivent générer la demande de signature de certificat (CSR) en utilisant le format PKCS #10 ([18]). Pour l'enregistrement automatique, l'entité doit fournir la CSR à la RA responsable spécifiée pendant la configuration. Pour l'enregistrement hors bande, la CSR doit être transportée par des moyens arbitraires jusqu'à la RA/CA responsable. La RA doit contrôler la validité de la demande en vérifiant la preuve de possession de la clé privée correspondante en vérifiant la signature PKCS #10 (CSR).

Si la demande est valide, la RA doit envoyer une demande à la CA respective. La CA doit générer un certificat de clé publique et le fournir à la RA en vue d'une distribution ultérieure à l'entité demandeuse. Le protocole utilisé pour la communication entre la RA et la CA n'entre pas dans le domaine d'application de la présente spécification.

Si la demande n'est pas valide, la RA ne doit pas envoyer de demande à la CA.



---

4 En cours d'élaboration. Stade au moment de la publication: IEC/RFDIS 62351-3:2023.

NOTE La RA et la CA peuvent être déployées ensemble comme une seule entité. Le processus décrit ci-dessous s'applique toujours.

Pour l'enregistrement automatique, l'infrastructure (RA/CA) doit prendre en charge les protocoles d'enregistrement suivants pour l'interaction avec les entités finales (clients qui s'enregistrent):

- protocole SCEP (RFC 8894) pour la rétrocompatibilité axée sur les certificats de clé publique RSA;
- protocole EST (RFC 7030) pour la prise en charge des certificats de clé publique RSA ou ECC en tant que protocole d'enregistrement préférentiel.

L'utilisation de l'enregistrement automatique permet d'obtenir une preuve d'identité en utilisant le code d'activation (OTP) ou un certificat déjà disponible et la clé privée correspondante, conjointement avec les données d'enregistrement sur la RA.

Une entité cliente (entité qui s'enregistre) peut prendre en charge au moins l'un des protocoles d'enregistrement.

Si l'enregistrement est protégé par TLS, il est recommandé d'utiliser le profil TLS défini dans l'IEC 62351-3. Si l'interface fonctionnelle d'une PKI utilisant TLS pour protéger la communication d'enregistrement peut prendre en charge l'IEC 62351-3, il est fortement recommandé de l'utiliser.

Les certificats doivent être transportés comme cela est défini dans les protocoles d'enregistrement. Pour le stockage des certificats, voir 7.3.2.

#### 7.3.7.2 Enregistrement d'entité en utilisant le protocole SCEP

L'application du SCEP doit respecter les exigences obligatoires décrites dans la RFC 8894. Les entités prenant en charge le SCEP doivent respecter l'obligation de mettre en œuvre la fonctionnalité décrite à la section 2.9 de la RFC 8894:

- interrogation des capacités de la CA (`GetCACaps`);
- distribution des certificats de CA (`GetCACert`);
- enregistrement de certificat (`PKCSReq`);
- renouvellement de certificat (`RenewalReq`).

Si l'enregistrement d'entité en utilisant le SCEP a été réussi, un événement de sécurité doit être déclenché ("avis: l'enregistrement avec SCEP a été effectué avec succès").

Si l'enregistrement d'entité en utilisant le SCEP a été annulé en raison d'une erreur, un événement de sécurité doit être déclenché ("erreur: l'enregistrement avec SCEP a été annulé en raison d'une erreur"). Il convient de fournir des informations détaillées conformément à la RFC 8894.

#### 7.3.7.3 Enregistrement d'entité en utilisant le protocole EST

Le support fonctionnel obligatoire exigé de l'EST concorde avec les exigences obligatoires décrites dans la RFC 7030 avec l'ajout de la prise en charge obligatoire de la récupération des attributs de certificat. Les entités qui utilisent l'EST doivent également prendre en charge:

- la distribution des certificats de CA en utilisant le point d'extrémité */cacerts* pour pouvoir interroger un certificat de CA basé sur une ancre de confiance implicite. Cette fonctionnalité permet tout d'abord la distribution de certificats de CA pour générer une base de données d'ancres de confiance, mais aussi la mise à jour des certificats de CA;
- l'échange *SimplePKIRequest/Response* (enregistrement simple) par l'intermédiaire du point d'extrémité */simpleenroll* au niveau de la RA peut prendre en charge l'échange

*FullPKIRequest/Response* par l'intermédiaire du point d'extrémité */fullcmc* au niveau de la RA;

- le réenregistrement (renouvellement des certificats ou des clés des certificats) en utilisant le point d'extrémité */simplereenroll* au niveau de la RA.

Facultativement, le message *csrattrs* (demande d'attribut de certificat) peut être pris en charge pour pouvoir demander les attributs de certificat CSR à utiliser dans la CSR pour permettre des exigences explicites depuis le côté d'une RA/CA. Ce message peut être omis si l'entité qui s'enregistre a reçu ces informations par d'autres moyens (par exemple par configuration).

Le traitement de la demande/réponse d'attribut de CSR est exigé afin de prendre en charge le choix des algorithmes de signature et des paramètres associés (longueur de clé, sélection de courbe pour les algorithmes à courbe elliptique).

Il convient que les implémenteurs prennent connaissance de l'IETF RFC 8951 qui clarifie les codages de transfert et ASN.1.

Si l'enregistrement d'entité en utilisant l'EST a été réussi, un événement de sécurité doit être déclenché ("avis: l'enregistrement avec EST a été effectué avec succès").

Si l'enregistrement d'entité en utilisant l'EST a été annulé en raison d'une erreur, un événement de sécurité doit être déclenché ("erreur: l'enregistrement avec EST a été annulé en raison d'une erreur"). Il convient de fournir des informations détaillées conformément à la RFC 7030.

### 7.3.8 Mise à jour des informations d'ancre de confiance

Si la mise à jour automatique des informations d'ancre de confiance est prise en charge, la mise à jour des certificats d'ancre de confiance doit être effectuée à l'aide de l'EST (voir 7.3.7.3) sur la base d'une ancre de confiance déjà existante en utilisant le point d'extrémité */cacerts*. La mise à jour d'ancres de confiance peut être effectuée en utilisant le protocole TAMP (RFC 5934).

Si le TAMP est utilisé, les messages TAMP suivants doivent être pris en charge:

- TAMP Status Query: ce message d'interrogation d'état permet de demander des informations concernant les ancres de confiance qui sont actuellement installées dans un magasin d'ancres de confiance, ainsi que la liste des communautés auxquelles appartient le magasin;
- TAMP Status Query Response: ce message envoyé par un magasin d'ancres de confiance est une réponse à un message TAMP Status Query valide. Il donne des informations concernant les ancres de confiance actuellement installées dans le magasin d'ancres de confiance, ainsi que la liste des communautés auxquelles appartient ce magasin, le cas échéant;
- Trust Anchor Update: ce message permet d'ajouter, de supprimer et de modifier les ancres de confiance de gestion et d'identité. Il ne peut pas être utilisé pour mettre à jour l'ancre de confiance supérieure;
- Trust Anchor Update Confirm: ce message envoyé par un magasin d'ancres de confiance est une réponse à un message Trust Anchor Update valide. Il donne des informations sur la réussite ou l'échec de chacune des mises à jour demandées;
- Apex Trust Anchor Update: ce message remplace la clé publique opérationnelle et, éventuellement, la clé publique de circonstance associée à l'ancre de confiance supérieure. Chaque magasin d'ancres de confiance contient exactement une ancre de confiance supérieure. Aucune contrainte n'est associée à l'ancre de confiance supérieure. L'identificateur de la clé publique opérationnelle permet d'identifier l'ancre de confiance supérieure dans les messages TAMP qui suivent. La signature numérique sur le message Apex Trust Anchor Update est validée avec la clé publique opérationnelle en cours ou avec la clé publique de circonstance en cours, conformément à la RFC;

- Apex Trust Anchor Update Confirm: ce message envoyé par un magasin d'ancres de confiance est une réponse à un message Apex Trust Anchor Update valide. Il donne des informations sur la réussite ou l'échec de la mise à jour de l'ancre de confiance supérieure;
- TAMP Error: ce message envoyé par un magasin d'ancres de confiance est une réponse à un message TAMP non valide. Il donne une indication de la raison de l'erreur.

## 7.4 Composants des certificats et vérification associée

### 7.4.1 Généralités

Les certificats d'entité, y compris les certificats de clé publique, les certificats d'attribut et les certificats de CA, doivent être vérifiés par la partie utilisatrice. Tous les composants des certificats doivent être autorisés conformément à l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019). Les paragraphes suivants décrivent les composants des certificats et leur vérification. Sur la base du résultat de vérification des certificats, des événements de sécurité spécifiques doivent être générés en cas d'erreur.

Il est recommandé de valider les certificats autosignés en utilisant des listes d'autorisation et de validation de certificats, comme décrit en 7.8.

### 7.4.2 Format et codage du certificat

La structure formelle du certificat doit être vérifiée. À cet effet, la partie utilisatrice doit valider le codage DER ASN.1, ainsi que la validité de la structure de certificat ASN.1.

Si un certificat ne peut pas être traité en raison d'erreurs de décodage, un événement de sécurité ("avertissement: format de certificat incompatible. Échec de la vérification.") doit être déclenché.

### 7.4.3 Vérification de la signature de certificat

Le processus de vérification de la signature de certificat pour les certificats de clé publique et les certificats d'attribut est similaire. Dans les deux cas, la signature est calculée sur les composants de certificat sur la base du `signatureAlgorithm` identifié par l'`AlgorithmIdentifier`. Voir aussi 7.4.4.4 pour ce qui est de la prise en charge de l'algorithme de signature.

En cas d'échec de la vérification de signature d'un certificat de clé publique par rapport au `CTBSCertificate`, un événement de sécurité doit être déclenché ("alarme: la signature du certificat de clé publique n'a pas pu être vérifiée").

En cas d'échec de la vérification de signature d'un certificat d'attribut sur le `TBSAttributeCertificate`, un événement de sécurité doit être déclenché ("alarme: la signature du certificat d'attribut n'a pas pu être vérifiée").

### 7.4.4 Composants des certificats de clé publique

#### 7.4.4.1 Généralités

Les composants des certificats de clé publique sont basés sur le contenu du Tableau 1.

#### 7.4.4.2 Version

Le numéro de `version` du certificat doit être 2 (indiquant X.509v3).

Si la version de certificat ne correspond pas à la version attendue, un événement de sécurité ("erreur: mauvaise version de certificat") doit être déclenché.

#### 7.4.4.3 Issuer

Le composant `issuer` identifie la CA qui a émis le certificat. Il contient le nom distinctif de la CA émettrice. Le nom distinctif de l'émetteur est constitué d'attributs définis dans le composant `subject` (voir 7.4.4.6). Le composant `issuer` est utilisé dans le processus de validation du chemin de certification (voir 7.4.4.8).

NOTE L'Annexe M de l'ISO/IEC 9594-8 | la Rec. UIT-T X.509 fournit une brève introduction concernant les noms distinctifs.

Si l'émetteur ne correspond pas à un émetteur connu et de confiance, un événement de sécurité ("erreur: émetteur du certificat de clé publique non digne de confiance") doit être déclenché.

#### 7.4.4.4 Signature

Les mises en œuvre revendiquant la conformité à la présente spécification doivent prendre en charge le traitement des algorithmes de signature suivants pour les certificats:

- RSA (voir aussi B.3.2);
- ECDSA (voir aussi B.3.4).

Une mise en œuvre peut prendre en charge l'EdDSA (voir aussi B.3.5). Il est à noter qu'au moment de la publication, l'EdDSA n'est pas utilisée dans la spécification de l'IEC 62351. Une prise en charge facultative peut être considérée dans le contexte de la crypto-agilité, si l'EdDSA est prise en compte dans les spécifications de l'IEC 6351.

Les paramètres associés sont la longueur de clé et les courbes elliptiques sélectionnées, toutes deux étant décrites ci-dessous.

La technologie RSA à clés de 2 048 bits doit être prise en charge. La longueur de clé de 2 048 est la longueur minimale à prendre en charge pour les signatures RSA. Sur la base des recommandations de NIST SP 800-131A Rev.2 [89] ou BSI TR01202-1 [58], il est fortement recommandé de prendre également en charge la longueur de clé RSA de 3072 bits et plus pour tenir compte des avancées réalisées en cryptographie. La prise en charge de la technologie RSA à clés de 1024 bits est déconseillée. Son usage est limité à la rétrocompatibilité. Le choix de l'algorithme de signature dépend de la politique de sécurité de l'organisation.

NOTE 1 Les recommandations concernant la longueur de clé exigée pour les algorithmes de signature sont constamment révisées et peuvent être consultées dans les documents NIST SP 800-57 ou BSI TR01202-1.

NOTE 2 Le traitement des clés RSA longues est un processus exigeant, ce qui peut être un problème pour les dispositifs contraints. L'utilisation de la cryptographie à courbe elliptique peut alors être envisagée (voir aussi B.3). Pour ECDSA, la longueur de clé minimale est de 256 bits (en combinaison avec SHA-256). L'OID pour `ecdsa-with-SHA256` à utiliser est: 1.2.840.10045.4.3.2 (iso(1) member-body(2) us(840) ansi-X9-62(10045) signatures(4) ecdsa-with-SHA2(3) 2).

La courbe à utiliser pour ECDSA doit être au minimum `secp256r1`. L'OID pour cette courbe est: 1.2.840.10045.3.1.7 (iso(1) member-body(2) us(840) ansi-X9-62(10045) curves(3) prime(1) prime256v1(7)).

Un support facultatif peut être fourni pour ECDSA en utilisant la courbe suivante:

- `brainpoolP256r1` (telle que définie dans la RFC 5639): L'OID pour cette courbe est: 1.3.36.3.3.2.8.1.1.7 (iso(1) identified-organization(3) teletrust(36) algorithm(3) signatureAlgorithm(3) ecSign(2) ecStdCurvesAndGeneration(8) ellipticCurve(1) versionOne(1) brainpoolP256r1(7)).

L'IEC 62351-5:20xx (dont la publication est prévue en 2022) exige la prise en charge de trois autres courbes:

- la courbe 22519 (telle que définie dans la RFC 7748 [43]);

- la courbe 448 (telle que définie dans la RFC 7748 [43]);
- la Secp256k1 (telle que définie en [60]): l'OID pour cette courbe est: 1.3.132.0.10 (iso(1) identified-organization(3) certicom(132) curve(0) ansip256k1(10)).

Il est à noter qu'au moment de la publication du présent document la prise en charge de la courbe 22519 et de la courbe 448 est uniquement prévue dans l'IEC 62351 dans le contexte de l'accord de clé ECDH (voir IEC 62351-5), et non dans le contexte des signatures numériques. Il est également à noter que si elles sont utilisées pour les signatures numériques, la courbe 22519 et la courbe 448 peuvent uniquement être utilisées dans l'EdDSA.

Les documents faisant référence au présent document peuvent spécifier des courbes supplémentaires à prendre en charge.

Si le `signatureAlgorihm` n'est pas pris en charge, un événement de sécurité ("erreur: algorithme de signature non pris en charge") doit être déclenché.

#### 7.4.4.5 Validité

La période de validité d'un certificat de clé publique dépend de la politique de sécurité de l'organisation pour les certificats opérationnels (certificats LevID), ainsi que de la politique de sécurité du fabricant pour les certificats de dispositifs initiaux (certificats IDevID). La période de validité est déterminée par les valeurs `not before` et `not after`. La partie utilisatrice doit vérifier que la date et l'heure actuelles sont comprises entre les valeurs `notBefore` et `notAfter` du certificat:

- si l'heure actuelle est postérieure à la valeur `notAfter`, un événement de sécurité ("erreur: certificat de clé publique expiré") doit être déclenché;
- si l'heure actuelle est antérieure à la valeur `notBefore`, un événement de sécurité ("erreur: certificat non encore valide") doit être déclenché.

Conformément à l'UIT-T X.509, les valeurs définies pour la validité du certificat (not before et not after) peuvent être spécifiées en `UTCTime` jusqu'à fin 2049. À partir de 2050, ces valeurs doivent être spécifiées en `GeneralizedTime`. Il est à noter que la RFC 5280 exige que la validité doive être spécifiée en `UTCTime` jusqu'à fin 2049.

NOTE Pour indiquer qu'un certificat n'a pas de date d'expiration bien définie, la RFC 5280 utilise la valeur `GeneralizedTime` de 99991231235959 comme valeur `notAfter`.

Les certificats non valides ne doivent pas être acceptés par les entités, par exemple lors de l'établissement de sessions TLS, comme cela est exigé dans l'IEC 62351-3.

Il est à noter que l'accès à une horloge synchronisée sur le temps atomique international (TAI) ou le temps universel coordonné (UTC) est exigé afin d'assurer une évaluation précise la date d'expiration du certificat. La synchronisation exacte de l'heure et de l'horloge dans un réseau est généralement assurée par les protocoles NTP ou PTP, qui sont respectivement définis dans la RFC 5905 [38] ou l'IEEE 1588 [4]. Il est à noter que pour NTP, les options de sécurité sont actuellement révisées par l'IETF en tant que NTS (application de sécurité du temps réseau, RFC 8915). Pour PTP, l'IEEE 1588 v2.1 définit des options de sécurité pour assurer une fourniture des informations temporelles tout en protégeant leur intégrité. Les exigences relatives à la synchronisation d'horloge sont décrites en 7.7.

#### 7.4.4.6 Subject

Le composant `subject` contient des informations relatives au nom distinctif (DN) du sujet. Ce composant doit être pris en charge.

Si le `subject` n'est pas contenu ou s'il ne contient que des attributs inconnus, un événement de sécurité ("avertissement: sujet non inclus dans le certificat de clé publique") doit être déclenché.

S'il a été contrôlé, la vérification de la totalité du nom distinctif ou de champs spécifiques doit correspondre aux identificateurs autorisés définis par la politique de sécurité de l'organisation. Les cas où la vérification du sujet peut être omise concernent le RBAC comme décrit dans l'IEC 62351-8, profil A. Dans ce cas, le sujet n'est utilisé qu'à des fins de journalisation.

Le contenu du composant `subject` est susceptible d'être différent pour les certificats opérationnels (certificats LevID) ou les certificats fournis par le fabricant (certificats IDevID). Le contenu des attributs utilisés dans le `subject` pour les certificats opérationnels est défini par un opérateur. Une mise en œuvre doit pouvoir valider et traiter au moins les éléments suivants:

- `CN`: nom courant, identificateur unique du dispositif. Il est à noter que si un dispositif possède plusieurs certificats, la finalité peut être reflétée dans le CN;
- `O`: nom de l'organisation à laquelle le dispositif peut être attribué (par exemple opérateur ou fabricant);
- `OU`: nom de l'unité organisationnelle incluant d'autres détails sur l'organisation;
- `C`: code de pays (ISO 3166) du responsable de l'organisation;
- `serialNumber`: numéro de série de l'entité (à ne pas confondre avec le numéro de série du certificat) tel que défini dans l'IEEE 802.1AR (comme `X520SerialNumber`.

Des attributs supplémentaires peuvent être spécifiés, par exemple par un fabricant pour un certificat de fabricant (certificat IDevID) contenant des informations spécifiques sur le dispositif, par exemple des informations sur le type de produit.

#### 7.4.4.7 Subject Public Key Info

Le composant `subjectPublicKeyInfo` contient deux sous-composants fournissant des informations sur l'algorithme (OID), ainsi que la clé publique du certificat.

L'échec de la recherche d'un algorithme de correspondance pour la clé publique doit déclencher un événement de sécurité ("erreur: algorithme de clé publique natif dans le certificat de clé publique non pris en charge").

#### 7.4.4.8 Identificateurs uniques

Les composants `issuerUniqueIdentifier` et `subjectUniqueIdentifier` doivent être absents. Ces composants sont déconseillés dans la X.509.

#### 7.4.4.9 Validation du chemin de certification

La partie utilisatrice doit valider le chemin de certification en commençant par le certificat de CA émis par l'ancre de confiance et en terminant par les certificats de clé publique de l'entité finale. Dans un cas particulier, le certificat de clé publique de l'entité finale peut être émis directement par l'ancre de confiance (généralement appelé un certificat de CA racine).

La non-découverte de l'ancre de confiance correspondante d'un certificat dans la configuration locale doit déclencher un événement de sécurité ("erreur: ancre de confiance non prise en charge").

NOTE Souvent, les certificats intermédiaires sont transférés à l'intérieur de messages de protocole comme dans TLS. Néanmoins, la partie utilisatrice doit être en mesure de trouver les certificats racines contenus dans la configuration locale.

Tous les certificats doivent être correctement signés par la CA émettrice respective, tandis que le certificat racine doit être autosigné. Les composants `issuer` et `authorityKeyIdentifier` sont tenus d'être respectivement égaux au nom du sujet et au `subjectKeyIdentifier` du certificat suivant dans le chemin.

L'échec de la validation des certificats le long du chemin de certification doit déclencher un événement de sécurité ("erreur: le chemin de certification n'a pas pu être vérifié").

#### 7.4.4.10 Extensions de certificat

##### 7.4.4.10.1 Généralités

Toutes les extensions doivent être respectées conformément à l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019):

- si l'extension est indiquée comme étant critique, le certificat de clé publique doit être réservé à l'une des finalités mentionnées;
- si l'extension est indiquée comme étant non critique, elle indique alors la ou les finalités prévues de la clé. Si cette extension est présente et que la partie utilisatrice reconnaît et traite le type d'extension, la partie utilisatrice doit alors s'assurer que le certificat de clé publique doit être réservé à l'une des finalités indiquées. Si l'extension n'est pas critique et n'est pas connue, elle peut être négligée.

En outre, les certificats incluant une extension critique inconnue, ou des extensions critiques contenant des informations qui ne peuvent pas être traitées, doivent être rejetés.

Si le certificat contient une extension non reconnue marquée comme étant critique, un événement de sécurité ("erreur: extension critique inconnue dans le certificat de clé publique") doit être déclenché.

Si le certificat contient des informations non reconnues dans une extension marquée comme étant critique, un événement de sécurité ("erreur: informations inconnues dans l'extension critique") doit être déclenché.

##### 7.4.4.10.2 Authority Key Identifier

L'`authorityKeyIdentifier` contient le `subjectKeyIdentifier` du certificat de la CA émettrice.

Si l'`authorityKeyIdentifier` n'est pas contenu dans le certificat, un événement de sécurité ("erreur: identificateur de clé d'autorité non inclus dans le certificat de clé publique") doit être déclenché.

##### 7.4.4.10.3 Subject Key Identifier

L'extension `subjectKeyIdentifier` permet d'identifier les certificats contenant une clé publique particulière. Conformément à la RFC 5280, elle est obligatoire pour émettre des certificats de CA et peut être facultativement prise en charge dans les certificats d'entité finale.

Si le `subjectKeyIdentifier` n'est pas contenu dans un certificat de CA, un événement de sécurité ("erreur: identificateur de clé de sujet non inclus dans le certificat de clé publique") doit être déclenché.

##### 7.4.4.10.4 Subject Alternative Name

Si l'extension `subjectAltName` est disponible dans le certificat, elle doit être traitée. Elle peut contenir un ou plusieurs noms alternatifs pour le sujet du certificat.

Pour les certificats de serveur TLS, il est exigé qu'au moins une extension `subjectAltName` corresponde au dNSName (par exemple FQDN) ou à l'IPAdress du serveur TLS présentant le certificat.

Si le certificat authentifiant un serveur TLS ne contient pas une extension `subjectAltName`, un événement de sécurité ("erreur: nom alternatif du sujet non inclus") doit être déclenché.

#### 7.4.4.10.5 Basic Constraints

L'extension `BasicConstraints` permet d'identifier si le certificat est un certificat de CA. Si le certificat est un certificat de CA, l'extension doit être marquée comme étant critique. Pour les entités finales, il est recommandé d'omettre cette extension.

Pour les CA émettrices, deux composants doivent être fournis:

- le composant `CA` doit être défini sur true. Il est à noter que cela exige également que l'utilisation de la clé soit mise à `keyCertSign` (voir 7.4.4.10.6);
- le composant `pathLenConstraint` peut être utilisé pour limiter la longueur du chemin de certification, en fonction de la politique de sécurité de l'organisation.

Si l'extension `BasicConstraints` n'est pas fournie dans un certificat de CA, un événement de sécurité ("erreur: contraintes de base non incluses dans le certificat de CA") doit être déclenché.

Si le composant `pathLenConstraint` est fourni et utilisé dans un certificat de CA sur le chemin de certification et est mis à zéro, le certificat suivant doit être un certificat d'entité finale. Si le certificat suivant n'est pas un certificat d'entité finale, un événement de sécurité ("erreur: contraintes de longueur de chemin dans le certificat de CA non respectées") doit être déclenché.

Les mises en œuvre conformes au présent document doivent prendre en charge un `pathLenConstraint` minimal de 2, autorisant deux CA intermédiaires.

#### 7.4.4.10.6 Key Usage

L'extension `keyUsage` est une extension critique et elle doit être vérifiée. Elle identifie l'utilisation prévue du certificat de clé publique, comme `digitalSignature`,`keyAgrement` ou `keyEncipherment`.

keyUsage dans le contexte de TLS:

- `digitalSignature`
  - doit être définie pour les certificats de clients TLS; et
  - doit être définie pour les certificats de serveurs TLS si des suites de chiffrement basées sur la signature sont utilisées.

  Si le certificat authentifiant un client TLS ou un serveur TLS est utilisé conjointement avec une suite de chiffrement exigeant des signatures numériques dans le protocole de transfert TLS pour un accord de clé ne contient pas l'utilisation de clé pour `digitalSignature`, un événement de sécurité ("erreur: utilisation de clé pour les signatures numériques non incluse") doit être déclenché;
- `keyEncipherment` doit être défini pour les certificats de serveurs TLS si des suites de chiffrement basées sur le transport de clé sont utilisées.

  Si le certificat authentifiant un serveur TLS est utilisé conjointement avec une suite de chiffrement utilisant le chiffrement de clé publique dans le protocole de transfert TLS pour un accord de clé ne contient pas l'utilisation de clé pour `keyEncipherment`, un événement de sécurité ("erreur: utilisation de clé pour le chiffrement de clé non incluse") doit être déclenché;

Key usage dans le contexte des opérations de PKI:

- `keyCertSign` doit être défini pour émettre des certificats pour une CA émettrice. Si le certificat émetteur ne contient pas d'informations `keyCertSign`, un événement de sécurité ("erreur: utilisation de clé non incluse pour la signature des certificats") doit être déclenché;

- `cRLSign` doit être défini pour la signature de CRL. Si le certificat utilisé pour signer une CRL ne contient pas d'informations d'utilisation `cRLSign`, un événement de sécurité ("erreur: utilisation de clé non incluse pour la signature de CRL") doit être déclenché;
- `digitalSignature` doit être défini pour les AA émettrices, si l'AA certifiée a été émise par une CA. Il doit également être défini pour les certificats CA utilisés dans le contexte du SCEP, car dans ce contexte les messages de réponse sont signés par la CA du SCEP.

#### 7.4.4.10.7 Extended Key Usage

Si elle est disponible, l'extension `extendedKeyUsage` doit être vérifiée. Elle identifie des finalités supplémentaires pour l'utilisation du certificat de clé publique (la prise en charge de ce composant est facultative, selon le cas d'utilisation).

extendedKeyUsage dans le contexte des certificats TLS:

- `ServerAuth` doit être défini pour les certificats de serveurs TLS. Si les informations `serverAuth` ne sont pas contenues dans un certificat de serveur TLS, un événement de sécurité ("erreur: utilisation de clé étendue non incluse pour le serveur TLS") doit être déclenché;
- `clientAuth` doit être défini pour les certificats de clients TLS. Si les informations `clientAuth` ne sont pas contenues dans un certificat de client TLS, un événement de sécurité ("erreur: utilisation de clé étendue non incluse pour le client TLS") doit être déclenché.

Extended key usage dans le contexte des opérations de PKI:

- `OCSPSigning` doit être défini pour la signature des réponses (1.3.6.1.5.5.7.3.9 – OCSPSigning). Si les informations `OCSPSigning` ne sont pas contenues dans un certificat utilisé pour signer des réponses OCSP, un événement de sécurité ("erreur: utilisation de clé étendue non incluse pour la signature OCSP") doit être déclenché.

Extended key usage dans le contexte des opérations du système:

- AVL Signing (`avlSign`): Si elle est définie, la clé privée correspondante peut être utilisée pour signer des AVL. L'agent d'autorisation utilise sa clé privée pour signer une AVL à soumettre à une entité d'AVL. Le certificat de clé publique correspondant doit comprendre l'extension d'utilisation de clé étendue avec la valeur id-`avlSign` (= 2.5.38.2). Cette extension d'utilisation de clé étendue et la valeur `id-avlSign` sont définies en 9.2.2.4 de l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019). Si les informations `id-avlSign` ne sont pas contenues dans le certificat utilisé pour signer une AVL, un événement de sécurité ("erreur: utilisation de clé étendue non incluse pour la signature d'une AVL") doit être déclenché.

#### 7.4.4.10.8 Extension d'autorisation et de validation

L'extension d'autorisation et de validation est spécifiée en 9.2.2.8 de l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019). La présence de cette extension dans un certificat de clé publique indique que ce certificat de clé publique doit uniquement être considéré comme étant valide s'il peut être vérifié avec succès par rapport à une AVL particulière. Le traitement d'une AVL relève de la responsabilité d'un opérateur.

Si cette extension est utilisée, elle doit toujours être indiquée comme étant critique.

Il convient que cette extension soit uniquement utilisée si toutes les entités devant valider ce certificat de clé publique sont gérées par le même agent d'autorisation (cette restriction n'est pas mentionnée dans l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019)).

#### 7.4.4.10.9 CRL Distribution Point

L'extension `cRLDistributionPoints` identifie le point de distribution de CRL auquel la partie utilisatrice peut accéder à la liste de distribution de CRL. La récupération de la CRL est spécifiée par le protocole indiqué dans la liste de distribution de CRL et il peut s'agir de HTTP ou LDAP dans le contexte de la présente spécification.

Les CA émettrices conformes au présent document doivent fournir des informations sur les points de distribution de CRL dans cette extension. Les mises en œuvre d'entités finales conformes au présent document prenant en charge des CRL doivent pouvoir utiliser les informations contenues pour récupérer les informations de révocation à partir d'un point de distribution de CRL.

Les mises en œuvre (CA émettrice et entité finale prenant en charge des CRL) conformes au présent document doivent prendre en charge la récupération des CRL en utilisant HTTP comme protocole de transport. La méthode par défaut est HTTP GET.

Il est à noter que le protocole LDAP facultatif peut être utilisé dans les environnements qui appliquent déjà LDAP pour les opérations liées aux certificats. Tel peut être le cas lorsque le RBAC utilisant le modèle PULL pour le profil A ou B défini dans l'IEC 62351-8 est pris en charge. Si la récupération de CRL est prise en charge par le protocole d'enregistrement utilisé et la PKI des opérateurs, elle peut également être utilisée.

NOTE La CRL elle-même est autonome (signée numériquement) et peut être transmise indépendamment de la sécurité du transport.

Si des CRL sont utilisées pour les contrôles de révocation de certificat par la partie utilisatrice et si les informations sur le point de distribution de CRL (CDP) ne sont pas contenues dans le certificat reçu, un événement de sécurité ("avertissement: point de distribution de CRL non contenu dans le certificat de clé publique") doit être déclenché.

#### 7.4.4.10.10 Authority Information Access

L'extension `authorityInformationAccess` indique la méthode d'accès aux informations et services de l'émetteur du certificat. Plus précisément, cette extension peut contenir des informations d'accès pour un répondeur OCSP chargé de fournir des informations sur l'état de révocation du certificat contenant l'extension. Dans le contexte de la présente spécification, les informations de support pour le répondeur OCSP sont exigées comme spécifié dans la RFC 5280.

Les CA émettrices conformes au présent document doivent fournir des informations sur le point de distribution OCSP indiqué par la méthode d'accès `id-ad-ocsp`.

Les mises en œuvre d'entités finales conformes au présent document prenant en charge l'OCSP doivent pouvoir utiliser les informations contenues pour récupérer les informations de révocation d'un répondeur OCSP en utilisant l'OCSP.

Si les informations du répondeur OCSP ne sont pas contenues dans le certificat reçu, un événement de sécurité ("avertissement: informations de répondeur OCSP non contenues dans le certificat de clé publique") doit être déclenché.

#### 7.4.4.10.11 Extension RBAC

L'IEC 62351-8 définit l'extension `IECUserRoles` pour le transport d'informations liées au sujet concernant le rôle que le sujet peut jouer, ainsi que des attributs supplémentaires fournissant plus d'informations sur l'utilisation du rôle, comme la zone de responsabilité. Voir IEC 62351-8, profil A, pour la définition de l'extension, ainsi que la validation.

#### 7.4.4.10.12 Extension de point de distribution d'AVL

L'utilisation de l'extension de certificat de clé publique définie dans l'Édition 1 est déconseillée. Les AVL sont un problème local concernant l'opérateur et peuvent être inconnues de la CA émettrice. Par conséquent, le traitement d'un point de distribution d'AVL est un problème de mise en œuvre locale et peut être effectué par exemple par le biais de la configuration. Les protocoles de gestion des AVL sont indiqués dans l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019).

#### 7.4.4.10.13 Extension d'identificateur SOA

L'extension `sOAIdentifier` est décrite dans l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019) et indique que le sujet du certificat de clé publique peut agir comme une source d'autorité (SOA). En tant que tel, le sujet du certificat de clé publique peut utiliser la clé privée pour émettre des certificats d'attribut qui confèrent des privilèges aux détenteurs.

La définition de cette extension relève de la responsabilité de la CA émettrice. Si la CA émet des certificats d'AA, cette extension doit être incluse. Elle permet d'établir un lien étroit entre la CA et l'AA.

En plus de l'extension `sOAIdentifier`, une AA a également besoin de la `digitalSignature` de l'utilisation de clé attribuée pour émettre des certificats d'attributs.

Dans le contexte de la présente spécification, seule une CA est autorisée à émettre un certificat d'AA incluant l'identificateur SOA.

### 7.4.5 Composants des certificats d'attributs

#### 7.4.5.1 Généralités

Les composants des certificats d'attributs sont basés sur le contenu du Tableau 2 du paragraphe 7.2.2. Si un certificat d'attribut ou l'un de ses composants ne peut pas être vérifié, le détenteur doit être traité comme n'ayant aucun rôle spécifique.

#### 7.4.5.2 Version

Le composant `version` du certificat d'attribut doit être 1 (indiquant v2).

Si la version de certificat ne correspond pas à la version attendue, un événement de sécurité ("erreur: mauvaise version de certificat d'attribut") doit être déclenché.

#### 7.4.5.3 Holder

Le composant `holder` doit transmettre l'identité du détenteur du certificat d'attribut. Conformément à l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019), différents composants sont possibles ici. Les mises en œuvre conformes au présent document doivent prendre en charge l'évaluation:

- du composant `baseCertificateID`. S'il est présent, il identifie un certificat de clé publique particulier utilisé pour authentifier l'identité du détenteur lors de l'assertion de privilèges avec ce certificat d'attribut sur la base du nom distinctif du `issuer` et du `serialnumber` du certificat de clé publique du détenteur. D'après la RFC 5755, le certificat de clé publique doit avoir un nom distinctif d'émetteur non vide, qui doit être présent dans le champ `directoryName` du composant `holder.baseCertificateID.issuer`;
- du composant `entityName`. S'il est présent, l'authentification du détenteur est effectuée par d'autres moyens qu'un certificat de clé publique.

NOTE La RFC 5755 permet également l'identification du détenteur d'un certificat de clé publique sur la base du composant `subject` ou de l'extension `subjectAltName` qui doit être contenu dans l'`entityName`. L'approche

adoptée dans le présent document est plus stricte et simplifie la distinction entre l'authentification du détenteur basée sur un certificat de clé publique et les autres types d'authentifications de détenteur.

- les attributs de l'`entityName` sont définis par un opérateur. Il est à noter que, conformément à l'IEC 62351-8, le détenteur peut se rapporter à un utilisateur humain ou à un utilisateur technique (par exemple une application). Une mise en œuvre doit être prête à recevoir et traiter au moins l'un des attributs suivants dans l'`entityName`:
  - `otherName`: nom alternatif du détenteur de toute forme identifiée par un OID et une valeur correspondante,
  - `directoryName`: nom distinctif du détenteur selon la Rec. UIT-T X.501 | l'ISO/IEC 9594-2.

D'après la recommandation formulée dans la RFC 5755, la présente spécification exige qu'un seul des composants doive être inclus dans un certificat d'attribut. De plus, en cas d'utilisation de l'`entityName`, un seul attribut pour ce composant doit être fourni. Ceci permet d'éviter toute confusion concernant le composant à traiter comme normatif.

Si l'authentification du détenteur est effectuée sur la base d'un certificat de clé publique correspondant, le certificat doit être vérifié comme décrit en 7.4.4.

Si le détenteur n'a pas pu être vérifié sur la base d'un certificat de clé publique correspondant, un événement de sécurité ("avertissement: le détenteur du certificat d'attribut ne peut pas être vérifié sur la base d'un certificat de clé publique") doit être déclenché. Dans ce cas, le détenteur doit être traité comme n'ayant aucun rôle spécifique.

Si l'authentification du détenteur est réalisée par d'autres moyens, le processus d'authentification doit être arrivé à terme avec succès avant de vérifier le certificat d'attribut.

Si le détenteur n'a pas pu être vérifié sur la base d'une authentification alternative non basée sur un certificat de clé publique, un événement de sécurité ("avertissement: le détenteur du certificat d'attribut n'a pas pu être vérifié sur la base d'une authentification alternative") doit être déclenché. Dans ce cas, le détenteur doit être traité comme n'ayant aucun rôle spécifique.

#### 7.4.5.4 Issuer

Le composant `issuer` identifie l'AA qui a émis le certificat d'attribut. Il contient le nom distinctif de l'AA émettrice. D'après la RFC 5755, il ne doit contenir qu'un seul nom distinctif dans le champ `directoryName` du composant `issuer`.

Selon la RFC 5755, une AA peut être déclarée de confiance par configuration ou par d'autres moyens.

Si l'émetteur ne correspond pas à un émetteur de confiance connu et configuré, un événement de sécurité ["erreur: émetteur de certificat d'attribut non digne de confiance (non configuré)"] doit être déclenché.

Dans le contexte de la présente spécification, les "autres moyens" sont fournis par une intégration étroite de la CA émettrice et de la CA émettrice d'un opérateur. La CA peut, dans ce cas, émettre des certificats d'AA. Le vérificateur d'un certificat d'attribut doit vérifier que le certificat d'AA émettrice:

- contient le `sOAIdentifier` (voir aussi en 7.4.4.10.13);
- contient la `digitalSignature` de l'utilisation de clé (voir aussi 7.4.4.10.6);
- a été émis par une CA de confiance.

Si l'AA ne peut pas être vérifiée en tant qu'émetteur de confiance sur la base de la vérification des extensions et de l'émetteur du certificat d'AA, un événement de sécurité ("erreur: émetteur de certificat d'attribut non digne de confiance (par une CA de confiance)") doit être déclenché.

#### 7.4.5.5 Signature

Le composant `signature` contient l'identificateur d'algorithme utilisé pour valider la signature du certificat d'attribut. Pour les certificats d'attribut, les mêmes conditions que pour les certificats de clé publique s'appliquent (voir 7.4.4.4).

#### 7.4.5.6 Attributes

Le composant `attributes` contient les attributs associés au détenteur.

Pour le système de puissance, l'IEC 62351-8 définit la valeur et le type d'attribut pour `IECUserRoles` pour le transport d'informations liées au sujet concernant le rôle que le sujet peut jouer, ainsi que des attributs supplémentaires fournissant plus d'informations sur l'utilisation du rôle, comme la zone de responsabilité. Voir IEC 62351-8, profil B, pour la définition de l'attribut, ainsi que la validation.

#### 7.4.5.7 attrCertValidityPeriod

La période de validité d'un certificat d'attribut dépend de la politique de sécurité de l'organisation. La période de validité est déterminée par les valeurs *not before* et *not after*. La partie utilisatrice doit vérifier que la date et l'heure actuelles sont comprises entre les valeurs `notBeforeTime` et `notAfterTime` du certificat.

- si l'heure actuelle est postérieure à la valeur `notAfterTime`, un événement de sécurité ("erreur: certificat d'attribut expiré") doit être déclenché;
- si l'heure actuelle est antérieure à la valeur `notBeforeTime`, un événement de sécurité ("erreur: certificat d'attribut non encore valide") doit être déclenché.

Contrairement à la RFC 5280, la RFC 5775 relative aux certificats d'attribut exige que les valeurs pour `not before` et `not after` doivent toujours être spécifiées en `GeneralizedTime`.

NOTE Pour indiquer qu'un certificat n'a pas de date d'expiration bien définie, la RFC 5280 utilise la valeur `GeneralizedTime` de 99991231235959 comme valeur `notAfter`.

#### 7.4.5.8 Extensions de certificats d'attributs

##### 7.4.5.8.1 Généralités

Les extensions définies pour les CA de certificats d'attributs fournissent des méthodes pour associer des attributs supplémentaires à des détenteurs.

Si le certificat d'attribut contient une extension non reconnue marquée comme étant critique, un événement de sécurité ("erreur: extension critique inconnue dans le certificat d'attribut") doit être déclenché.

Si le certificat contient des informations non reconnues dans une extension marquée comme étant critique, un événement de sécurité ("erreur: informations non reconnues dans l'extension critique du certificat d'attribut") doit être déclenché.

Si le certificat contient une extension non reconnue, un événement de sécurité ("avertissement: extension inconnue dans le certificat d'attribut") doit être déclenché.

##### 7.4.5.8.2 Authority Key Identifier

Le composant `authorityKeyIdentifier` peut être inclus dans un certificat d'attribut afin d'aider le vérificateur de certificat d'attribut à contrôler la signature du certificat d'attribut par une AA émettrice.

Si l'`authorityKeyIdentifier` est contenu dans le certificat d'attribut sans qu'il soit possible de le vérifier, un événement de sécurité ("erreur: l'identificateur de clé d'autorité dans le certificat d'attributs n'a pas pu être vérifié ") doit être déclenché.

#### 7.4.5.8.3 Traitement de la révocation des certificats d'attributs

Les certificats d'attributs sont destinés à améliorer temporairement les attributs d'un détenteur. Ils peuvent être émis avec une courte période de validité.

Si la politique de sécurité de l'organisation permet une courte période de validité n'exigeant pas de révocation, cela doit être indiqué en utilisant l'extension `noRevAvail` (pas de révocation disponible). Dans ce cas, les extensions CRL Distribution Point et Authority Information Access ne doivent pas être définies.

Si les informations `noRevAvail` sont contenues dans le certificat d'attribut reçu, un événement de sécurité ("avis: aucune information de révocation prévue dans le certificat d'attribut") doit être déclenché.

La même approche que pour les certificats de clé publique est appliquée pour le traitement des extensions CRL Distribution Point (voir 7.4.4.10.9) et Authority Information Access (voir 7.4.4.10.10), mais en se référant à l'AA au lieu de la CA.

Si la politique de sécurité de l'organisation exige des informations de révocation, l'AA émettrice conforme au présent document doit fournir des informations sur le point de distribution de CRL dans l'extension `cRLDistributionPoints` et des informations concernant le point de distribution d'OCSP dans l'extension `authorityInformationAccess`.

Les mises en œuvre d'entités finales conformes au présent document prenant en charge des CRL doivent pouvoir utiliser les informations contenues pour récupérer les informations de révocation à partir d'un point de distribution de CRL.

Les mises en œuvre (CA émettrice et entité finale prenant en charge des CRL) conformes au présent document doivent prendre en charge la récupération des CRL en utilisant HTTP comme protocole de transport. La méthode par défaut est HTTP GET.

Il est à noter que le protocole LDAP facultatif peut être utilisé dans les environnements qui appliquent déjà LDAP pour les opérations liées aux certificats. Tel peut être le cas lorsque le RBAC utilisant le modèle PULL pour le profil A ou B défini dans l'IEC 62351-8 est pris en charge.

NOTE La CRL elle-même est autonome (signée numériquement) et peut être transmise indépendamment de la sécurité du transport.

Si des CRL sont utilisées pour les contrôles de révocation de certificat par la partie utilisatrice et si les informations sur le point de distribution de CRL (CRLDP) ne sont pas contenues dans le certificat reçu, un événement de sécurité ("avertissement: point de distribution non contenu dans le certificat d'attribut") doit être déclenché.

Les mises en œuvre d'entités finales conformes au présent document prenant en charge l'OCSP doivent pouvoir utiliser les informations contenues pour récupérer les informations de révocation d'un répondeur OCSP en utilisant l'OCSP.

Si les informations du répondeur OCSP ne sont pas contenues dans le certificat d'attribut reçu, un événement de sécurité ("avertissement: informations de répondeur OCSP non contenues dans le certificat d'attribut") doit être déclenché.

### 7.4.6 Vérification de l'état de révocation des certificats

Une mise en œuvre revendiquant la conformité au présent document doit pouvoir contrôler l'état de révocation d'un certificat reçu. Si le contrôle de la révocation du certificat est positif, un événement de sécurité ("alarme: le certificat a été révoqué") doit être déclenché. Si elle est disponible, la raison de la révocation peut être ajoutée dans l'événement de sécurité. Les codes de raison possibles tels que définis dans l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019) sont:

- `unspecified (0;`
- `keyCompromise (1;`
- `cACompromise (2);`
- `affiliationChanged (3);`
- `superseded (4);`
- `cessationOfOperation (5);`
- `certificateHold (6);`
- `removeFromCRL (8);`
- `privilegeWithdrawn (9);`
- `aACompromise (10);`
- `weakAlgorithmOrKey (11).`

La fourniture de la CRL est une question locale et elle peut être effectuée localement (sur fichier) ou en fonction des informations URI du composant CRL Distribution Point dans le certificat et peut utiliser HTTP ou LDAP comme indiqué en 7.4.4.10.9.

L'indisponibilité d'une CRL doit déclencher un événement de sécurité ("avertissement: point de distribution de CRL inaccessible").

L'expiration d'une CRL doit déclencher un événement de sécurité ("avertissement: la CRL a expiré").

L'échec de validation de la signature d'un CRL doit déclencher un événement de sécurité ("avertissement: la signature des CRL n'a pas pu être vérifiée").

En variante, une entité peut utiliser le protocole d'état de certificat en ligne (OCSP) pour vérifier l'état de révocation d'un certificat.

Si l'OCSP est utilisé pour acquérir une réponse OCSP pour un certificat reçu afin d'effectuer un contrôle de révocation de certificat, l'entité utilisatrice doit utiliser l'URI trouvé dans le champ `accessLocation` de la `id-ad-ocsp accessMethod` de l'extension Authority Info Access dans le certificat homologue afin d'interroger le répondeur OCSP. Différents cas d'erreur peuvent se produire:

- l'inaccessibilité du répondeur OCSP à une entité doit déclencher un événement de sécurité: ("avertissement: répondeur OCSP inaccessible");
- un délai de réponse pour une connexion de l'entité à un répondeur OCSP doit déclencher un événement de sécurité: ("avertissement: délai de connexion au répondeur OCSP"). Il est à noter que cet événement peut se rapporter à un délai TCP/IP normal ou qu'il peut être causé par une attaque par déni de service;
- si le répondeur OCSP ne reconnaît pas le certificat dans les messages de demande, il répond avec la valeur d'état de certificat "unknown" (inconnu). Cette condition doit déclencher un événement de sécurité ("avertissement: certificat non connu du répondeur OCSP"). Il est à noter que cet événement peut indiquer un émetteur non reconnu qui n'est pas desservi par ce répondeur.

L'expiration d'une réponse OCSP à une entité doit déclencher un événement de sécurité ("avertissement: réponse OCSP expirée").

L'échec de la validation de la signature d'un message de réponse OCSP doit déclencher un événement de sécurité ("avertissement: la signature d'une réponse OCSP n'a pas pu être vérifiée").

Comme indiqué en 7.4, la politique de sécurité d'une organisation détermine la réaction finale à un événement de sécurité. Les certificats révoqués ne doivent pas être acceptés par les entités, par exemple lors de l'établissement d'une session TLS, comme cela est exigé dans l'IEC 62351-3.

### 7.5 Révocation des certificats

Le certificat d'une entité doit être révoqué au moins dans les conditions suivantes, en utilisant les codes de raison spécifiés en 9.5.3.1 de l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019).

- la clé privée de l'entité a été compromise;
- la CA a été compromise;
- l'affiliation de l'entité a changé;
- l'entité est arrivée en fin de vie.

D'autres raisons justifiant la révocation d'un certificat peuvent être données par la politique de révocation de l'organisation. La révocation des certificats qui ont été renouvelés ou qui ont expiré n'est pas exigée.

La fourniture d'informations de révocation est une exigence relative à l'environnement PKI, et non d'une entité particulière.

L'infrastructure de clé publique (PKI) doit prendre en charge au moins les deux méthodes sécurisées de révocation suivantes:

- listes de révocation de certificats (CRL);
- protocole d'état de certificat en ligne (OCSP).

La PKI doit propager les informations de révocation au moins toutes les 24 heures (ce délai peut varier en fonction de la politique PKI de l'organisation, de la criticité des dispositifs particuliers et du nombre de connexions homologues).

Les entités doivent prendre en charge la CRL et/ou l'OCSP pour récupérer des informations de révocation. Les informations de révocation sont exigées pour effectuer des contrôles de vérification de certificat.

Il est à noter qu'il y a un compromis à trouver quant au choix de la méthode de révocation (CRL ou OCSP). Ce choix dépend du dispositif cible et de l'environnement cible.

- La taille des CRL peut augmenter et peut ne plus pouvoir être traitée par des IED contraints. Le traitement inclut la vérification initiale de la CRL, mais aussi le stockage des informations jusqu'à la prochaine mise à jour de la CRL. Une CRL est généralement récupérée toutes les 24 heures. Les fabricants peuvent déclarer une taille maximale d'une CRL prise en charge.
- Il est exigé que les IED prenant en charge l'OCSP traitent les réponses OSCP (ce qui inclut la vérification et la mise en mémoire cache) par certificat. Ce traitement peut exiger une communication plus fréquente avec un répondeur OCSP afin de récupérer des réponses OCSP pour les certificats reçus. Il est à noter que la mise en mémoire cache des réponses OCSP (traitée par la politique de sécurité d'une organisation) peut réduire la communication avec le répondeur OCSP, mais qu'elle peut également accroître les besoins de stockage local pour la mise en mémoire cache des réponses OCSP.

NOTE Il est prévu que le comportement du système en cas d'indisponibilité des CRL, des mises à jour de CRL ou des répondeurs OCSP soit défini dans une autre partie de l'IEC 62351 ou qu'il fasse partie intégrante de la politique de sécurité menée par l'organisation.

### 7.6 Expiration et renouvellement des certificats

Le présent document n'impose aucune durée de vie minimale ou maximale pour les certificats de clé publique. Il convient de choisir la date d'expiration du certificat en fonction du type de certificat et des politiques de sécurité locales (voir 7.3.4).

Les certificats de clé publique peuvent facultativement inclure une extension d'utilisation de clé privée, qui spécifie la période pendant laquelle la clé privée correspondante peut être utilisée par son propriétaire. Cette période est normalement définie pour être plus courte que la période de validité du certificat afin de s'assurer que les certificats restent valides pendant une période minimale après avoir été utilisés par leur propriétaire. Des informations détaillées sur l'extension de l'utilisation des clés privées sont fournies en 9.2.2.5 de l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019).

Les entités doivent générer une nouvelle paire de clés et effectuer une CSR dans les environnements PKI dès que les dates d'expiration de leurs certificats de clé publique se rapprochent de la date de fin de validité des certificats. Il est à noter que le traitement est généralement spécifié par les politiques de certification de l'organisation. Les entités doivent renouveler leurs certificats de clé publique avant leur expiration et doivent journaliser leurs actions de renouvellement de ces certificats (réussite ou échec).

Les entités doivent permettre de configurer la politique de renouvellement des certificats de clé publique, en indiquant notamment:

- si le renouvellement automatique est pris en charge ou pas;
- le délai avant l'expiration du renouvellement de certificat de clé publique.

### 7.7 Synchronisation et exactitude des horloges

La synchronisation et l'exactitude des horloges doivent être assurées pour permettre une vérification fiable des informations relatives à la validité du certificat.

La synchronisation temporelle est donc cruciale et il est fortement recommandé de mettre en œuvre les options de sécurité pour les protocoles de synchronisation temporelle:

- si PTP (IEEE 1588 v2.1) est utilisé pour la synchronisation temporelle, il convient d'utiliser l'option de sécurité intégrée. Il est à noter que pour l'automatisation des sous-stations basées sur l'IEC 61850, un profil PTP a été défini dans l'IEC 61850-9-3/IEEE c37.248. Il est également à noter que la sécurité sera incluse dans la prochaine révision de ce profil;
- si NTP est utilisé pour la synchronisation temporelle, il convient de prendre en compte l'application de sécurité du temps réseau (NTS, RFC 8915);
- en variante, d'autres protocoles de sécurité peuvent être utilisés pour assurer la synchronisation temporelle.

### 7.8 Listes d'autorisation et de validation

#### 7.8.1 Généralités

La prise en charge des AVL et des protocoles associés est facultative, mais si elle est utilisée, elle doit satisfaire aux exigences de l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019) et l'ISO/IEC 9594-11:2020 | la Rec. UIT-T X.510 (2020) comme suit:

- l'Article 11 de l'ISO/IEC 9594-08:2020 | la Rec. UIT-T X.509 (2019) spécifie les AVL;
- les Articles 8 à 12 de l'ISO/IEC 9594-11:2020 | la Rec. UIT-T X.510 (2020) spécifient un protocole-enveloppe qui assure la sécurité des autres protocoles;

- l'Article 13 de l'ISO/IEC 9594-11:2020 | la Rec. UIT-T X.510 (2020) spécifie le protocole de gestion des autorisations et des validations (AVMP), qui est utilisé pour la communication entre un agent d'autorisation et les entités d'AVL qu'il prend en charge. Il utilise les services du protocole-enveloppe;
- l'Article 14 de l'ISO/IEC 9594-11:2020 | la Rec. UIT-T X.510 (2020) spécifie le protocole d'abonnement à une CA (CASP). Ce protocole est utilisé par un agent d'autorisation afin de s'abonner aux informations d'état des certificats de clé publique provenant des CA pertinentes. Il est uniquement utilisé par l'agent d'autorisation pour les environnements contraints (voir 5.11.3). Ce protocole peut également être utilisé par les entités finales comme méthode alternative pour obtenir des informations d'état de clé publique. Il exige que les CA soient mises à jour afin de prendre en charge le protocole CASP et les capacités associées.

### 7.8.2 Syntaxe des listes d'autorisation et de validation (AVL) pour les certificats de clé publique

Les différents composants du type de données `TBSCertAVL` sont décrits dans l'Article 11 de l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019). Seuls les aspects spécifiques aux AVL pertinents pour les systèmes de puissance sont décrits ci-dessous:

- le composant `serialNumber` permet d'obtenir des fonctionnalités plus avancées en laissant un agent d'autorisation placer plusieurs AVL dans une entité d'AVL. Il est également possible pour plusieurs agents d'autorisation de placer une ou plusieurs AVL dans la même entité d'AVL;
- le composant `constrained` doit prendre la valeur **FALSE** pour indiquer que l'entité d'AVL est une entité d'AVL non contrainte. Il convient de définir cette valeur jusqu'au moment où les CA prennent en charge l'abonnement à l'état du certificat de clé publique;
- le composant `entries` doit contenir un élément pour chaque certificat de clé publique et/ou groupe d'entités représenté par l'AVL. Chaque élément doit être spécifié comme suit.

Le composant `idType` doit choisir l'une des deux alternatives suivantes:

- si l'alternative `certIdentifier` est choisie, il doit identifier le certificat de clé publique particulier représenté par cette entrée. Cette identification peut être effectuée en spécifiant le nom d'émetteur de la CA, ainsi que le numéro de série du certificat de clé publique, ou par une empreinte digitale du certificat de clé publique ou uniquement de la clé publique. Si le certificat de clé publique est un certificat autosigné, seule une empreinte digitale peut être utilisée pour identifier le certificat de clé publique;
- si l'alternative `entityGroup` est choisie, il doit contenir la partie d'un nom distinctif qui est commune à toutes les entités dans le groupe. Cette alternative est uniquement pertinente pour un environnement non contraint.

Les extensions supplémentaires de l'AVL sont les suivantes:

- le composant `entryExtensions`, le cas échéant, doit contenir une ou plusieurs extensions spécifiques pour l'entrée concernée. Les extensions spécifiées de 7.8.3 à 7.8.6 peuvent y être incluses. Si un type d'extension particulier est utilisé comme extension d'entrée, il ne doit pas être inclus dans le composant `avlExtensions`;
- le composant `avlExtensions`, lorsqu'il est présent, doit contenir une ou plusieurs extensions applicables aux entrées dans l'AVL. Les extensions spécifiées de 7.8.3 à 7.8.6 peuvent y être incluses. Si un type d'extension particulier est inclus, il ne doit pas être présent dans le composant `entryExtension` d'une quelconque entrée.

Il est à noter que des extensions peuvent être ajoutées aux entrées individuelles et à l'AVL dans son ensemble. Ces extensions sont spécifiées de 7.8.3 à 7.8.6.

### 7.8.3 Restriction du domaine d'application des AVL

Le domaine d'application des AVL est destiné à limiter l'applicabilité des certificats de clé publique. Dans certains scénarios, il peut être souhaitable d'inclure le domaine d'application dans un certificat de clé publique, cette extension étant facultative. Les contraintes réelles du domaine d'application sont définies dans un type séparé. Cela permet d'utiliser également la contrainte de manière séparée.

La syntaxe de l'extension **scopeConstraints** est définie comme suit:

```
scopeConstraints EXTENSION::= {
  SYNTAX        ScopeConstraints
  IDENTIFIED BY { avl62351Extion 1 } }

ScopeConstraints =:: SEQUENCE Of (SIZE (1..MAX)) OF ScopeConstraint

ScopeConstraint::= SEQUENCE { - contains the scope information
  aor       UTF8String (SIZE(1..64)),
  --Def. of "Area of Responsibility" of CA cert
  revision  INTEGER (0..255) OPTIONAL
  - optional revision if aor changes
}
```

La zone de responsabilité (**aor**) limite l'applicabilité d'un certificat de clé publique à une certaine zone géographique ou organisationnelle. Le présent document définit le champ et le format de la zone de responsabilité **aor** comme suit:

| Nom du champ | Codage, longueur max. (octet) | Exemple |
|---|---|---|
| Area of responsibility | UTF8, 64 | BAVARIA.DE |

L'**aor** est un identificateur et il définit un espace de noms hiérarchique ou une référence à l'espace de noms. Il est à noter que ces identificateurs sont généralement alphanumériques. L'**aor** est fournie en fonction de la politique (par exemple par l'opérateur responsable).

NOTE La notion d'**aor** (ou de domaine d'application) visant à décrire une restriction géographique ou organisationnelle a déjà été introduite dans l'IEC 62351-8 et est réutilisée dans le présent document.

### 7.8.4 Extension de restriction de protocole d'AVL

L'extension d'entrée de restriction de protocole d'AVL est définie en 9.7.2 de l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019). Elle est utilisée pour énumérer les protocoles associés à utiliser lors de la communication avec l'entité (ou les entités avec **entityGroup**). Si cette extension d'entrée d'AVL est omise, il n'y a aucune restriction concernant les types de protocoles à employer.

Un protocole doit être identifié par la norme de communication spécifiant ce protocole. L'identificateur d'objet doit respecter le principe spécifié en A.2 et A.4 de l'ISO/IEC 9834-1 | la Rec. UIT-T X.660.

Par exemple, l'IEC 61850-8-1 est représentée par l'identificateur d'objet:

{ iso(1) standard(0) iec61850(61850) series(8) part(1) }

NOTE L'ISO/IEC 9834-1 | la Rec. UIT-T X.660 ne prend pas en compte le cas où le numéro de norme est constitué de trois éléments. Cet exemple reprend la notation utilisée par la présente partie de l'IEC 62351.

L'utilisation des identificateurs d'objets ci-dessous, définis dans l'Édition 1 du présent document, est déconseillée. Pour la rétrocompatibilité, il convient que les mises en œuvre soient en mesure de gérer ces identificateurs d'objet.

```
id-P-IEC61850-T    OBJECT_IDENTIFIER::= { id-IEC62351prot 1 }
```

```
id-P-IEC61850-A   OBJECT_IDENTIFIER::= { id-IEC62351prot 2 }
id-P-IEC60870-5-T OBJECT_IDENTIFIER::= { id-IEC62351prot 3 }
id-P-IEC60870-5-A OBJECT_IDENTIFIER::= { id-IEC62351prot 4 }
id-P-IEC62325-T   OBJECT_IDENTIFIER::= { id-IEC62351prot 5 }
id-P-IEC62325-A   OBJECT_IDENTIFIER::= { id-IEC62351prot 6 }
id-P-IEEE1518-T   OBJECT_IDENTIFIER::= { id-IEC62351prot 7 }
id-P-IEEE1518-A   OBJECT_IDENTIFIER::= { id-IEC62351prot 8 }
```

### 7.8.5 Marquage AVL du certificat et de l'identificateur associé

Cette extension d'entrée d'AVL **pinningId** associe un identificateur distinct à un certificat de clé publique. Cet identificateur est très probablement l'adresse IP, mais il peut également s'agir d'un autre identificateur. Il existe deux cas dans lesquels un certificat de clé publique n'est à utiliser que sur une adresse IP dédiée. L'extension **pinningId** prend en charge l'association d'un certificat à l'adresse IP (ou à un autre identificateur) si le certificat lui-même ne donne pas d'informations relatives à l'adresse IP dans l'extension **subjectAltName**.

```
pinningId EXTENSION::= {
  SYNTAX        PinningId
  IDENTIFIED BY { avl62351Extion 2 } }

PinningId::= GeneralNames

GeneralNames::= SEQUENCE SIZE (1..MAX) OF GeneralName

GeneralName::= CHOICE {
  otherName                   [0]  OtherName,
  rfc822Name                  [1]  IA5String,
  dNSName                     [2]  IA5String,
  x400Address                 [3]  ORAddress,
  directoryName               [4]  Name,
  ediPartyName                [5]  EDIPartyName,
  uniformResourceIdentifier   [6]  IA5String,
  iPAddress                   [7]  OCTET STRING,
  registeredID                [8]  OBJECT IDENTIFIER,
  ... }
```

Le type de données **GeneralNames** est défini et les alternatives de **GeneralName** sont décrites en 9.3.2.1 de l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019). Le type de données **GeneralNames** est copié dans la présente partie pour faciliter la consultation.

L'**iPAddress** doit être stockée dans la chaîne d'octets de "network byte order" (ordre d'octet du réseau), comme spécifié dans la RFC 791. Le bit de poids faible (LSB) de chaque octet est celui de l'octet correspondant dans l'adresse de réseau. Pour l'IPv4 (Internet Protocol version 4), spécifiée dans la RFC 791, la chaîne d'octets doit contenir exactement quatre octets. Pour l'IPv6 (Internet Protocol version 6), spécifiée dans la RFC 2460, la chaîne d'octets doit contenir exactement seize octets.

### 7.8.6 Extensions des certificats de clé publique liées à l'utilisation d'AVL

En particulier si les AVL sont utilisées dans un contexte de PKI, l'utilisation de certaines extensions de certificats de clé publique est recommandée. Il est à noter que si des certificats autosignés sont utilisés, l'inclusion de ces extensions peut ne pas être réalisable dans la pratique. Les extensions de certificats de clé publique pertinentes pour les AVL sont décrites en 7.4, à savoir:

- l'extension d'autorisation et de validation (voir 7.4.4.10.8);
- l'utilisation de clé étendue pour émettre des AVL (voir 7.4.4.10.7).

Il est à noter que l'extension de certificat de clé publique définie dans l'Édition 1 de l'IEC 62351-9 (2017) pour un point de distribution d'AVL a été déconseillée comme indiqué en 7.4.4.10.12. Le point de distribution d'AVL est géré par l'opérateur et peut être inconnu de la CA émettrice. Si des AVL sont utilisées, ces informations peuvent être fournies par configuration.

### 7.8.7 Émission d'une AVL

L'émission d'une AVL relève de la responsabilité de l'opérateur (agent d'autorisation) d'un système de puissance dédié. L'AVL est censée être compilée en fonction des données techniques de déploiement d'un système particulier. En fonction de ces données techniques, les relations de communication sont connues, de même que les protocoles utilisés entre les entités qui communiquent. L'AVL peut être compilée simultanément, en partant de l'hypothèse selon laquelle les certificats de clé publique spécifiques à une entité sont déjà disponibles. Si des composants spécifiques à une entité ne sont pas disponibles au moment de l'ingénierie, la CA qui délivre les certificats peut être configurée avec les informations techniques et donner ces informations aux entités pendant l'enregistrement.

Il est à noter qu'une AVL nécessite d'être mise à jour si les homologues de communication changent dans le système. Tel peut être le cas si des composants du système sont retirés ou échangés, ou bien viennent d'être introduits. Il est présumé qu'au moins le retrait ou l'introduction de composants est accompagné d'une ingénierie.

L'Article 21 de l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019) spécifie les procédures de gestion des AVL.

### 7.8.8 Traitement de bout en bout des AVL

Le processus de vérification d'un certificat de clé publique reçu lors de l'établissement d'une connexion (par exemple lors du protocole de transfert TLS), de validation du certificat de clé publique lui-même et de contrôle par rapport à une AVL disponible en local, est exposé dans l'ISO/IEC 9594-8:2020 | la Rec. UIT-T X.509 (2019).

## 8 Gestion des clés de groupe (normative)

### 8.1 Exigences relatives au GDOI

Une présentation informative du GDOI est fournie en 5.6.4.2.

La méthode du domaine d'interprétation de groupe (GDOI) de la RFC 6407 et la RFC 8052 "Prise en charge du protocole d'interprétation de domaine de groupe (GDOI) pour services de sécurité IEC 62351)" doivent être utilisées pour la distribution de clés de groupe aux membres du groupe (GM).

La prise en charge de la méthode GROUPKEY-PULL est obligatoire, la méthode GROUPKEY-PUSH étant facultative.

Le GDOI définit l'application de GROUPKEY-PUSH pour envoyer des informations de contrôle aux membres du groupe. Cela concerne par exemple le renouvellement de clé de l'association de sécurité (SA) existante, mais aussi la modification des membres d'un groupe. Pour pouvoir acquitter les informations reçues d'un GCKS, le message d'acquittement GROUPKEY-PUSH a été spécifié dans la RFC 8263. L'aptitude à utiliser les accréditations du certificat de clé publique du client GDOI échangées au moment de l'établissement de la connexion pour l'authentification des membres du groupe est obligatoire.

Le centre de distribution de clés (KDC) doit garder en lieu sûr le référentiel des clés et les paramètres associés.

Le KDC doit configurer la politique de renouvellement de la clé de session, qui doit reposer sur la durée de vie de la clé.

Il est à noter que chaque fois que la clé publique est mentionnée (dans le GDOI), il est sous-entendu qu'elle fait partie intégrante du certificat. Elle a une clé privée correspondante, qui est utilisée pour générer des signatures.

## 8.2 Protocole d'échange de clés Internet version 1 (IKEv1)

La RFC 6407 relative au GDOI fournit une mise en œuvre d'ISAKMP dans laquelle le GDOI est un nouveau domaine d'interprétation (DOI) ISAKMP. La RFC 6407 définit la manière dont IKEv1 (RFC 2409) est utilisé comme protocole de phase 1 pour le GDOI. Le KDC doit utiliser IKEv1 pour son protocole GDOI de phase 1. Il n'est pas nécessaire que le KDC prenne en charge la totalité des fonctionnalités et fonctions d'IKEv1, mais il doit au moins prendre en charge les exigences IKEv1 énumérées dans le Tableau 3. Il est à noter que des algorithmes d'authentification supplémentaires permettant l'application de l'ECDSA pour IKEv1 et IKEv2 sont définis dans la RFC 4754 [32].

**Tableau 3 – Exigences IKEv1 du KDC**

| Description | Valeurs |
|---|---|
| Échanges ISAKMP pris en charge | 2 – Mode principal (protection d'ID)<br>5 – Informationnel |
| Types de charges utiles d'ID de GM pris en charge | 9 – ID_DER_ASN1_DN (ID d'objet du certificat d'identité du GM) |
| Échange de clé pris en charge | Diffie-Hellman par l'intermédiaire de l'attribut de description de groupe (voir ci-dessous) |
| Port IP du serveur | UDP 848 (configurable) |
| 1 – Attribut d'algorithme de chiffrement | 7 – AES-CBC (longueurs de clé: 128/256) |
| 2 – Attribut d'algorithme de hachage | 4 – SHA2-256<br>5 – SHA2-384<br>6 – SHA2-512 |
| 3 – Attribut de méthode d'authentification | 2 – Signatures DSA<br>3 – Signatures RSA (obligatoire)<br>9 – ECDSA avec SHA-256 sur la courbe P-256 (RFC 4754) |
| 4 – Attribut de description de groupe | 14 – MODP-2048<br>15 – MODP-3072<br>16 – MODP-4096<br>17 – MODP 6144 (RFC 3526)<br>18 – MODP-8192 (RFC 3526) |
| 11 – Type de vie (facultatif) | 1 – Secondes |
| 12 – Durée de vie (facultatif) | 120:86400 secondes (valeur par défaut: 120) |
| 14 – Longueur de clé | AES-CBC (longueur de clé: 128/256) |



NOTE MODP-1024 et MODP-1538, qui ont été mentionnés dans la version IEC 62351-9:2017, ont été déconseillés conformément aux recommandations du NIST et du BSI allemand. Les mises en œuvre conformes à la présente spécification et la mise en œuvre du GDOI doivent prendre en charge les algorithmes suivants énumérés dans le Tableau 3:

- chiffrement: AES-CBC-128 (également obligatoire pour la charge utile TEK IEC 61850); Si un bourrage doit être effectué pour correspondre à la longueur du bloc de l'algorithme, le bourrage PKCS #7 doit être utilisé;
- hachage: SHA2-256;
- authentification: signatures RSA-2048;
- groupe DH:
  - les mises en œuvre de serveur GDOI (KDC) conformes doivent prendre en charge les groupes DH 14 à 18. Le groupe DH 14, MODP-2048, n'est pas considéré comme suffisamment sécurisé conformément à la norme allemande BSI TR 02102-3 et à la NIST SP 800-56A, et il convient de l'utiliser uniquement pour les cas de rétrocompatibilité;
  - les mises en œuvre conformes du client GDOI doivent prendre en charge les groupes DH 16 à 18 et peuvent éventuellement prendre en charge les groupes DH 14 à 15.

Les autres algorithmes indiqués dans le Tableau 3 peuvent facultativement être pris en charge.

La détection de la proposition d'autres algorithmes que les algorithmes obligatoires ou facultatifs énumérés dans le Tableau 3 doit générer un événement de sécurité ("erreur: algorithme cryptographique non spécifié proposé dans IKEv1 de phase 1").

IKEv1 utilise une approche MAC-then-encrypt après que le secret DH a été établi et que les clés de session ont été calculées. Le chiffrement est indiqué dans les paragraphes suivants par la notation HDR* pour indiquer que les données après l'en-tête sont chiffrées. La détection d'erreurs de bourrage pendant le déchiffrement dans l'un des messages chiffrés doit déclencher un événement de sécurité ("erreur: erreur de bourrage pendant le déchiffrement du protocole de transfert IKEv1") et doit annuler le protocole de transfert GDOI.

## 8.3 Échange IKEv1 de phase 1 de type 2 en mode principal

### 8.3.1 Généralités

IKEv1 (RFC 2409) définit deux méthodes pour effectuer un échange de phase 1: "mode principal" et "mode agressif". La prise en charge de l'échange IKEv1 en mode principal est obligatoire, celle de l'échange IKEv1 en mode agressif étant interdite. L'échange IKEv1 en mode principal est une instanciation de l'échange de protection d'identité ISAKMP. L'échange IKEv1 en mode principal utilise l'échange ISAKMP de type 2.

IKEv1 spécifie quatre échanges en mode principal en fonction de la méthode d'authentification (attribut de SA des méthodes d'authentification IKE, valeur 3) fournie dans la charge utile SA initiale:

- authentification avec signatures numériques;
- authentification avec chiffrement à clé publique;
- méthode d'authentification révisée avec chiffrement à clé publique;
- authentification avec clé prépartagée.

La prise en charge de l'authentification avec des signatures numériques RSA avec clés RSA 2 048 bits est obligatoire conformément à la RFC 6407. La signature numérique RSA est la valeur 3 de l'attribut des méthodes d'authentification IKEv1. Outre le RSA, d'autres algorithmes de signature tels que DSA et ECDSA peuvent facultativement être pris en charge comme indiqué dans la section 5.3.6 de la RFC 6407. Il est à noter que d'un point de vue opérationnel, il est fortement recommandé de ne prendre en charge qu'un seul schéma de signature au sein d'un groupe ou d'un domaine de KDC afin de permettre le transfert unidiffusion et multidiffusion des messages GDOI GROUPKEY-PUSH. De plus, la prise en charge de différents schémas de signature dans un domaine peut exiger la prise en charge de plusieurs certificats par composant, ce qui augmente les coûts de gestion. L'authentification avec chiffrement à clé publique et l'authentification avec clé prépartagée sont facultatives et ne sont pas spécifiées dans le présent document.

L'échange en mode principal avec les signatures numériques est défini sur la Figure 19.

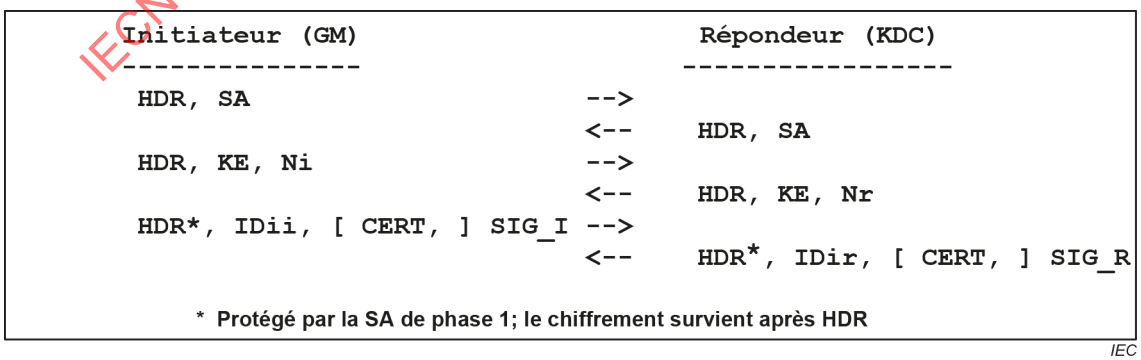



**Figure 19 – Échange IKEv1 en mode principal (RFC 2409) avec des signatures numériques RSA**

Comme indiqué sur la Figure 19, le membre du groupe est toujours à l'initiative de l'échange. Le centre de distribution de clés (KDC) est toujours le répondeur de l'échange. Les notations

(c'est-à-dire HDR, SA, KE, etc.) sont issues de la RFC 2409. Se reporter à la section 3[5] de la RFC 2409 pour les définitions des notations.

Les paragraphes 8.3.2 à 8.3.5 décrivent tous les messages et indiquent dans quelle mesure le KDC se distingue du protocole IKEv1 ou le limite.

### 8.3.2 Charge utile Certificate Request

Le GM et le KDC ne doivent pas utiliser la charge utile Certificate Request décrite dans l'ISAKMP (RFC 2408) et l'IKEv1 (RFC 2409). Cela est dû à l'exigence stipulée en 8.3.5.3 selon laquelle la charge utile Certificate doit toujours être envoyée dans le troisième protocole de transfert de l'échange de phase 1. Sur la base de cette exigence, il convient que ni le GM ni le KDC ne s'attendent à recevoir une charge utile Certificate Request dans le cadre de l'utilisation du GDOI conforme à l'IEC 62351-9.

### 8.3.3 Échange d'association de sécurité (1)

#### 8.3.3.1 Généralités

Les deux premiers messages échangés doivent être utilisés pour déterminer l'association de sécurité pour la SA IKE, comme représenté à la Figure 20. L'échange repéré en rouge indique l'étape actuellement discutée.

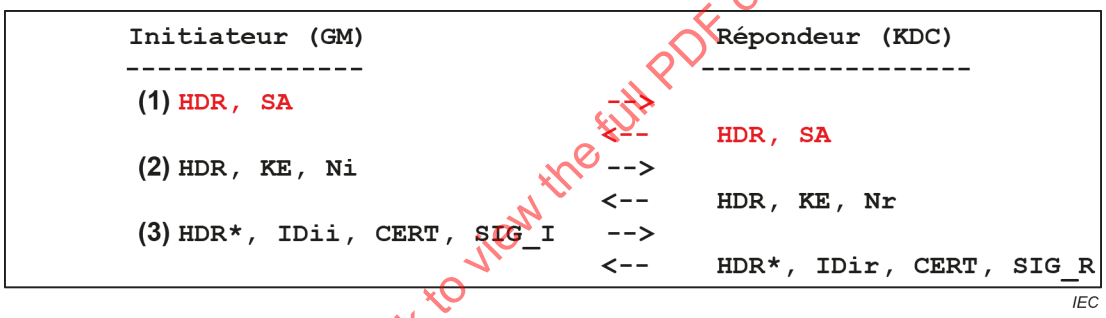



**Figure 20 – Échange IKEv1 en mode principal et messages d'association de sécurité**

Le GM doit envoyer une liste des transformations proposées dans l'ordre de priorité, et le KDC de clé doit choisir la première qui est compatible.

Le message initial envoyé par le GM doit contenir une charge utile SA, telle que définie dans la RFC 2409, section 5, Échanges. Une charge utile SA IKEv1 est définie comme étant une ou plusieurs charges utiles Transform intégrées dans une charge utile Proposal qui est elle-même intégrée dans une charge utile SA. Plusieurs charges utiles de proposition ne doivent pas être prises en charge.

#### 8.3.3.2 Charge utile SA

La charge utile Security Association (SA) doit être identique à celle de la RFC 2408, section 3.4. Le KDC doit prendre en charge le GDOI (RFC 6407, section 2.1) qui exige de définir le champ SA payload DOI de phase 1 sur GDOI (2). Le champ SA payload Situation est de 4 octets et doit être défini sur 0. Toutes les autres valeurs doivent générer une erreur.



5 http://tools.ietf.org/html/rfc2409#page-3.

#### 8.3.3.3 Charge utile Proposal

La charge utile Proposal doit être identique à celle de la RFC 2408, section 3.5. Le KDC attend une taille de SPI nulle, mais si elle ne l'est pas, le contenu du SPI doit être ignoré. Le KDC doit prendre en charge plusieurs transformations.

#### 8.3.3.4 Charge utile Transform

La charge utile Transform doit être identique à celle de la RFC 2408, section 3.6. Les attributs de SA exigés sont définis à la section 4 de la RFC 2409 et sont copiés ci-dessous:

- algorithme de chiffrement, valeur 1;
- longueur de clé, valeur 14. Exigé si l'algorithme de chiffrement ne spécifie pas de longueur de clé (c'est-à-dire AES_CBC);
- algorithme de hachage, valeur 2;
- méthode d'authentification, valeur 3;
- description de groupe, valeur 4.

Le KDC peut prendre en charge les attributs de SA facultatifs suivants:

- type de vie, valeur 11;
- durée de vie, valeur 12.

Le domaine de chaque attribut est défini dans le Tableau 3. Une charge utile Transform avec des attributs supplémentaires doit être rejetée par le KDC.

### 8.3.4 Échange de clés (2)

Une fois l'association de sécurité négociée, le GM et le KDC doivent être en mesure d'échanger les valeurs publiques Diffie-Hellman (DH) en s'appuyant sur la description du groupe DH convenue dans la SA. La séquence d'échange de clés est représentée à la Figure 21. L'échange repéré en rouge indique l'étape actuellement discutée.

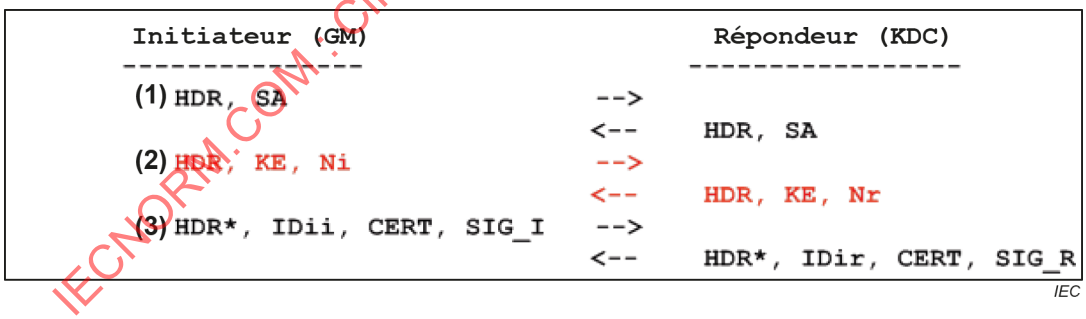



**Figure 21 – Échange IKEv1 en mode principal: messages d'échange de clés**

Dans cette séquence d'échange de clés, le KDC doit recevoir un message ISAKMP avec une charge utile Key Exchange (KE) et une charge utile Nonce (Ni). La charge utile KE contient la valeur publique DH du GM, la charge utile Ni étant le nonce du GM. Sur la base de la valeur publique DH du GM reçue, le secret DH peut être calculé. De même, le GM reçoit la clé publique DH de la KDC dans une charge utile Key Exchange (KE) et un nonce du KDC contenu dans la charge utile Nr. Les nonces doivent être sécurisés par cryptographie et différents pour chaque SA.

Le secret DH et les informations de nonce reçues de la charge utile Ni sont ensuite utilisés pour dériver d'autres clés en phase 1. Le paramètre DH public, le nonce et d'autres informations provenant du protocole de transfert sont ensuite utilisés pour calculer les signatures basées sur HASH_I (initiateur) et HASH_R (répondeur) du troisième protocole de transfert de l'échange de phase 1 (voir 8.3.5).

Comme indiqué dans la RFC 2409, section 5, le nonce doit être compris entre 8 octets et 256 octets. Le KDC doit créer un nonce qui est au moins égal à la moitié de la taille du bloc de hachage.

Les charges utiles KE et Ni/Nr doivent être identiques à celles respectivement définies dans la RFC 2408, sections 3.8 et 3.14.

### 8.3.5 Échange d'authentification d'ID (3)

#### 8.3.5.1 Généralités

Une fois la connexion sécurisée, le dernier échange de messages de sécurité procède à une authentification mutuelle. La séquence d'authentification d'ID est représentée à la Figure 22. L'échange repéré en rouge indique l'étape actuellement discutée.

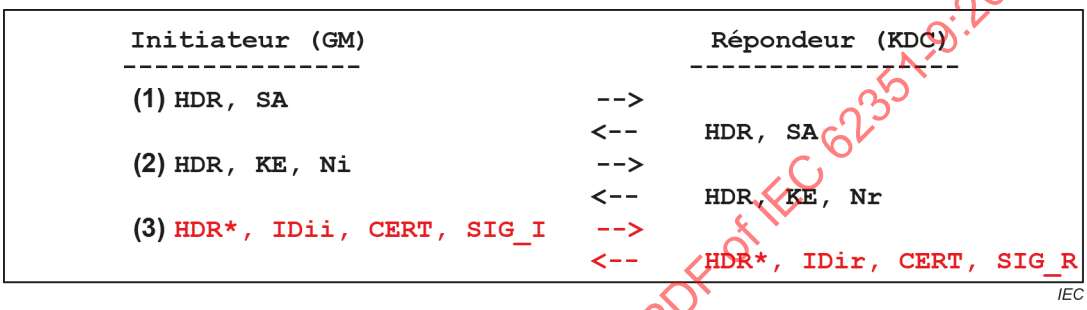



**Figure 22 – Échange IKEv1 en mode principal: messages d'authentification ID**

Le dernier message reçu du GM doit contenir l'identification ISAKMP du GM (charge utile Identification), son certificat de clé publique (charge utile Certificate) et la signature de HASH_I ou HASH_R respectivement en utilisant la clé privée de son certificat de clé publique X.509 (charge utile Signature).

#### 8.3.5.2 Charge utile Identification

La charge utile ID (IDii, IDir) doit être identique à celle définie dans la RFC 2408, section 3.8. D'un point de vue technique, les types d'ID sont spécifiques au DOI. Le GDOI de la RFC 6407 ne spécifie pas ses propres types, mais implique d'utiliser les types définis pour le DOI IPSec dans le GDOI. Ces types sont gérés par l'IANA sous IKEv2 (RFC 5996 [39]). Par conséquent, le type d'ID pris en charge doit être ID_DER_ASN1_DN, dont la valeur est 9. Le contenu de la charge utile doit être l'ID d'objet à codage DER provenant du certificat de clé publique X.509 du GM. Tous les autres types d'ID ne doivent pas être pris en charge et doivent générer une erreur.

Il est à noter que l'ID d'objet fait également partie de la charge utile Certificate décrite en 8.3.5.3. Néanmoins, il est conservé dans le présent document pour maintenir la compatibilité avec la RFC 2409.

#### 8.3.5.3 Charge utile Certificate

La charge utile Certificate (CERT) doit contenir le certificat de clé publique X.509 à codage DER et doit identique à celle définie dans la RFC 2408, section 3.9. Le type de codage de certificat *Certificats de clé publique X.509 – Signature* dont la valeur est 4, doit être pris en charge. Tous les autres types de codages ne doivent pas être pris en charge et doivent générer une erreur.

Même si la RFC 2409 spécifie qu'au moins une charge utile Certificate peut éventuellement être incluse, le KDC doit toujours inclure une seule charge utile Certificate dans le message ISAKMP.

Si le GM fournit un certificat qui ne peut pas être vérifié par le KDC, le KDC doit envoyer un échange informationnel de phase 2 contenant une charge utile Delete, qui est chiffrée en utilisant le matériel de clé de SA de phase 1 (voir aussi 8.4.3).

#### 8.3.5.4 Charge utile Signature

La charge utile Signature (SIG_I, SIG_R) ne doit pas être modifiée par rapport à celle de la RFC 2408, section 3.12. Le contenu doit être la signature qui est générée par l'utilisation de la clé privée des initiateurs ou répondeurs correspondant au certificat de clé publique utilisé pour l'identification de l'expéditeur, respectivement de HASH_I (initiateur) ou HASH_R (répondeur) comme défini dans la RFC 2409, section 5, et représenté à la Figure 23.

```
HASH_I = prf(SKEYID, g^xi | g^xr | CKY-I | CKY-R | SAi_b | IDii_b )

HASH_R = prf(SKEYID, g^xr | g^xi | CKY-R | CKY-I | SAi_b | IDir_b )
```

IEC

**Figure 23 – Calcul d'IKEv1 HASH_I**

Se reporter à la section 3 de la RFC 2409 pour la définition des notations.

Il est important de noter que la signature RSA n'est pas une signature au format PKCS #1 normal, mais qu'elle est simplement chiffrée à l'aide de la clé privée RSA correspondante du certificat, tel que décrit dans la RFC 2409, section 5.1.

> ***"Puisque l'algorithme de hachage utilisé est déjà connu, il n'est pas nécessaire de coder son OID dans la signature. De plus, il n'y a pas de liaison entre les OID utilisés pour les signatures RSA dans PKCS #1 et celles utilisées dans le présent document. Par conséquent, la signature RSA DOIT être codée comme un chiffrement de clé privée au format PKCS #1 et non comme une signature au format PKCS #1 (qui inclut l'OID de l'algorithme de hachage)."***

### 8.4 Échange informationnel ISAKMP de phase 1/2 de type 5

#### 8.4.1 Généralités

Le KDC doit utiliser les échanges informationnels ISAKMP pour informer ses homologues qu'une erreur s'est produite. Il ne doit pas utiliser ces échanges pour indiquer l'état des SA tel que défini dans l'IKEv1 (voir RFC 2409, section 5.7).

Un échange informationnel peut être envoyé lors d'un échange de phase 1 ou 2. Un échange informationnel de phase 1 n'est pas chiffré et la valeur zéro est attribuée au champ Message ID de l'en-tête ISAKMP. Un échange informationnel de phase 2 est chiffré et comporte un ID de message unique. Un ID de message unique doit être fourni de manière à pouvoir calculer un vecteur d'initialisation (IV) cryptographique applicable.

#### 8.4.2 Échange informationnel de phase 1

##### 8.4.2.1 Généralités

Un échange informationnel de phase 1 peut être initié lors de l'établissement de la connexion ISAKMP afin d'indiquer qu'une erreur s'est produite dans l'échange de phase 1. L'échange informationnel de phase 1 est représenté à la Figure 24.


**Figure 24 – Échange informationnel de phase 1 (voir RFC 2408, section 4.8)**

Le GM et le KDC peuvent être les initiateurs d'un échange informationnel de phase 1. Le message d'échange informationnel doit comporter une seule charge utile Notification (N), comme défini dans la RFC 2408, section 3.14.

La charge utile Delete (D) définie dans la section 3.15 de la RFC 2408 doit être prise en charge pour un échange informationnel de phase 1 tel que défini dans la RFC 2408, section 3.15.

La charge utile Hash ne doit pas être prise en charge pour un échange informationnel de phase 1. Par conséquent, le KDC ne doit envoyer aucun échange informationnel de phase 1 contenant une charge utile Hash et doit ignorer toutes les charges utiles Hash reçues pendant un échange informationnel de phase 1. Le champ Message ID de l'en-tête ISAKMP du message d'échange informationnel de phase 1 doit être défini sur 0 et ce message ne doit pas être chiffré.

#### 8.4.2.2 Charge utile Notification

La charge utile Notification est définie dans la RFC 2408, section 3.14. Pour un échange informationnel de phase 1, les valeurs de champ de la charge utile sont définies comme suit:

- Domain of Interpretation (domaine d'interprétation): valeur 2, GDOI;
- Protocol-ID (ID de protocole): valeur 0;
- SPI Size (taille de SPI): valeur 0. I-Cookie et R-Cookie font office de SPI;
- Notify Message Type (notification du type de message): l'une des valeurs définies dans la RFC 2408, section 3.14.1;
- SPI: exclu de la charge utile;
- Notification Data (données de notification): exclues de la charge utile.

#### 8.4.2.3 Charge utile Delete

La charge utile Delete est définie dans la RFC 2408, section 3.15. La charge utile Delete est utilisée pour réagir aux erreurs potentielles au cours de la négociation de phase 1, telles qu'un délai de SA, des erreurs de certificat ou des erreurs liées à Diffie-Hellman (par exemple MODP erroné, etc.), aucune concordance entre les ensembles de transfert, etc. Pour un échange informationnel de phase 1, les valeurs de champ de la charge utile sont définies comme suit:

- Domain of Interpretation: valeur 2, GDOI;
- Protocol-ID: valeur 0;
- SPI Size: valeur 16;
- Number of SPI (nombre de SPI): le nombre de SPI contenus dans la charge utile Delete;
- SPI: SPI à supprimer. I-Cookie et R-Cookie font office de SPI.

### 8.4.3 Échange informationnel de phase 2

Un échange informationnel de phase 2 peut être initié pour fournir des charges utiles Notification ou Delete après de l'établissement de la connexion ISAKMP, afin d'indiquer qu'une erreur s'est produite dans l'échange de phase 2. L'échange informationnel de phase 2 est représenté à la Figure 25.

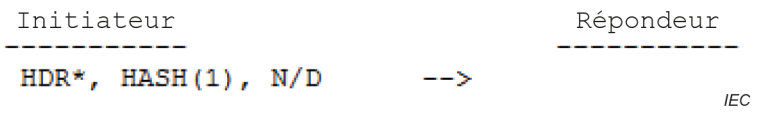



**Figure 25 – Échange informationnel de phase 2 (voir RFC 2409, section 5.7)**

La RFC 6407 relative au GDOI mentionne un échange informationnel se rapportant à un GM qui n'accepte pas une politique de SA. Si la politique de la charge utile SA n'est pas acceptable pour le GM, "il convient que le GM supprime la session de phase 1 après avoir notifié le KDC avec un échange informationnel ISAKMP contenant une charge utile Delete" (RFC 6407, section 3.3).

Toutefois, aucune précision n'est apportée quant à la manière dont il convient que le KDC ou le GM traite les cas de hachage incorrect, de GM sans droit d'accès au groupe ou d'expiration de la durée de vie de la SA spécifique. Pour couvrir ces situations, le présent document ajoute l'exigence selon laquelle un KDC doit intégrer une charge utile Notification dans l'échange GROUPKEY-PULL lui-même. En cas de défaillance lors du traitement d'un message GROUPKEY-PULL provenant du GM, le KDC doit renvoyer un message GROUPKEY-PULL dans le cadre du même échange avec une seule charge utile Notification indiquant le code d'erreur. L'échange GROUPKEY-PULL doit prendre fin.

Le KDC doit prendre en charge la réception d'une charge utile Notification intégrée dans l'échange GROUPKEY-PULL et la réception d'un échange informationnel avec une charge utile Delete. Si le SPI de la charge utile Notification/Delete correspond à une SA d'un échange GROUPKEY-PULL existant, il doit arrêter l'échange.

Si cette erreur est signalée à l'aide d'un échange informationnel avec une charge utile Delete, elle doit contenir une valeur HASH. Le message doit être sécurisé en s'appuyant sur le matériel de clé établi SKEYID_e et SKEYID_a issu de la phase 1. La valeur HASH doit être calculée comme défini pour HASH(1) dans la section 5.7 de la RFC 2409 et représentée à la Figure 26 basée sur SKEYID_a.

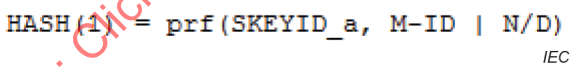



**Figure 26 – Calcul d'IKEv1 HASH(1)**

De plus, lorsque l'échange informationnel avec une charge utile Delete est envoyé, il doit être chiffré à l'aide du SKEYID_e résultant de la SA de phase 1, comme décrit dans la section 5.7 de la RFC 2409.

## 8.5 Échange GROUPKEY-PULL du GDOI de phase 2 de type 32

### 8.5.1 Généralités

Le GDOI (RFC 6407) définit l'échange GROUPKEY-PULL de phase 2 permettant aux membres du groupe d'extraire les associations de sécurité partagées pour un seul groupe multidiffusion sécurisé. L'échange GROUPKEY-PULL est effectué après sécurisation de la connexion basée sur la SA IKE établie au cours de l'échange IKEv1 de phase 1 en mode principal. La RFC 6407, section 3.2, définit l'échange GROUPKEY-PULL représenté à la Figure 27.

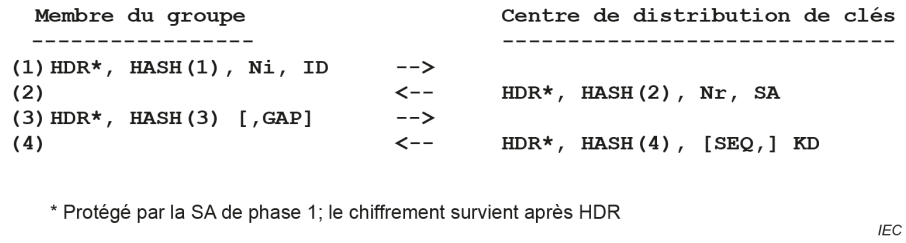



**Figure 27 – Échange GROUPKEY-PULL du GDOI tel que défini dans la RFC 6407**

Voir la RFC 2409, section 3.2, et la RFC 6407, section 1.3, pour la notation, les acronymes et les abréviations.

Un membre du groupe doit toujours initier un échange GROUPKEY-PULL. Le KDC doit toujours être le répondeur d'un échange GROUPKEY-PULL.

Lorsque le KDC reçoit le message d'échange GROUPKEY-PULL initial de la part du GM, ce message est validé et un nouvel échange GROUPKEY-PULL commence. Les charges utiles sont extraites et HASH(1) est calculé et validé. Un calcul de hachage incorrect au niveau du KDC doit se traduire par l'échec de l'échange, et un échange informationnel de phase 2 doit être initié par le KDC en indiquant un type d'erreur INVALID_HASH_INFORMATION (valeur 23).

Un calcul de hachage incorrect au niveau du GM doit se traduire par l'échec de l'échange, et un message de réponse GROUPKEY-PULL doit être renvoyé au GM avec une charge utile Notification indiquant un type d'erreur INVALID_HASH_INFORMATION (valeur 23).

L'ID de groupe multidiffusion sécurisé doit être extrait et, si le groupe multidiffusion sécurisé est introuvable ou si le GM n'est pas autorisé, l'échange doit échouer et un message de réponse GROUPKEY-PULL doit être renvoyé au GM avec une charge utile Notification indiquant l'erreur INVALID-ID-INFORMATION (valeur 18) et AUTHENTICATION_FAILURE (valeur 24), respectivement.

Les membres du groupe doivent s'assurer que les protocoles de transfert de phase 2 pour les clés doivent être terminés avant toute autre nouvelle demande de clé dans cette SA.

Si un membre du groupe n'a pas été en mesure d'extraire la même clé de groupe pour un groupe spécifique après plusieurs tentatives, il doit envoyer un échange informationnel de phase 2 contenant une charge utile Delete, qui est chiffré en utilisant le matériel de clé de la SA de phase 1 (voir aussi 8.4.3). Ceci met pratiquement fin à la SA avec ce GM. Le nombre de nouvelles tentatives dépend de la politique de sécurité locale.



### 8.5.2 Calculs de hachage

Les hachages calculés pour l'échange GROUPKEY-PULL doivent utiliser l'algorithme de hachage de sécurité provenant de la SA de phase 1. Les hachages doivent être sécurisés avec le secret d'authentification de phase 1, SKEYID_a, tel que défini dans la RFC 2409, section 5. Les différents hachages utilisés dans l'échange GROUPKEY-PULL doivent être calculés en fonction des informations spécifiées à la Figure 28.

```
HASH(1) = prf(SKEYID_a, M-ID | Ni | ID)

HASH(2) = prf(SKEYID_a, M-ID | Ni_b | Nr | SA)

HASH(3) = prf(SKEYID_a, M-ID | Ni_b | Nr_b [ | GAP ])

HASH(4) = prf(SKEYID_a, M-ID | Ni_b | Nr_b [ | SEQ ] | KD)
```

IEC

**Figure 28 – Calculs de hachage GROUPKEY-PULL**

### 8.5.3 Algorithme de chiffrement multi-expéditeur et de mode compteur

Les SA de l'IEC 61850 (voir 8.5.7) doivent prendre en charge les algorithmes de mode compteur AES-GCM et AES-GMAC. Les algorithmes de mode compteur offrent la possibilité d'avoir plusieurs expéditeurs sur une seule SA. Il s'agit de s'assurer que tous les expéditeurs utilisent des vecteurs d'initialisation (IV) uniques pour chaque paquet envoyé. La RFC 6407, section 3.5, relative au GDOI, spécifie que le KDC met en œuvre la RFC 6054 afin d'attribuer une partie de l'espace IV à chaque expéditeur présent dans le groupe [6].

La prise en charge de la demande d'ID d'expéditeur formulée par un GM pour le mode compteur en fonction des groupes de clés IEC 61850 n'est pas exigée. Si une charge utile de la politique associée au groupe (GAP) est reçue d'un GM détenant l'attribut Sender-ID GAP (valeur 3), le KDC doit interrompre l'échange GROUPKEY-PULL. Un type d'erreur ATTRIBUTES-NOT-SUPPORTED (valeur 13) doit être renvoyé dans l'échange informationnel.

NOTE Une édition ultérieure du présent document devra déterminer si plusieurs expéditeurs d'une même SA DATA seront pris en charge.

### 8.5.4 Prise en charge des charges utiles liées au téléchargement de clé SA KEK, SEQ, KEK/LKH

GROUPKEY-PUSH étant facultatif, la prise en charge d'une charge utile SA KEK, d'une charge utile SEQ ou des paquets de clé KEK/LKH par l'échange GROUPKEY-PULL dans la charge utile Key Download n'est pas exigée. Par conséquent, dans le contexte d'un GROUPKEY-PULL, il convient de ne pas utiliser les clés SA KEK. Les clés KEK doivent être réservées au GROUPKEY-PUSH conformément à la RFC 6407, section 5.3.2.

Le renouvellement de clé d'une SA existante avec GROUPKEY-PUSH est décrit en 8.6.

Les informations relatives aux affectations pour les charges utiles GDOI, en particulier les types de téléchargements de clé, sont disponibles auprès de l'IANA [52]. D'après cette définition, le type de téléchargement de clé pour TEK est 1, tandis que le type de téléchargement de clé pour KEK est 2.

NOTE Le moment de la prise en charge de l'échange GROUPKEY-PUSH par les protocoles de sécurité de l'IEC 61850 est à déterminer.

### 8.5.5 Échange de demande de SA du groupe GROUPKEY-PULL

#### 8.5.5.1 Généralités

L'échange de demande de SA initiale est représenté à la Figure 29. L'échange repéré en rouge indique l'étape actuellement discutée.



6 RFC 6054 – Using Counter Modes with Encapsulating Security Payload (ESP) and Authentication Header (AH) to Protect Group Traffic (http://tools.ietf.org/html/rfc6054) (disponible en anglais seulement).

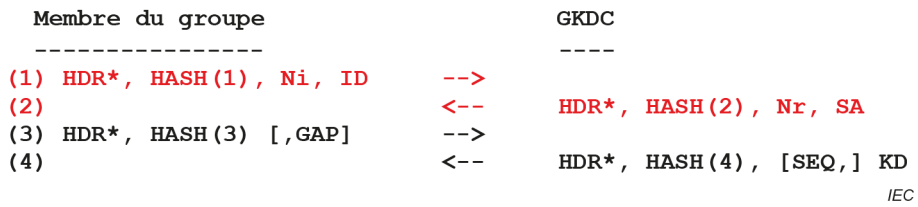



**Figure 29 – Échange de demande de SA initiale du groupe GROUPKEY-PULL**

Le GM doit initier un échange GROUPKEY-PULL en envoyant un message (1) au KDC. Les charges utiles HASH (1) et Ni sont définies dans la RFC 6407, section 3. La charge utile ID a été étendue pour prendre en charge les groupes multidiffusion sécurisés de l'IEC 61850.

#### 8.5.5.2 Charge utile Identification

##### 8.5.5.2.1 Généralités

La charge utile Identification (ID) pour l'échange de phase 2 est la même que celle utilisée dans l'échange de phase 1 (voir 8.3.5.2). La différence est que, dans l'échange de phase 1, les données d'identification contenues sont celles du membre du groupe, alors que pour l'échange de phase 2, l'ID identifie le groupe de clés des SA à extraire (voir Figure 30). Le KDC doit prendre en charge les groupes multidiffusion sécurisés de l'IEC 61850.

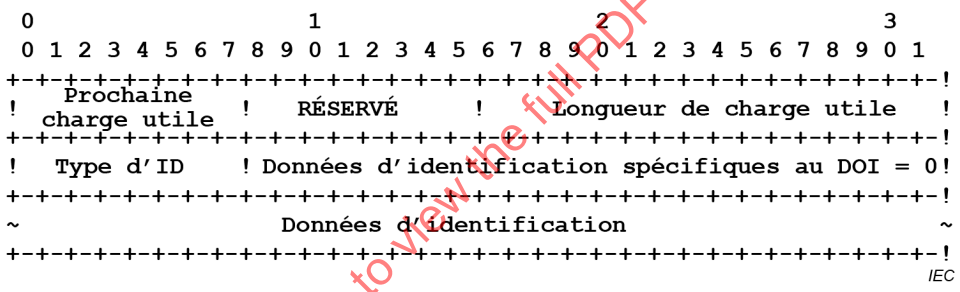



**Figure 30 – Charge utile Identification (RFC 6407)**

Le format de la charge utile Identification est composé des éléments suivants:

- **Prochaine charge utile**: voir la RFC 2408 pour la définition et l'utilisation;
- **Réservé**: voir la RFC 2408 pour la définition et l'utilisation;
- **Longueur de charge utile**: voir la RFC 2408 pour la définition et l'utilisation;
- **Type d'ID**: un groupe multidiffusion sécurisé IEC 61850 est identifié par le type d'ID ID_OID défini dans la RFC 8052, section 2.1, Prise en charge du protocole d'interprétation de domaine de groupe (GDOI) pour les services de sécurité IEC 62351. Le type d'ID ID_OID a une valeur de 13. Les données d'identification de l'ID_OID représentent un ID d'objet (OID) [7] ASN.1 et ses données spécifiques à l'OID. L'OID et les données OID sont codées à l'aide des règles de codage DER [8]. Les définitions des champs dans la charge utile Données d'identification sont données et représentées à la Figure 31. Par exemple, un OID peut représenter un protocole GOOSE ou Sampled Value, l'IEC 61850 spécifiant également dans d'autres cas une adresse de destination multidiffusion particulière à décrire dans le champ Charge utile spécifique à l'OID;
- **Données d'identification spécifiques au DOI**: doivent être définies sur 0;



---

7 Le format d'ID d'objet est décrit dans l'UIT-T-X.683 – http://www.itu.int/ITU-T/studygroups/com17/languages/X.683-0207.pdf.

8 Les règles de codage distinctives (DER) sont définies dans l'UIT-T X.690 – http://www.itu.int/ITU-T/studygroups/com17/languages/X.690-0207.pdf.

- **Données d'identification**: spécifiques à l'ID_OID et décrites en 8.5.5.2.2.

#### 8.5.5.2.2 Données d'identification des objets IEC 61850

La charge utile Identification du GDOI (par exemple demande de phase 2) d'un échange GROUPKEY-PULL de GDOI permet au membre du groupe (GM) de déclarer le groupe qu'il souhaiterait rejoindre. Un groupe est défini par une charge utile ID telle que définie dans la RFC 6407 relative au GDOI. La Figure 31 a été extraite de la RFC 8052 et définit une valeur Type d'ID spécifique et le format des données d'identification.

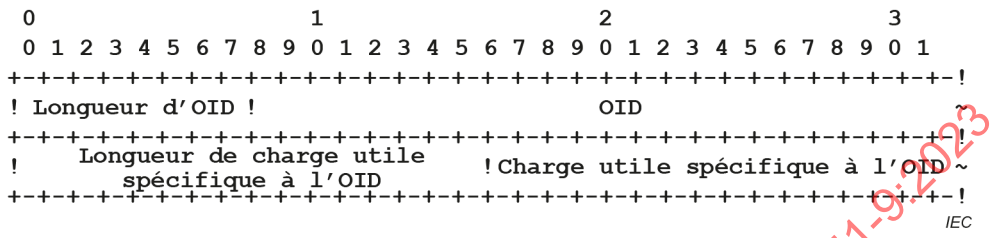



**Figure 31 – Données d'identification ID_OID**

- **Longueur d'OID**: cette longueur doit être une valeur entière non signée et doit spécifier le nombre d'octets de l'OID codé ASN.1, dont la valeur est conforme à la longueur;
- **OID**: cet ensemble d'octets représente un identificateur d'objet codé ASN.1. La valeur de l'identificateur définit la charge utile qui suit. L'utilisation de cet identificateur d'objet va permettre aux autres organisations ou normes d'utiliser ce processus d'extension de charge utile sans collision dans le contexte de la définition. Les valeurs de l'identificateur d'objet sont définies comme étant obligatoires (o) ou facultatives (f) dans le Tableau 4;
- **Longueur de charge utile spécifique à l'OID** (2 octets): longueur de la charge utile spécifique à l'OID. Sa valeur est nulle si l'OID n'exige pas de charge utile spécifique à l'OID;
- **Charge utile spécifique à l'OID** (variable): sélecteur spécifique à l'OID codé en DER. Si la longueur de charge utile spécifique à l'OID est définie sur zéro, ce champ n'apparaît pas dans la charge utile ID.

Au plus fort d'un échange GROUPKEY-PULL, le serveur de clés doit avoir envoyé la politique de groupe à tous les membres de groupe autorisés, leur permettant de participer à des communications de groupe sécurisées.

Le KDC doit prendre en charge les ID d'objet IEC 61850 définis dans le Tableau 4, qui identifient le flux IEC 61850 pour lequel le GM demande le matériel de clé.

**Tableau 4 – ID d'objet IEC 61850: obligatoire (o) et facultatif (f)**

| Nom de l'identificateur d'objet | Description | Valeur | o/f |
|---|---|---|---|
| 61850_ETHERNET_GOOSE | Indique que la charge utile demande une clé pour une APDU GOOSE IEC 61850-8-1. | 1.0.62351.9.61850.8.1.1 | o |
| 61850_UDP_ADDR_GOOSE | Indique que la charge utile demande une clé pour une APDU GOOSE acheminable envoyée à une adresse IP de destination particulière. | 1.0.62351.9.61850.8.1.2 | o |
| 61850_UDP _Tunnel | Indique que la charge utile demande une clé pour une APDU Tunnel acheminable IEC 61850 envoyée à une adresse IP de destination particulière. | 1.0.62351.9.61850.8.1.4 | f |
| 61850_ETHERNET_SV | Indique que la charge utile demande une clé pour une APDU SV IEC 61850-9-2. | 1.0.62351.9.61850.9.2.1 | o |
| 61850_UDP_ADDR_SV | Indique que la charge utile demande une clé pour une APDU SV IEC 61850. | 1.0.62351.9.61850.9.2.2 | o |

| Nom de l'identificateur d'objet | Description | Valeur | o/f |
|---|---|---|---|
| 61850_IP_ISO9506 | Indique que la charge utile demande une clé pour un point d'extrémité IEC 61850-8-1 ISO 9506. La définition de cette charge utile est hors du domaine d'application. | 1.0.62351.9.61850.8.1.4 | f |
| 61850_9_3_PTP | Indique que la charge utile demande une clé pour un domaine PTP IEC 61850-9-3. | 1.0.62351.9.61850.9.3.1 | f |
| Les OID utilisés ont été définis dans l'espace IEC 62351 qui commence par 1.0.62351.9.xxx par opposition aux OID utilisés dans l'espace IEC 61850-90-5 qui débute par 1.2.840.10070.xxx. | | | |

#### 8.5.5.2.3 Charges utiles spécifiques à l'OID IEC 61850

Les charges utiles des versions UDP des OID [9] GOOSE (61850_UDP_ADDR_GOOSE) et SV (61850_UDP_ADDR_SV) sont identiques. La séquence ASN.1 des données spécifiques aux OID 61850_UDP_ADDR_GOOSE et 61850_UDP_ADDR_SV est spécifiée à la Figure 32.

```
IecUdpAddrPayload ::= SEQUENCE
{
      version     INTEGER { v1(1) },
      ipAddress   IPADDRESS,
      dsRef       VisibleString SIZE(1..128))
}
```

IEC

**Figure 32 – 61850_UDP_ADDR_GOOSE/SV ASN.1 BNF**

- `Version` est une valeur à un seul octet qui représente la version de la charge utile particulière. Sauf spécification contraire, la valeur de la VERSION doit être un (1).
- `IpAddress` est la composante d'une valeur permettant d'utiliser une adresse de destination IPv4 ou IPv6 pour l'entité associée à la clé demandée. La séquence ASN.1 IPADDRESS est spécifiée à la Figure 33.
- La composante de valeur `dsRef` permet de spécifier une référence d'ensemble de données (**DSRef**) IEC 61850 (voir IEC 61850-7-2).

```
IPADDRESS ::= SEQUENCE
{
   typeOfAddress ENUMERATED { IPv4(0), IPv6(1) },
   address  CHOICE {
      ip      OCTECT STRING (SIZE(4 | 16)),
      dns     VisibleString (SIZE(1..65536))
      }
}
```

IEC

**Figure 33 – IPADDRESS ASN.1 BNF**

La Figure 34 est un exemple de charge utile Adresse OID 61850 pour l'OID 61850_UDP_ADDR_GOOSE ou 61850_UDP_ADDR_SV.



[9] Les charges utiles pour les versions IP de GOOSE et SV sont identiques.

```
groupaddr IecUdpAddrPayload ::=
{
   version v1,
   ipAddress
   {
        typeOfAddress IPv4,
        address dns: "www.iec.org"
   }
   dsRef "@somedataref"
}

DER Encoding:

30230201 0130100A 01001A0B 7777772E
6965632E 6F72671A 0C40736F 6D656461
74617265 66
```

IEC

**Figure 34 – Exemple de données ASN.1 IecUdpAddrPayload avec codage DER**

La séquence ASN.1 des données spécifiques à l'OID 61850_UDP_TUNNEL est spécifiée à la Figure 35.

```
IecUdpTunnelPayload ::= SEQUENCE
{
    version      INTEGER { v1(1) },
    ipAddress    IPADDRESS
}
```

IEC

**Figure 35 – Charge utile ASN.1 BNF 61850_UDP_TUNNEL**

Le champ `dsRef` est absent de cette charge utile, car plusieurs trames multidiffusion Ethernet (par exemple GOOSE ou SV) peuvent être envoyées dans une SPDU Tunnel. Par conséquent, l'adresse IP de destination elle-même distingue les ensembles de données.

La charge utile OID des versions Ethernet de GOOSE (61850_ETHERNET_GOOSE) et SV (61850_ETHERNET_SV) sont identiques. La séquence ASN.1 des données spécifiques à l'OID 61850_ETHERNET_GOOSE/SV est spécifiée à la Figure 36.

```
IecEthernetAddrPayload ::= SEQUENCE
{
    version   INTEGER { v1(1) },
    dstMAC    OCTET STRING (SIZE(6)),
    dsRef     VisibleString (SIZE(1..256))
}
```

IEC

**Figure 36 – Charge utile ASN.1 BNF 61850_ETHERNET_GOOSE/SV**

- Version est une valeur à un seul octet qui représente la version de la charge utile particulière. Sauf spécification contraire, la valeur de la VERSION doit être un (1).
- `dstMAC` est une valeur composée de six (6) octets. La valeur doit être dans l'ordre de transmission Ethernet.

- La composante de valeur `dsRef` permet de spécifier une référence d'ensemble de données (DSRef) IEC 61850 (voir IEC 61850-7-2).

#### 8.5.5.2.4 Charges utiles spécifiques à l'IEC 61850-9-3

Les charges utiles pour l'IEC 61850-9-3 (c'est-à-dire le profil de puissance de PTP) doivent spécifier le domaine temporel de PTP (`ptpDomain`) et un identificateur pour ce domaine en plus de l'ID de domaine. Cet identificateur doit être une valeur mutuellement convenue entre les GM et le KDC, de sorte que le même numéro de domaine puisse être réutilisé dans d'autres parties de l'infrastructure avec différentes clés et politiques.

```
Iec61850PTPPayload::= SEQUENCE
{
  ptpDomain       INTEGER  - defined by the allowed range per IEEE 1588
  ptpSubDomain    [0] IMPLICIT INTEGER  OPTIONAL - defined by IEEE 1588
  keyMngtDomain   VisibleString (Size(1..256)
}
```

La valeur `keyMngtDomain` est utilisée pour permettre le déploiement des mêmes identificateurs `ptpDomain` et `ptpSubDomain` dans deux zones non liées de l'entreprise de service public et offrir la possibilité de fournir différentes clés et politiques à ces entités. Cette valeur est un problème de configuration locale et ne doit pas être nulle ou vide. L'identification unique des clés demandées dans le GROUPKEY-PULL est un unique triplet de valeurs comprenant les valeurs de `ptpDomain`,`ptpSubDomain` et `keyMngtDomain`.

Toutes les instances PTP pour un domaine PTP particulier ou un sous-domaine PTP sont censées utiliser le même protocole de gestion de clés.

### 8.5.6 Charge utile SA TEK

La charge utile SA TEK doit contenir les attributs de sécurité d'un seul ensemble de politiques associées à la clé de chiffrement du trafic (TEK) d'un groupe. Le type de politique à utiliser avec la TEK est décrit par un champ Protocol-ID inclus dans la SA TEK. Comme représenté à la Figure 37, extraite de la RFC 6407, chaque champ Protocol-ID décrit une définition particulière de la charge utile spécifique du protocole TEK.

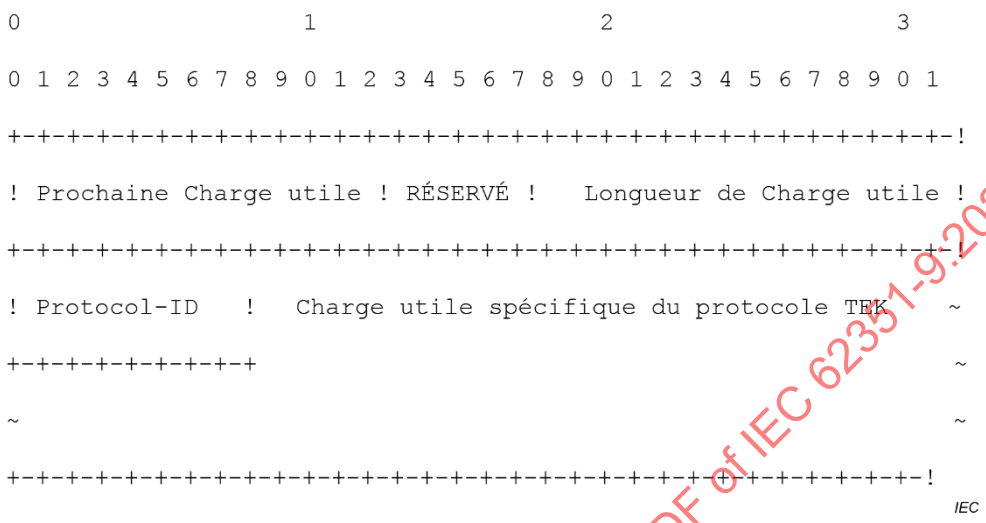



**Figure 37 – Charge utile SA TEK (RFC 6407)**

Le KDC doit respecter le format de charge utile SA TEK IEC-61850 défini dans la RFC 8052, section 2.2. La charge utile SA TEK IEC-61850 définit deux attributs de données de SA facultatifs: SA_ATD (retard d'heure d'activation) et SA_KDA (assurance de livraison de clé). Les valeurs de ces attributs sont définies dans la RFC 8052, section 4.0 (IANA Considerations).

Le nom de l'ID de protocole de GDOI_PROTO_IEC_61850 est défini dans la RFC 8052, section 2.2, et possède la valeur 3.

### 8.5.7 Charge utile SA TEK IEC 61850

La SA TEK GDOI_PROTO_IEC_61850 doit contenir un OID et (éventuellement) une charge utile spécifique à l'OID qui définissent ensemble les sélecteurs du trafic réseau. Les champs de sélecteur doivent être suivis des champs de politique de sécurité indiquant la manière dont le trafic spécifié doit être protégé. Comme représenté à la Figure 38, extraite de la RFC 8052, chaque charge utile TEK IEC 61850 décrit une définition particulière de charge utile spécifique du protocole TEK.

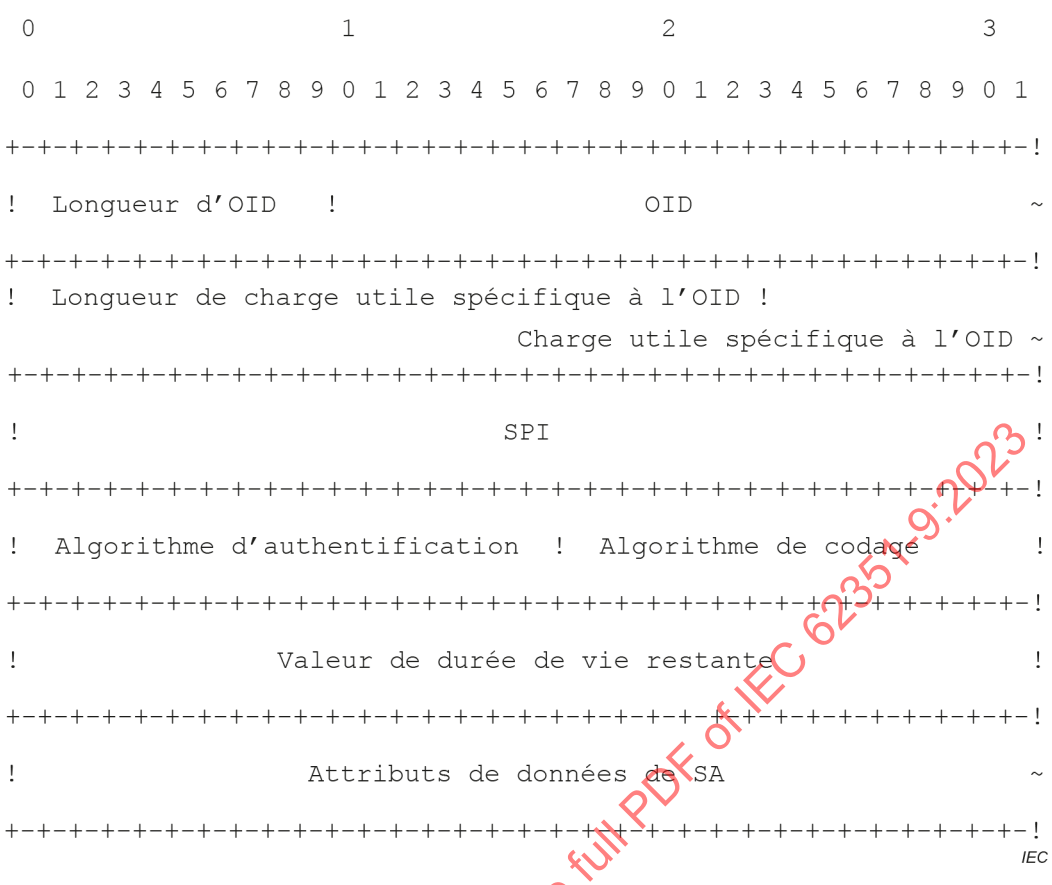



**Figure 38 – Charge utile SA TEK IEC-61850**

Les champs Charge utile SA TEK GDOI_PROTO_IEC_61850 sont définis dans la RFC 8052 comme suit:

a) longueur d'OID (1 octet): longueur du champ OID;

b) OID (variable): identificateur d'objet ASN.1 codé DER. Les OID définis dans l'IEC 61850 déclarent le type de message IEC 61850 à protéger (voir Tableau 4);

c) longueur de charge utile spécifique à l'OID (2 octets): longueur de la charge utile spécifique à l'OID. La valeur de ce champ est nulle si la politique n'inclut aucune charge utile spécifique à l'OID;

d) charge utile spécifique à l'OID (variable): sélecteur de trafic (par exemple adresse multidiffusion) spécifique à l'OID codé DER. Certains paramètres de politique OID n'exigent pas d'utiliser une charge utile spécifique à l'OID, auquel cas ce champ est exclu de la TEK et la longueur de charge utile spécifique à l'OID est définie sur zéro (0);

e) SPI (4 octets): identificateur de la clé en cours. Ce champ représente un SPI;

f) algorithme d'authentification (2 octets): ID de l'algorithme d'authentification. Les valeurs valides sont définies dans la RFC 8052, section 2.2.2. HMAC-SHA256-128 est obligatoire. De plus, le présent document exige également une prise en charge de HMAC-SHA256, AES-GMAC-128 et AES-GMAC-256;

g) algorithme de codage (2 octets): ID de l'algorithme de confidentialité. Les valeurs valides sont définies dans la RFC 8052, section 2.2.3. AES-CBC-128 est obligatoire;

h) valeur de durée de vie restante (4 octets): nombre de secondes restantes avant l'expiration de la TEK. Une valeur nulle (0) doit indiquer que la TEK ne présente aucune heure d'expiration. La durée de vie maximale doit être corrélée au temps d'actualisation de la liste de révocation de certificat, afin de s'assurer que les membres du groupe possédant des certificats expirés ou révoqués sont gérés conformément à la politique de sécurité de l'organisation;