ETSI TR 119 100 V1.1.1 (2016-03)

TECHNICAL REPORT

Electronic Signatures and Infrastructures (ESI); Guidance on the use of standards for signature creation and validation

Reference

DTR/ESI-0019100

Keywords

e-commerce, electronic signature, security,
trust services

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Copyright Notification

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Report (TR) has been produced by ETSI Technical Committee Electronic Signatures and Infrastructures (ESI).

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Introduction

ETSI TR 119 000 [i.1]: "The framework for standardization of signatures: overview", describes the structure of a general framework for digital signatures standardization (hereinafter denoted as Rationalized Framework or Framework) outlining existing and potential standards related to the implementation of digital signatures and the provision of related trust services by trust service providers. This framework identifies six areas of standardization with a list of existing and potential future standards in each area.

ETSI TR 119 000 [i.1] includes a set of guidance documents to assist business stakeholders, users and their suppliers in mapping or deriving from their business driven requirements the appropriate selection of digital signature standards and their options. Each guide addresses a particular area as identified in the aforementioned Rationalized Framework. A complete solution will need to address requirements in most of these areas.

This series is based on the selection of the business scoping parameters for each area of standardization. The selection of these scoping parameters is based on a process involving an analysis of the business requirements and associated risks leading to an identification of the policy and security requirements and to an analysis of the resulting business scoping parameters from which the appropriate standards and options can be selected. From the requirements expressed in terms of business scoping parameters for an area, each guidance document provides assistance in selecting the appropriate standards and their options for that area. Where standards and their options within one area make use of another area this is stated in terms of scoping parameters of that other area.

This general process of the selection of standards and options is described further in ETSI TR 119 000 [i.1], clause 4.2.6.

1 Scope

The present document, which addresses area 1 of the Framework [i.1], provides a **business driven guided process for implementing generation and validation of digital signatures in business' electronic processes**. Starting from a business analysis and risk analysis of the business' electronic processes, stakeholders are guided for making the best choice among the wide offer of standards in order to ensure the best implementation of digital signatures within the addressed application/business electronic processes.

The target audience includes enterprise/business process architects, application architects, application developers, and signature policy issuers.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference/.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

Not applicable.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI TR 119 000: "Electronic Signatures and Infrastructures (ESI); The framework for standardization of signatures: overview".

[i.2] ETSI EN 319 122-1: "Electronic Signatures and Infrastructures (ESI); CAdES digital signatures; Part 1: Building blocks and CAdES baseline signatures".

[i.3] ETSI EN 319 122-2: "Electronic Signatures and Infrastructures (ESI); CAdES digital signatures; Part 2: Extended CAdES signatures".

[i.4] ETSI EN 319 132-1: "Electronic Signatures and Infrastructures (ESI); XAdES digital signatures; Part 1: Building blocks and XAdES baseline signatures".

[i.5] ETSI EN 319 132-2: "Electronic Signatures and Infrastructures (ESI); XAdES digital signatures; Part 2: Extended XAdES signatures".

[i.6] ETSI EN 319 142-1: "Electronic Signatures and Infrastructures (ESI); PAdES digital signatures; Part 1: Building blocks and PAdES baseline signatures".

[i.7] ETSI EN 319 142-2: "Electronic Signatures and Infrastructures (ESI); PAdES digital signatures; Part 2: Additional PAdES signatures profiles".

[i.8] ETSI EN 319 162-1: "Electronic Signatures and Infrastructures (ESI); Associated Signature Containers (ASiC); Part 1: Building blocks and ASiC baseline containers".

[i.9] ETSI EN 319 162-2: "Electronic Signatures and Infrastructures (ESI); Associated Signature Containers (ASiC); Part 2: Additional ASiC containers".

[i.10] ETSI EN 319 102 (all parts): "Electronic Signatures and Infrastructures (ESI); Procedures for Creation and Validation of AdES Digital Signatures; Part 1: Creation and Validation".

[i.11] ETSI TS 119 101: "Electronic Signatures and Infrastructures (ESI); Policy and security requirements for applications for signature creation and signature validation".

[i.12] CEN EN 419 111-1: "Protection Profiles for signature creation & validation application; Part 1: Introduction to the European Norm".

[i.13] CEN EN 419 111-2: "Protection Profiles for signature creation & validation applications; Part 2: Signature creation application - Core PP".

[i.14] CEN EN 419 111-3: "Protection Profiles for signature creation & validation applications; Part 3: Signature creation application - Possible Extensions".

[i.15] CEN EN 419 111-4: "Protection Profiles for signature creation & validation applications; Part 4: Signature verification application - Core PP".

[i.16] CEN EN 419 111-5: "Protection Profiles for signature creation & validation applications; Part 5: Signature verification application - Possible Extensions".

[i.17] ETSI TS 119 172-1: "Electronic Signatures and Infrastructures (ESI); Signature Policies; Part 1: Building blocks and table of contents for human readable signature policy documents".

[i.18] ETSI TS 119 172-2: "Electronic Signatures and Infrastructures (ESI); Signature Policies; Part 2: XML format for signature policies".

[i.19] ETSI TS 119 172-3: "Electronic Signatures and Infrastructures (ESI); Signature Policies; Part 3: ASN.1 formant for signature policies".

[i.20] ETSI TS 119 172-4: "Electronic Signatures and Infrastructures (ESI); Signature Policies; Part 4: Signature validation policy for European qualified electronic signatures/seals using trusted lists".

[i.21] CEN EN 419 103: "Electronic Signatures and Infrastructures (ESI); Conformity Assessment for Signature Creation & Validation Applications (& Procedures)".

[i.22] ETSI TS 119 124 (all parts): "Electronic Signatures and Infrastructures (ESI); CAdES digital signatures Testing Conformance and Interoperability".

[i.23] ETSI TS 119 134 (all parts): "Electronic Signatures and Infrastructures (ESI); XAdES digital signatures Testing Conformance and Interoperability".

[i.24] ETSI TS 119 144 (all parts): "Electronic Signatures and Infrastructures (ESI); PAdES digital signatures Testing Conformance and Interoperability".

[i.25] ETSI TS 119 164 (all parts): "Electronic Signatures and Infrastructures (ESI); Associated Signature Containers (ASiC) Testing Conformance and Interoperability".

[i.26] Regulation (EU) No 910/2014 of the European Parliament and of the Council on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC.

[i.27] CEN TR 419 200: "Electronic Signatures and Infrastructures (ESI); Guidance on the use of standards for signature creation and other related devices".

[i.28] ETSI TR 119 300: "Electronic Signatures and Infrastructures (ESI); Guidance on the use of standards for Cryptographic Suites".

[i.29] ETSI TS 119 312: "Electronic Signatures and Infrastructures (ESI); Cryptographic Suites".

[i.30] Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures.

[i.31] Commission Decision 2009/767/EC of 16 October 2009 setting out measures facilitating the use of procedures by electronic means through the 'points of single contact' under Directive 2006/123/EC of the European Parliament and of the Council on services in the internal market.

[i.32] Commission Decision 2010/425/EC of 28 July 2010 amending Decision 2009/767/EC as regards the establishment, maintenance and publication of trusted lists of certification service providers supervised/accredited by Member States.

[i.33] Commission Decision 2013/662/EU of 14 October 2013 amending Decision 2009/767/EC as regards the establishment, maintenance and publication of trusted lists of certification service providers supervised/accredited by Member States.

[i.34] Commission Implementing Decision 2014/148/EU of 17 March 2014 amending Decision 2011/130/EU establishing minimum requirements for the cross-border processing of documents signed electronically by competent authorities under Directive 2006/123/EC of the European Parliament and of the Council on services in the internal market.

[i.35] IETF RFC 5280: "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile".

[i.36] ETSI TR 119 001: "Electronic Signatures and Infrastructures (ESI); The framework for standardization of signatures; Definitions and abbreviations".

[i.37] W3C Recommendation: "XML Signature Syntax and Processing Version 1.1". April 2013.

[i.38] Commission Decision 2011/130/EU of 25 February 2011 establishing minimum requirements for the cross-border processing of documents signed electronically by competent authorities under Directive 2006/123/EC of the European Parliament and of the Council on services in the internal market.

[i.39] ETSI TS 101 903: "Electronic Signatures and Infrastructures (ESI); XML Advanced Electronic Signatures (XAdES)".

[i.40] ETSI TS 101 733: "Electronic Signatures and Infrastructures (ESI); CMS Advanced Electronic Signatures (CAdES)".

[i.41] ETSI TS 102 778 (all parts): "Electronic Signatures and Infrastructures (ESI); PDF Advanced Electronic Signature Profiles".

[i.42] ETSI TS 102 918: "Electronic Signatures and Infrastructures (ESI); Associated Signature Containers (ASiC)".

[i.43] ETSI TS 103 171: "Electronic Signatures and Infrastructures (ESI); XAdES Baseline Profile".

[i.44] ETSI TS 103 172: "Electronic Signatures and Infrastructures (ESI); PAdES Baseline Profile".

[i.45] ETSI TS 103 173: "Electronic Signatures and Infrastructures (ESI); CAdES Baseline Profile".

[i.46] ETSI TS 103 174: "Electronic Signatures and Infrastructures (ESI); ASiC Baseline Profile".

[i.47] IETF RFC 2315: "PKCS #7: Cryptographic Message Syntax. Version 1.5".

[i.48] IETF RFC 5652: "Cryptographic Message Syntax (CMS)".

[i.49] ISO 32000-1: "Document management -- Portable document format -- Part 1: PDF 1.7".

[i.50] IETF RFC 3851: "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2. Message Specification".

[i.51] IETF RFC 3986: "Uniform Resource Identifier (URI): Generic Syntax".

[i.52] IETF RFC 3161: "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ETSI TR 119 001 [i.36] and the following apply:

NOTE: The definitions being imported in the present document for the sake of reader's convenience.

advanced electronic seal: As defined in Regulation (EU) No 910/2014 [i.26].

advanced electronic signature: As defined in Regulation (EU) No 910/2014 [i.26].

business scoping parameter: specific parameter scoped in the light of the business process(es) where digital signatures or trust services are to be implemented, which implementers need to take into consideration for appropriately addressing the related business requirements in their implementation

CAdES signature: digital signature that satisfies the requirements specified within ETSI EN 319 122-1 [i.2] or ETSI EN 319 122-2 [i.3]

claimed signing time: time of signing claimed by the signer which on its own does not provide independent evidence of the actual signing time

(signature) commitment type: signer-selected indication of the exact implication of a digital signature

data object: actual binary/octet data being operated on (transformed, digested, or signed) by an application

NOTE: This definition is part of the definition of this term within XMLDSIG [i.37].

digital signature: data appended to, or a cryptographic transformation of a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery e.g. by the recipient

digital signature value: result of the cryptographic transformation of a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery e.g. by the recipient

detached (digital) signature: digital signature that, with respect to the signed data object, is neither enveloping nor enveloped.

enveloped (digital) signature: digital signature embedded within the signed data object

enveloping (digital) signature: digital signature embedding the signed data object

evidence: information that can be used to resolve a dispute about various aspects of authenticity of archived data objects

evidence record: unit of data, which can be used to prove the existence of an archived data object or an archived data object group at a certain time

legacy ASiC 102 918 container: associated signature container generated according to ETSI TS 102 918 [i.42]

legacy ASiC baseline container: digital signature generated according to ETSI TS 103 174 [i.46]

legacy ASiC container: legacy ASiC 10 918 container or legacy ASiC baseline container

legacy CAdES 101 733 signature: digital signature generated according to ETSI TS 101 733 [i.40]

legacy CAdES baseline signature: digital signature generated according to ETSI TS 103 173 [i.45]

legacy CAdES signature: legacy CAdES 101 733 signature or a legacy CAdES baseline signature

legacy PAdES 102 778 signature: digital signature generated according to ETSI TS 102 778 [i.41]

legacy PAdES baseline signature: digital signature generated according to ETSI TS 103 172 [i.44]

legacy PAdES signature: legacy PAdES 102 778 signature or a legacy PAdES baseline signature

legacy XAdES 101 903 signature: digital signature generated according to ETSI TS 101 903 [i.39]

legacy XAdES baseline signature: digital signature generated according to ETSI TS 103 171 [i.43]

legacy XAdES signature: legacy XAdES 101 903 signature or legacy XAdES baseline signature

PAdES signature: digital signature that satisfies the requirements specified within ETSI EN 319 142-1 [i.6] or ETSI EN 319 142-2] [i.7]

PDF serial signature: specific digital signature where the second (and subsequent) signers of a PDF not only sign the document but also the signature of the previous signer and any modification that can also have taken place (e.g. form fill-in)

PDF signature: DER-encoded binary data object based on the PKCS #7 (IETF RFC 2315 [i.47]) or the CMS (IETF RFC 5652 [i.48]) or related syntax containing a digital signature and other information necessary to validate the digital signature such as the signer's certificate along with any supplied revocation information placed within a PDF document structure

NOTE: As specified in ISO 32000-1 [i.49], clause 12.8.

proof of existence: evidence that proves that an object existed at a specific date/time

qualified electronic seal: As defined in Regulation (EU) No 910/2014 [i.26].

qualified electronic signature: As defined in Regulation (EU) No 910/2014 [i.26].

qualified electronic signature/seal creation device: As specified in Regulation (EU) No 910/2014 [i.26].

secure cryptographic device: device which holds the user's private key, protects this key against compromise and performs signing or decryption functions on behalf of the user

signature attribute: signature property

signature augmentation: process of incorporating to a digital signature information aiming to maintain the validity of that signature over the long term

NOTE: Augmenting signatures is a co-lateral process to the validation of signatures, namely the process by which certain material (e.g. time stamps, validation data and even archival-related material) is incorporated to the signatures for making them more resilient to change or for enlarging their longevity.

signature augmentation policy: set of rules, applicable to one or more digital signatures, that defines the technical and procedural requirements for their augmentation, in order to meet a particular business need, and under which the digital signature(s) can be determined to be conformant

NOTE: This covers collection of information and creation of new structures that allows performing, on the long term, validations of a signature.

signature creation application: application within the signature creation system, complementing the signature creation device, that creates a signature data object

signature creation device: configured software or hardware used to implement the signature creation data and to create a digital signature value

signature creation policy: set of rules, applicable to one or more digital signatures, that defines the technical and procedural requirements for their creation, in order to meet a particular business need, and under which the digital signature(s) can be determined to be conformant

signature policy: signature creation policy, signature augmentation policy, signature validation policy or any combination thereof, applicable to the same signature or set of signatures

signature policy authority: entity responsible for the drafting, registering, maintaining, issuing and updating of a signature policy

signature policy document: document expressing one or more signature policies in a human readable form

signature validation: process of verifying and confirming that a digital signature is valid

signature verification: process of checking the cryptographic value of a signature using signature verification data

signer: entity being the creator of a digital signature

time assertion: time-stamp token or an evidence record

time-stamp: data in electronic form which binds other electronic data to a particular time establishing evidence that these data existed at that time

XAdES signature: digital signature that satisfies the requirements specified within ETSI EN 319 132-1 [i.4] or ETSI EN 319 132-2 [i.5]

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

ASiC	Associated Signature Containers
ASN.1	Abstract Syntax Notation 1
BER	Basic Encoding Rules
BPMN	Business Process Management and Notation
BSP	Business Scoping Parameter
CA	Certification Authority
CD	Commission Decision
CMS	Cryptographic Message Syntax
CRL	Certificate Revocation List
DA	Driving Application
DER	Distinguished Encoding Rules
DSS	Document Security Store
DTBS	Data To Be Signed
DTBSR	Data To Be Signed Representation
EC	European Commission
ETSI CTI	ETSI Centre for Testing and Interoperability
IN_{MI}	INput for Message Imprint computation
ISMS	Information Security Management System
ISO	International Organization for Standardization
OCSP	Online Certificate Status Protocol
ODF	Open Document Format
OID	Object IDentifier
PKI	Public Key Infrastructure
POE	Proof Of Existence
QES	Qualified Electronic Signature
S/MIME	Secure/Multipurpose Internet Mail Extensions
SAML	Security Assertion Markup Language
SCA	Signature Creation Application.
SCDev	Signature Creation Device
SCS	Signature Creation System
SHA	Secure Hash Algorithm
SVA	Signature Validation Application.
TL	Trusted List
TSA	Time-Stamping Authority
TSP	Trust Service provider
UML	Unified Modelling Language
URI	Uniform Resource Identifier
VRI	Validation Related Information
XFA	Xml Forms Architecture
XML	eXtensible Markup Language
XMP	eXtensible Metadata Platform

4 Introduction to the guided implementation process

4.1 How to use the present document

The present document is one of a series of guidance documents on selection of standards and options for implementing digital signatures and/or trust services. All these documents share a general approach, suitably profiled and developed by each one. This general approach starts from a pre-required analysis of the business requirements and involves the analysis of business scoping parameters specific to each area of standardization. These scoping parameters are essential elements to be addressed and for which business driven choices need to be made facilitating the selection of the appropriate standards and their options in a way which meets, as far as possible, the business requirements.

The present document proposes a guided process (that is driven by business) for implementing generation and validation of digital signatures in business electronic processes.

The present document specifically addresses the implementation of digital signatures, in particular generation, validation, and augmentation of digital signatures. Any other aspect within other areas related to the implementation of digital signatures (like cryptographic devices, cryptographic suites, supporting TSPs, etc.) is out of its scope. Nevertheless, it addresses readers to the suitable guidance documents within the Rationalized Framework that deal with other areas.

The present clause provides some suggestions on how to read the present document depending on the reader's profile (business managers, application architects, developers and signature policy issuers):

1) Enterprise/business process architects, and managers should read until clause 7 included. These clauses are the part of the process that aims at describing at a high level the conditions and rules under which digital signatures will be used within a business or application domain and process. These clauses focus on areas that are familiar to the aforementioned profiles, i.e. business processes modelling, risk assessment, business requirements, regulatory/legal framework requirements, policy and security requirements, business rules and business scoping parameters, which will jointly condition the actual implementation of digital signatures within the business.

2) Application architects should read the whole document. They will find material that will guide them throughout the process of designing a system that fully and properly satisfies all the business and legal/regulatory requirements specific to digital signatures, and who will gain a better understanding on how to select the proper standards to be implemented and/or used.

3) Application developers should read the whole document. They will find an understanding of the business driven approach underlying the decisions made by the enterprise/business process architects, and application architects on the relevant business scoping parameters when creating and validating digital signatures in the concerned business processes. They will also better understand why the managed signatures incorporate certain components. They finally will gain a proper knowledge of what standards exist in the field (that they are supposed to know in detail for a proper development).

4) Signature policy issuers should read the whole document. A signature policy document is a declaration of the practices and rules (to be) used when creating, preserving, validating and augmenting digital signatures in a specific context (e.g. business process) and is usually a document resulting from the execution of the implementation process described in the present document. Signature policy issuers will find in the present document guidance on the decision-making process for specifying the aforementioned rules to be imposed within a specific context.

4.2 An overview of the guided implementation process

The present clause aims at providing a summary of the guided implementation process proposed within the present document and also at briefly uncovering its relationships with other relevant guidance documents within the Rationalized Framework [i.1].

Figure 1 graphically summarizes the most relevant phases of the guided implementation process. It also shows two relevant elements, which may have a great impact, despite the fact that they cannot be considered, strictly speaking, as being part of the process. These two elements are addressed at the end of the present clause.

The proposed guided implementation process is likely to be iterative by nature, as indicated by the arrow that goes back from the last phase to the beginning. The present document does not make any consideration about the degree of completion of the different phases in each iteration, which is entirely left to the implementers.

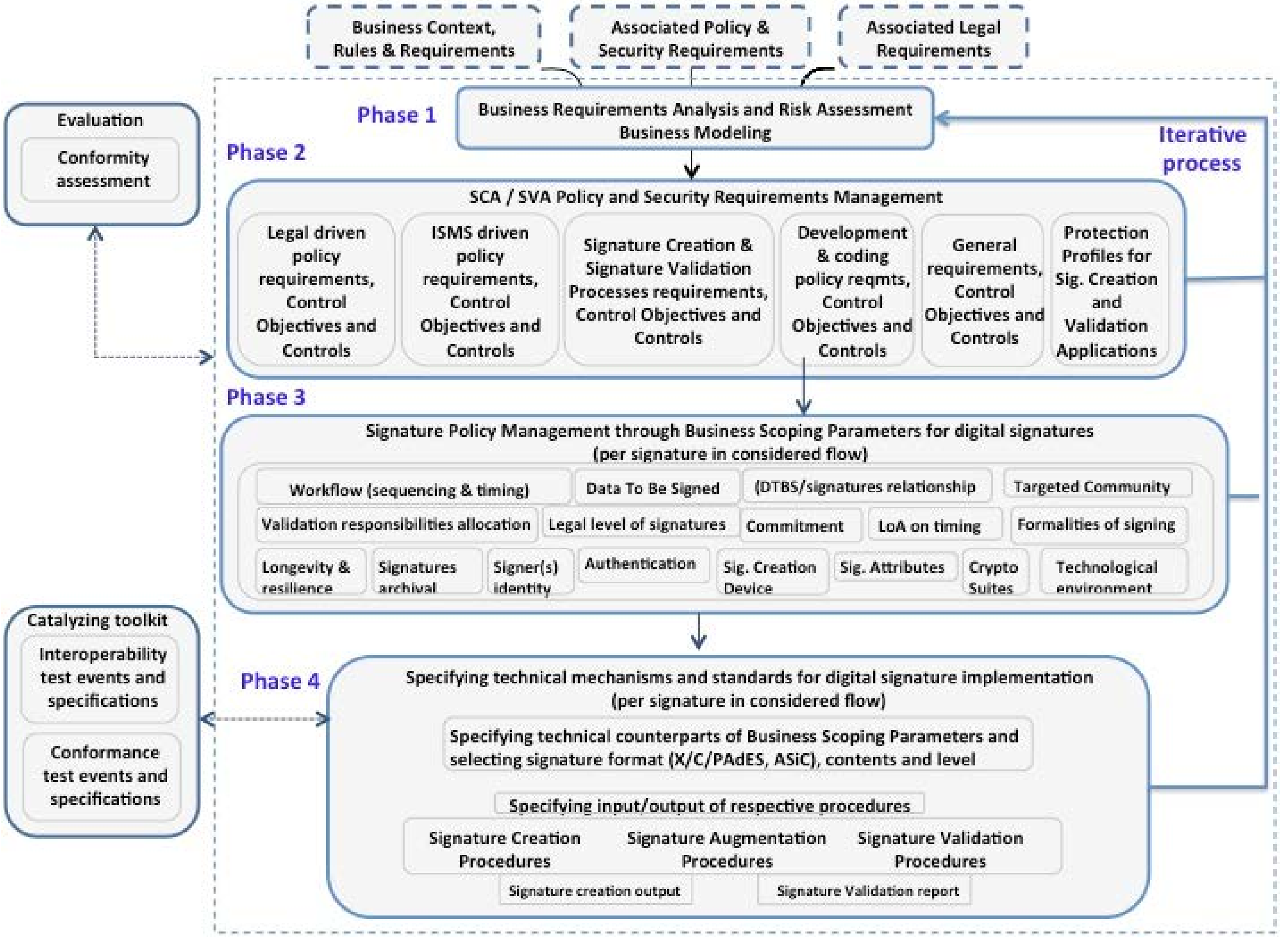

Business Context, Rules & Requirements
Associated Policy & Security Requirements
Associated Legal Requirements
Evaluation
Conformity assessment
Phase 1
Business Requirements Analysis and Risk Assessment
Business Modeling
Iterative process
Phase 2
SCA / SVA Policy and Security Requirements Management
Legal driven policy requirements, Control Objectives and Controls
ISMS driven policy requirements, Control Objectives and Controls
Signature Creation & Signature Validation Processes requirements, Control Objectives and Controls
Development & coding policy reqmts, Control Objectives and Controls
General requirements, Control Objectives and Controls
Protection Profiles for Sig. Creation and Validation Applications
Phase 3
Signature Policy Management through Business Scoping Parameters for digital signatures
(per signature in considered flow)
Workflow (sequencing & timing)
Data To Be Signed
(DTBS/signatures relationship
Targeted Community
Validation responsibilities allocation
Legal level of signatures
Commitment
LoA on timing
Formalities of signing
Longevity & resilience
Signatures archival
Signer(s) identity
Authentication
Sig. Creation Device
Sig. Attributes
Crypto Suites
Technological environment
Catalyzing toolkit
Interoperability test events and specifications
Conformance test events and specifications
Phase 4
Specifying technical mechanisms and standards for digital signature implementation
(per signature in considered flow)
Specifying technical counterparts of Business Scoping Parameters and selecting signature format (X/C/PAdES, ASiC), contents and level
Specifying input/output of respective procedures
Signature Creation Procedures
Signature Augmentation Procedures
Signature Validation Procedures
Signature creation output
Signature Validation report

Figure 1: Iterative process for implementing generation and validation of digital signatures

As a pre-requisite to the present guided implementation process (phase 1 in Figure 1), implementation of digital signatures should start with a proper, complete and as detailed as possible analysis of the business processes (description and modelling of complex business electronic processes) within which one or more digital signatures need to be implemented. This aims to ensure that all the details related to crucial aspects of the business electronic process are actually well captured and that the implementation of digital signatures does not miss any of them. It also includes a risk assessment, as a way of getting the needed information from which policy and security requirements are identified, so that once they are satisfied, stakeholders are sure that the implementation of digital signature is done in such a way that it actually counters the identified risks. The present document, however does not aim at providing a complete guide on these topics but at making readers aware of their relevance.

Phase 2 aims at elaborating the different sources of policy requirements and security requirements into controls' objectives, and controls to be implemented in the system. The present document does not aim at providing a complete guide on these topics; instead it makes readers aware of their existence and relevance and refers to ETSI TS 119 101 [i.11] and CEN EN 419 111 [i.12], [i.13], [i.14], [i.15], [i.16], which properly deals with these issues.

NOTE 1: Within the European Union legislation exists addressing the most relevant issues of digital signatures and that the Regulation (EU) No 910/2014 [i.26] has been published to achieve a more uniform legislative coverage. Additionally, Signature Creation Applications and Signature Validation Applications already exist on the market, which have been developed abiding by suitable security and policy requirements, simplifying their usage and integration within complex systems.

Phase 3 of the process aims at addressing and analysing the essential business scoping parameters in the light of the context where is conducted the business in which digital signatures have to be implemented. They will condition the whole implementation lifecycle from its inception to its deployment and maintenance. These parameters may actually come from different sources:

1) From the business electronic process itself. These are business scoping parameters inherent to the particularities of the business electronic process in which digital signatures have to be implemented. They are related to:

    - the data object to be signed;
    - the relationship between the signatures and the data objects to be signed;
    - the workflow of the documents and signed documents that is required by the business electronic process;
    - the requirements on the timing and sequencing of signatures generation and proof of timely generation;
    - requirements established by the business electronic process on privileges that a signer has to detain;
    - the time period after their generation, during which there is the need of being able to validate the signatures (longevity & resilience in Figure 1);
    - the archival requirements imposed by the business electronic process;
    - the specific community where the digital signatures will be exchanged;
    - the allocation of signature validation responsibilities, done by the business electronic process;
    - the fact that the business electronic process might envisage the generation/validation of digital signatures within mobile environment.

2) From the legal and/or regulatory framework where the business process is conducted. Lack of consideration of parameters depending on legal/regulatory framework when defining the strategy for implementing digital signatures would likely lead to implementations that do not properly satisfy what is established by the applicable legal and/or regulatory framework with all the negative consequences that this would bring. These business scoping parameters include: the quality level that the legal/regulatory framework imposes to certain signatures of certain business processes, parameters derived from what the legal/regulatory framework establishes with regards to the scope and purposes of signatures, parameters related to the formalities of signing, and those that come from requirements on the time period after signatures' generation, during which there is the need of being able to validate them.

3) Regarding the actor that actually generates the signature. These are business scoping parameters inherent to the actor, including its type (i.e. whether it is a natural person or a legal person), the type of the signing certificate owned by the signer, and the signer device.

4) Other. These are business scoping parameters coming from a variety of sources. Some of them might require the introduction of additional information within the signatures not already introduced. Other might require restricting the cryptographic suites.

The three aforementioned phases collectively aim at describing the conditions under which digital signatures will be used within a business or application domain and process, including the identification of the resulting digital signatures flow that will be considered in the context of:

1) a specific business application domain and/or process, with its own context and requirements;

2) its associated set of policies (e.g. corporate IT and security policies) including any existing signature policy to which the to be designed signature policy is subordinate;

3) its associated legal requirements; and

4) the associated risk assessment identifying risks for which digital signatures can be a mitigation tool but also risks induced by the use of digital signatures themselves in the business or application process.

Phase 4 aims at deciding at the technical level the means to be used for fulfilling all the business context related requirements that come from the business scoping parameters identified in the previous phase, and what standards within the Rationalized Framework are best suited for this. More specifically in this phase implementers will find guiding material that will help them in deciding:

1) The formats, contents, and levels of the digital signatures.

2) The technical procedures for generating, augmenting and validating digital signatures.

3) The protection profiles which their applications generating and/or validating digital signatures will be compliant with.

The table of contents for signature policy documents provided in ETSI EN 319 172-1 [i.17] should be used as a way to document the various decisions taken while executing the business driven digital signature implementation process for which guidance is provided in the present document. At the end of this iterative process, this would help to finalize and formalize the declaration of the practices and rules (to be) used when creating, augmenting, validating and preserving digital signatures in the concerned specific context (e.g. business process) into such a standardized signature policy document, if required.

Implementers may also use a set of available catalysing tools for assessing the conformance of their implementations to referenced standards (and consequently speeding up their production). This includes technical specifications for conformance testing and interoperability testing, and events for testing interoperability and conformance. This usage is shown in Figure 1 as a bidirectional dotted line connecting this phase with the round rectangle showing these tools. These tools are presented in clause 9.

Finally, it is quite likely that the applications to be put in place need to pass an evaluation process in order to be compliant with the regulatory/legal framework in force for the business context. Figure 1 shows this fact as a bidirectional dotted line connecting the round rectangle showing the evaluation with the dotted square enclosing the process itself. Some hints on the evaluation process are given in clause 10.

NOTE 2: Signature Generation Applications and Signature Validation Applications do exist in the market. The present document highlights a number of relevant aspects to consider when assessing the suitability of using one of these within a business process.

5 Analysing the Business Requirements

An accurate and complete business analysis, covering the entirety of the electronic business processes conducted, is essential for implementing digital signatures. Without such analysis it is highly unlikely that the implemented solution effectively supports the electronic business as it would be expected by its business managers and sponsors.

As mentioned before, it is not necessary to wait until the completion of the business analysis to start with the next tasks. This analysis, very likely, will be distributed among different iterations. However, it should have been completed at the end of all the iterations, in order to ensure that the whole set of requirements have actually been captured. When dealing with business with a certain degree of complexity this analysis should include the production of a business model, as a way of capturing all its relevant aspects.

The present document does not provide any further recommendations neither on the techniques used for analysing the business nor on how to distribute their completion throughout the different process iterations, as these issues are not within its scope.

The present document does not provide further recommendations neither on the techniques used for modelling the business nor on how to distribute its production throughout the different process iterations, as these issues are not within its scope. However, it signals the existence of tools for building these models that implementers may take into account, namely the Unified Modelling Language (UML) and some extensions specifically devoted to build up businesses models, or Business Process Management and Notation (BPMN).

A risk assessment should be conducted with regards to the usage of digital signatures as part of a business electronic process scenario. It aims at identifying risks for which digital signatures can be a mitigation tool but also risks induced by the use of digital signatures themselves in the business or application process. Implementers should also identify the relevant outputs of such assessment to be considered as input to the next phase, i.e. the establishment of the policy and security requirements for digital signatures generation and validation applications, as well as for the business rules to be accomplished by the implementation of digital signatures.

It is out of the scope of the present document to provide any further recommendation on risk analysis methodologies.

6 Managing the policy and security requirements

The second phase of the proposed guided implementation process is the management of the policy and security requirements that apply to the business electronic process and to the aimed integration of digital signatures within. This management includes the following tasks:

1) Identification of the relevant requirements imposed by different sources (among which the different policies in force within the business context).

2) Specification of the objectives to be achieved by the controls to put in place for satisfying the identified requirements.

3) Selection of the controls for achieving the aforementioned objectives.

While identifying the relevant requirements, implementers should take into account all their possible sources. Below follows the list of these potential sources of requirements:

1) Policies within the applicable regulatory or/and legal framework.

2) Policies concerned with the information security management of information technology risks (e.g. ISMS policies).

3) Specific processes for generating, augmenting and validating digital signatures.

4) Development and coding of applications dealing with the generation, augmentation and/or validation of digital signatures.

A complete set of these requirements is the starting point for the implementation of a solution that effectively supports the electronic business modelled.

The completion of this phase may be distributed among several iterations, and it may receive feedback from results and findings of ulterior phase. ETSI TS 119 101 [i.11] should be used to perform this task. This document provides general security and policy requirements to be considered when implementing applications for signature creation and signature validation.

7 Business scoping parameters

7.1 Introduction

The present clause provides details of the third phase of the proposed guided implementation process, which aims at properly addressing and analysing essential business scoping parameters in the light of the results of the two previous phases with regards to the specific business aspects and requirements of the business process where the digital signatures have to be implemented.

The business scoping parameters to be taken into account when implementing creation and validation of digital signatures are grouped as follows and addressed in the next clauses:

1) parameters mainly related to the specific application or business electronic process;

2) parameters mainly related to the regulatory/legal framework where the business will be conducted;

3) parameters mainly related to the different types of signing entities; and

4) other aspects that do not fall within the above three listed categories but are important to be addressed when implementing digital signatures.

7.2 Business scoping parameters mainly related with the business process

7.2.1 Introduction

When attempting to implement digital signatures in a business context, a number of business scoping parameters purely inherent to this context need to be taken into account, otherwise the risk of deploying a system that does not properly support the business in one way or the other is extremely high. These business scoping parameters will condition the whole system lifecycle from its inception to its deployment and maintenance. They, in consequence, will highly impact the selection of the right standards that deal with the direct management of digital signatures, namely with:

- their generation;
- their formats;
- their contents;
- their relative placement and relationship;
- their placement with respect to the signed data object(s);
- their resilience to time (longevity); or
- to cryptanalysis advances; and
- their validation.

This clause enumerates and provides details of the business scoping parameters mainly related with the business process itself that have a direct impact on the selection of standards.

7.2.2 BSP (a): Workflow (sequencing and timing) of digital signatures

7.2.2.1 Introduction

It is not unusual that business processes deal with workflows where different documents are generated and signed (by one or several signers) in different time instants and in a specific order that may or may not be changed. These inherent parameters of the workflow also have an impact in the selection of the suitable standards, and in consequence, implementers should take them into account. Below follow the most relevant ones:

1) Whether the time when a signature was applied is relevant or not (see clause 7.2.2.3).

2) For the not unusual situations where there are data objects that have to be signed by more than one signers, implementers should take into account the following aspects:

   - Whether the order in which the signatures are applied is relevant or not (see clause 7.2.2.3).

   - Whether all the signatures sign the same (the data object to be signed) or something different (the data object to be signed and one or more signatures previously applied to it, or even only one or more previously applied signatures) (see clause 7.2.2.2).

7.2.2.2 Multiple signatures

One data object can require more than one signature for having the required effect. In certain occasions this is actually required by the legal or regulatory Framework. When facing these situations, implementers should differentiate between:

1) Parallel signatures. These are signatures applied exactly to the same data object(s). They are mutually independent. Implementers should, in the cases where this type of signatures is required, identify what parallel signatures are required by the business process and/or its regulatory or legal framework, and where they have to appear, for giving the signed data object(s) its full effect.

2) Serial signatures. These are signatures applied to different data object(s) and whose order of generation is relevant. Implementers should, in the cases where this type of signatures is required, identify what serial signatures are required and what data object(s) each one should apply to. Implementers should clearly identify the order in which the different signatures have to be computed and where these signatures have to appear (sequencing of signatures is addressed within clause 7.2.2.3).

3) Counter-signatures. These are a special type of serial signatures, used in business processes that establish that a certain signature does not have any effect unless it is signed in turn by another signature, usually generated by a certain entity entitled for conferring such an effect to the first one. Countersignatures are applied one after the other and are used where the order in which the signatures are applied is important. They can be used to provide signatures from different parties with different signed attributes, or to provide multiple signatures from the same party using alternative signature algorithms, in which case the other attributes, excluding time values and information, will generally be the same. When such type of signatures appear in the workflow, implementers should take into account:

   - The relative position of countersignature and countersigned signature. Most of signature formats allow embedding the countersignature within the countersigned signature. However, some formats also allow keeping them physically detached and still indicating that a certain signature is actually a countersignature of another signature.

   - The actual meaning of a signature's countersignature, as this can impact the type of commitment endorsed by the counter-signer (see clause 7.3.3).

   - Whether there is the requirement of validating the to-be-countersigned signature before generating the countersignature.

   - Whether the counter-signer is required by the business process to countersign only the previously existing signature(s), or sign these ones and the signed data object(s), or even to add additional data object(s) and also sign it (them).

Implementers should also take into account that complex business processes would likely require to manage combinations of the different signature types aforementioned. A clear differentiation of the signatures types in each combination is crucial for properly selecting the most suitable standards and mechanisms.

Implementers should also identify whether the business process is actually demanding bulk signing, i.e. generate a significantly high number of serial signatures, as this may have an impact on, among other things, requirements for using devices specially designed for these purposes (e.g. hardware security modules).

7.2.2.3 Timing and sequencing

Implementers should identify those constraints on the timing and sequence of signatures generation imposed by the business process and/or its regulatory or legal framework for giving to the documents and signatures its full effect.

These constraints can, depending on the business process, be of very different nature: a mere specification of a deadline for the generation of each signature, a mere specification of the order in which documents and/or signatures have to be generated, detailed ranges of allowed time periods between the occurrence of the aforementioned events, specification of the order in which the signatures have to be validated, etc.

Implementers should also take into account the actual scope of these constraints, as they could apply to individual signatures, individual documents, multiple signatures, or multiple documents, depending of the workflow defined for the business process.

Special care should be paid when the business process and/or its regulatory or legal framework requires capability to prove that certain documents and/or signatures had been generated before a certain given time instant, as the satisfaction of this constraint would lead to use time assertions (like time-stamping techniques), significantly impacting the system being built. Should this be the case, implementers should carefully consider the level of assurance of the timing evidences (see clause 7.3.4).

Finally, implementers should also take into account any specific relationships that may appear between constraints in the sequencing of the generation of each signature and constraints established on potential roles/attributes to be held by its corresponding signer (see BSP (l) in clause 7.4.1).

7.2.3 BSP (b): Data Object(s) to be signed

Implementers of digital signatures in an application/business processes should clearly identify all the relevant aspects of the data object(s) to be signed. These aspects include:

1) The nature and the format of the data to be signed (e.g. binary, structured data, xml, PDF document, editable documents such as Word or ODF, multimedia packages, images, etc.). One crucial aspect for instance is the threat of existence of corruption agents (any code that changes the visualization of the data object to be signed) in these documents, which obviously should be avoided. The type of format for the data object to be signed can also be influenced by business risks or legal provisions, for example, when a specific provision is imposed on the formalities of signing (e.g. what you see is what you sign, see BSP(i) in clause 7.3.5).

NOTE: At present, digital signatures may be generated following XML, ASN.1 or PDF syntax. Although implementers could think that where XML data objects need to be signed, XAdES should be used, that where PDF documents need to be signed, PAdES should be used, and where ASN.1 or binary data objects need to be signed, CAdES should be used, in fact the decision on the signature syntax to be used mainly depends on the specificities of the business process where these signatures are going to be implemented: for instance, under certain circumstances there could be good reasons for taking a PDF document and build an XAdES signature enveloping it, or conversely for including a XML document within a PDF document and use PAdES signatures. Implementers should, in consequence, take into account the specificities of the business process before making any decision on the format(s) of the signature(s) to be implemented.

2) In those cases where the data object involved in a signing process is structured, it is worth identifying whether the whole data object or only certain part(s) have to be signed, as this is strongly related to the features offered by the different digital signature formats and would impact the final choice.

7.2.4 BSP (c): Relationships of signatures with signed data object(s) and signature(s)

Implementers of digital signatures in an application/business processes should pay attention to the relationships between each signature and its corresponding signed data object(s) and other signatures in the workflow. More specifically, they should consider:

1) The number of data objects that one signature actually signs. While all the signature formats are able to deal with one data object without any additional manipulation, the generation of a signature covering more than one object requires the application of different techniques depending on the signature format ranging from manipulating the data objects to be signed, to just take advantage of native mechanisms within the signature format for dealing with this kind of situations.

2) In special cases like bulk signatures (i.e. situations where there is a high number of data objects collectively signed by one signature), the benefits of using referencing mechanisms (like using signed `ds:Manifest` within XAdES signatures) which, in case of failure in the checks performed on some of the signed data objects, still would allow to affirm that the signature on the rest of the signed data objects is valid.

3) The recommended (as per the application/business processes) relative position of the signed data object and its signature. Three different situations can appear:

   - The signature is part of the data object that it signs (enveloped signature hereinafter).
   - The signature envelops the data object that it signs (enveloping signature hereinafter).
   - Signature and signed data object are detached (detached signature hereinafter).

   Also here the features offered by the different signature formats vary from one to the other, ranging from formats that by its own nature only cover one of the former situations, to formats that incorporate mechanisms for dealing with all of them.

   When one signature has to sign different data objects, the situation might become more complicated, as theoretically the application/business processes might require that the signature envelops some of the signed data object, and simultaneously be enveloped by another one and even be detached from others signed data objects. Although these so highly complex situations are not likely to be frequent, they should not be discarded by principle.

7.2.5 BSP (d): Targeted community

Implementers should clearly identify the community each document and its (their) signature(s) is (are) addressed to. Once this has been done, the implementers should identify any specific community rules in place. These rules could, for instance, state the conditions under which a certain signature can be relied upon, or include provisions relating to the intended effectiveness of signatures, where multiple signatures are required. These rules could greatly impact not only the formats of the signatures and their relationships with the signed documents, but also the specific standards and/or profiles to be used.

7.2.6 BSP (e): Allocation of responsibility of signatures validation and augmentation

When analysing the management of digital signatures within business processes, implementers should pay attention to the allocation of the responsibility of validating such digital signatures. Implementers should clearly distribute this responsibility among the following entities, according to the specificities of the business process:

1) Party relying on the signature. Although this is a common allocation, implementers should not assume that this would always be the most suitable one. In certain occasions it would merely be impractical or even too expensive. In consequence in certain scenarios it could be better to assign this responsibility to a subset of parties taking part in the transaction.

2) Digital signature Validation Trusted Services. This alternative would release the different relying parties of all the complexities associated with the validation of digital signatures and allocate them to specialized services conveniently supervised and/or accredited, ensuring the suitable level of trust in the validations performed.

3) Business processes where countersignatures are generated, could impose that counter-signing parties are required to perform a validation of the signatures to be counter-signed before actually countersigning them, as part of the data flow.

These three types of allocations are not necessarily exclusive, being it possible that some of them coexist within complex business processes.

Augmenting a digital signature is the process by which certain material (e.g. time-stamps, validation data and even archival-related material) is incorporated to the digital signatures for making them more resilient to change or for enlarging their longevity. Implementers should, in consequence, also identify requirements for augmenting digital signatures as they are validated and progress in the business process data flow.

7.3 Business scoping parameters mainly influenced by legal/regulatory framework where the business process is conducted

7.3.1 Introduction

The following BSPs may not strictly be influenced by legal provisions only but may also be driven by business considerations inherent to the concerned business process and its expectations with regards to the type of evidences resulting from the implementation of digital signatures.

7.3.2 BSP (f): Legal Effect of the signatures

For each signature identified in the concerned workflow, implementers should specify the signature's legal effect required in the context of the business process and the associated legal/regulatory requirements.

This parameter has an impact on the level of assurance on the authentication (i.e. the certification of the identification) of the actor generating a digital signature, on the class and policy requirements on the TSP providing such level of assurance, on the class of signature creation device used by such actors, on the use of a specific trust model for TSP issuing certificates (e.g. Trusted Lists, specific Trust Anchors in PKI hierarchy, use of CA certificate stores).

NOTE: Within the European Union, each type of electronic signatures has a different legal effect. Below follow the different types of electronic signatures:

- In accordance with Regulation (EU) No 910/2014 [i.26]: qualified electronic signatures (QES), advanced electronic signatures supported by a qualified certificate (AdES$_{QC}$), advanced electronic signatures, qualified electronic seals, advanced electronic seals supported by a qualified certificate, and advanced electronic seals.
- In accordance with Directive 1999/93/EC [i.30], CD 2009/767/EC [i.31], and CD 2011/130/EU [i.38] as amended by CD 2014/148/EU [i.34]: qualified electronic signatures (QES), advanced electronic signatures supported by a qualified certificate (AdES$_{QC}$), and advanced electronic signatures (AdES).

7.3.3 BSP (g): Commitment assumed by signer

Implementers should identify and describe the expected purpose of each signature and hence the meaning and the precise nature of the responsibility assumed by signing, or in other words the type of commitment for each digital signature in the considered business scenario and identified digital signature(s) flow. The description of such digital signature commitment types may be useful for avoiding potential ambiguity due to the fact that digital signatures may not provide equivalent contextual information as in the paper world leading to uncertainty about the signer's intention.

Implementers should also take into account that digital signatures supported by Public Key Infrastructures technologies uniquely link them to their signers.

Below follow some examples of different commitments:

1) digital signatures intended for data authentication purposes only;

2) electronic seals generated by legal persons;

3) digital signatures intended for entity authentication purposes only;

4) digital signatures created with the intention to sign the associated data (signed data object(s)):

   - as a draft;
   - as an acknowledgement of receipt;
   - as an intermediate approval as part of a decision process;
   - to indicate authorship or responsibility for a document (signed data);
   - to indicate having reviewed a document (signed data);
   - to certify that a document is an authentic copy;
   - to indicate witnessing of someone else signature on the same document (signed data);
   - having read, approving and being bound accordingly to the content of the data object that is signed;
   - etc.

The commitment type can be indicated in the digital signature:

- explicitly using a commitment type indication in the digital signature; or
- implicitly or explicitly from the semantics of the signed data object.

If the indicated commitment type is explicit by means of a commitment type indication in the digital signature, acceptance of a verified signature implies acceptance of the semantics of that commitment type. The semantics of explicit commitment types indications are specified either as part of the signature policy or can be registered for generic use across multiple policies.

The commitment type can be:

- defined as part of the signature policy, in which case the commitment type has precise semantics that is defined as part of the signature policy;
- a registered type, in which case the commitment type has precise semantics defined by registration, under the rules of the registration authority. Such a registration authority may be a trading association or a legislative authority.

The definition of a commitment type includes an identifier (URI or OID) and an optional sequence of qualifiers, which may provide additional information (for instance information about the context, be it contractual/legal/application specific).

If a digital signature does not contain a recognized commitment type then the semantics of the digital signature depends on the data object being signed and the context in which it is being used. How commitment is indicated using the semantics of the data object being signed depends on the specific business process context (for instance, some documents can explicitly indicate this commitment within the document itself).

7.3.4 BSP (h): Level of assurance of timing evidences

For each signature identified in the concerned workflow (see BSP(a) in clause 7.2.2) implementers should describe and specify the requirement on the level of assurance on the required timing evidences. This component is closely related to the components BSP(a) in clause 7.2.2, BSP(j) in clause 7.3.6, and BSP(k) in clause 7.3.7.

Implementers should distinguish between claimed assertions with regards to time information, and trusted time evidence, such as time assertions (time-stamps provided by trust service providers issuing time-stamp tokens, or evidence records issued by trusted services).

When trusted time evidence are required, implementers should consider the requirements and level of assurance associated respectively to the time-stamp tokens (whether they are qualified or not qualified time-stamp tokens, for instance), or evidence records and the providers, and on which type of information the time-stamp tokens, or evidence records, are generated (e.g. time information only, signed data object(s), signature(s), signature(s) and validation data, etc.).

7.3.5 BSP (i): Formalities of signing

One of the most important characteristics of a signature is the manner of its creation. Often referred to as the "ceremony of signing", it is the way the attention of the signer is drawn to the significance of the commitment that is being undertaken by performing this act of signing.

Implementers should identify requirements on any type of evidence of the will or intention to sign that would have an influence on the manner the digital signature is created. Implementers should also specify how the act of signing is presented to the signer in order to draw signer's attention to the significance of the commitment that is being undertaken under the signing process.

Such requirements will likely impact the signer interface design. Below follow some possible consequences:

1) Provide users with a "What You See Is What You Sign" environment.

2) Provide users with proper advice and information on the application's signature process.

3) Provide users with proper advice and information on the legal consequences.

4) Design the user interface in a way to guarantee, to the extent possible, a valid legal signature environment, including:

    - Implementation allowing and demonstrating clear expression of a will to sign and the user's intention to be bound by the signature.
    - Implementation allowing and demonstrating an informed consent.
    - Consistence between the use of the appropriate signature creation and verification data, signature creation device, the data to be signed and the expected scope and purpose of the signature (or the act of signing).

This BSP can impact the selection of appropriate protection profiles and conformity assessment schemes against which the signature creation application will be designed and assessed.

7.3.6 BSP (j): Longevity and resilience to change

Certain business processes and/or their regulatory or legal framework require that signatures have a certain longevity, i.e. that the signatures can be validated a certain time after their generation, being it possible in certain occasions that the implied elapsed time since their generation until their potential re-validation is of a certain number of years. Clauses 8.7.2.4 and 8.11 of the present document further elaborate the technical implications of achieving digital signatures whose validity needs to be reassessed long after they have been generated. The present document also uses the terms "long term digital signatures" or "long term signatures" for referring to these signatures.

Time passing has two different effects on the digital signatures: firstly, the validation material used for generating and validating them (certificates) can expire or even not be available anymore; secondly, the cryptographic algorithms (also including digest algorithms) can become weak as cryptology techniques and computer capabilities improve.

Longevity and resilience to change (understood as the resistance of digital signatures to the uncovering of weaknesses of their algorithms) are in consequence strongly related to each other.

Implementers should identify those signatures whose re-validation is required some time after their generation, as well as the time period during which their re-validation has to be made possible. These factors will help implementers in making right decisions when planning the means to be put in place for ensuring the required longevity of the signatures.

7.3.7 BSP (k): Archival

Archival is related with the longevity of the signatures. Regarding this issue, implementers should identify requirements on the archival of the signed data objects, their signatures and the material used for their validation, including requirements on whether archiving them together or not.

Implementers should respect the prerequisites of electronic archiving from the early stages of the design of new developments as well as when integrating digital signature solutions in current products. This aims to ensure proper implementation of electronic archiving where it is legally recognized and facilitate compliance with future regulations applicable on electronic archival.

7.4 Business scoping parameters mainly related to the actors involved in generating the signature

7.4.1 BSP (l): Identity (and roles/attributes) of the signer

In most cases, a signature is worthless if it cannot be attributed to the purported signer. Implementers should identify and specify:

1) who are the anticipated signers;

2) the associated signer identification rules;

3) if any, the rules applicable to the roles and/or attributes of the signers; and

4) if any, the requirements on an associated proof of authority.

They should, in consequence, identify and describe what are the necessary elements to ensure that a signature is that of a specified individual (whether a physical or legal person, a business or transactional functional entity, a machine, an application or server, etc.), i.e. what is the required identification element (identity attributes) for each type of signer.

EXAMPLE 1: For instance where a contract names an individual as a party to be bound by its terms, what is required as signer identification elements; names, date of birth, unique identification number, etc.

In some business scenarios, attributes owned by or the role played by a signer are at least as important as his identity.

EXAMPLE 2: For instance, some document (i.e. a contract) may only have the required effect if signed by an entity that plays a particular role, e.g. a Sales Director. In many cases, who the sales Director really is, is not that important, but being sure that the signer is empowered by his company to be the Sales Director is fundamental.

Under these circumstances, the term "signer role" does not refer to the "signing" role played by the signer in the digital signature supported business process (e.g. primary signature, countersignature) but relates to roles such as "official representative of a legal person" or "sales director", which can be claimed or certified, but which implies some attribute(s) associated with the signer. Implementers should describe the set of attributes, authorities and responsibilities which are associated with each signer, his access rights, or authority to sign, to act on behalf of the organization he purports to represent, etc.

Implementers should state the type of proof of authority to sign that is acceptable. This may include, among others:

1) proof that an employee or representative is authorized to enter into transactions over a specified value;

2) proof that delegation to sign has been authorized.

7.4.2 BSP (m): Level of assurance required for the authentication of the signer

Implementers should identify what is the level of assurance required for the authentication for the signer in each signature to be generated within the business process, i.e. what are the expectations in terms of trust on the signer identification (e.g. quality level of certificate).

EXAMPLE: For instance, certificates can be required to be qualified certificates and/or issued by an accredited, supervised, certified, or audited certification authority, or be issued according to a specific Certificate Policy, etc.

This, very likely, will not impact the specific contents of the signature itself but the signing application; nevertheless, a failure in reaching the level required by the legal/normative framework would lead to the potential rejection of the signatures in case of auditing or dispute.

7.4.3 BSP (n): Signature creation devices

Implementers should also identify any existing requirement on the signature creation devices (e.g. sole control) that will be used for generating the signatures within the business process, in order to ensure their fulfilment. Again, a failure to satisfy these requirements would lead to the potential rejection of the signatures in case of auditing or dispute.

7.5 Other Business scoping parameters

7.5.1 Introduction

The present clause addresses business scoping parameters that are not mainly related either to the business process, the legal/regulatory framework, and the signer.

7.5.2 BSP (o): Other information to be included within the signatures

Implementers should indicate, if considered necessary, any other applicable signature attributes, such as:

1) Geographic location where the signature was created. In some transactions, the purported location of the signer at the time the signature was created may need to be indicated. The incorporation of such a signature attribute (the location or jurisdiction in which the signature was made), might have legal consequences in the event of a dispute, in determining where the dispute should be heard and/or in determining the applicable jurisdiction.

2) Claimed signing time. Another example of applicable signature attribute is the signer's claim on the time at which he generated the signature. This is only to be considered as a claim and should not be considered as trusted unless the corresponding time is provided as the result of a trusted time service provided by a trusted time-stamping service provider.

3) Content time-stamp. Time-stamp tokens on the signed data object(s) can be incorporated into digital signatures using time-stamp tokens containers. In this way, a trusted secure time can be obtained before the document is signed and incorporated into the digital signature. This may not represent the precise signing time, since it can be obtained in advance. The signer can use these time-stamp tokens to prove that the signed object existed before the date included in the time-stamp token.

4) Indication of the signed data object(s) format. This could be necessary where it is important that when presenting the signed data object to a human user there is no ambiguity as to its presentation to the relying party, if the format is not implicit within the signed data object (for instance because a signature policy has established that the relying party system has to use one specific format for presenting the data object to the relying party as a mandatory requirement for successfully validating the signature). In order for the appropriate representation (text, sound or video) to be selected by the relying party such an indication can be incorporated into the signature by the signer.

7.5.3 BSP (p): Cryptographic suites

Implementers should describe and specify requirements on the robustness of cryptographic suites used to generate or augment each digital signature in the concerned business process. Implementers should carefully read ETSI TR 119 300 [i.28], the guidance document that specifically addresses area 3 (Cryptographic Suites) of the framework for standardization of signatures. They will find in this document guidance on how to select the cryptographic suites that properly fulfil the aforementioned requirements, and how to use ETSI TS 119 312 [i.29]. ETSI TS 119 312 [i.29] specifies cryptographic suites used for digital signature creation and verification algorithms.

7.5.4 BSP (q): Technological environment

From the business process specification, implementers should also pay attention to the technological environment where the data objects to be signed and the signatures will be managed, as this may have an impact on a number of technological decisions to be made, among which the signature formats to be used.

In particular, it is suggested to identify whether it is required (or even could be required in the future) to support the generation and/or validation of signatures within mobile or distributed environments. In case this requirement exists, implementers should clearly identify which type(s) of document(s) and which signatures within them need to also be managed within mobile/distributed environments. This is extremely relevant, as the mobility aspect may require making use of specific services for supporting these tasks, and in consequence, to use specific sets of standards.

8 Selecting the most appropriate standards, options, and technical mechanisms

8.1 Introduction

The framework for standardization of signatures includes standards defining three digital signature formats:

1) CAdES (defined in ETSI EN 319 122-1 [i.2] and ETSI EN 319 122-2 [i.3]);

2) XAdES (defined in ETSI EN 319 132-1 [i.4] and ETSI EN 319 132-2 [i.5]);

3) PAdES (defined in ETSI EN 319 142-1 [i.6] and ETSI EN 319 142-2 [i.7]).

It also includes one standard defining a container able to embed several data objects and detached digital signatures that selectively sign some of them: the ASiC container (defined in ETSI EN 319 162-1 [i.8] and ETSI EN 319 162-2 [i.9]).

NOTE: When making references to specific parts of XAdES, PAdES, CAdES and ASiC specifications, the present document uses the clauses numbering of ETSI EN 319 1x2, which differs, in most of the cases, from the numbering implemented in the ETSI TSs specifying legacy CAdES signatures [i.40] and [i.45], legacy PAdES signatures [i.41] and [i.44], legacy XAdES signatures [i.39] and [i.43], and legacy ASiC containers [i.42] and [i.46]. Nevertheless, whenever this occurs, the text within the present document makes it easy to identify what is the relevant part of the aforementioned specifications the text is referencing, and in consequence, it is not difficult to identify the referenced material even in the aforementioned ETSI TSs.

Hereinafter, when referring to elements or properties of XAdES signatures, their prefixed qualified names will be used. Table 1 shows the prefixes used for the different URI namespaces used in the XML Schema specified by ETSI EN 319 122-1 [i.2].

Table 1: Prefixes assigned to namespaces' URIs

XML Namespace URI	Prefix
http://www.w3.org/2000/09/xmldsig#	ds
http://uri.etsi.org/01903/v1.3.2#	xades
http://uri.etsi.org/01903/v1.4.1#	xadesv141

8.2 Format of signatures: CAdES, XAdES or PAdES

8.2.1 Introduction

The suitable format of signature strongly depends on the business process itself. Under certain circumstances it clearly makes one option much better suited than the others. Under other circumstances, though, the advantages of a choice among other choices are not so clear and even arguable.

This clause lists some considerations that implementers may use to decide the format(s) of digital signatures to be implemented in their business processes.

However, it is worth to address first PAdES signatures as they represent a special case, because they actually are built on different formats. PAdES signatures conformant to ETSI EN 319 142-1 [i.6] and to ETSI EN 319 142-2 [i.7], clause 5, build on CAdES signatures. PAdES signatures conformant to ETSI EN 319 142-2 [i.7], clause 4, build on CMS signatures. Finally, PAdES signatures conformant to ETSI EN 319 142-2 [i.7], clause 6, build on XAdES signatures. Clause 6 of ETSI EN 319 142-2 [i.7] defines two profiles groups: one for XAdES signatures on XML documents embedded within PDF containers, and another one for XAdES signatures on XFA forms.

Hereinafter, the following acronyms will be used for clearly indicating the PAdES signatures types that are addressed in the text:

1) PAdES will be used in sentences that apply to signatures conformant with ETSI EN 319 142-1 [i.6] or with ETSI EN 319 142-2 [i.7].

2) PAdES-CMS, will be used in sentences that apply only to PAdES signatures conformant with ETSI EN 319 142-2 [i.7], clause 4 ("Profile for CMS digital signatures in PDF").

3) PAdES-OnCAdES will be used in sentences that apply only to signatures conformant to ETSI EN 319 142-1 [i.6] or ETSI EN 319 142-2 [i.7], clause 5.

4) PAdES-NoXML will be used in sentences that apply only to signatures conformant with ETSI EN 319 142-1 [i.6] or ETSI EN 319 142-2 [i.7] except clause 6 of ETSI EN 319 142-2 [i.7].

5) PAdES-XML will be used in sentences that apply only to PAdES signatures conformant with ETSI EN 319 142-2 [i.7], clause 6 ("Profiles for XAdES Signatures signing XML content in PDF").

6) PAdES-XML-EMB will be used in sentences that apply only to PAdES signatures conformant with ETSI EN 319 142-2 [i.7], clause 6.2 ("Profiles for XAdES signatures of signed XML documents embedded in PDF containers").

7) PAdES-XML-XFA will be used in sentences that apply only to PAdES signatures conformant with ETSI EN 319 142-2 [i.7], clause 6.3 ("Profiles for XAdES signatures on XFA forms").

8) Wherever there is the need to signal one specific level of PAdES signature, the present document will use the level identifier specified within ETSI EN 319 142-1 [i.6] or ETSI EN 319 142-2 [i.7].

NOTE: This happens for instance in clause 8.8, where an explicit reference to PAdES-E-BES signatures is made.

8.2.2 Format of the document

This is one of the first elements that implementers have to take into account. In principle, the closer the formats of signatures and documents are, the better.

Under this perspective, for XML documents, XAdES signatures would be the natural option.

Also in principle PAdES-NoXML signatures would be the natural option for embedding digital signatures within PDF documents. PAdES-XML-XFA would be the natural option for signing XFA forms, and PAdES-XML-EMB would be the natural option for signing XML documents that are embedded within a PDF container.

CAdES is also in principle the natural option for signing data objects whose structure has been defined in ASN.1, and that have been encoded in DER or BER.

For other binary formats, both XAdES and CAdES would initially work properly. Nevertheless, depending on the specific business process, one format could present advantages that would make that format more advisable. Implementers should, in consequence, analyse at least the aspects that are mentioned in subsequent clauses.

Despite what it has been said before, there are a number of additional considerations that modulate the former assertions and even, under certain circumstances, could fully justify selecting a signature format not considered initially as "the natural option".

These considerations are addressed in subsequent clauses 8.2.3 and 8.2.4.

8.2.3 Relative placement of signatures and signed data objects

8.2.3.1 Introduction

This clause provides information on how the different formats can manage different combinations with regards to the relative placement of signatures and signed data objects.

In essence, one may distinguish three pure relative placements of signatures with regards to where the signed data objects may appear: enveloped, enveloping and detached signatures. A certain business process can require some form of combination of these placements (for instance, the business process can require that one of the signatures of a signed data object is enveloped by the object, while it also requires that another signature is actually detached or even enveloping the signed data object; it could even be possible that a certain signature is required to be enveloped in one signed data object, and at the same time, detached from a second signed data object signed by the same signature). Under these circumstances, implementers should carefully analyse the features provided by each format and also consider the potential benefits that a packaging mechanism like the one provided by ASiC could bring to the solution.

8.2.3.2 Enveloped signatures

PAdES-NoXML signatures are, by their own document-centric nature, enveloped signatures, i.e. they are embedded within the PDF document they sign. Also PAdES-XML signatures can be embedded within the object they sign.

CAdES signatures can be embedded within objects whose structure is defined in ASN.1 as long as this structure defines fields for embedding them, or within S/MIME [i.50] messages. However, neither CMS nor CAdES specifications defines a mechanism for explicitly referencing signed data objects that are external to the signature. This means that very likely, under these circumstances, the parts of the enveloping data object actually signed have to be specified separately, when specifying the syntax and semantics of the enveloping data object itself. In terms of implementation, this means that an application that manages CAdES signatures would require additional software for knowing what the CAdES signature is actually signing if it is embedded within an ASN.1-defined object.

XAdES signatures may be embedded within XML documents. Unlike CAdES, XAdES inherits the XML Signature [i.37] mechanisms for explicitly referencing any signed data object, and in consequence, a standardized way of retrieving such data objects (the `ds:Reference` element). This referencing mechanism allows explicitly referring to (and actually sign) the whole XML document or only parts of it. The important consequence is that any XAdES application based on another one claiming conformance against XML Signature W3C Recommendation does not require any additional software for identifying what the signature is actually signing.

8.2.3.3 Enveloping signatures

PAdES-NoXML signatures are not allowed to envelop the data object they sign, by their own document-centric nature. However, PAdES-XML-EMB can envelope the data object they sign.

CAdES signatures, as they are built on CMS signatures, can envelop the signed data object, by encapsulating it within the `encapContentInfo`'s `eContent` field. CAdES applications built on applications claiming conformance to CMS do not require additional software for identifying what the signature is actually signing.

XAdES signatures can also envelop the signed data object. When this is a binary object, it is previously base64 encoded, which increases its size, and encapsulated within a `ds:Object` element. Additionally, if the signed data object is XML the signature can also sign part(s) of the object using the referencing mechanisms specified in XML Signature [i.37]. XAdES applications claiming conformance against the XML Signature W3C Recommendation [i.37] do not require additional software for identifying what the signature is actually signing.

8.2.3.4 Detached signatures

PAdES-NoXML signatures are not allowed to exist detached from the PDF document they sign, by their own document-centric nature. However, PAdES-XML can be detached from the data objects they sign.

CAdES signatures can be detached from the signed data object, by leaving the `encapContentInfo`'s `eContent` field empty. However, neither CMS nor CAdES incorporate mechanisms that make it explicit any hint on how to retrieve the detached signed data object. This means that the location of the detached signed data object has to be specified separately (as it happens, for instance in S/MIME [i.50]). This also can be done using ASiC containers (see clause 8.2).

XAdES signatures also can be detached from the signed data object. Unlike CAdES, XAdES inherits the XML Signature mechanisms (URI references) for explicitly referencing any signed data object, included the detached ones, and in consequence, a standardized way of retrieving such data objects. As specified in IETF RFC 3986 [i.51], URI references can be absolute or relative. Use of absolute URIs does not allow changing the location of the signed data objects. Use of relative URIs does allow changing the location of the signed data objects as long as it is ensured that the URI obtained after completing the reference resolution process is the URI of the new location of the data object. This can be achieved for instance, changing properly also the XAdES signature location.

ASiC containers allow carrying within a container both XAdES signatures and detached signed data objects using relative URI references. Within these packages the relative positions between signatures and signed data objects are preserved even if the location of the package (and in consequence of the signatures and the signed data objects) is changed.

8.2.3.5 Simultaneous multiple relative positions

Due to the referencing mechanism inherited from XML Signature [i.37], one XAdES signature can be, at the same time, enveloping one of the data objects that it signs, be enveloped by another data object that it signs, and be detached from another data object that it signs.

PAdES-XML-EMB signatures can be at the same time, enveloped within one XML signed document, and detached from another signed data object.

8.2.4 Number of signatures and signed data objects

8.2.4.1 Introduction

One of the elements to be also taken into account when specifying the signature format to be implemented is the cardinality of the relationship between signed data objects and its (their) signature(s). Different situations can appear, depending on the business case, which are explored in clauses 8.2.3.2, 8.2.3.3 and 8.2.3.4.

8.2.4.2 One document is signed by only one signature

The three formats deal well with this situation.

8.2.4.3 One document is signed by more than one signature

When one document requires to be signed by more than one signature, implementers should take into account a number of considerations that are presented below.

Any PAdES-NoXML signature signs any other PAdES-NoXML signature already present within the document when it is created: they are always serial signatures; PAdES-NoXML signatures do not allow generation of parallel signatures. More than one PAdES-XML signature can be used for signing the same data object. In addition to that, as they are XAdES signatures, any combination of parallel and serial signatures is allowed.

As CAdES signatures build on CMS signatures, they also incorporate within its specification native means for managing parallel signatures on one data object. CMS and CAdES signatures can also incorporate countersignatures as an unsigned attribute, which allows a sequence of countersignatures on one of the parallel signatures. However, arbitrary combinations of parallel and serial signatures are not easily implementable, as CMS and CAdES lack mechanisms for explicitly referencing signed data objects, and in consequence, applications should be configured for properly managing each specific combination.

XAdES signatures inherit from XML Signatures their native mechanisms for explicitly referencing and processing the data objects they sign (including other XML or XAdES signatures). XAdES signatures can also incorporate an unsigned property that encapsulates a countersignature (be it a XML Signature or a XAdES signature), or can countersign a detached XML or XAdES signature (in which case, the `Type` attribute of the `ds:Reference` element referencing the countersigned signature has the value "http://uri.etsi.org/01903#CountersignedSignature"). This makes any XAdES application fully compliant with XML Signature W3C Recommendation inherently able to manage any number of signatures signing one XML document (completely or partially), with any combination of serial and parallel signatures, and without any restriction on the relative placement of signatures and the signed data object. However, unlike CAdES, no standard mechanism is defined within XML Signatures W3C Recommendations or XAdES specifications for placing together a set of parallel XAdES signatures. This requires additional specifications. At present there are several examples on how this can be achieved; below follows some of them:

1) Embed several XAdES signatures within a XML document, each one being a parallel signature of the document itself or certain parts of the document.

2) Define containers that specify elements where parallel XAdES signatures on the same data object are placed (e.g. ASiC).

Several XAdES signatures can also sign one binary data object. However, in this case, XAdES signatures can only sign the complete data object.

8.2.4.4 One signature is required to sign more than one data object

PAdES-NoXML signatures only sign a PDF container by their own document-centric nature. Anything that is within the PDF container is signed, but nothing else. PAdES-XML signature, being XAdES signatures, can sign more than one data object within the XML content of the PDF container. Additionally, PAdES-XML-EMB can also sign data objects that are outside the PDF container.

CAdES signatures are not able by their own, to sign more than one data object. This requires doing some previous work on the signed data objects or use CAdES within appropriate containers. Below follow some examples on how to achieve this:

1) Sign a multi-part MIME object, as specified in S/MIME [i.50].

2) Define containers that specify elements where one CAdES signature can indirectly sign several data objects within the container (e.g. ASiC).

XAdES signatures incorporate native mechanisms for signing more than one data object. Additionally, the usage of signed `ds:Manifest` element also allows that if the validation of the collective digital signature succeeds and some check of certain signed data objects fails, applications can still decide that the rest of the data objects are correctly signed and proceed with their processing. In other words, this mechanism allows that failures in some individual checks of the signed data objects do not invalidate the whole collective signature.

8.3 A container for packaging together signatures and detached signed data objects

Whenever the business process analysis shows that the business electronic processes require to generate and manage detached signatures, and advices that, in order to facilitate such a management, it is worth to embed both the signatures and their signed objects within a container, implementers are referred to ETSI EN 319 162-1 [i.8] and ETSI EN 319 162-2 [i.9].

ETSI EN 319 162-1 [i.8] specifies containers that hold one or more detached signatures (XAdES or CAdES) and the data objects signed by these signatures. These containers allow managing detached signatures and their signed data objects in a standardized way. ASiC containers can also encapsulate time-stamp tokens and evidence records.

If there is only one document to be signed by several detached signatures, implementers should use the ASiC Simple (ASiC-S) container type. Implementers are referred to clause 5 of ETSI EN 319 162-1 [i.8], clause 4.3.

If, on the contrary, there are more than one data objects signed by detached signatures, then implementers should consider using the ASiC Extended (ASiC-E) container type. Implementers are referred to clause 6 of ETSI EN 319 162-1 [i.8], clause 4.4. An ASiC-E container can include several data objects and several signatures, detached from the aforementioned data objects, each signature selectively signing some of them. Objects of any format are allowed. Either CAdES or XAdES signatures are allowed within one ASiC container.

If the embedded signatures are CAdES signatures, the ASiC-E container incorporates one additional XML file (known as ASiCManifest file) for each CAdES signature embedded within the container. Each ASiCManifest file references (using URIs) a list of the files present within the container. The ASiCManifest file also contains the digest values of the aforementioned referenced files. Each ASiCManifest file also references one of the files containing a CAdES signature. The CAdES signature referenced from the ASiCManifest file, signs this ASiCManifest, including the digest values of the referenced files, which makes the CAdES signature an indirect signature of the referenced files within the container. Consequently, the ASiCManifest file standardizes a mechanism for referencing data objects indirectly signed by detached CAdES signatures within ASiC containers.

An ASiCManifest file can also reference a file containing a time-stamp token instead a CAdES signature. This allows to incorporate within the ASiC container a time-stamp token on a set of files present within the container.

If the embedded signatures are XAdES signatures, ASiC relies on the native mechanisms of XML Signatures (i.e. the usage of `ds:Reference` elements) for referencing all the documents signed by them. The XAdES signatures themselves appear within one or more files whose names follow the pattern "*signatures*.xml". ASiC containers provide a standardized way of packaging together parallel XAdES signatures.

8.4 Baseline or extended/additional?

ETSI EN 319 122-1 [i.2], clause 6, ETSI EN 319 132-1 [i.4], clause 6, and ETSI EN 319 142-1 [i.6], clause 6 specify baseline signatures. ETSI ETSI ETSI EN 319 162-1 [i.8], clause 5 specifies baseline ASiC containers.

Baseline signatures and containers are meant to minimize the number of options in the usage of CAdES, PAdES, XAdES signatures and ASiC containers, as well as to maximize interoperability.

ETSI EN 319 122-2 [i.3] and ETSI EN 319 132-2 [i.5] specify extended signatures. ETSI EN 319 142-2 [i.7] specifies additional PAdES signatures profiles (this term is due to the fact that historically PAdES specification was built as a set of profiles, instead as a unique specification, as originally happened with CAdES and XAdES). ETSI EN 319 162-2 [i.9] specifies additional ASiC containers. These signatures and containers offer a higher degree of optionality than the baseline signatures and containers.

All the digital signatures and containers specified in all these documents aim at supporting electronic signatures, advanced electronic signatures, qualified electronic signatures, electronic seals, advanced electronic seals, and qualified electronic seals as per Regulation (EU) No 910/2014 [i.26].

Implementers should, first check whether the business context, and the regulatory/legal framework explicitly require the usage of the baseline signatures and/or baseline containers. If this is not the case, implementers should check whether the requirements imposed by the business process, and the legal/regulatory framework (including digital signatures life-cycle management related issues) can be satisfied with the functionality provided by baseline signatures and/or baseline containers. If so implementers should consider their usage. Otherwise, implementers should proceed to use the specifications for extended signatures or additional containers, deciding what specific contents should be incorporated to the signatures/containers.

8.5 Selecting the proper quality of the signature

Where the legal/regulatory framework requires that digital signatures satisfy certain legal requirements, implementers should put in place the corresponding technical mechanisms for ensuring that these requirements are met.

Implementers should take into consideration that for ensuring a certain quality for the signature(s), they have to ensure that the following elements fulfil the legal requirements:

1) the Signing Device,

2) the Certificate Issuance,

3) the Independent Assurance on (1) and (2),

4) the Signature Cryptographic Suite,

5) the Signature Application, and

6) the desired longevity of the signatures,

7) the desired protection features (level) for the signatures, and

8) the Independent Assurance on (7).

8.6 Mapping formalities of signing to the electronic domain

Implementers should ensure that the provided signing environment gives satisfaction to the right subset of characteristics listed within clause 7.3.5 as applicable to the specific legal/regulatory framework and business process.

8.7 Satisfying timing and sequencing requirements

8.7.1 Satisfying sequencing requirements

8.7.1.1 Introduction

As mentioned before, certain business processes can impose constraints in the order to be followed for generating signatures on specific data objects.

Although these constraints always apply to counter-signatures (it is obvious that a counter-signature will be generated after the counter-signed signature), they can also be imposed to parallel signatures. In this later case any specific requirement on their sequencing can lead to the addition of a generation time indication (see next clause) or even to the specification of their relative placement.

8.7.1.2 Including counter-signatures

Implementation in CAdES, PAdES and XAdES signatures

PAdES, CAdES and XAdES signatures allow counter-signing. In all the cases, the counter-signatures can be in turn PAdES, CAdES or XAdES signatures respectively.

Implementers are referred to clause 5.2.7 of ETSI EN 319 132-1 [i.4] when implementing counter-signatures for XAdES signatures. This format allows managing counter-signatures in two ways:

1) Embedded within the counter-signed signature. Implementers are referred to clause 5.2.7.2 of ETSI EN 319 132-1 [i.4]. It specifies `xades:CounterSignature` unsigned property, a container for a `ds:Signature` element which can be a regular XML signature or a XAdES signature counter-signing the embedding signature. `xades:CounterSignature` signs the `ds:SignatureValue` element of the countersigned XAdES signature but can also sign other data objects (for instance the data object(s) that the countersigned XAdES signature signs).

2) Not embedded within the counter-signed signature. This is achieved by setting the `Type` attribute of the counter-signature's `ds:Reference` element referencing the counter-signed signature, to a pre-defined value. This allows to effectively detaching both signatures while making it explicit that one is a counter-signature of the other. Implementers are referred to clause 5.2.7.1 of ETSI EN 319 132-1 [i.4].

Implementers are referred to clause 5.2.7 of ETSI EN 319 122-1 [i.2], when implementing CAdES signatures, which specifies the `counter-signature` unsigned attribute, a container for a regular CMS or a CAdES signature counter-signing the embedding signature. This unsigned attribute signs the `signature` field.

When PAdES signatures are used, implementers should take into account the following considerations:

1) Counter-signatures for PAdES-NoXML signatures are other serial PAdES-NoXML signatures added afterwards. They sign all the previously existing data within the PDF container, including signed data objects and any signature. Usage of the `counter-signature` attribute is not allowed.

2) PAdES-XML signatures allow the usage of the `xades:CounterSignature` unsigned property (clauses 6.2.2.6 and 6.3.2.5 of ETSI EN 319 142-2 [i.7]).

8.7.2 Satisfying timing requirements

8.7.2.1 Introduction

PAdES, CAdES and XAdES signatures provide containers including information of different nature about the time when the signature and/or the signed data objects have been generated. Implementers can:

1) Include within a digital signature one or more time-stamp token(s) on the data objects to be signed, before the signature is actually generated, in case it is (they are) required to prove that certain data object(s) to be signed had been generated before a certain given time instant.

2) Include within a digital signature an indication of the claimed signature generation time. This is understood as a claim made by the signer and as such is generally treated by the relying parties, i.e. it does not deserve, generally speaking, the same confidence as a trusted time indication like, for instance, a time-stamp token generated by a Time-stamp service provider (unless the signer is an entity entitled for being trusted when claiming that time –a certain Registered Electronic Mail Management Domain could be an example).

3) Include within a digital signature one or more time-stamp tokens on the signature generated. Each time-stamp token, which is calculated on the signature, proves that the signature was generated before the time indicated within the time-stamp token.

Clauses 8.7.2.2, 8.7.2.3 and 8.7.2.4 provide additional details of these mechanisms.

8.7.2.2 Time-stamping the data objects to be signed before signature generation

PAdES, CAdES and XAdES signatures provide containers for including time-stamp tokens on the data objects to be signed before the actual signature is generated.

Implementers are referred to clauses 5.2.8.1 and 5.2.8.2 of ETSI EN 319 132-1 [i.4], when implementing XAdES signatures. Clause 5.2.8.1 specifies `xades:AllDataObjectsTimeStamp` signed property, a container for a time-stamp token that collectively time-stamps all the data objects referenced in the `ds:SignedInfo` element within the XAdES signature, except the `xades:SignedProperties`. Clause 5.2.8.2 specifies `xades:IndividualDataObjectsTimeStamp`, a container for a time-stamp token on one or several of the data objects referenced within the `ds:SignedInfo` or within a signed `ds:Manifest` element.

Implementers are referred to clause 5.2.8 of ETSI EN 319 122-1 [i.2], when implementing CAdES signatures, which specifies the `content-time-stamp` signed attribute, a container for a time-stamp token on the signed data object.

When PAdES signatures are used, implementers should take into account the following considerations:

1) Clause 5.4.2 of ETSI EN 319 142-1 [i.6] specifies the `Document Time-Stamp` dictionary, a special type of PDF signature dictionary that contains a time-stamp on all the previously existing data within the PDF container.

2) PAdES-XML signatures make use of the `xades:AllDataObjectsTimeStamp` and `xades:IndividualDataObjectsTimeStamp` signed properties (ETSI EN 319 142-2 [i.7], clauses 6.2.2.5 and 6.3.2.4).

8.7.2.3 Including claimed signing time

Rationale

It is a common use case that the signer wishes to make a claim of the time when generated the signature. This time, is not, in general, a trusted time.

Implementation in CAdES, PAdES and XAdES

CAdES, PAdES, and XAdES signatures provide mechanisms for incorporating as signed information, an indication of this claimed signing time.

Implementers are referred to clause 5.2.1 of ETSI EN 319 132-1 [i.4], when implementing XAdES signatures, which specifies the `xades:SigningTime` signed property.

Implementers are referred to clause 5.2.1 of ETSI EN 319 122-1 [i.2], when implementing CAdES signatures, which specifies the `signing-time` signed attribute.

When PAdES are used, implementers should take into account the following considerations:

1) PAdES-OnCAdES signatures, requiring incorporation of the claimed signing time, use the M entry of the signature dictionary (ETSI EN 319 142-1 [i.6], clause 6.3, and ETSI EN 319 142-2 [i.7], clause 5.3).

2) Within PAdES-XML-EMB signatures, the claimed signing time, if required, will be indicated within `xades:SigningTime` signed property (ETSI EN 319 142-2 [i.7], clause 6.2.2.5).

3) Within PAdES-XML-XFA signatures, the claimed signing time, if required, will be indicated by the content of the `CreateDate` element defined within the XMP `ns.adobe.com/xap/1.0/` namespace (ETSI EN 319 142-2 [i.7], clause 6.3.2.4).

8.7.2.4 Including time-stamp token on the digital signature value

Rationale

Signature time stamping is strongly related with the longevity of digital signatures. The longevity of a digital signature is the time period during which it is ensured the capability of reassessing its technical validity (or in other words, of providing long term evidence of its validity). It is not uncommon that it is required to enlarge the longevity of a digital signature until a time that goes beyond the expiration or the revocation of any of the certificates within the certification path of the signer's certificate, and beyond the break of any of the algorithms (including digest algorithms) used for its generation.

A signer, verifier or both can be required to provide on request, proof that a digital signature was created or validated during the validity period of all the certificates that make up the certificate path. In this case, the signer, verifier or both will also be required to provide proof that all the end entities and CA certificates used were not revoked when the signature was created or validated (it would be quite unacceptable to consider a signature as invalid even if the keys or certificates were only compromised later). Time-stamp tokens generated by trusted TSAs can provide such type of proof.

The time indicated within the time-stamp token defines a lower time boundary for the existence of the time-stamped digital signature. Finally, time-stamping a digital signature allows distinguishing:

1) first between signatures generated before the end of the validity period of the signer's certificate and signatures generated after the end of this period; and

2) second between signatures generated before the revocation of the signer's certificate and signatures generated after the revocation of this period.

Additionally, the signature time-stamp enlarges the signature's longevity at most until the first expiration of a certificate within the time-stamp token certification path (if there has not been any revocation before). In fact, the first measure within ETSI digital signature formats to allow that the technical validity of a digital signature can be reassessed during a period of time that goes beyond of the expiration or the revocation of any of the certificates within the certification path of the signer's certificate, and beyond the break of any of the algorithms (including digest algorithms) used for its generation, is the incorporation of a time-stamp token on the signature before any of the aforementioned events occur.

Validators can, in consequence, prove that the signature was valid when generated, even beyond the validity period of any of the certificates within the certification path of the signer's certificate, as long as:

1) they have access to the validation material of the certificates within the certification path of the signer's certificate, and that this material actually proves that at the time indicated within the signature time-stamp token none of them was revoked; and

2) none of the certificates within the certification path of the time-stamp token signing certificate, have expired or have been revoked at the time when the validation is performed.

If an entity wants to keep the capability of reassessing the validity of a digital signature, this entity will have to ensure that it has obtained a valid time-stamp for it, before the signer's certificate (and any certificate involved in the validation) expires or is revoked. The sooner the time-stamp is obtained after the signing time, the better.

It is important to note that signatures can be generated "off-line" and time-stamped at a later time by anyone, for example by the signer or any recipient interested in the signature. The time-stamp can thus be provided by the signer together with the signed data object, or obtained by the recipient following receipt of the signature.

The validation mandated by the signature policy can specify a maximum acceptable time difference which is allowed between the time instant indicated in the claimed signing time element (see clause 8.7.2.3 of the present document) and the time indicated by the time-stamp token on the signature.

If there is the requirement of proving the correctness of the status of the certificates within the time-stamp token certification path beyond this time, then there is the need of protecting this time-stamp token and, by doing so, enlarging the signature's longevity. See clause 8.11 for more details on the technical mechanisms available in CAdES, PAdES and XAdES for enlarging signatures longevity and supporting their lifecycles.

Implementation in CAdES, PAdES and XAdES

Implementers are referred to clause 5.3 of ETSI EN 319 132-1 [i.4], when implementing XAdES signatures, which specifies the `xades:SignatureTimeStamp` unsigned property.

Implementers are referred to clause 5.3 of ETSI EN 319 122-1 [i.2], when implementing CAdES signatures, which specifies the `signature-time-stamp` unsigned attribute.

When PAdES are used, implementers should take into account the following considerations:

1) PAdES-CMS signatures can incorporate a time-stamp token as specified in ISO 32000-1 [i.49], clause 12.8.3.3.1 (ETSI EN 319 142-2 [i.7], clause 4.2.4).

2) PAdES-OnCAdES signatures make use of the `signature-time-stamp` unsigned attribute (ETSI EN 319 142-1 [i.6], clause 6.3, and ETSI EN 319 142-2 [i.7], clause 5.3).

3) PAdES-XML signatures make use of the `xades:SignatureTimeStamp` unsigned property (ETSI EN 319 142-2 [i.7], clauses 6.2.2.5 and 6.3.2.4).

8.8 Including indication of commitments assumed by the signer

Implementation in CAdES, PAdES and XAdES

CAdES, PAdES, and XAdES signatures provide mechanisms for indicating the commitment made by the signer.

Implementers are referred to clause 5.2.3 of ETSI EN 319 132-1 [i.4], when implementing XAdES signatures. The signed property `xades:CommitmentTypeIndication` uses URI values as the way for indicating the commitment made by the signer. Implementers should also take into account that as one XAdES signature can collectively sign different data objects, each instance of this signed property identifies the data object(s) it refers to.

Implementers are referred to clause 5.2.3 of ETSI EN 319 122-1 [i.2], when implementing CAdES signatures. The signed attribute `commitment-type-indication` uses OID values as the way for indicating the commitment made by the signer.

Annex B of ETSI TS 119 172-1 [i.17] lists a set of pre-defined pairs of [URI, OID], each pair corresponding to a specific commitment, whose semantics is precisely defined. URIs are defined for being used in `xades:CommitmentTypeIndication` XAdES signed property. OIDs are defined for being used in `commitment-type-indication` CAdES signed attribute.

If ASiC containers are used, implementers should include commitment indications in each CAdES and XAdES signature where their presence is required, using the aforementioned signed attribute/signed property.

When PAdES signatures are used, implementers should take into account the following considerations:

1) Within PAdES-CMS a string within the signed entry `Reason`, in the signature dictionary, can identify the commitment made by the signer. Implementers are referred to ETSI EN 319 142-2 [i.7], clauses 4.1 and 5.3, and ISO 32000-1 [i.49], clause 12.8.1, Table 252 for further details.

2) Within PAdES-E-BES signatures, the commitment made by the signer can be signalled either in the entry Reason as indicated above, or by the signed attribute `commitment-type-indication`. These mechanisms are exclusive. Implementers are referred to ETSI EN 319 142-2 [i.7], clauses 5.3.

3) Within PAdES-OnCAdES signatures that are not PAdES-E-BES signatures, the commitments made by the signer can be signalled in two different ways (implementers are referred to ETSI EN 319 142-1 [i.6], clause 6.3 additional requirements d) and m) for further details), namely:

   - The signed entry `Reason` within the signature dictionary can be used only if these signatures do not contain neither the `signature-policy-identifier` signed attribute nor the `commitment-type-indication` signed attribute.

   - The signed attribute `commitment-type-indication` can be used only if the signed entry `Reason` is not present. When the `signature-policy-identifier` signed attribute is present and there is the need of indicating the commitment made by the signer, the `commitment-type-indication` signed attribute is used instead the signed entry `Reason` because the explicit signature policy document can establish specific constraints for each commitment made by the signer, which makes imperative that, if a certain commitment is made by the signer, this one is signalled using the `commitment-type-indication` signed attribute.

4) Within PAdES-XML-EMB signatures, the commitments made by the signer is indicated using the `xades:CommitmentTypeIndication` signed property (ETSI EN 319 142-2 [i.7], clause 6.2.2.5).

5) Within PAdES-XML-XFA signatures, the commitments made by the signer can be signalled in two different ways (ETSI EN 319 142-2 [i.7], clause 6.3.2.4):

   - The `description` child of `ds:SignatureProperties` element, if these signatures do not contain the `signature-policy-identifier` signed attribute. The description element is defined within the Dublin Core http://purl.org/dc/elements/1.1/namespace.

   - The `xades:CommitmentTypeIndication` signed property if these signatures contain the `xades:SignaturePolicyIdentifier` signed property.

8.9 Including and protecting indication of signer's identity, signer's roles and/or attributes

8.9.1 Including and protecting indication of signer's identity

Rationale

In many real-life environments, users will be able to get from different CAs or even from the same CA, different certificates containing the same public key for different names. The prime advantage is that a user can use the same private key for different purposes. Multiple use of the private key is an advantage when a smart card is used to protect the private key, since the storage of a smart card is always limited. When several CAs are involved, each different certificate can contain a different identity, e.g. as a citizen of a nation or as an employee from a company. Thus, when a private key is used for various purposes, the certificate is needed to clarify the context in which the private key was used when generating the signature. Where there is the possibility that multiple private keys are used, it is necessary for the signer to indicate to the verifier the precise certificate to be used.

Many current schemes simply add the certificate after the signed data and thus are subject to various substitution attacks. An example of a substitution attack is a "bad" CA that would issue a certificate to someone with the public key of someone else. If the certificate from the signer was simply appended to the signature and thus not protected by the signature, any one could substitute one certificate by another and the message would appear to be signed by someone else. In order to counter this kind of attack, the identifier of the certificate is protected by the digital signature from the signer.

A number of signed attributes/properties, enclosing, among other things the digest value of the signer's certificate, are designed to prevent the simple substitution of the certificate.

Implementation in CAdES, PAdES, and XAdES

All the digital signature formats standardized by ETSI, with the exception of PAdES-CMS, force to protect either the signer's certificate or the digest of the signer's certificate with the signature itself.

Implementers are referred to clause 5.2.2 of ETSI EN 319 132-1 [i.4], when implementing XAdES baseline signatures. This clause specifies the `xades:SigningCertificateV2` signed property, the container that includes a reference to the signer's certificate and optionally references to certificates within the certification path of the signer's certificate. As each reference contains the digest of the referenced certificate, this one is actually protected by the signature itself.

NOTE: Property `xades:SigningCertificateV2` substitute the previous `xades:SigningCertificate` specified in ETSI TS 101 903 [i.39] because XML Sig Version 1.1 [i.37] deprecated the `ds:X509IssuerSerial`, used within `xades:SigningCertificate`, as a number of XML Schema validation tools do not support integer types with decimal data exceeding 18 decimal digits, which is not an uncommon fact in certificates issued by CAs that randomly generate the certificates serial numbers.

XAdES extended signatures can protect the signer's certificate incorporating the `xades:SigningCertificateV2` signed property (as XAdES baseline signatures do), or incorporating the actual base-64 encoding of the DER-encoded X.509 signer's certificate within one `ds:X509Data` child of `ds:KeyInfo` element and adding one `ds:Reference` element that ensures that the signer's certificate is actually signed. Implementers are referred to ETSI EN 319 132-2 [i.5], clause 4.2.

Implementers are referred to ETSI EN 319 122-1 [i.2], clause 5.2.2, when implementing CAdES signatures (regardless they are baseline or extended). This clause specifies that two different attributes can use for incorporating a reference to the signer's certificate, namely `ESS-signing-certificate` (clause 5.2.2.2), and `ESS-signing-certificate-v2` (clause 5.2.2.3). The first attribute assumes that the digest algorithm is always SHA-1. The second one incorporates a field that contains an indication of the digest algorithm and, consequently, this one can be a different algorithm than SHA-1. See clause 8.13 for details on how to get guidance on cryptographic suites.

When PAdES signatures are used, implementers should take into account the following considerations:

1) ETSI EN 319 142-2 [i.7], clause 4 for PAdES-CMS does not mandate the inclusion of either `ESS-signing-certificate` or `ESS-signing-certificate-v2`.

1) In PAdES-OnCAdES signatures, the presence of either `ESS-signing-certificate` or `ESS-signing-certificate-v2` is mandatory (ETSI EN 319 142-1 [i.6], clause 6.3, ETSI EN 319 142-2 [i.7], clause 5.3).

2) Within PAdES-XML signatures, it is mandatory either to incorporate the base-64 encoding of the DER-encoded X.509 signer's certificate into `ds:KeyInfo` and cover the signer's certificate with the signature or to incorporate `xades:SigningCertificateV2` signed property into the signature. (ETSI EN 319 142-2 [i.7], clauses 6.2.2.4.1 and 6.3.2.3.1).

8.9.2 Including signer's roles and/or attributes

Implementation in CAdES, PAdES and XAdES

CAdES, PAdES, and XAdES signatures provide mechanisms for indicating the role played by the signer, which entitles him with certain attributes.

This indication can be:

1) a mere claim stated by the signer, which the relying party can trust or not as his own discretion; or

2) it can be a "certified" statement, issued by an Attribute Authority (e.g. attribute certificate or a signed SAML assertion signed by an Attribute Authority); or

3) it can be an assertion signed by an entity that is not an Attribute Authority (e.g. a signed SAML assertion).

Implementers should assess, for each data object to be signed and for each signature, whether the inclusion of an indication of the signing role of the signer or the indication that the signer is in possession of certain attribute(s), is required or not. Implementers should take into account the legal/regulatory framework of the business process while doing this assessment. For those signatures requiring an indication of the role played by the signer or of the attributes in possession of the signer, implementers should assess whether a claimed indication is enough or a signed assertion or a certified indication is required.

Implementers are referred to clause 5.2.6 of ETSI EN 319 132-1 [i.4], when implementing XAdES signatures. This clause specifies the `xades:SignerRoleV2` signed property, which can include a set of claimed attributes or roles, a set of certified attributes or roles, and/or a set of signed assertions.

Implementers are referred to clause 5.2.6 of ETSI EN 319 122-1 [i.2], when implementing CAdES signatures. This clause 6.2.6.1 specifies the `signer-attributes-v2` signed attribute, which can include a set of claimed attributes, a set of certified attributes issued by an Attribute Authority, and/or a set of signed assertions.

When PAdES signatures are used, implementers should take into account the following considerations:

1) Attribute certificates should not be included within PAdES-CMS signatures (ETSI EN 319 142-2 [i.7], clause 4.2.1).

2) In PAdES-OnCAdES signatures, the signer roles/attributes, if required, are indicated within the `signer-attribute-v2` signed attribute (ETSI EN 319 142-1 [i.6], clause 6.3, ETSI EN 319 142-2 [i.7], clause 5.3).

3) Within PAdES-XML signatures, the signer roles, if required, are indicated within the `xades:SignerRoleV2` signed property. (ETSI EN 319 142-2 [i.7], clauses 6.2.2.5 and 6.3.2.4).

8.10 Including additional signed information

8.10.1 Introduction

Clauses 8.10.2, 8.10.3 and 8.10.4 provide guidance on how to include additional information that is also signed by the signer. Any piece of signed information (including signer commitment and signer role addressed above) further qualifies the signed data object(s), the signer or the digital signature itself.

8.10.2 Including explicit indication of the signature policy

Rationale

Signature policies are fundamental for ensuring consistency of signature validation.

Signature policies can be issued by a wide variety of entities. They can be explicitly identified or can be implied by the semantics of the data object(s) being signed and some other information, e.g. national laws or private contractual agreements, that mention that a given signature policy has to be used for this type of data content.

In general the signature policy needs to be available in human readable form so that it can be assessed to meet the requirements of the legal and contractual context in which it is being applied. To facilitate the automatic processing of a digital signature, it is worth that the parts of the signature policy, which specify the electronic rules for the creation, validation and augmentation of the digital signature, be comprehensively defined and in a computer-processable form (e.g. in XML or ASN.1).

An explicit signature policy has a globally unique reference, which is bound to a digital signature by the signer as part of the digital signature value calculation. In these cases, for a given explicit signature policy there will be one definitive form that has a unique binary encoded value. See ETSI TS 119 172-1 [i.17] for more details on the signature policy building blocks. See ETSI TS 119 172-4 [i.20] defining a policy for digital signatures to be considered successfully verified as an advanced electronic signatures (AdES), advanced electronic seals, advanced electronic signatures supported by a qualified certificate (AdES$_{QC}$), advanced electronic seals supported by a qualified certificate, qualified electronic signatures (QES), or qualified electronic seals against EU Member States trusted lists as defined in CD 2009/767/EC [i.31] as amended by CD 2010/425/EU [i.32] and by CD 2013/662/EU [i.33] in the context of European Directive 1999/93/EC [i.30].

The explicit indication of the signature policy will usually include: the unique identifier of the signature policy itself, and a digest of the signature policy document. It can also contain additional qualifying information.

By including the explicit indication of the signature policy within a digital signature, the signer explicitly declares that the identified signature policy is the one that has governed its generation and is required to govern its validation.

Implementation in CAdES, PAdES and XAdES

CAdES, PAdES, and XAdES signatures provide mechanisms for incorporating explicit information of the signature policy that actually governs their generation and validation.

Within XAdES and CAdES signatures, this information consists in a unique identifier of the signature policy and a digest value computed on the whole or certain part of the unique binary representation of the signature policy document. Optionally additional information can also be provided, as indicated below:

1) Pointers to sites where such a binary representation can be reached.

2) User notices with information that is intended for being displayed while the signature is being validated.

3) An identifier that indicates the specification the binary representation of the signature policy pointed is compliant with. Binary representation can be in human readable form, XML or ASN.1. ETSI TS 119 172-1 [i.17] specifies a format for the human readable form; ETSI TS 119 172-2 [i.18] (not yet produced at the time the present document was written) will specify a format for the XML form, and ETSI TS 119 172-3 [i.19] (not yet produced at the time the present document was written) will specify a format for the ASN.1 form.

Implementers are referred to clause 5.2.9 of ETSI EN 319 132-1 [i.4], when implementing XAdES signatures, which specifies the `xades:SignaturePolicyIdentifier` signed property.

Implementers are referred to clause 5.2.9 of ETSI EN 319 122-1 [i.2], when implementing CAdES signatures, which specifies the `signature-policy-identifier` signed attribute.

When PAdES is used, implementers should take into account the following considerations:

1) Within PAdES-OnCAdES signatures, the signature policy identifier, if required, will appear within the `signature-policy-identifier` signed attribute (ETSI EN 319 142-1 [i.6], clause 6.3, ETSI EN 319 142-2 [i.7], clause 5.4).

2) Within PAdES-XML signatures, the signature policy identifier, if required, will appear within the `xades:SignaturePolicyIdentifier` signed property (ETSI EN 319 142-2 [i.7], clauses 6.2.2.5 and 6.3.2.4).

8.10.3 Including indication of the signed data object format

Implementation in CAdES, PAdES and XAdES

CAdES, XAdES and PAdES-XML-EMB digital signatures provide mechanisms for incorporating an indication of the format of the signed data object as signed information.

Implementers are referred to clause 5.2.4 of ETSI EN 319 132-1 [i.4], when implementing XAdES signatures, which specifies the `xades:DataObjectFormat` signed property. This property can contain among other information, the mime type and the encoding of each signed data object.

Implementers are referred to clause 5.2.4 of ETSI EN 319 122-1 [i.2], when implementing CAdES signatures. This clause specifies two signed attributes, namely: `content-hints`, which is to be used for multi-layered CAdES signatures, and `mime-type`, which can also be used in not multi-layered CAdES signatures. Both attributes allow indicating the mime type of the signed data object. Should a CAdES signature collectively sign a multipart mime structure, each of these parts can individually indicate its own mime type.

When PAdES is used, implementers should take into account the following considerations:

1) Signed attributes `content-hints` and `mime-type` are not allowed within PAdES-NoXML signatures: what they sign is a PDF container (ETSI EN 319 142-1 [i.6], clause 5.2).

2) PAdES-XML signatures can incorporate `xades:DataObjectFormat` signed property (ETSI EN 319 142-2 [i.7], clauses 6.2.2.5 and 6.3.2.4).

By specifying the mime-type, it is possible to counter attacks based on adding html commands into a pdf, jpg, bmp, etc. file, and changing the filetype in "html". This attack would change the data object presentation, since this file would likely be opened as an html file.

8.10.4 Including indication of the signature production place

Implementation in CAdES, PAdES and XAdES

CAdES, PAdES, and XAdES signatures provide mechanisms for incorporating, as signed information, an indication of the location where signer claims that the signature has been generated.

Implementers are referred to clause 5.2.5 of ETSI EN 319 132-1 [i.4], when implementing XAdES signatures, which specifies the `xades:SignatureProductionPlaceV2` signed property.

Implementers are referred to clause 5.2.5 of ETSI EN 319 122-1 [i.2], when implementing CAdES signatures, which specifies the `signer-location` signed attribute.

When PAdES signatures are used, implementers should take into account the following considerations:

1) PAdES-OnCAdES signatures make use of the `Location` entry within the signature dictionary (ETSI EN 319 142-1 [i.6], clause 6.3, and ETSI EN 319 142-2 [i.7], clause 5.3).

2) PAdES-XML signatures make use of the `xades:SignatureProductionPlaceV2` signed property (ETSI EN 319 142-2 [i.7], clauses 6.2.2.5 and 6.3.2.4).

8.11 Supporting signatures lifecycle

8.11.1 Introduction

The clauses above have provided details on how the signer can incorporate into the signature signed attributes/properties that further qualify the signature, the signer, or the signed data objects.

However, business processes can require that the technical validity of certain digital signatures can be reassessed during a period of time long enough as to allow expiration or compromise of some PKI tokens (e.g. certificates) used for the validation process itself, or even the breach of some cryptographic algorithm used in their generation.

These digital signatures, before being destroyed, go through more complex cycles than the simple cycle generation-initial validation by the signer– almost immediate validation by the relying party. Instead, some other entities (e.g. arbitrator in case of conflict between the signer and the relying party) can need to perform ulterior validations during a certain (long) period before the obligation of allowing this validity reassessing ceases. The digital signatures formats specified by ETSI satisfy this type of requirements allowing that additional data are added to the signatures after they have been generated for supporting their lifecycles. The process of incorporating additional data to a digital signature previously generated is called signature augmentation. This additional data can be validation data, i.e. data that has to be used for validating the signature (e.g. certificates, OCSP responses, etc). Part of this data can also be data for increasing signatures' longevity (for instance time-stamp tokens that can extend the longevity of the signature beyond the expiration or revocation time of some of the certificates in the signer's certificate path). See clause 8.11.6 for details of digital signatures lifecycle.

The signer can add part of this information; other information can be added by the relying parties or even by third parties specifically entitled for doing that.

Clauses 8.11.2 to 8.11.7 provide details on the different types of data that can be added for augmenting a digital signature throughout its lifecycle.

8.11.2 Including time-stamp tokens on the digital signature value

Clause 8.7.2.4 of the present document provides rationale for time-stamping the signature as well as details of signature time-stamp containers for the different formats of digital signatures standardized by ETSI.

As mentioned in clause 8.7.2.4, since a time-stamp token has a limited validity period, it can be required to protect the signature time-stamp token itself. This can be achieved by using another time-stamp token that protects the first one, which in turn enlarges signature's longevity.

Clauses 8.11.4 and 8.11.5.3 provide details on techniques for protecting the components of a digital signature, enlarging its longevity, including time-stamp tokens already incorporated, by incorporation of new time-stamp tokens.

8.11.3 Including references to validation data

8.11.3.1 Rationale

When dealing with digital signatures in the long term, all the data used in the verification (namely, certificate path and revocation information) of such signatures are stored and conveniently time-stamped for arbitration purposes. Similar considerations apply to attribute certificates if they appear within the signature. In some environments, it can be convenient to add these data to the digital signature (as unsigned attributes/properties) for archival purposes.

Certain business processes though, can advise to archive validation data outside the digital signature itself, e.g. to prevent redundant storage and to reduce the size of the signatures. In such cases each digital signature can incorporate references to all these data within the signature, in order to keep the size of the digital signatures to a minimum. These references need to incorporate means for unambiguously identifying the validation data they are references of. This would facilitate these parties to store the validation data outside the signatures, and still allow their identification and retrieval when validating the signature.

ETSI formats allow augmenting the signature by incorporating the following references:

- the sequence of references to the full set of CA certificates used to validate the digital signature up to (but not including) the signer's certificate;
- the sequence of references to the full set of revocation data used in the validation of the signer and CA certificates;
- the references to the full set of certificates required for verifying any time-stamp token incorporated into the signature at the time the unsigned attribute/property encapsulating these references is incorporated;
- the references to the full set of revocation data required for verifying any time-stamp token incorporated into the signature at the time the unsigned attribute/property encapsulating these references is incorporated;
- the references to the full set of certificates used to validate the attribute certificate(s) or signed assertion(s), if present;
- the references to the full set of revocation data used in the validation of the attribute certificate(s) or signed assertion(s), if present.

The full sets of references to the revocation data that have been used in the validation of the signer, any attribute certificate, signed assertion, and the signing certificate of any already incorporated time-stamp token, as well as their corresponding CAs certificates, provide means to retrieve the actual revocation data archived elsewhere in case of dispute and, in this way, to illustrate that the verifier has taken due diligence of the available revocation information.

Currently two major types of revocation data are managed in most of the systems, namely CRLs and responses of on-line certificate status servers, obtained through protocols designed for these purposes, like OCSP protocol. In consequence, the ETSI formats for digital signature standards provide means for referencing both types of revocation data.

Each reference contains the digest value of the validation data, computed with a certain hash algorithm, which allows the unambiguous identification of the corresponding validation data, and optionally explicit identifiers of such validation data, which can facilitate their management (searches in databases for instance).

Within the European Union, each Member State publishes a Trusted List (TL) listing, among others, all the qualified Trust Service Providers issuing certificates, and all the services that they provide (be them supervised or accredited). Among other details, the TL includes the certificate of the TSP itself, which allows to use the TL itself as a container of potential source of trusted certificates.

XAdES and CAdES specify containers for references to validation data. PAdES signatures do not incorporate such type of references, as this format intends to be a self-contained package in terms of validating a signature in the long term.

8.11.3.2 Including references to certificates

Implementation in CAdES and XAdES

Both CAdES and XAdES signatures define containers for references to:

1) CA certificates within the certification path of the signer's certificate;

2) attribute authorities certificates (required when the signer signs attribute certificates) and the certificates within its certification path;

3) assertions signing certificates (required when the signer signs signed assertions) and the certificates within their certification paths; and

4) time-stamp tokens certificates already present in the signature at the time of generating these containers, and the certificates within their certification paths.

Each reference contains the digest value computed on the referenced certificate using a specific digest algorithm and an optional identifier. Relying parties can use the digest value for checking that the certificate retrieved is actually the referenced one.

Implementers are referred to clause A.1.1 of ETSI EN 319 132-1 [i.4], when implementing XAdES signatures. This clause specifies the `xadesv141:CompleteCertificateRefsV2` unsigned property, the container for references to certificates within the certification path of the signer's certificate, the time-stamp tokens certificates and the certificates within their certification paths. Implementers are also referred to clause A.1.3 of ETSI EN 319 132-1 [i.4] when the signature contains attribute certificates or signed SAML assertions. This clause specifies the `xadesv141:AttributeCertificateRefsV2` unsigned property, the container for references to Attribute Authorities' certificates, or certificates of signers of signed assertions, and the certificates within their certification paths.

NOTE: Properties `xadesv141:CompleteCertificateRefsV2` and `xadesv141:AttributeCertificateRefsV2` substitute the previous `xades:CompleteCertificateRefs` and `xades:AttributeCertificateRefs` both specified in ETSI TS 101 903 [i.39] because XML Sig Version 1.1 [i.37] deprecated the `ds:X509IssuerSerial`, used within `xades:CompleteCertificateRefs` and `xades:AttributeCertificateRefs` as a number of XML Schema validation tools do not support integer types with decimal data exceeding 18 decimal digits, which is not an uncommon fact in certificates issued by CAs that randomly generate the certificates serial numbers.

Implementers are referred to clause A.1.1.1 of ETSI EN 319 122-1 [i.2], when implementing CAdES signatures. This clause specifies the `complete-certificate-references` unsigned attribute, the container for references to certificates within the certification path of the signer's certificate, the time-stamp tokens certificates and the certificates within their certification paths. Implementers are referred to clause A.1.3 of ETSI EN 319 122-1 [i.2] when the signature contains attribute certificates or signed SAML assertions. This clause specifies the `attribute-certificate-references` unsigned attribute, the container for references to Attribute Authorities' certificates, or certificates of signers of signed assertions, and the certificates within their certification paths.

8.11.3.3 Including references to certificate status data

Implementation in CAdES and XAdES

CAdES and XAdES define containers for references to certificate status data. Both define references to OCSP responses and CRLs. They also define a placeholder for references to other types of certificate status data. These containers can include references to certificate status data corresponding to:

1) CA certificates within the certification path of the signer's certificate;

2) Attribute Authorities certificates (required when the signer signs attribute certificates) and the certificates within its certification path;

3) assertions signing certificates (required when the signer signs signed assertions) and the certificates within its certification path; and

4) time-stamp tokens certificates already present in the signature at the time of generating these containers, and the certificates within their certification paths.

Each reference contains an identifier of the referenced certificate status data and a digest value computed on it using a specific digest algorithm. Relying parties can use this value for checking that the certificate status data retrieved is actually the referenced one.

Implementers are referred to clause A.1.2 of ETSI EN 319 132-1 [i.4], when implementing XAdES signatures. This clause specifies the `xades:CompleteRevocationRefs` unsigned property, the container for references to certificate status data corresponding to certificates within the certification path of the signer's certificate, the time-stamp tokens certificates and the certificates within their certification paths. Also, implementers are referred to clause A1.4 of ETSI EN 319 132-1 [i.4] when the signature contains attribute certificates or signed SAML assertions. This clause specifies the `xades:AttributeRevocationRefs` unsigned property, a container able to contain references to certificate status data corresponding to attribute certificates, Attribute Authorities' certificates, certificates of signers of signed assertions, and the certificates within their certification paths.

Implementers are referred to clause A1.2.1 of ETSI EN 319 122-1 [i.2], when implementing CAdES signatures. This clause specifies the `complete-revocation-references` unsigned attribute, the container for references to certificate status data corresponding to certificates within the certification path of the signer's certificate, the time-stamp tokens certificates and the certificates within their certification paths. Implementers are referred to clause A.1.4 of ETSI EN 319 122-1 [i.2] when the signature contains attribute certificates or signed SAML assertions. This clause specifies the `attribute-revocation-references` unsigned attribute, the container for references to certificate status data corresponding to attribute certificates, Attribute Authorities' certificates, certificates of signers of signed assertions, and the certificates within their certification paths.

As mentioned before, PAdES signatures do not incorporate references to validation data.

8.11.4 Time-stamping references to validation data

Rationale

Digital signatures incorporating time-stamp tokens on validation data references are needed when the signature incorporates references to the validation material and there is a requirement to safeguard against the possibility of a CA key in the certificate chain ever being compromised. A verifier can be required to provide, on request, proof that the certification path and the revocation information used at the time of the signature were valid, even in the case where one of the issuing keys or OCSP responder keys is later compromised.

Time-stamping CA certificates references will stop any attacker from issuing bogus CA certificates that could be claimed to exist before the CA key was compromised. Any bogus time-stamped CA certificates references will show that the certificate was created after the legitimate CA key was compromised. In the same way, time-stamping CA CRLs references will stop any attacker from issuing bogus CA CRLs that could be claimed to exist before the CA key was compromised.

For protecting the signature against this threat, ETSI digital signature standards allow the incorporation of two additional types of time-stamp containers, namely:

- A time-stamp token container that encapsulates a time-stamp token on the sequence formed by the digital signature value, the time-stamp token on the digital signature value (if present), and the unsigned attributes/properties encapsulating references to the validation material.

- A time-stamp token container that encapsulates a time-stamp token on the unsigned attributes/properties encapsulating references to the validation material only.

ETSI digital signature standards allow that signer, verifier or another entity can request, obtain and augment the signature incorporating some of the time-stamp tokens mentioned above to the digital signature. With this type of signature augmentation it can be proved that at the time instant indicated within the time-stamp token the signature was safeguarded against the possibility of a CA key in the certificate chain ever being compromised.

If the business process advises to time-stamp the references on validation material, in case an OCSP response is used, it is necessary to time-stamp in particular that response in the case the key from the responder would be compromised. Since the information contained in the OCSP response is user specific and time specific, an individual time-stamp is needed for every signature received. Instead of placing the time-stamp only over the certification path references and the revocation information references, which include the OCSP response reference, the time-stamp token is computed on the digital signature value, the signature time-stamp token on the signature if present, and all the unsigned attributes/properties encapsulating references to validation material. For the same cryptographic price, this will provide an integrity mechanism over the digital signature. Any modification can be immediately detected. It should be noticed that other means of protecting/detecting the integrity of the digital signature exist and can be used.

When CRLs are used, time-stamping each digital signature with the complete validation data references as defined above cannot be efficient, particularly when the same set of CA certificates and CRL information is used to validate many signatures. Time-stamping references to commonly used certificates and CRLs, can be done centrally, e.g. inside a company or by a service provider. This method reduces the amount of data the verifier has to time-stamp, for example it could reduce to just one time-stamp per day (i.e. in the case were all the signers use the same CA and the CRL applies for the whole day). As indicated before, the information that needs to be time-stamped is not the actual certificates and CRLs but the unambiguous references to those certificates and CRLs. Nevertheless, using time-stamp tokens that cover both the references and the signature elements, is also allowed in scenarios where the revocation data are CRLs.

Implementation in CAdES and XAdES

XAdES and CAdES define two types of containers for time-stamp tokens on references to validation data.

Implementers are referred to clause A.1.5 of ETSI EN 319 132-1 [i.4], when implementing XAdES signatures. This clause specifies two unsigned properties. The first one is `xadesv141:SigAndRefsTimeStampV2`, a container for a time-stamp token computed on the `ds:SignatureValue`, any present `xades:SignatureTimeStamp`, and any present container of references to validation data. The second one is `xadesv141:RefsOnlyTimeStampV2`, a container for a time-stamp token computed on any present container of references to validation data only.

NOTE: Properties `xadesv141:SigAndRefsTimeStampV2` and `xadesv141:RefsOnlyTimeStampV2`substitute the previous `xades:SigAndRefsTimeStamp` and `xades:RefsOnlyTimeStamp` both specified in ETSI TS 101 903 [i.39] because both of them time-stamp `xades:CompleteCertificateRefsV2` and `xades:AttributeCertificateRefsV2` instead `xades:CompleteCertificateRefs` and `xades:AttributeCertificateRefs`, as `xades:SigAndRefsTimeStamp` and `xades:RefsOnlyTimeStamp` did.

Implementers are referred to clause A.1.5 of ETSI EN 319 122-1 [i.2], when implementing CAdES signatures. This clause specifies two unsigned properties. The first one is `time-stamped-certs-crls-references`, a container for a time-stamp token computed on any present container of references to validation data only. The second one is `CAdES-C-time-stamp`, a container for a time-stamp token computed on the OCTET STRING of the `signature` field within `SignerInfo`, any present `signature-time-stamp`, and any present container of references to validation data.

Although there is no mandatory constraint on the scenarios where to use one or the other, a good practice is to use the `xades:SigAndRefsTimeStamp` or `CAdES-C-time-stamp` when references to OCSP responses are used, while `xades:RefsOnlyTimeStamp` or `time-stamped-certs-crls-references` are better for references to CRLs.

8.11.5 Enlarging longevity and resilience to change

8.11.5.1 Introduction

Certain business processes require to allow that the technical validity of a digital signature can be reassessed during a period of time that goes far beyond the expiration or the revocation of any of the certificates within the certification paths of the time-stamp token on the signature or the time-stamps on references to validation material, or the breach of some of the algorithms used for their generation (a fact that experience has proved to be not so uncommon).

Before any of these situations occur, the augmented signature needs to be protected, in the case of near breach of some of the algorithms, with stronger algorithms. CAdES, XAdES, and PAdES signatures provide means for protecting the augmented signatures, and consequently for enlarging their longevity. Below follow the required steps for this augmentation:

1) To incorporate any missing validation material to the signature, including the missing validation material from any previously incorporated time-stamp token.

2) To protect all the material required for validating the signature (including the signed data objects, even if they are detached from the signature, and the validation material) generating a new time-stamp token using a stronger digest algorithm if required. This time-stamp token actually provides a proof of existence of all the time-stamped material and at the same time protects its integrity.

3) To incorporate the new time-stamp token into the signature encapsulated in a suitable container.

This type of time-stamp tokens is known as time-stamp tokens for long term availability and integrity of validation material.

NOTE: In ETSI TS 101 903 [i.39], ETSI TS 101 733 [i.40], ETSI TS 102 778 [i.41], ETSI TS 103 171 [i.43], ETSI TS 103 172 [i.44], and ETSI TS 103 173 [i.45], this type of time-stamp tokens where known as archive time-stamps. This term is not used any longer for separating their specification and usage from the most general problem of archival of digital signatures. Nevertheless, the names of the XAdES properties and CAdES attributes have not been changed due to the backwards compatibility problem that this change would have caused.

At a minimum, these signatures will incorporate all the validation data required for its validation and one or more of this type of time-stamp tokens (each one time-stamping anything in the signature present at the time of generating the archive time-stamp tokens).

Consequently, these signatures will require at least two specific components:

1) Containers for validation data values.

2) Containers for archival time-stamp tokens.

ETSI specifications allow complex combinations of attributes/properties that can be secured with archive time-stamp tokens. The following CAdES, PAdES, and XAdES signatures incorporate this type of time-stamp tokens:

1) XAdES-B-LTA baseline signatures as specified in ETSI EN 319 132-1 [i.4], clause 6.3 and XAdES-E-A extended signatures as specified in ETSI EN 319 132-2 [i.5], clauses 4.3 and A.2.

2) CAdES-B-LTA baseline signatures as specified in ETSI EN 319 122-1 [i.2], clause 6.3 and CAdES-E-A extended signatures as specified in ETSI EN 319 122-2 [i.3], clauses 4.3 and A.2.

3) PAdES-B-LTA baseline signatures as specified in ETSI EN 319 142-1 [i.6], clause 6.3 and PAdES-E-LTV signatures with DSS dictionary and at least one DocumentTimeStamp dictionary, as specified in ETSI EN 319 142-2 [i.7], clause 5.5.

Besides the above mentioned mechanisms, services offered by Trust Service Providers using different techniques to preserve the digital signatures within archival systems that do not require the incorporation of additional material within the signatures themselves, or require the incorporation of part of it, can also be used.

Clauses 8.11.5.2 and 8.11.5.3 provide guidance on the mechanisms used within each format. Clause 8.11.5.2 provides details on containers for validation data values. Clause 8.11.5.3 provides details for containers that embed time-stamps long term availability and integrity of validation material.

8.11.5.2 Incorporating containers for validation material

Rationale

A verifier will have to verify that the certification path was valid, at the time of the creation of the signature, up to a trust point according to the naming constraints and the certificate policy constraints from a certain implicitly or explicitly identified signature validation policy. For achieving this long after the signature was generated, it will be necessary to capture all the validation material required for verifying the certification path, starting with those from the signer and ending up with a trust anchor, as well as the certificates used for validating any attribute certificate and/or time-stamp present within the digital signature.

When dealing with long term digital signatures, all the data used in the validation (including the certification paths of the signing certificate, any incorporated countersignature, any attribute certificate and/or time-stamp, as well as their corresponding revocation data, and the material required for verifying such revocation data), need to be conveniently stored and time-stamped.

For dealing with long term signatures, it is also needed to store and conveniently time-stamp all the revocation data used in the validation of such signatures.

When using CRLs to get revocation information, a verifier will have to make sure that he gets at the time of the first validation the appropriate certificate revocation information from the signer's CA. This involves checking that the signer certificate serial number is not included in the CRL. The signer, the verifier or any other third party can obtain this CRL. If obtained by the signer, then it will be conveyed to the verifier. Additional CRLs for the CA certificates in the certificate path need to also be checked by the verifier. It can be convenient to incorporate these CRLs within the digital signature for ease of subsequent validation or arbitration.

When using OCSP to get revocation information, a verifier will have to make sure that he gets at the time of the first validation an OCSP response. The signer, the verifier or any other third party can fetch this OCSP response. Since OCSP responses are transient and thus are not archived by any TSP including CA, it is the responsibility of every verifier to make sure that it is stored in a safe place.

ETSI digital signature formats specify mechanisms for incorporating this validation material into the signatures themselves.

Implementation in CAdES signatures

CAdES signatures have evolved with time since its first version was published as ETSI TS 101 733 [i.40]. This has resulted in changes in the containers of validation data, the containers of the archive-time-stamp tokens, and the containers of ancillary information.

CAdES signatures compliant with ETSI EN 319 122-1 [i.2] embed the certificates values and certificate status values required for validating the signature, any present attribute certificate or signed SAML assertion, and any present time-stamp tokens within `SignedData.certificates` and `SignedData.crls` fields. See ETSI EN 319 122-1 [i.2], clause 5.5.

Business processes can require implementations to be able to validate legacy CAdES signatures that use different containers (currently superseded by ETSI EN 319 122-1 [i.2]). In such cases, implementers should take into account that these signatures could contain the following containers:

1) Unsigned attributes `certificate-values` and `revocation-values` (specified in ETSI EN 319 122-1 [i.2], clauses A.1.1.2 and A.1.2.2 respectively). These were containers for validation data required for validating the signature and any present attribute certificate or signed SAML assertion or any time-stamp token not containing all needed information before the first archive time-stamp token (or `long-term-validation` attribute) was added to the signature.

2) Fields `extraCertificates` and `extraRevocation` embedded within the `long-term-validation` unsigned attribute. These were containers for extra validation data after the first `long-term-validation` attribute was added (see ETSI EN 319 122-1 [i.2], clause A.2.5).

Implementation in XAdES signatures

XAdES signatures have also evolved with time since its first version was published as ETSI TS 101 903 [i.39]. This has resulted in changes in the containers of validation data, the containers of the archive-time-stamp tokens, and the containers of ancillary information.

ETSI EN 319 132-1 [i.4] identifies the following containers for certificates and certificate status data:

1) `ds:KeyInfo` element, and unsigned properties `xades:CertificateValues` (clause 5.4.1), `xades:RevocationValues` (see clause 5.4.2), `xades:AttrAuthoritiesCertValues` (see clause 5.4.3), and `xades:AttributeRevocationValues` (see clause 5.4.4). These are containers for validation data required for validating the signature, any incorporated countersignature and any present attribute certificate or signed assertions.

2) Fields `xadesv141:TimeStampValidationData`. This is a container for validation data corresponding to one or more time-stamp tokens present within the signature (see clause 5.5.1).

Implementation in PAdES signatures

ETSI EN 319 142-1 [i.6], clause 5.4.2 specifies two PDF dictionaries as containers for validation data in long term PAdES signatures. All the types, except PAdES-CMS make use of them when long term signatures need to be managed:

1) Document Security Store (DSS) dictionary. This dictionary is designed as a single container for all validation data of some or all signatures in the document (see ETSI EN 319 142-1 [i.6], clause 5.4.2.2).

2) Validation Related Information (VRI) dictionary. This dictionary acts as a container for validation data related to one specific signature in the document (see ETSI EN 319 142-1 [i.6], clause 5.4.2.3). This is an optional dictionary in long term PAdES signatures.

8.11.5.3 Incorporating time-stamp tokens for long term availability and integrity of the validation material

Rationale

Advances in computing increase the probability of being able to break algorithms and compromise keys. There is therefore a requirement to be able to protect digital signatures against this possibility.

Over a period of time weaknesses can occur in the cryptographic algorithms used to create a digital signature (e.g. due to the time available for cryptanalysis, or improvements in crypto analytical techniques). Furthermore, if the digital signature incorporates some time-stamp token, some of the crypto algorithms used by the TSA can become weak. Before such weaknesses become likely, a verifier should take extra measures to maintain the validity of the digital signature.

Several techniques could be used to achieve this goal depending on the nature of the weakened cryptography. In order to simplify matters, ETSI digital signature standards specify a technique that covers all the cases.

This technique consists in incorporating into the signature all the required certificate values and revocation data values, generate a time-stamp token covering the components of the digital signature, and augment the signature with an unsigned attribute/property encapsulating the aforementioned time-stamp token. The complete validation data is necessary if the hash function and the crypto algorithms that were used to create the signature are no longer secure.

If the digital signature incorporated some previous time-stamp tokens, the corresponding validation material (certificates and certificates revocation status data) for these time-stamp tokens is incorporated to the signature before computing the message imprint to be submitted to the TSA, so that this material is also protected by the new time-stamp token. This is needed for proving the precise status of the already present time-stamp tokens (time-stamp tokens on the signature value and/or time-stamp tokens on the references to validation material) when the additional time-stamp token is incorporated. New time-stamp tokens can be incorporated to the signature for increasing its longevity, before the expiration or revocation of the certificate of the last time-stamp token incorporated, or before the breach of some of the algorithms used for computing the last time-stamp incorporated.

The potential for Trusted Service Provider (TSP) (like TSAs) key compromise should be significantly lower than for user keys, because TSP(s) are expected to use stronger cryptography and better key protection. It can be expected that new algorithms (or old ones with greater key lengths) will be used. In such a case, a sequence of time-stamp tokens will protect against forgery. Each time-stamp token needs to be affixed before either the expiration or revocation of its certificates, or of the breach of the algorithms used by the TSA. TSAs should have long keys and/or a "good" or different algorithm. Consequently, this kind of signatures can incorporate multiple embedded time-stamps.

Implementation in CAdES signatures

ETSI EN 319 122-1 [i.2], clause 5.5.3, specifies that CAdES signatures embed the `archive-time-stamp-v3` unsigned attribute as container for the archive time-stamp token.

As before, business processes can require implementations to be able to validate legacy CAdES signatures that use different containers (currently superseded by ETSI EN 319 122-1 [i.2]). In such cases, implementers should take into account that these signatures could contain the following time-stamp tokens containers:

1) `timeStamp` field within the `long-term-validation` unsigned attribute (see ETSI EN 319 122-1 [i.2], clause A.2.5).

2) Archive time-stamp unsigned attribute whose OID is: `{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 48`. See version v2.2.1 of ETSI TS 101 733 [i.40] for details.

3) Archive time-stamp unsigned attribute whose OID is: `object identifier { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 27`. See version v1.2.2 of ETSI TS 101 733 [i.40] for details.

ETSI EN 319 122-1 [i.2] also specifies ancillary data. ETSI EN 319 122-1 [i.2], clause 5.5.2, requires embedding the `ats-hash-index-v3` as an unsigned attribute within `archive-time-stamp-v3`'s signature. That attribute contains sequences (`SEQUENCE OF` ASN.1 structures) of digest values of all the certificates, certificate status data and unsigned attributes within the digital signature that the archive time-stamp actually covers.

It serves two purposes: first it unambiguously identifies what parts of the validation material and what parts of the unsigned attributes (as each attribute can have several instances of `AttributeValue` type) present in the signature are actually covered by the time-stamp token; secondly, it solves the problem associated to the fact that the unsigned attributes, the `SignedData.certificates,` and `SignedData.crls` fields are contained within `SET OF` ASN.1 structures. These structures do not define an inner order among their components, which has historically caused problems to interoperability. The solution is achieved by concatenating the contents of the aforementioned `ats-hash-index-v3` to the archive time-stamp's message imprint computation input, instead of individually concatenating the different pieces of validation data and unsigned attributes.

It is emphasized that this technique allows to add a new instance of `AttributeValue` type to a certain unsigned attribute after a certain `archive-time-stamp-v3` attribute has been incorporated. It also allows adding a new `archive-time-stamp-v3` time after (which would cover all the instances o AttributeValue type in all the unsigned attributes) without breaking message imprint from any of the former `archive-time-stamp-v3`.

See ETSI EN 319 122-1 [i.2], clause 5.5.2 for further details.

Implementation in XAdES signatures

ETSI EN 319 132-1 [i.4] requires that XAdES signatures embed the `xadesv141:ArchiveTimeStamp` unsigned attribute as container for the archive time-stamp token (see ETSI EN 319 132-1 [i.4], clause 5.5.2).

Business processes can require implementations to be able to validate legacy XAdES signatures that use different containers (that were already superseded by ETSI TS 101 903v1.4.2 [i.39]). In such cases, implementers should take into account that these signatures could contain the following time-stamp tokens containers:

1) `xades:ArchiveTimeStamp` unsigned property (see ETSI EN 319 132-1 [i.4], annex C).

Implementation in PAdES signatures

ETSI EN 319 142-1 [i.6], clause 5.4.3 specifies the Document Time-stamp dictionary as a special type of signature dictionary, which contains a time-stamp token time-stamping the entire document (and consequently any present signature), including the Document Time-stamp dictionary but excluding the time-stamp token present within this dictionary.

8.11.6 Digital signatures lifecycle

8.11.6.1 Generation, validation and augmentation of digital signatures.

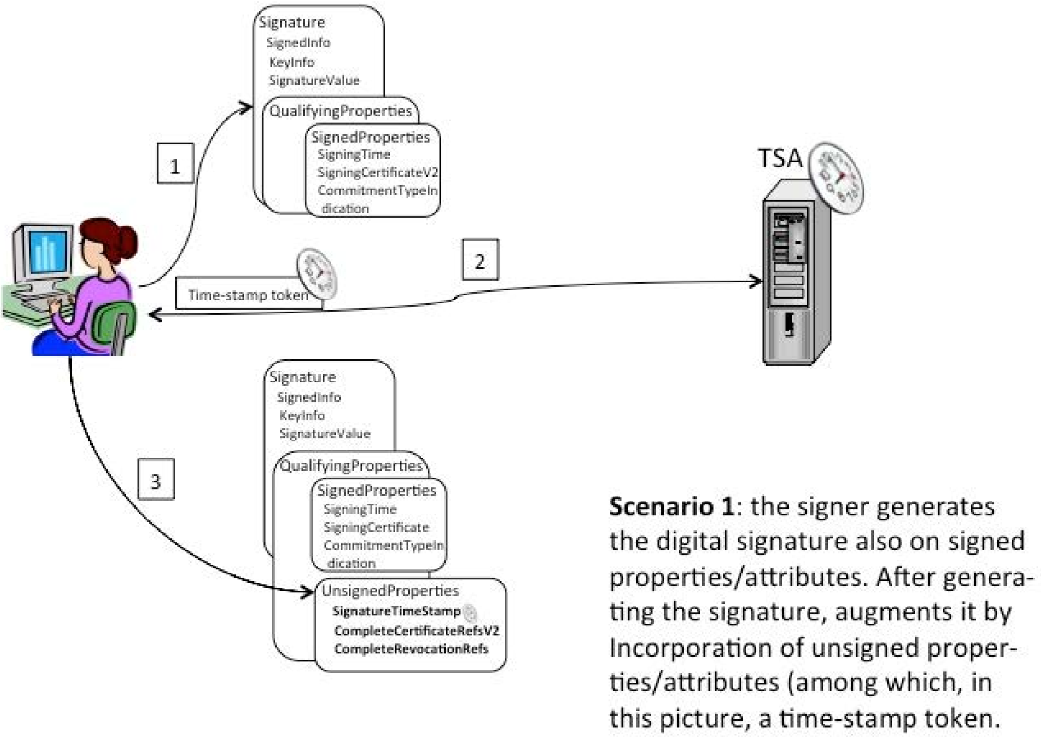
A digital signature, since the moment it is generated until the moment when its usage is definitively discarded, can go through a number of stages, some of which can even change its contents.

The lifecycle of a digital signature includes, in the most general case its generation and a set of augmentations, each one incorporating new unsigned attributes/properties to the generated or previously augmented signature, until the moment the augmented signature is discarded.

Given the high number of different augmentations that a digital signature, compliant with ETSI ENs 319 1x2 (x = 2, 3, and 4), can suffer during its life, the present clause addresses some interesting examples. For a more exhaustive list of augmentations for the three formats, see clauses 8.11.6.4 and 8.11.7 of the present document.

Figures below show signers, verifiers, and other entities (like trusted services –as time-stamp authorities or preservation systems- or arbitrators -acting in case of dispute on a certain signature), generating, augmenting, and validating the signature in different stages of its lifecycle. These figures show XAdES signatures. CAdES and PAdES signatures can go through similar stages (with the exception that PAdES signatures do not neither incorporate references to validation material nor time-stamp tokens on them, but dictionaries).

NOTE: Figures in the present clause do not show prefixes for the names of the different elements and XAdES properties for space reasons. Nevertheless the explanatory text in the present document shows the qualified names of the elements wherever necessary according to the rules stated in clause 8.1, which is enough to unambiguously identify the elements and XAdES properties involved in the generation and augmentation processes illustrated.



Scenario 1: the signer generates the digital signature also on signed properties/attributes. After generating the signature, augments it by Incorporation of unsigned properties/attributes (among which, in this picture, a time-stamp token.

Figure 2: Signer generating the signature, requesting time-stamp token on the digital signature value, and augmenting it with unsigned properties/attributes

Figure 2 shows a scenario where the signer:

1) **generates** a XAdES signature. The signature incorporates some signed properties, namely: `xades:SigningTime`, `xades:SigningCertificateV2` and `xades:CommitementTypeIndication`;

2) **requests** a time-stamp token to a TSA on the digital signature value; and

3) **augments** the generated signature, once she gets this time-stamp token, by incorporation into the signature of:

   - the obtained time-stamp token encapsulated within the `xades:SignatureTimeStamp` unsigned property; and

   - the references to the validation material within `xadesv141:CompleteCertificateRefsV2`, and `xades:CompleteRevocationRefs` unsigned properties.

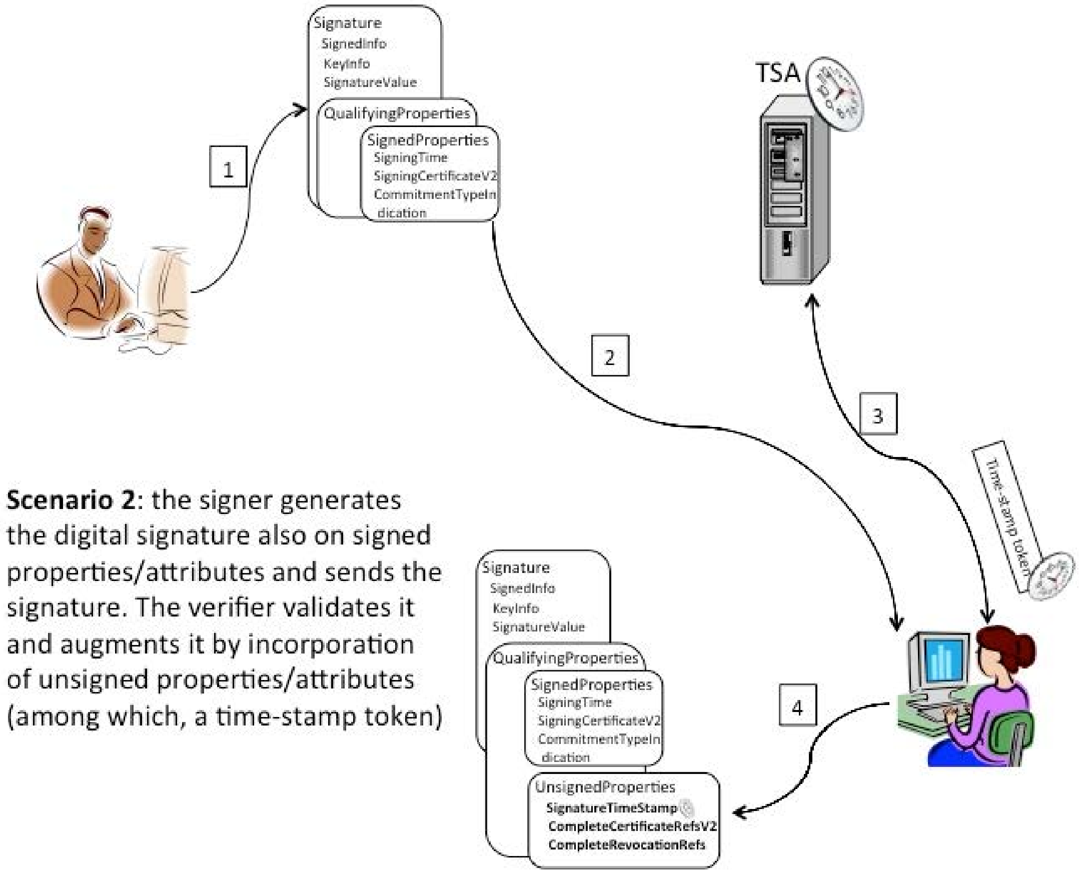
However, the signer could also have opted by not augmenting the signature, and it could be the verifier who, after its validation, has decided to augment it. Figure 3 shows a scenario where:

1) the signer **generates** a XAdES signature, which incorporates the same signed properties as the signature generated in Figure 2;

2) the verifier **validates** the XAdES signature;

3) the verifier **requests** a time-stamp token to a TSA on the digital signature value, and once she gets this time-stamp token;

4) the verifier **augments** the validated signature by incorporation of the obtained time-stamp token and references to the validation material (`xades:SignatureTimeStamp`, `xadesv141:CompleteCertificateRefsV2`, and `xades:CompleteRevocationRefs`) into the signature as unsigned properties.



Scenario 2: the signer generates the digital signature also on signed properties/attributes and sends the signature. The verifier validates it and augments it by incorporation of unsigned properties/attributes (among which, a time-stamp token)

Figure 3: Signer generating the signature, verifier validating it, requesting time-stamp token on the digital signature value, and augmenting it with unsigned properties/attributes

As mentioned several times the legal or regulatory framework that applies to the business process can mandate to be able to validate a signature long time after its generation. ETSI digital signature formats offer this possibility using signature augmentation techniques. Figure 2 and Figure 3 show the first step for achieving long term signatures: the augmentation process when the signer (in Figure 2) or the verifier (in Figure 3) request a time-stamp token to the TSA and they incorporate it into the signature encapsulated within the `xades:SignatureTimeStamp` unsigned property.

A second step for achieving long term signatures from the augmented signatures in Figure 2 and Figure 3 can be to time-stamp the references on the validation material and incorporate the issued time-stamp token into the signature, as shown in Figure 4; this would constitute a proof that at the time indicated in the time-stamp token, the references were present in the signature. Figure 4 shows how the verifier, after validating the signature, can indeed request a time-stamp token on the references to the validation material, and augment the signature by incorporating this time-stamp token, encapsulated within the `xadesv141:RefsOnlyTimeStampV2` unsigned property, into the signature.

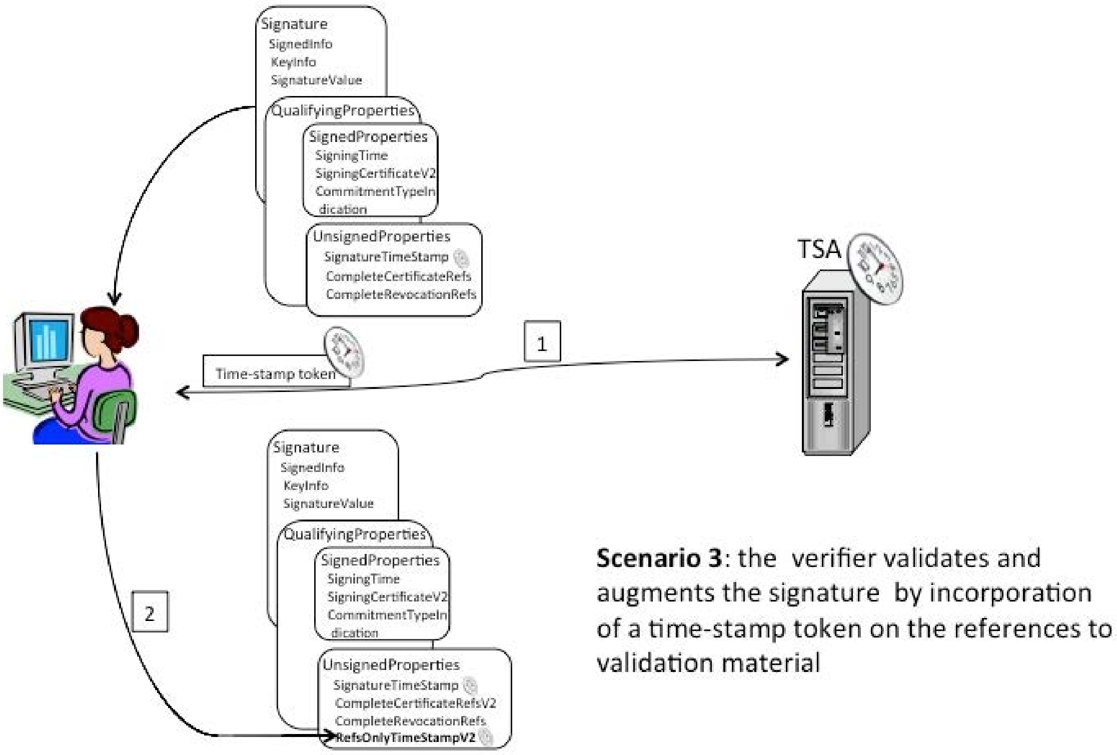


Figure 4: Verifier validating a signature with unsigned properties/attributes, requesting time-stamp token on references to validation material, and augmenting the signature with this time-stamp token

A third step towards a long term digital signature can be the incorporation into the signature of the validation material and a time-stamp token on all the components of the signature and the signed data object(s) (even if they are detached from the signature). By doing that the signature incorporates and secures all the material required for its validation, and entities trying to validate it should not have to search for it. The longevity of the signature is enlarged until the revocation or expiration of the certificates required for validating this time-stamp token, or until the break of one of the algorithms used for generating this time-stamp token. Figure 5 shows how the verifier:

1) After validating the signature **augments** it incorporating into the signature validation material (in bold in step 1 of Figure 5). This includes:

   - Certificates and certificates status values (CRLs, OCSP responses) encapsulated within `xades:CertificateValues` and `xades:RevocationValues` respectively.

   - All the validation material required for validating the time-stamp tokens already incorporated, namely `xades:SignatureTimeStamp` and `xadesv141:RefsOnlyTimeStampV2`. For XAdES this validation material is encapsulated within `xadesv141:TimeStampValidationData` unsigned property. The figure shows one `xadesv141:TimeStampValidationData` unsigned property after the two aforementioned time-stamp token containers. When all the validation material required for validating a certain time-stamp token incorporated into the signature are present elsewhere in the signature (a previously incorporated `xadesv141:TimeStampValidationData` or even fields of the `SignedData` instance of another time-stamp token) or in the time-stamp token itself, no `xadesv141:TimeStampValidationData` is required for this time-stamp token.

2) **Requests a time-stamp token** that covers all the components present in the signature after completing step 1) for achieving long term availability and integrity of the validation material incorporated into the signature.

3) **Augments** the signature incorporating into the signature the aforementioned generated time-stamp token embedded into a `xadesv141:ArchiveTimeStamp` unsigned property. This time-stamp token ensures the integrity of every piece of data within the signature for a period of time that would end at the time instant being the minimum of the expiration date of the time-stamp token signing certificate or some of the certificates in its path, the date of the revocation of any of these certificates, and the date the algorithms used for its computation are broken.

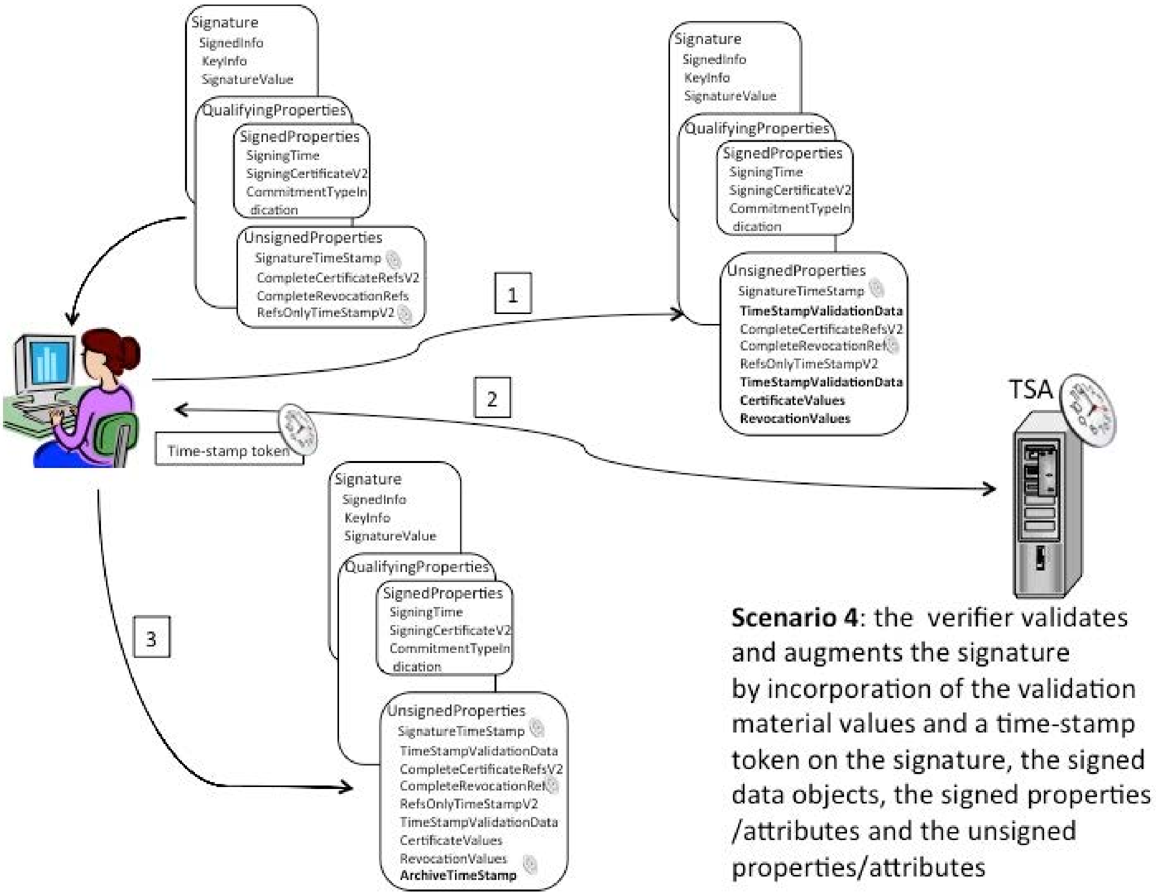


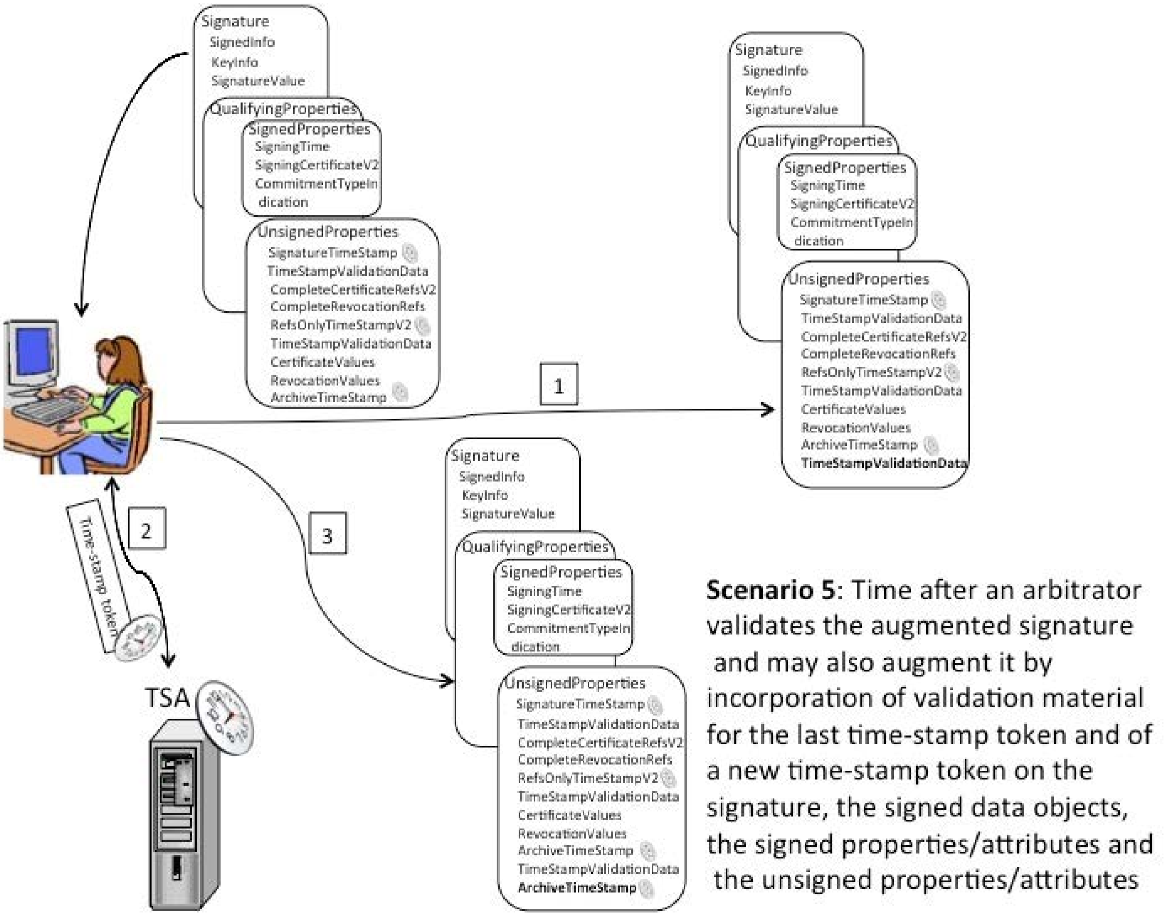
Figure 5: Verifier validating a signature with unsigned properties/attributes, requesting an archive time-stamp token, and augmenting the signature with the validation material and the archive time-stamp token

As mentioned before, the signature obtained in Figure 5 is a long term signature and offers the possibility of being validated for a longer period of time. If this period of time needs to be extended or if the algorithms used for computing the time-stamp token encapsulated within `ArchiveTimeStamp` unsigned property are about to be broken, an entity (for instance a potential arbitrator that has to resolve a potential dispute between the signer and the verifier on the validity of the signature, or a trusted service) can respond to any of the two aforementioned facts properly augmenting the signature as indicated in Figure 6. Figure 6 shows how the arbitrator:

1) After validating the signature **augments** it if needed, the signature incorporating into the signature validation material required for validating the time-stamp token embedded within the already existing `xadesv141:ArchiveTimeStamp`. This material is included within the last `xadesv141:TimeStampValidationData` unsigned property shown in bold in step 1 of Figure 6. If all the validation material required for validating that time-stamp token is already present in the signature, this augmentation is not required.

2) **Requests a time-stamp token** that covers all the components present in the signature after completing step 1) for achieving long term availability and integrity of the validation material incorporated into the signature.

3) **Augments** the signature incorporating into the signature the aforementioned generated time-stamp token embedded into a `xadesv141:ArchiveTimeStamp` unsigned property, shown in bold in step 3 of Figure 6. This time-stamp token ensures the integrity of every piece of data within the signature for a period that is longer than the period of time ensured by the time-stamp embedded in the previous `xadesv141:ArchiveTimeStamp` unsigned property (it would end at the time instant being the minimum of the expiration date of the time-stamp token signing certificate or some of the certificates in its path, the date of the revocation of any of these certificates, and the date the algorithms used for its computation are broken).



Scenario 5: Time after an arbitrator validates the augmented signature and may also augment it by incorporation of validation material for the last time-stamp token and of a new time-stamp token on the signature, the signed data objects, the signed properties/attributes and the unsigned properties/attributes

Figure 6: Arbitrator validating a signature with unsigned properties/attributes, requesting a second archive time-stamp token, and augmenting the signature with the validation material and the archive time-stamp token

The cycle shown in Figure 2 to Figure 6 is actually the longest one but not the unique one. In fact, it is also possible that the business process does not require incorporating into the signature neither the references to validation material nor the time-stamp token on them. Under such circumstances, the signature lifecycle can be the one shown in Figure 7.

Figure 7 illustrates the following steps:

1) The signer **generates** a digital signature incorporating only the signed properties and sends it to the verifier.

2) The verifier **validates** the signature after receiving it and requests a time-stamp token on the signature value.

3) The verifier, after receiving the aforementioned time-stamp token, **augments** the validated signature incorporating this time-stamp token embedded within a `xades:SignatureTimeStamp` unsigned property. The signature is then handed to the arbitrator.

4) The arbitrator **validates** the augmented signature. The arbitrator, before the expiration or revocation of some of the certificates in the path of signing certificate of the time-stamp token encapsulated within `xades:SignatureTimeStamp,` or before the break of the algorithms used for generating such time-stamp token, **augments** the signature by incorporation of the required validation material for validating the signature and the signature time-stamp token in `xadesv141:TimeStampValidationData`, `xades:CertificateValues`, and `xades:RevocationValues` unsigned properties.

5) The arbitrator then requests a new time-stamp token on all the components present in the signature.

6) Finally the arbitrator **augments** once again the signature incorporating the received time-stamp token embedded in a a `xadesv141:ArchiveTimeStamp` unsigned property.

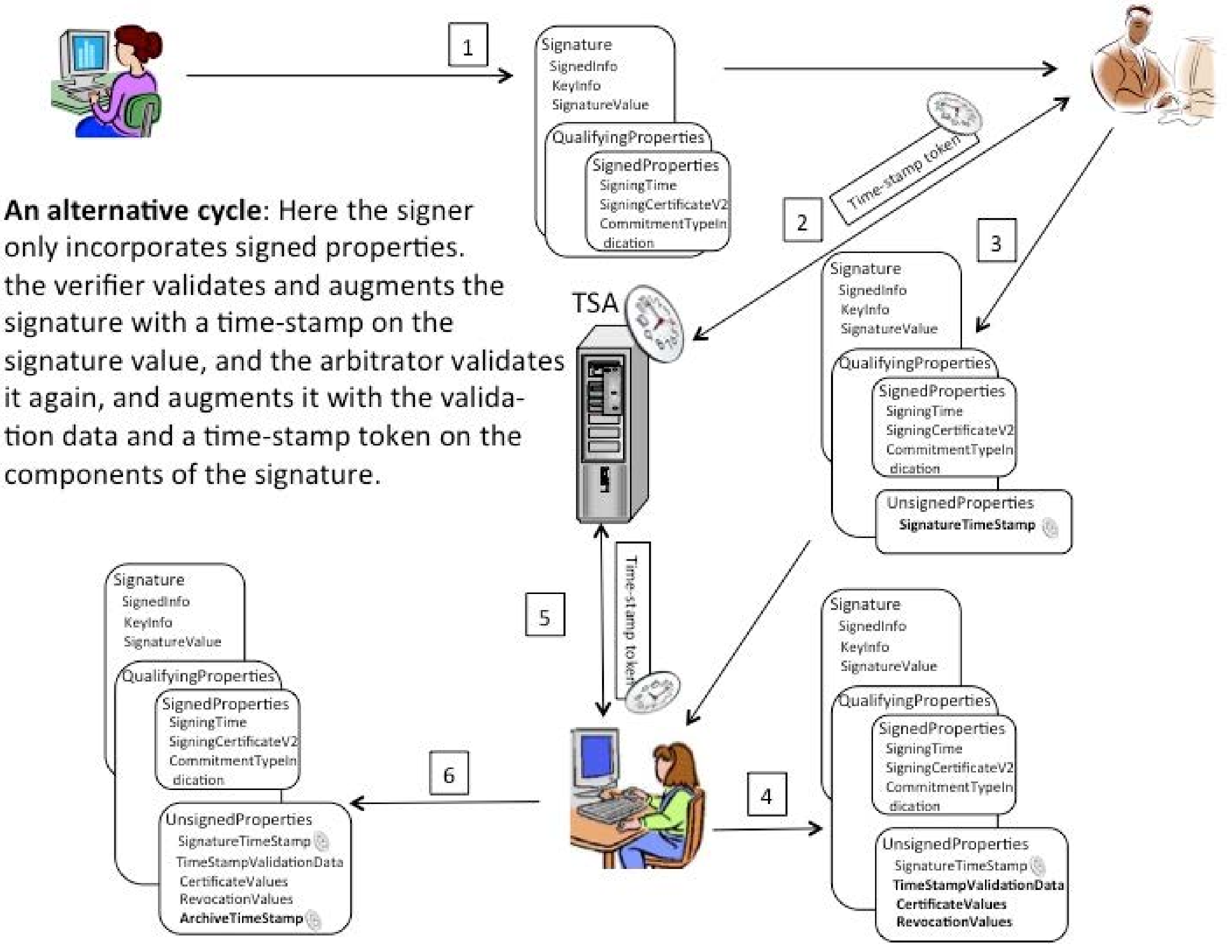
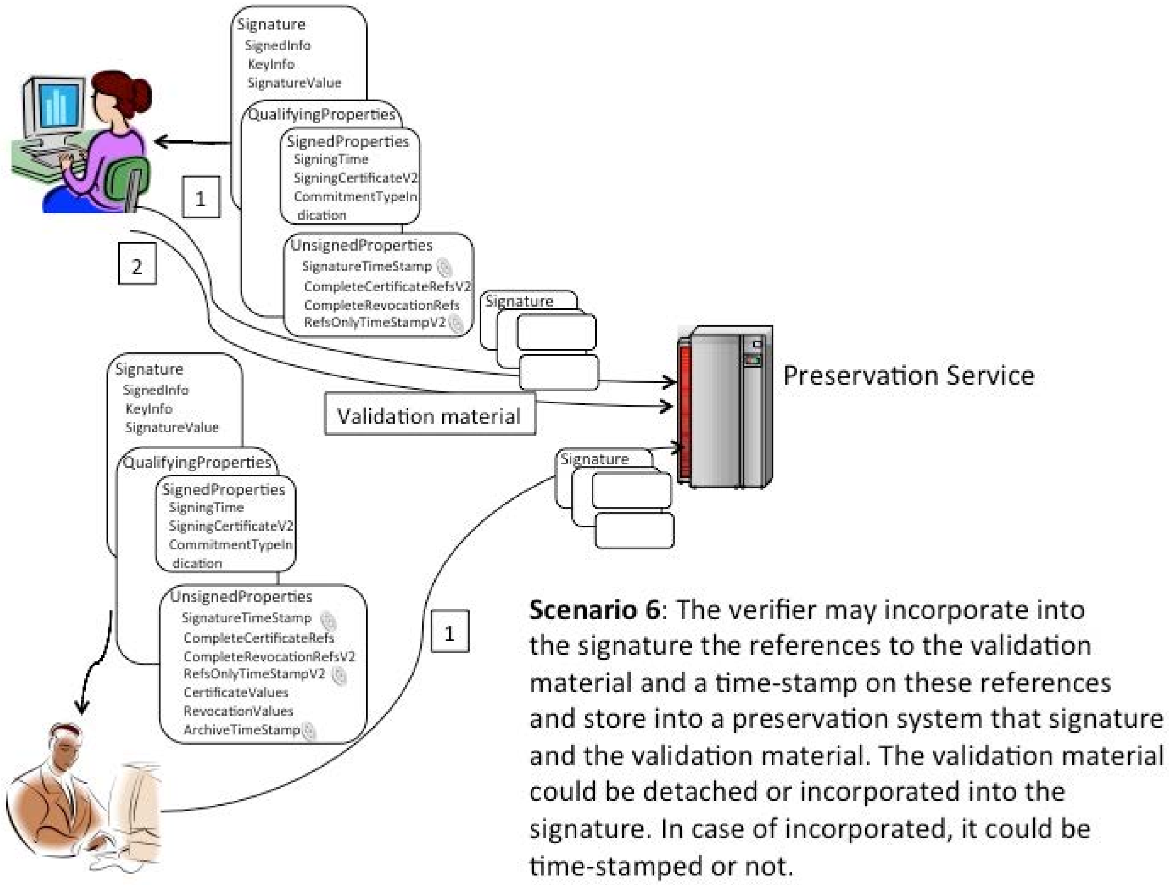

1
Signature
SignedInfo
KeyInfo
SignatureValue
QualifyingProperties
SignedProperties
SigningTime
SigningCertificateV2
CommitmentTypeIn
dication
Time-stamp token
2
3
An alternative cycle: Here the signer only incorporates signed properties. the verifier validates and augments the signature with a time-stamp on the signature value, and the arbitrator validates it again, and augments it with the validation data and a time-stamp token on the components of the signature.
TSA
UnsignedProperties
SignatureTimeStamp
5
Time-stamp token
6
4
TimeStampValidationData
CertificateValues
RevocationValues
ArchiveTimeStamp

Figure 7: Alternative signature lifecycle resulting in augmented signature without references to validation material

Signatures and validation material can also be preserved by a trusted service that ensures the integrity of what it preserves for a long time. In this case, the preservation system is responsible for ensuring the integrity of whatever it preserves for long periods of time, using suitable techniques. Figure 8 shows two verifiers using such kind of service. The first one gives to the preservation service an augmented signature that incorporates references to the validation material and a time-stamp token on these references (step 1). Under such circumstances the validation material is preserved separately from the signature, and consequently the verifier passes this validation material in step 2. The second verifier, though, sends to the preservation system a signature with all the validation material and one `ArchiveTimeStamp` and there is no need to store the validation material separately.



Scenario 6: The verifier may incorporate into the signature the references to the validation material and a time-stamp on these references and store into a preservation system that signature and the validation material. The validation material could be detached or incorporated into the signature. In case of incorporated, it could be time-stamped or not.

Figure 8: Verifiers validating signatures with unsigned properties/attributes and storing them in a trusted preservation service

8.11.6.2 Lifecycle and levels of digital signatures

ETSI EN 319 1x2 (with x = 2, 3, and 4) specifying signature formats define several levels for the signatures. Each level, within the aforementioned ETSI ENs, is defined by:

1) a certain combination of signed and unsigned attributes in CAdES, of signed and unsigned properties in XAdES, or signed and unsigned attributes, and dictionaries in PAdES; and

2) a set of specific requirements for the attributes/properties/dictionaries in each level.

These levels actually define a common technical language for exchanging knowledge about the relevant contents of a certain digital signature within its lifecycle.

These levels build a technical taxonomy for the signatures, and that this taxonomy is independent of the legal taxonomy defined by any regulatory framework as for instance the one defined by the Regulation (EU) No 910/2014 [i.26].

However, this allows that these levels specified in the ETSI ENs, can be explicitly referenced in the components of the regulatory frameworks (like secondary legislation within the EU for instance) as the formats to be used for technically implementing electronic signatures (or electronic seals) reaching certain legal levels.

For baseline signatures, each specification defines four levels addressing incremental requirements to maintain the validity of the signatures over the long term, in a way that a certain level always addresses all the requirements addressed at levels that are below it. Below follows the list of levels defined in each document and an outline of their main goals. The actual identifiers assigned for the levels in each specification, are the ones below preceded by CAdES-, PAdES-, and XAdES-:

a) B-B level provides requirements for the incorporation of signed and some unsigned attributes/qualifying properties when the signature is generated.

b) B-T level provides requirements for the generation and inclusion, for an existing signature, of a trusted token proving that the signature itself actually existed at a certain date and time.

c) B-LT level provides requirements for the incorporation of all the material required for validating the signature in the signature document. This level aims to tackle the long term availability of the validation material.

d) B-LTA level provides requirements for the incorporation of electronic time-stamps that allow validation of the signature long time after its generation. This level aims to tackle the long term availability and integrity of the validation material.

ETSI EN 319 162-1 [i.8] defines four levels for ASiC baseline containers addressing incremental requirements to maintain the validity of the signatures and time-assertions within the containers over the long term, in a way that a certain level always addresses all the requirements addressed at levels that are below it. Below follow some details on ASiC baseline containers:

1) For ASiC-S containers with one XAdES signature, ETSI EN 319 162-1 [i.8], clauses 5.1, 5.2, 5.3.1, 5.3.2.1, and 5.3.2.3 specify ASiC-B-B, ASiC-B-T, ASiC-B-LT, and ASiC-B-LTA containers. The level of the ASiC container is the level of the embedded XAdES signature.

2) For ASiC-S containers with one CAdES signature, ETSI EN 319 162-1 [i.8], clauses 5.1, 5.2, 5.3.1, 5.3.2.1, and 5.3.2.2 specify ASiC-B-B, ASiC-B-T, ASiC-B-LT, and ASiC-B-LTA containers. The level of the ASiC container is the level of the embedded CAdES signature.

3) For ASiC-E containers with several XAdES signatures, ETSI EN 319 162-1 [i.8], clauses 5.1, 5.2, 5.3.1 and 5.3. 3 specify ASiC-B-B, ASiC-B-T, ASiC-B-LT, and ASiC-B-LTA containers. The level of the ASiC container will be the level of that XAdES signature whose level is the lowest one among all the XAdES signatures present within the ASiC Container.

No baseline containers are specified for ASiC-E containing several CAdES signatures.

For CAdES/XAdES extended signatures, PAdES additional signature profiles, and ASiC additional containers, the number of levels is higher.

In a good number of occasions, the signatures created by the signer incorporate signed attributes/ properties and no validation data in unsigned attributes/properties. ETSI EN 319 102-1 [i.10] calls them **Basic** signatures. Below follows the list of signature levels that can be included within this generic denomination:

1) CAdES signatures of levels **CAdES-B-B**, **CAdES-E-BES**, and **CAdES-E-EPES**. CAdES B-B level is defined in ETSI EN 319 122-1 [i.2], clause 6.3, and gets its name from "Baseline Basic". CAdES-E-BES and CAdES-E-EPES are defined in ETSI EN 319 122-2 [i.3], clause 4.3; they get their names from "Extended Basic", and "Extended with Explicitly Policy based" (as it mandatorily incorporates `signature-policy-identifier` signed attribute), respectively. See the aforementioned references for checking the mandatory and optional signed attributes for each level.

2) XAdES signatures of levels **XAdES-B-B**, **XAdES-E-BES**, and **XAdES-E-EPES**. XAdES B-B level is defined in ETSI EN 319 132-2 [i.5], clause 6.3. XAdES-E-BES and XAdES-E-EPES (where it is mandatory to incorporate the `xades:SignaturePolicyIdentifier` signed property) are defined in ETSI EN 319 132-1 [i.4], clause 4.3. The origin of their names is as names for CAdES levels. See the aforementioned references for checking the mandatory and optional signed properties for each level.

3) PAdES signatures of levels **PAdES-B-B**, **PAdES-E-BES**, and **PAdES-E-EPES**. PAdES-B-B level is defined in ETSI EN 319 132-2 [i.5], clause 6.3. PAdES-E-BES, and PAdES-E-EPES (where it is mandatory to incorporate the `signature-policy-identifier` signed attribute into the CAdES signature present in the signature dictionary PDF object) are defined in ETSI EN 319 142-2 [i.7], clauses 5.3 and 5.4 respectively. See the aforementioned references for checking the mandatory and optional signed attributes for each level.

NOTE: Strictly speaking all the CAdES and XAdES levels listed above can incorporate unsigned attributes/properties encapsulating validation data and/or time-stamp tokens, as in fact, their corresponding specifications only recommend not incorporating them (but they do not prohibit it); nevertheless this is a recommendation that is widely followed by implementers. Similarly, all the PAdES levels listed above can incorporate some unsigned attributes within their CAdES signatures present in the signature dictionary PDF object, or can also incorporate DSS, VRI, and document time-stamps PDF dictionaries, as their specifications only recommend not incorporating them (but they do not prohibit it). As before, this is a widely followed recommendation by implementers and they are not incorporated.

Figure 9 shows examples of Basic CAdES, PAdES and XAdES signatures.

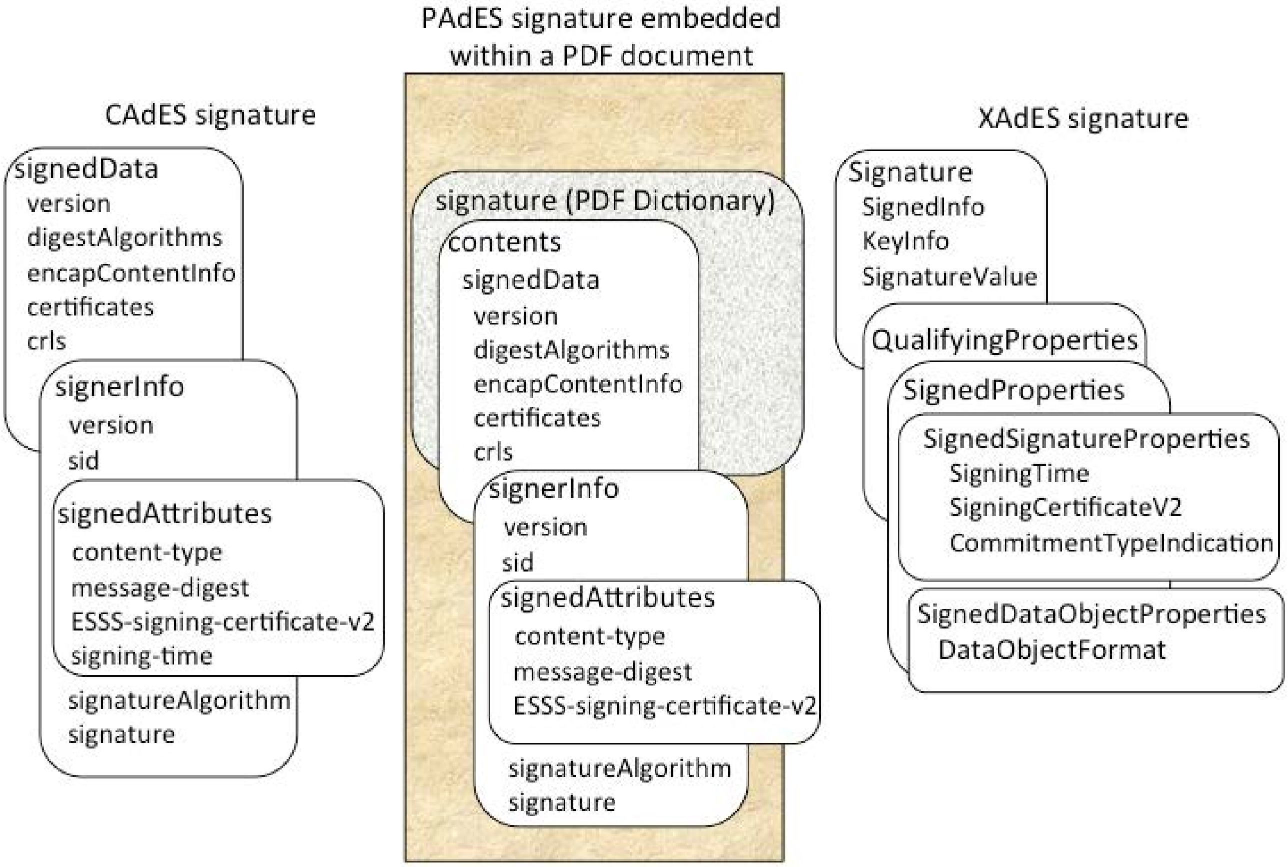


Figure 9: CAdES, PAdES and XAdES Basic signatures

ETSI EN 319 102-1 [i.10] calls **signatures with time** those ones resulting from incorporating a time-stamp token into the basic signature. Below follows the list of signature levels that can be included within this generic denomination:

1) CAdES signatures of levels **CAdES-B-T** (defined in ETSI EN 319 122-1 [i.2], clause 6.3), and **CAdES-E-T** (defined in ETSI EN 319 122-2 [i.3], clause 4.3). They build respectively on CAdES-B-B, and CAdES-E-BES or CAdES-E-EPES by incorporation into the signature of one or more time-stamp tokens on the signature value encapsulated within `signature-time-stamp` unsigned attributes.

2) XAdES signatures of levels **XAdES-B-T** (defined in ETSI EN 319 132-1 [i.4], clause 6.3) and **XAdES-E-T** (defined in ETSI EN 319 132-2 [i.5], clause 4.3). They build respectively on XAdES-B-B, and XAdES-E-BES or XAdES-E-EPES by incorporation into the signature of one or more time-stamp tokens on the signature value encapsulated within `xades:SignatureTimeStamp` unsigned properties.

3) PAdES signatures of levels **PAdES-B-T** (defined in ETSI EN 319 142-1 [i.6], clause 6.3). They build on PAdES-B-B by incorporation into the signature of:

   - one or more time-stamp tokens on the signature value encapsulated within `signature-time-stamp` unsigned attributes of the CAdES signature present within the signature dictionary of PAdES; or

   - one or more time-stamp tokens on the PAdES document as specified in ETSI EN 319 142-1 [i.6], encapsulated within the document time-stamp dictionary.

Figure 10 shows examples of CAdES, PAdES, and XAdES signatures with time.

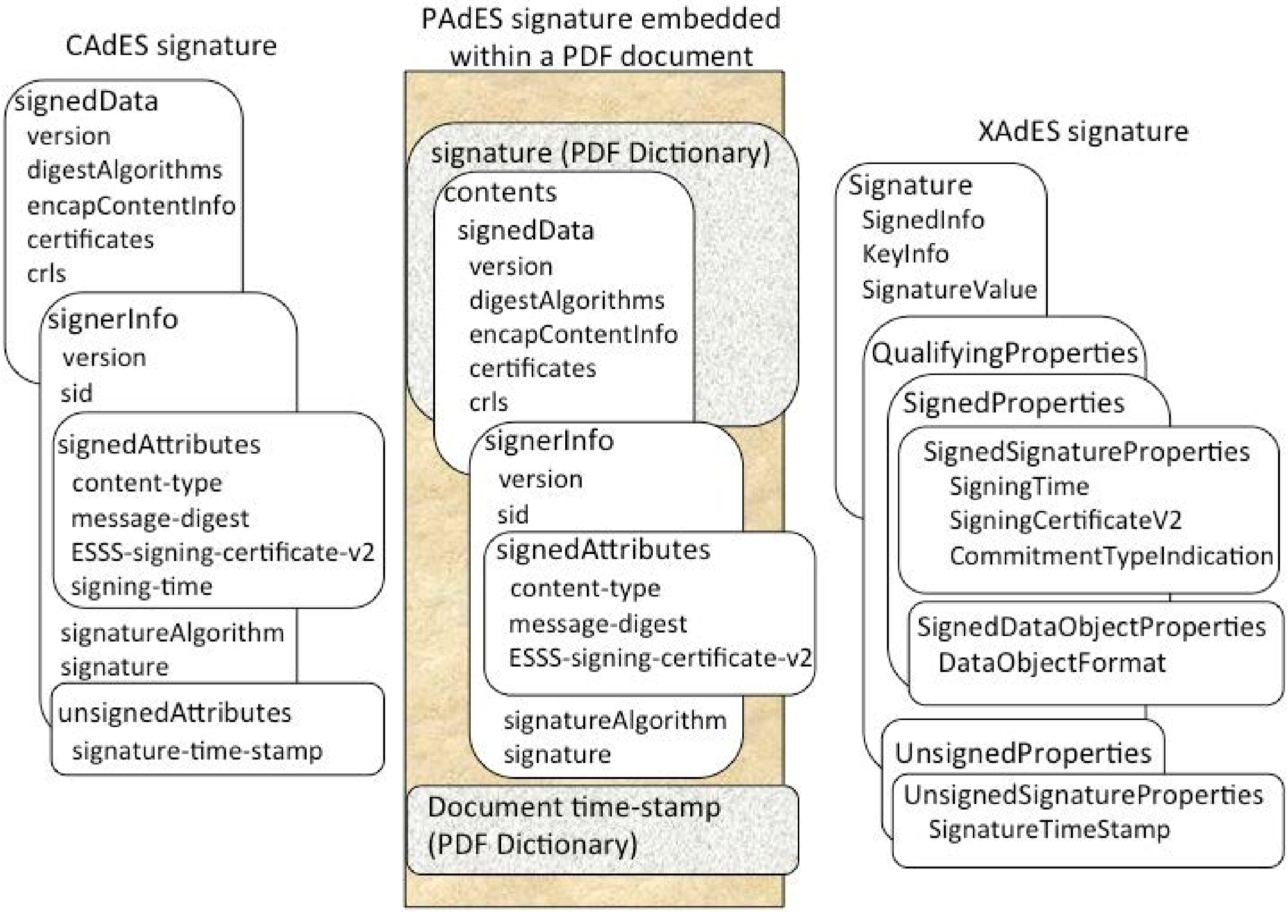


Figure 10: CAdES, PAdES and XAdES signatures with time

ETSI EN 319 102-1 [i.10] calls **signatures with long term validation material** those ones resulting from incorporating validation material (or/and references to this validation material) to signatures with time. Below follows the list of signature levels that can be included within this generic denomination:

1) CAdES signatures of levels **CAdES-B-LT** (defined in ETSI EN 319 122-1 [i.2], clause 6.3), **CAdES-E-C, CAdES-E-X, CAdES-E-X-Long**, and **CAdES-E-X-L** (defined in ETSI EN 319 122-2 [i.3], clause A.1). CAdES-B-LT builds on CAdES-B-T by incorporation of validation material for the signature. The CAdES-E-C signatures build on CAdES-E-T signatures by incorporation of references to validation material. The CAdES-E-X build on CAdES-E-C by incorporation of time-stamp tokens on these references and validation material. The CAdES-E-X-Long signatures build on CAdES-E-C signature by incorporation of certificates and revocation values. Finally CAdES-E-X-L signatures are built on CAdES-E-X signatures by incorporation of certificates and revocation values. See the aforementioned references for all the details.

2) XAdES signatures of levels **XAdES-B-LT** (defined in ETSI EN 319 132-1 [i.4], clause 6.3), **XAdES-E-C, XAdES-E-X, XAdES-E-X-Long**, and **XAdES-E-X-L** (defined in ETSI EN 319 132-2 [i.5], clause A.1). XAdES-B-LT builds on XAdES-B-T by incorporation of validation material for the signature. The XAdES-E-C signatures build on XAdES-E-T signatures by incorporation of references to validation material. The XAdES-E-X build on XAdES-E-C by incorporation of time-stamp tokens on these references and validation material. The XAdES-E-X-Long signatures build on XAdES-E-C signature by incorporation of certificates and revocation values. Finally XAdES-E-X-L signatures are built on XAdES-E-X signatures by incorporation of certificates and revocation values. See the aforementioned references for all the details.

3) PAdES signatures of level **PAdES-B-LT** (defined in ETSI EN 319 142-1 [i.6], clause 6.3) and **PAdES-E-LTV** (defined in ETSI EN 319 142-2 [i.7], clause 5.5) without document time-stamp. PAdES-B-LT builds on PAdES-B-T by incorporation of validation material within DSS dictionary PDF object (and optionally within VRI dictionary objects). PAdES-E-LTV builds on PAdES-E-BES or PAdES-E-PES by incorporation of validation material within DSS dictionary PDF object (and optionally within VRI dictionary objects).

Figure 11 shows examples of CAdES, PAdES, and XAdES signatures with long term validation material. Note that validation material in CAdES is added in fields `certificates` and `crls` of the instance of `SignedDataType` instead within any unsigned attributes.

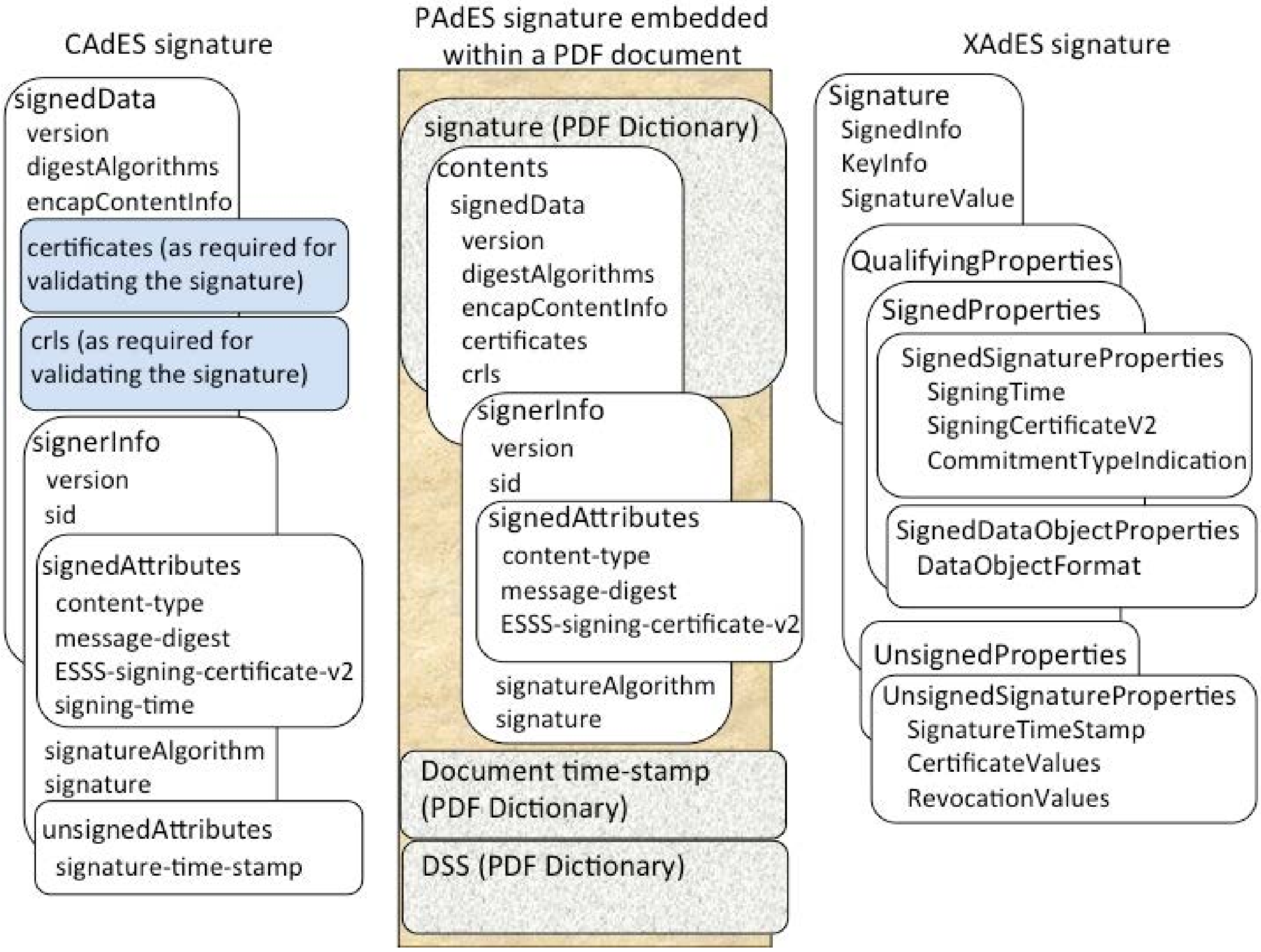


Figure 11: CAdES, PAdES and XAdES signatures with long term validation material

Finally, ETSI EN 319 102-1 [i.10] calls signatures for **long term availability and integrity of validation data** those ones resulting from incorporation of a time-stamp token covering the content of the signature to signatures with long term validation data. Below follows the list of signature levels that can be included within this generic denomination:

1) CAdES signatures of levels **CAdES-B-LTA** (defined in ETSI EN 319 122-1 [i.2], clause 6.3), and **CAdES-E-A** (defined in ETSI EN 319 122-2 [i.3], clauses 4.3 and A.1). CAdES-B-LTA builds on CAdES-B-LT by incorporation of a time-stamp token on the components of the signature, encapsulated in `archive-time-stamp-v3` unsigned attribute. CAdES-E-A signatures build on CAdES-E-T or any of the levels built on CAdES-E-T by incorporation of a time-stamp token on the components of the signature, encapsulated in `archive-time-stamp-v3` unsigned attribute. See the aforementioned references for all the details.

2) XAdES signatures of levels **XAdES-B-LTA** (defined in ETSI EN 319 132-1 [i.4], clause 6.3), and **XAdES-E-A** (defined in ETSI EN 319 132-2 [i.5], clauses 4.3, and A.1). XAdES-B-LTA builds on XAdES-B-LT by incorporation of a time-stamp token on the components of the signature, encapsulated in `xadesv141:ArchiveTimeStamp` unsigned property. XAdES-E-A signatures build on XAdES-E-T or any of the levels built on XAdES-E-T by incorporation of a time-stamp token on the components of the signature, encapsulated in `xadesv141:ArchiveTimeStamp` unsigned property. See the aforementioned references for all the details.

3) PAdES signatures of levels **PAdES-B-LTA** (defined in ETSI EN 319 142-1 [i.6], clause 6.3), and **PAdES-E-LTV** with a document time-stamp PDF dictionary (defined in ETSI EN 319 142-2 [i.7], clause 5.5). XAdES-B-LTA builds on XAdES-B-LT by incorporation of a time-stamp token on the contents of the PDF document, encapsulated within a document time-stamp PDF dictionary. PAdES-E-LTV signatures can also incorporate a time-stamp token on the contents of the document encapsulated within a document time-stamp PDF dictionary.

Figure 12 shows CAdES, PAdES, and XAdES signatures long term availability and integrity of validation data.

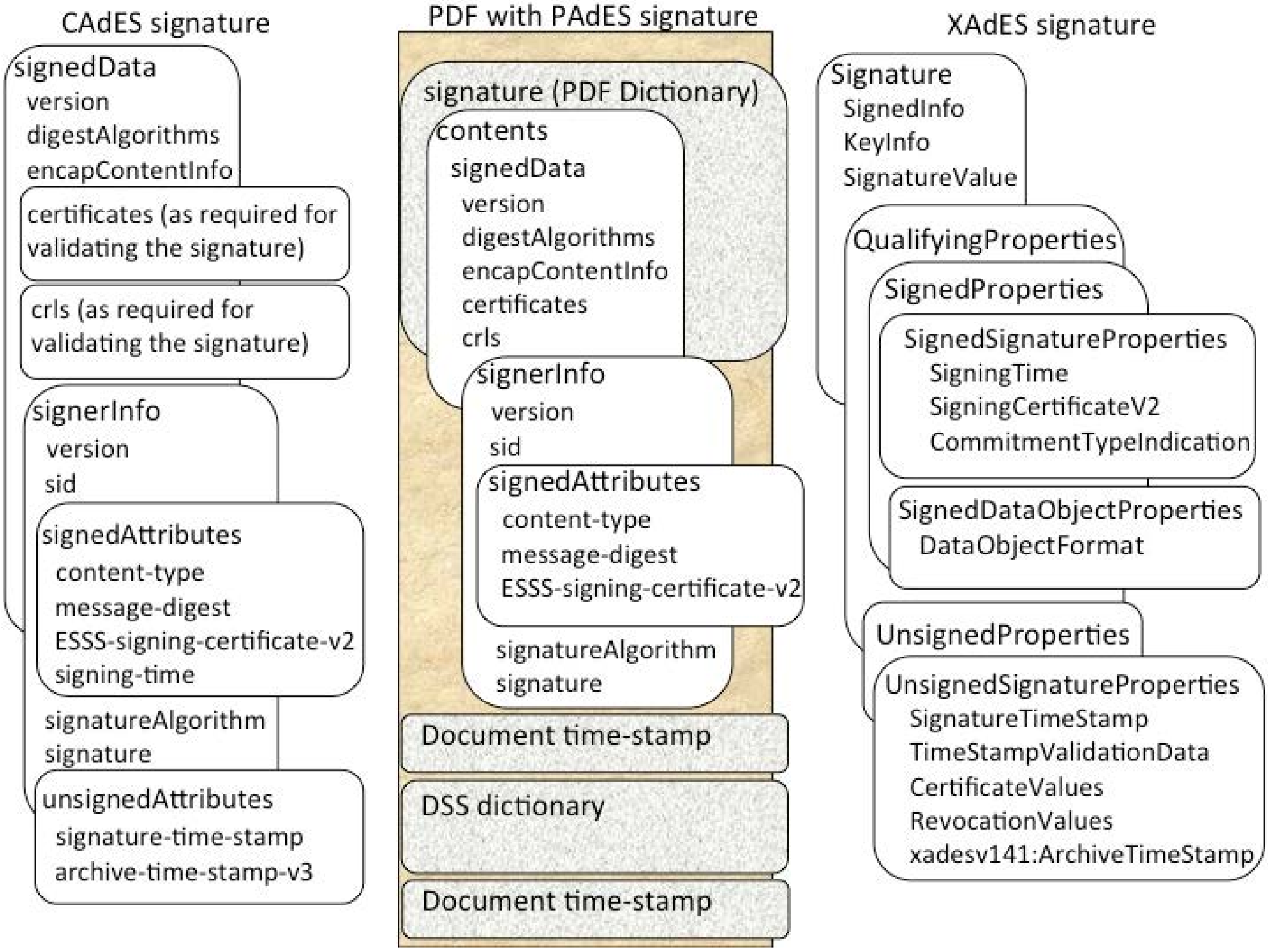


Figure 12: CAdES, PAdES and XAdES signatures for long term availability and integrity of validation data

Figure 10 to Figure 12 show the 4 transitions that CAdES, PAdES, and XAdES baseline signatures can suffer during their life from their initial levels, (C/P/X)AdES-B-B to (C/P/X)AdES-B-LTA levels.

8.11.6.3 Transitions between levels of baseline signatures and containers

A signature specified in ETSI EN 319 122-1 [i.2], ETSI EN 319 132-1 [i.4], or ETSI EN 319 142-1 [i.6] and an ASiC container specified in ETSI EN 319 162-1 [i.8] can sequentially go from *-B-B level to *-B-T, to *-B-LT, and to *-B-LTA (where * respectively stands for CAdES, XAdES, PAdES, and ASiC).

8.11.6.4 Transitions between levels of extended signatures

The present clause shows maps of the possible transitions between levels that CAdES and XAdES extended signatures can go through during their life.

Figure 13 shows the possible augmentations that an initial XAdES-E-EPES signature (the same paths would have been shown in the case the initial signature would have been a XAdES-E-BES signature) can go through for arriving to the XAdES-E-A level. The figure only shows the incorporation of the first `xadesv141:ArchiveTimeStamp` to the signature for achieving the XAdES-E-A level, and does not show how the longevity of the signatures can be enlarged by adding additional `xadesv141:ArchiveTimeStamp` unsigned properties.

Figure 13 shows the different paths which, starting in a XAdES-E-EPES level, can lead to XAdES-E-A level. Obviously not all the signatures generated will need to be augmented up to the XAdES-E-A level: the specific electronic business and the applicable regulatory framework will be determining the level that the signatures managed need to achieve and the path(s) within Figure 13 that the augmentations need to follow. The figure shows in bold those properties that are incorporated into the signature during the different augmentations.

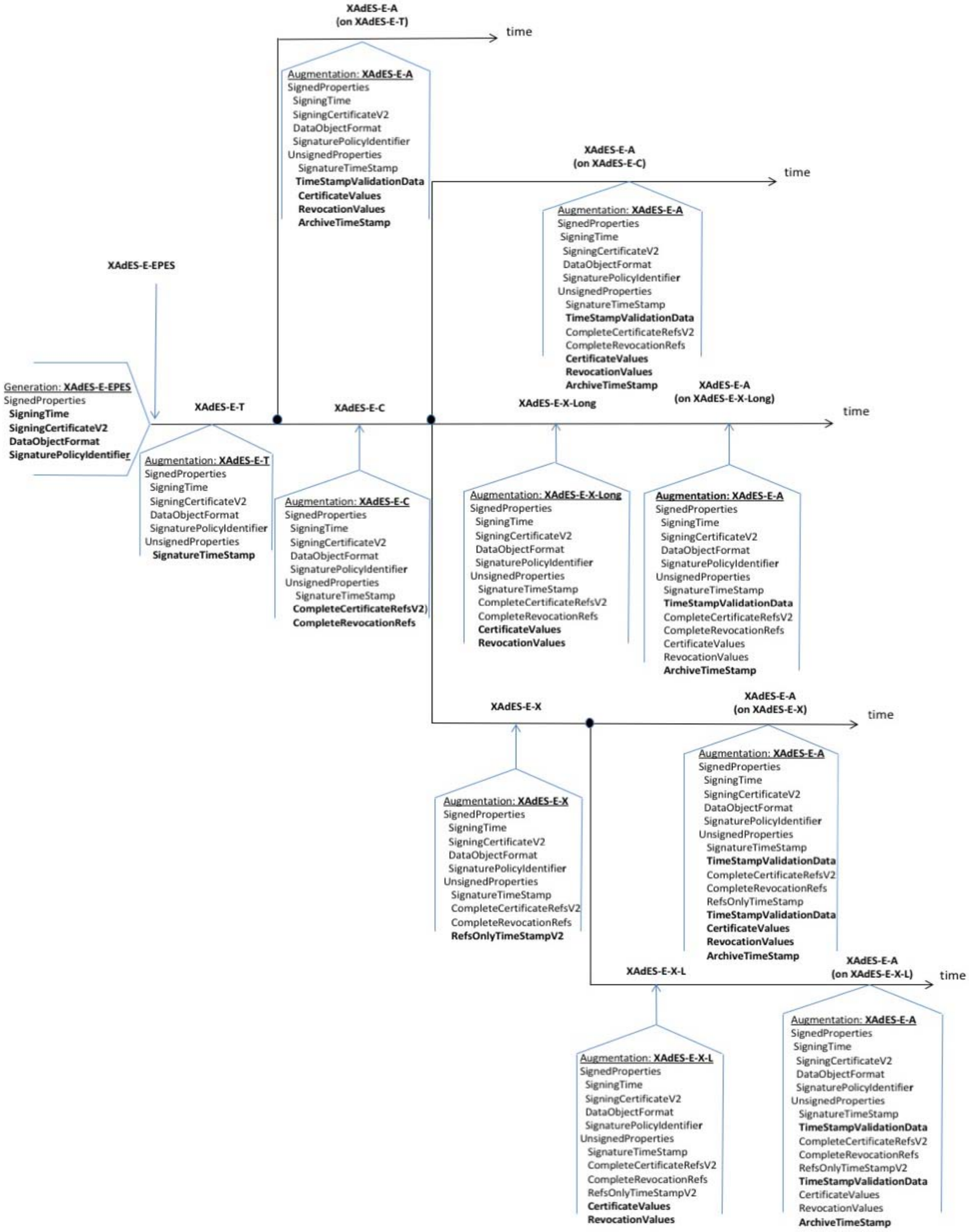


Figure 13: Transitions between levels for XAdES extended signatures

Figure 13 shows that regardless the final level a XAdES-E signature needs to achieve, the first augmentation will be the one that generates a XAdES-E-T signature by incorporation of the `xades:SignatureTimeStamp` unsigned property.

From XAdES-E-T level it is possible to directly augment the signature to a XAdES-E-A by incorporation of all the required validation material and one time-stamp token embedded within a `xadesv141:ArchiveTimeStamp` unsigned property. It is also, however, possible to augment the signature incorporating references to validation material, to XAdES-E-C level, which opens different alternative paths towards XAdES-E-A level. XAdES-E-A signatures built on XAdES-E-Long are similar to the XAdES-E-A signatures directly built on XAdES-E-C, as both of them build on a XAdES-E-C by incorporation of validation material and one time-stamp token embedded within a `xadesv141:ArchiveTimeStamp` unsigned property.

But it is also possible to augment a XAdES-E-C signature towards the XAdES-E-X level by incorporating a time-stamp token either on the references (as shown in Figure 13) or on the signature value and the references. XAdES-E-X signatures can either be directly augmented to XAdES-E-A level or to XAdES-E-X-L level and from there to XAdES-E-A. It can be noticed that these two last XAdES-E-A signatures are similar in terms of contents.

Figure 14 shows the different augmentations that a CAdES-E-EPES signature can go through.

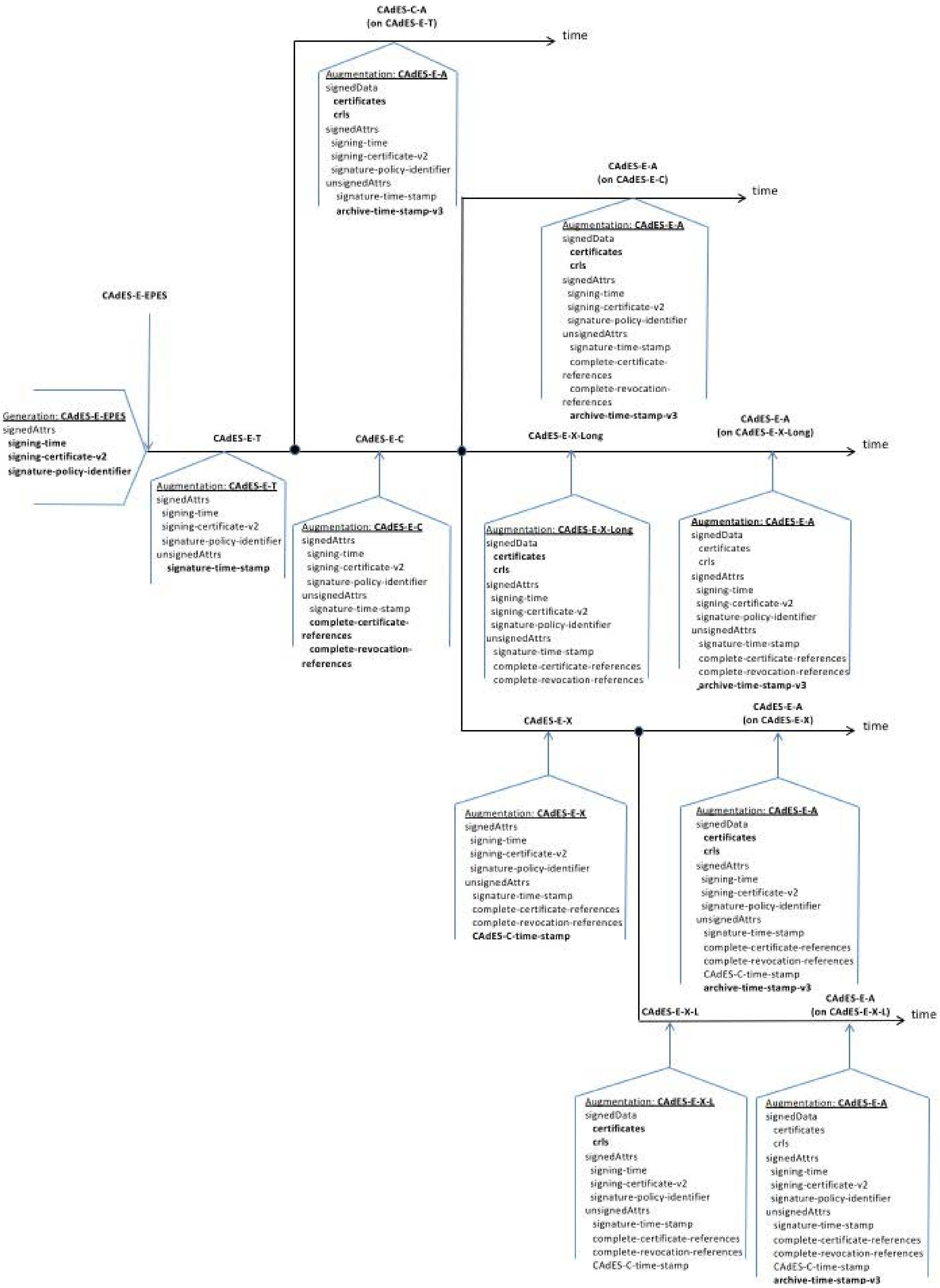


Figure 14: Transitions between levels for CAdES extended signatures

The augmentations in CAdES signatures are conceptually similar to the augmentations in XAdES signatures. The only remarkable difference is that the validation material in CAdES signatures is incorporated in the `certificates` and `crls` fields of the instance of `SignedData` type. This is the reason why Figure 14 explicitly shows these fields in those augmentations where this validation material is incorporated into the signature. This does not mean that these fields are not present in the initial CAdES-E-EPES signature, only that at those augmentations their content change by incorporation of additional material.

8.11.7 ASiC containers lifecycle

The present clause shows maps of the possible transitions between levels corresponding that ASiC containers can go through during they lifecycle.

Figure 15 shows the transitions for an ASiC-S container with one XAdES signature. This figure shows how the longevity of the signature within the package can be enlarged by using the augmentation techniques specified in ETSI EN 319 132-1 [i.4], without any further additions.

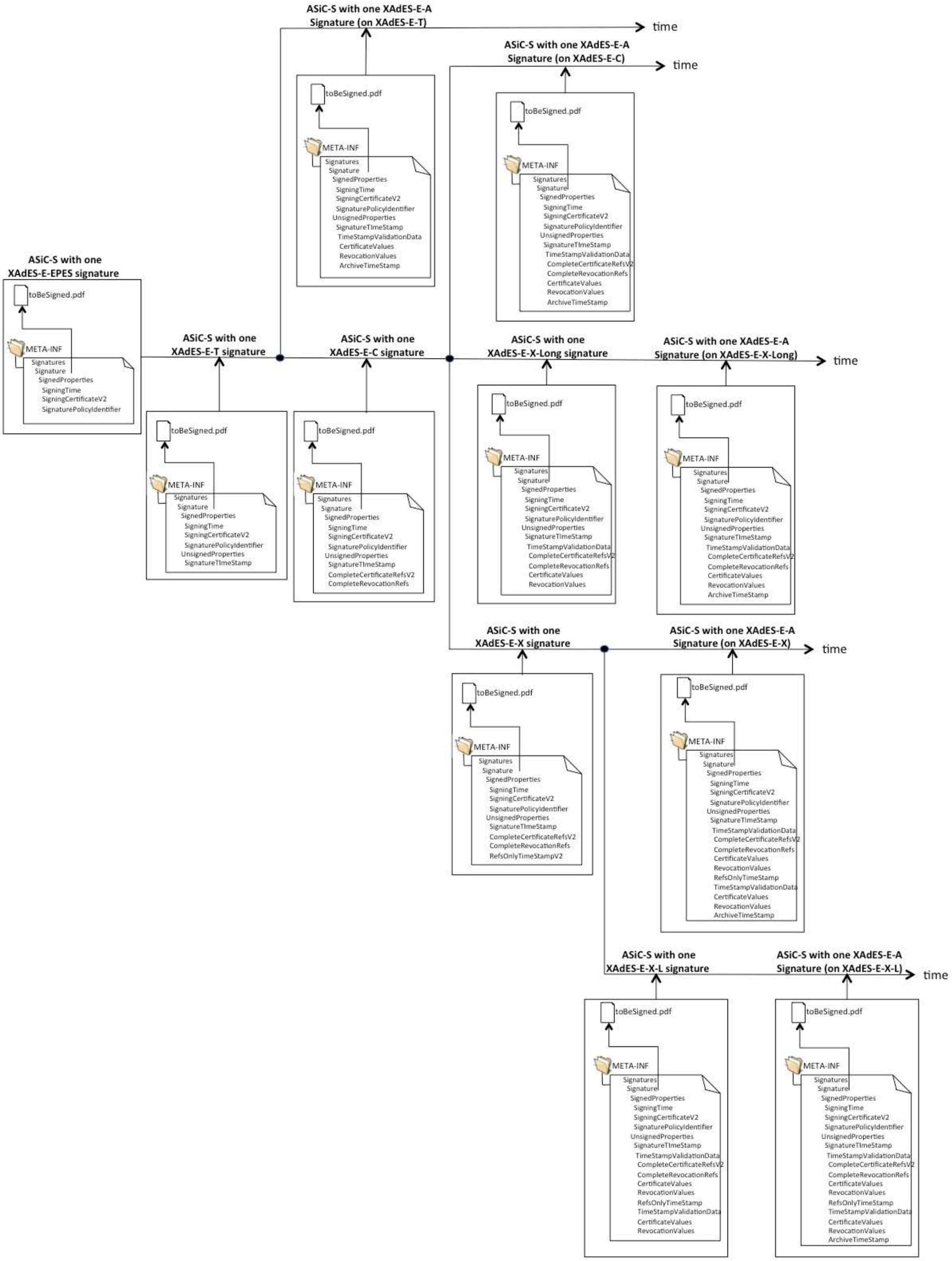


Figure 15: Transitions between levels in ASiC-S with XAdES signatures

Figure 16 shows the transitions for an ASiC-S container with one CAdES. This figure shows how the longevity of the signature within the package can be enlarged by using the augmentation techniques specified in ETSI EN 319 122-1 [i.2] without any further additions.

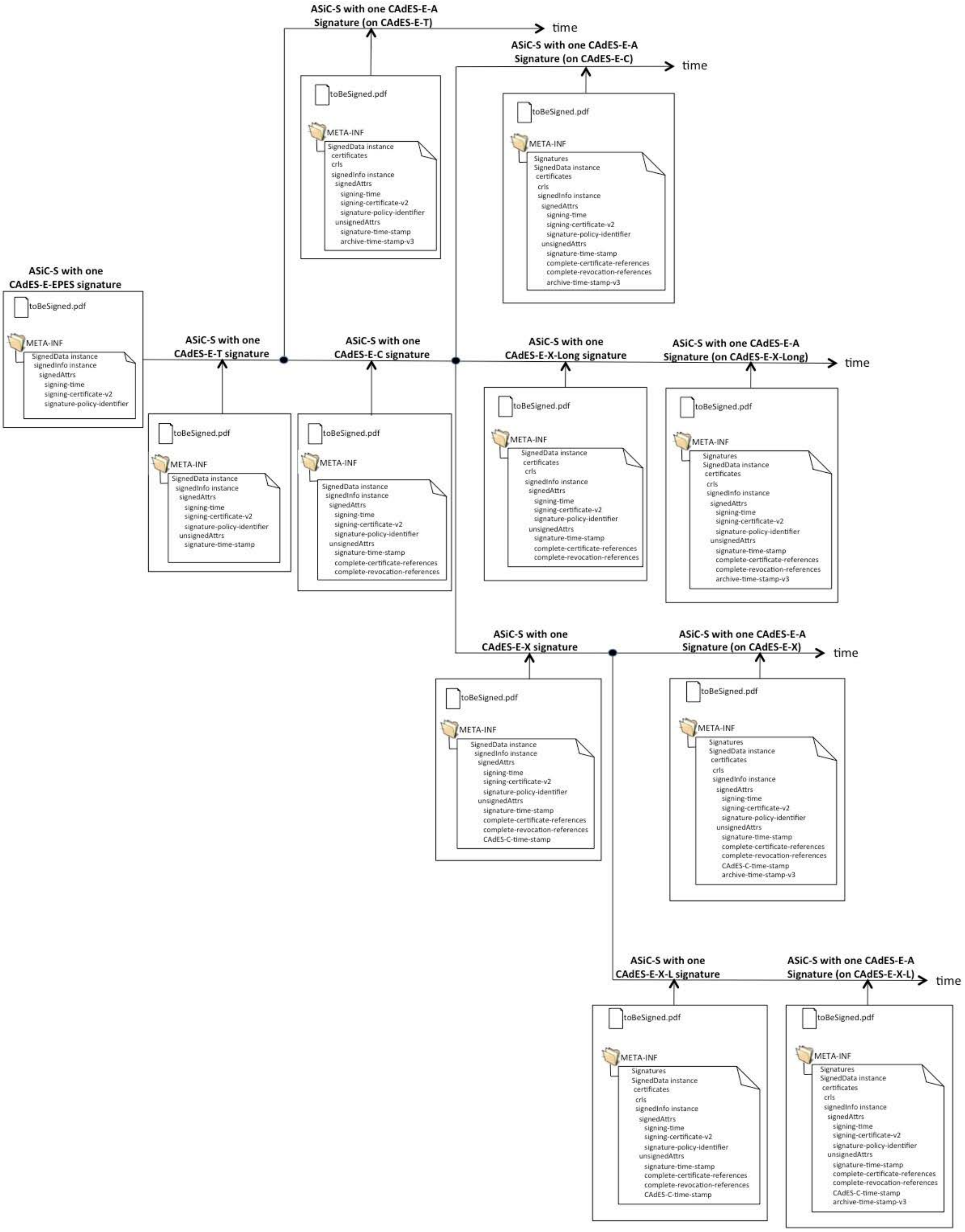

ASiC-S with one CAdES-E-EPES signature
toBeSigned.pdf
META-INF
SignedData instance
signedInfo instance
signedAttrs
signing-time
signing-certificate-v2
signature-policy-identifier
ASiC-S with one CAdES-E-T signature
unsignedAttrs
signature-time-stamp
ASiC-S with one CAdES-E-A Signature (on CAdES-E-T)
certificates
crls
archive-time-stamp-v3
time
ASiC-S with one CAdES-E-C signature
complete-certificate-references
complete-revocation-references
ASiC-S with one CAdES-E-A Signature (on CAdES-E-C)
Signatures
ASiC-S with one CAdES-E-X-Long signature
ASiC-S with one CAdES-E-A Signature (on CAdES-E-X-Long)
ASiC-S with one CAdES-E-X signature
CAdES-C-time-stamp
ASiC-S with one CAdES-E-A Signature (on CAdES-E-X)
ASiC-S with one CAdES-E-X-L signature
ASiC-S with one CAdES-E-A Signature (on CAdES-E-X-L)

Figure 16: Transitions between levels for ASiC-S containers with CAdES signatures

Figure 17 to Figure 21 show some transitions for one ASiC-E container embedding one CAdES signature on two pdf files.

The starting point of this example is an ASiC-E container as shown in Figure 17, embedding one CAdES-E-EPES signature, two pdf files (the data objects that are indirectly signed) and one ASiCManifest file. This file contains one ASiCManifest XML element. The first child of this element is the SigReference element, whose URI attribute points to the file that contains the CAdES-E-EPES signature. The rest of ASiCManifest's children are DataObjectReference elements. There are as many DataObjectReference elements as data object files signed by the CAdES signature. Each DataObjectReference element contains the following information, corresponding to one of these data object files: an URI to this file (within the DataObjectReference's URI attribute), the digest value of this file, and an indication of the digest algorithm used for computing the aforementioned digest value. Finally, the CAdES signature signs the ASiCManifest file, which implies that this CAdES signature is an indirect signature of the two data object files.

Figure 17 shows how the ASiC-E container is if the CAdES signature is augmented to CAdES-E-T and finally the contents of the ASiC-E if the container is augmented for achieving availability and integrity of validation data.

The archive-time-stamp-v3 unsigned attribute used in isolated CAdES signatures, cannot be used in augmentations for achieving containers for availability and integrity of validation data (and consequently long-term containers). In general, an ASiC-E container can embed more than one CAdES signature within different files, and each one signing a different subset of the data object files present within the container. As there is no native mechanism within CAdES allowing to explicitly identify the data object files signed by a certain CAdES signature, the corresponding archive-time-stamp-v3 unsigned attributes of these signatures could not be properly verified.

For achieving ASiC-E containers embedding CAdES signatures able to deal with availability and integrity of validation material, two new files are added within the ASiC-E container as shown in Figure 17, namely:

1) An ASiCArchiveManifest file, which also contains one ASiCManifest XML, which is built as indicated below:

   - The URI attribute of the SigReference child points to the file containing the time-stamp token added for enlarging the longevity of the CAdES signatures within the ASiC-E container, and whose message imprint is computed as indicated in 2).

   - It contains one DataObjectReference element for each data object file having been signed, one DataObjectReference element for each file enclosing CAdES signatures (one in this case), and one for each ASiCManifest already present within the ASiC-E container before requesting a new time-stamp token (one in the present case). Consequently, this second ASiCManifest file contains digest values of all the data object files already signed, all the already existing ASiCManifest files, and all the files enclosing CAdES signatures.

2) A new file enclosing a time-stamp token. The message imprint of this time-stamp token is the digest value of the second ASiCManifest file described in 1). The IETF RFC 3161 [i.52] time-stamp token is actually computed on digest values of each of the components present within the container, and consequently indirectly time-stamps them, including any CAdES signature, its corresponding ASiCManifest, and the data object files that it signs.

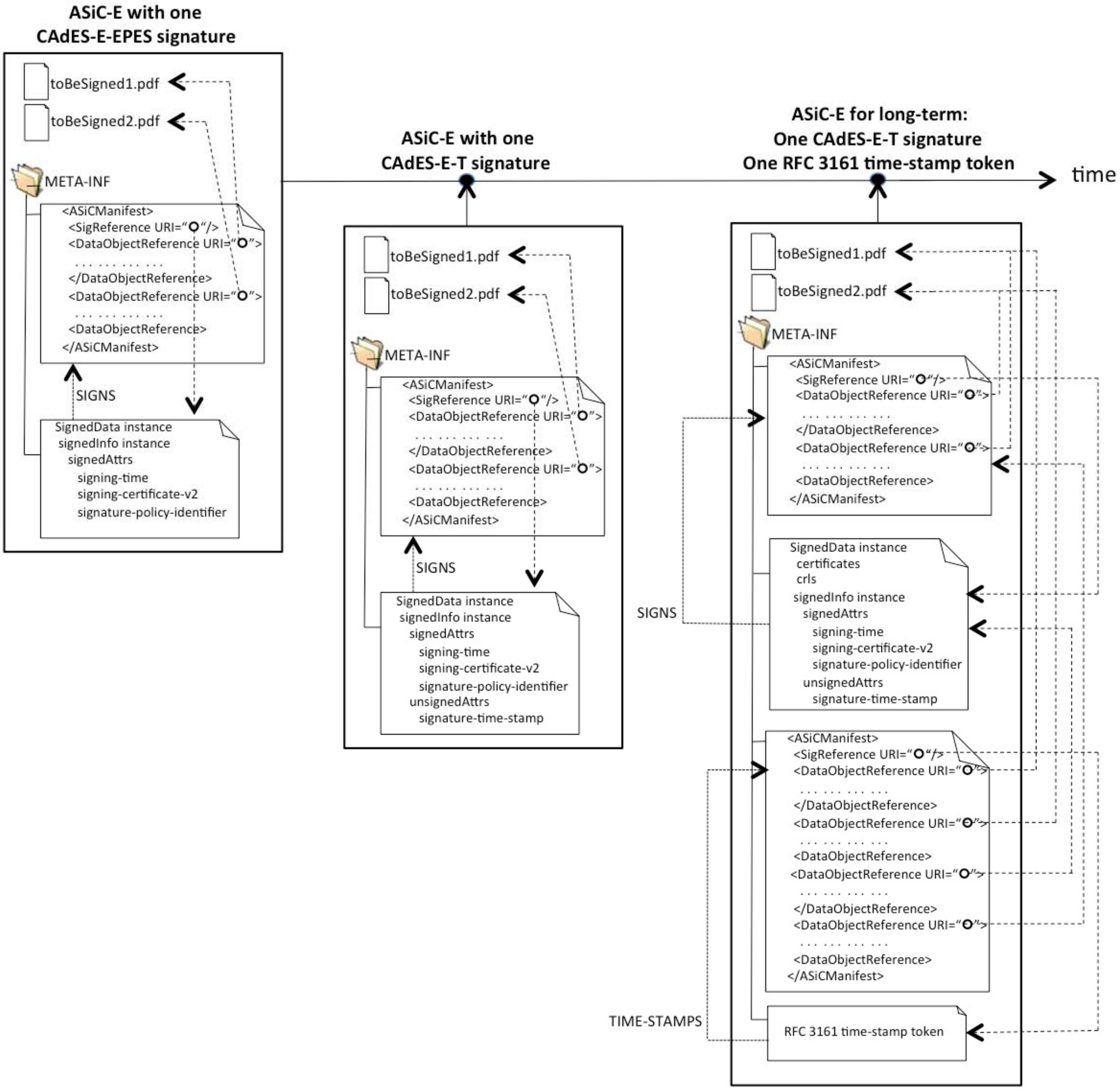


Figure 17: Transitions for an ASiC-E with one CAdES-E-EPES signature up to ASiC-E with availability and integrity of validation material

Figure 18 shows how an ASiC-E container with an augmented CAdES-E-T signature changes if the embedded CAdES signature is evolved to CAdES-E-C, and how the resulting ASiC-E container changes for dealing with availability and integrity of validation data. This last step requires again the incorporation of all the validation material within the XAdES signature, of the ASiCArchiveManifest file, and the file with the time-stamp token.

The direct transition from ASiC-E with CAdES-E-T to the ASiC-E for availability and integrity of validation data is also possible. The final ASiC-E container then contains a CAdES-E-T signature instead a CAdES-C as appears in the figure (the `complete-certificate-references` and `complete-revocation-references` attributes are not present).

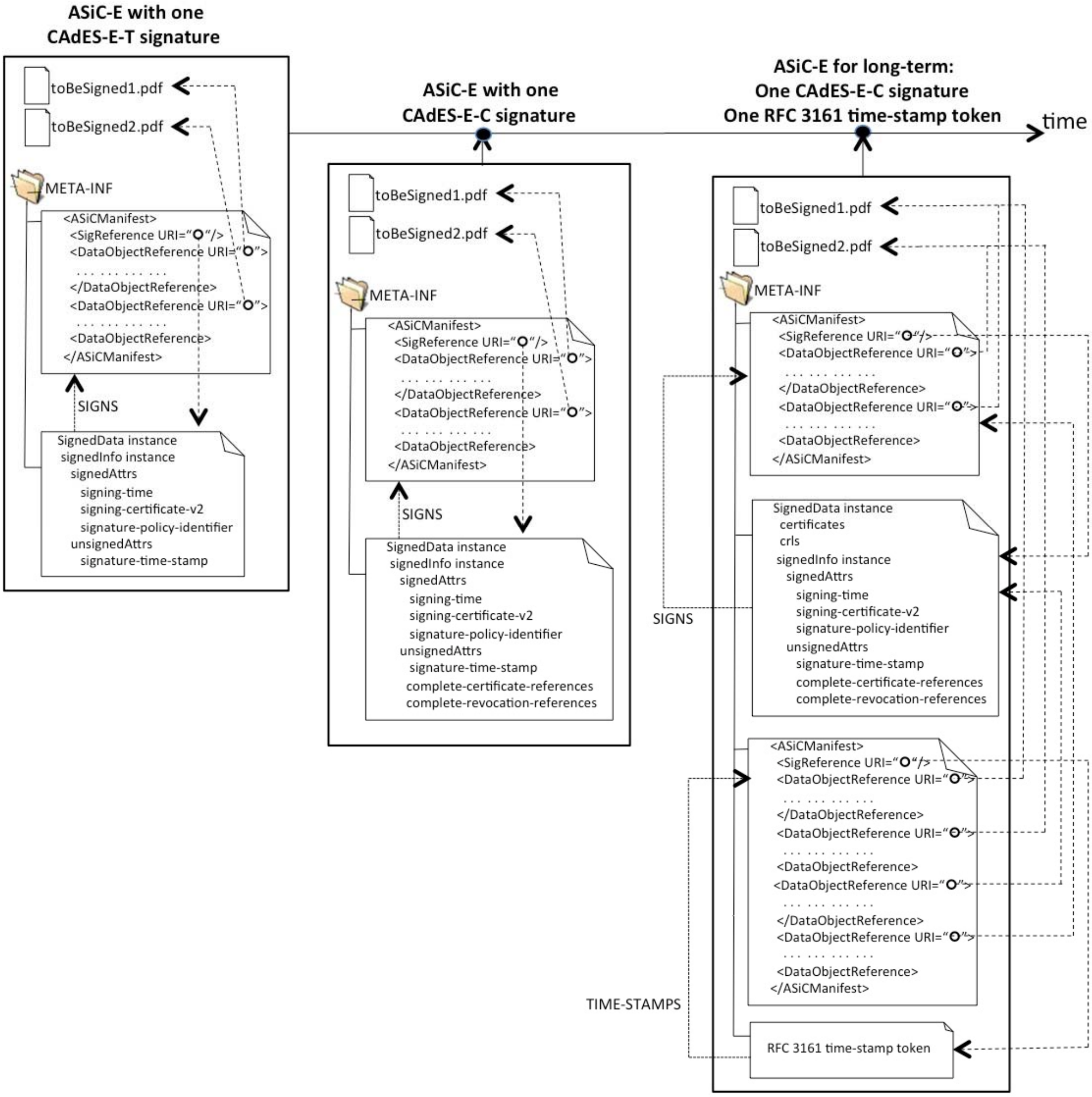

ASiC-E with one CAdES-E-T signature
toBeSigned1.pdf
toBeSigned2.pdf
META-INF
<ASiCManifest>
<SigReference URI="O"/>
<DataObjectReference URI="O">
</DataObjectReference>
<DataObjectReference URI="O">
<DataObjectReference>
</ASiCManifest>
SIGNS
SignedData instance
signedInfo instance
signedAttrs
signing-time
signing-certificate-v2
signature-policy-identifier
unsignedAttrs
signature-time-stamp
ASiC-E with one CAdES-E-C signature
complete-certificate-references
complete-revocation-references
ASiC-E for long-term: One CAdES-E-C signature One RFC 3161 time-stamp token
time
certificates
crls
TIME-STAMPS
RFC 3161 time-stamp token

Figure 18: Transitions for an ASiC-E with one CAdES-E-T signature up to ASiC-E with availability and integrity of validation material

Figure 19 shows how an ASiC-E container with an augmented CAdES-E-C signature changes if the embedded CAdES signature is evolved to CAdES-E-X-Long, and how the resulting ASiC-E container changes for dealing with availability and integrity of validation data. This last step requires again the incorporation of all the validation material within the CAdES signature, of the ASiCArchiveManifest file, and the file with the time-stamp token.

Figure 19 also shows that the direct transition from an ASiC-E with a CAdES-E-C signature to an ASiC-E container for availability and integrity of validation data, is also possible, by augmenting the CAdES signature with the validation data, and the incorporation of the IETF RFC 3161 [i.52] time-stamp token and the ASiCArchiveManifest file. The resulting package is similar to the one obtained in the previous path.

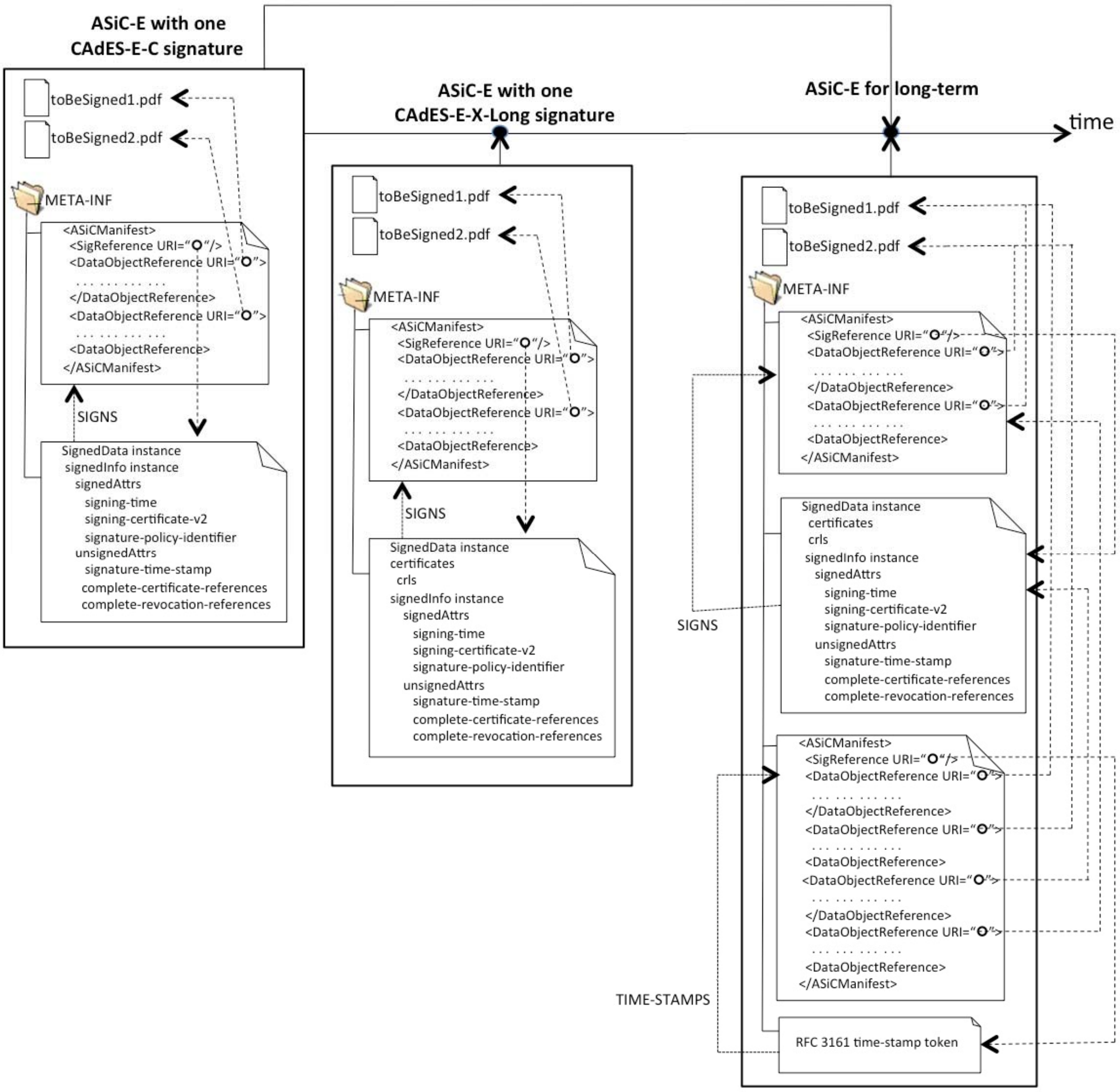


Figure 19: Transitions from an ASiC-E with one CAdES-E-C signature up to ASiC-E with availability and integrity of validation material, without time-stamp tokens on references to validation material

Figure 20 shows how an ASiC-E container with an augmented CAdES-E-C signature changes if the embedded CAdES signature is evolved to CAdES-E-X, and how the resulting ASiC-E container changes for dealing with availability and integrity of validation data. This last step requires again the incorporation of all the validation material within the CAdES signature, of the ASiCArchiveManifest file, and the file with the time-stamp token.

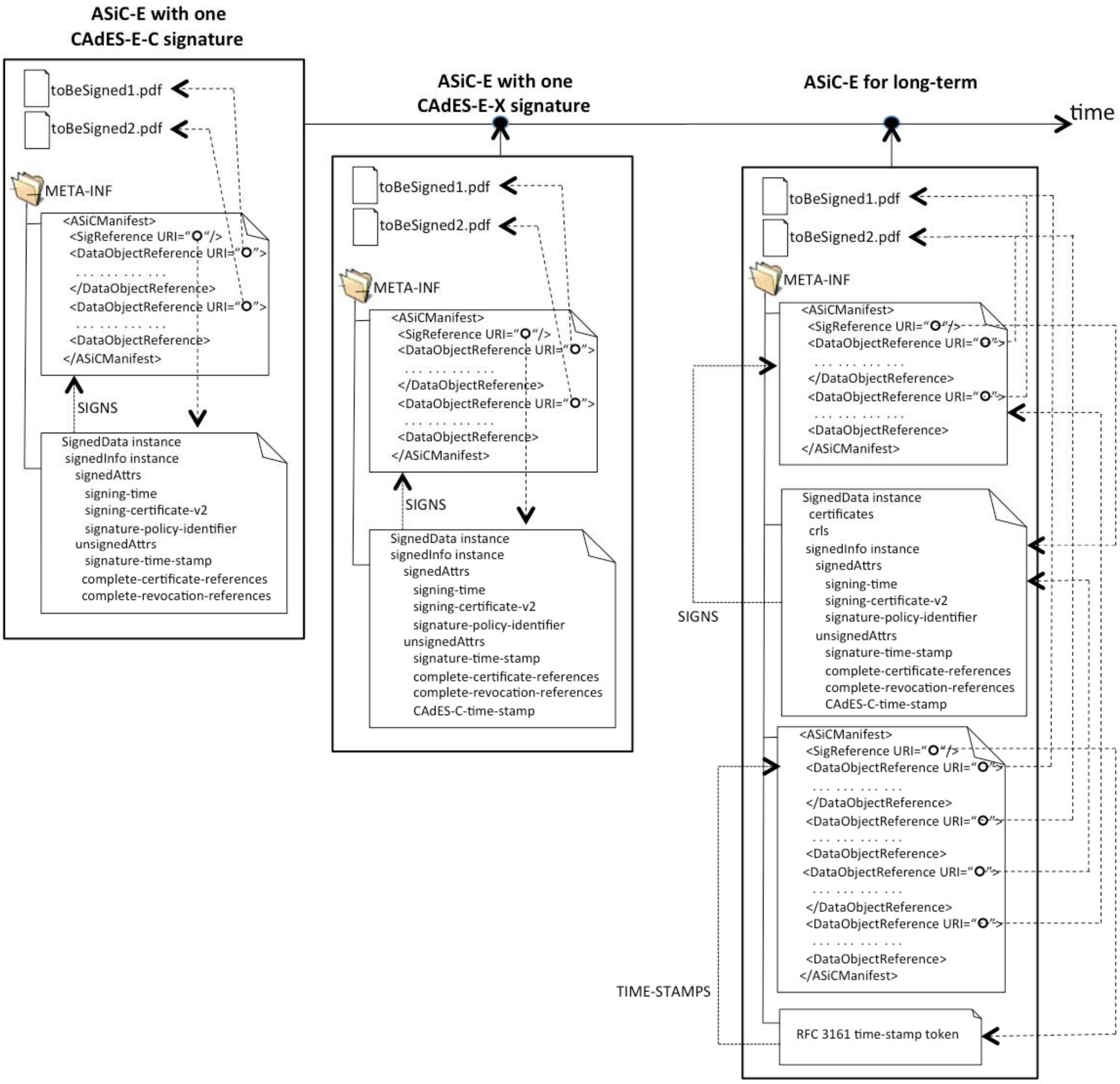


Figure 20: Transitions for an ASiC-E with one CAdES-E-C to an ASiC-E with one CAdES-E-X signature and to an ASiC-E with availability and integrity of validation material

Figure 21 shows how an ASiC-E container with an augmented CAdES-E-X signature changes if the embedded CAdES signature is evolved to CAdES-E-X-L, and how the resulting ASiC-E container changes for dealing with availability and integrity of validation data. This last step requires again the incorporation of all the validation material within the CAdES signature, of the ASiCArchiveManifest file, and the file with the time-stamp token.

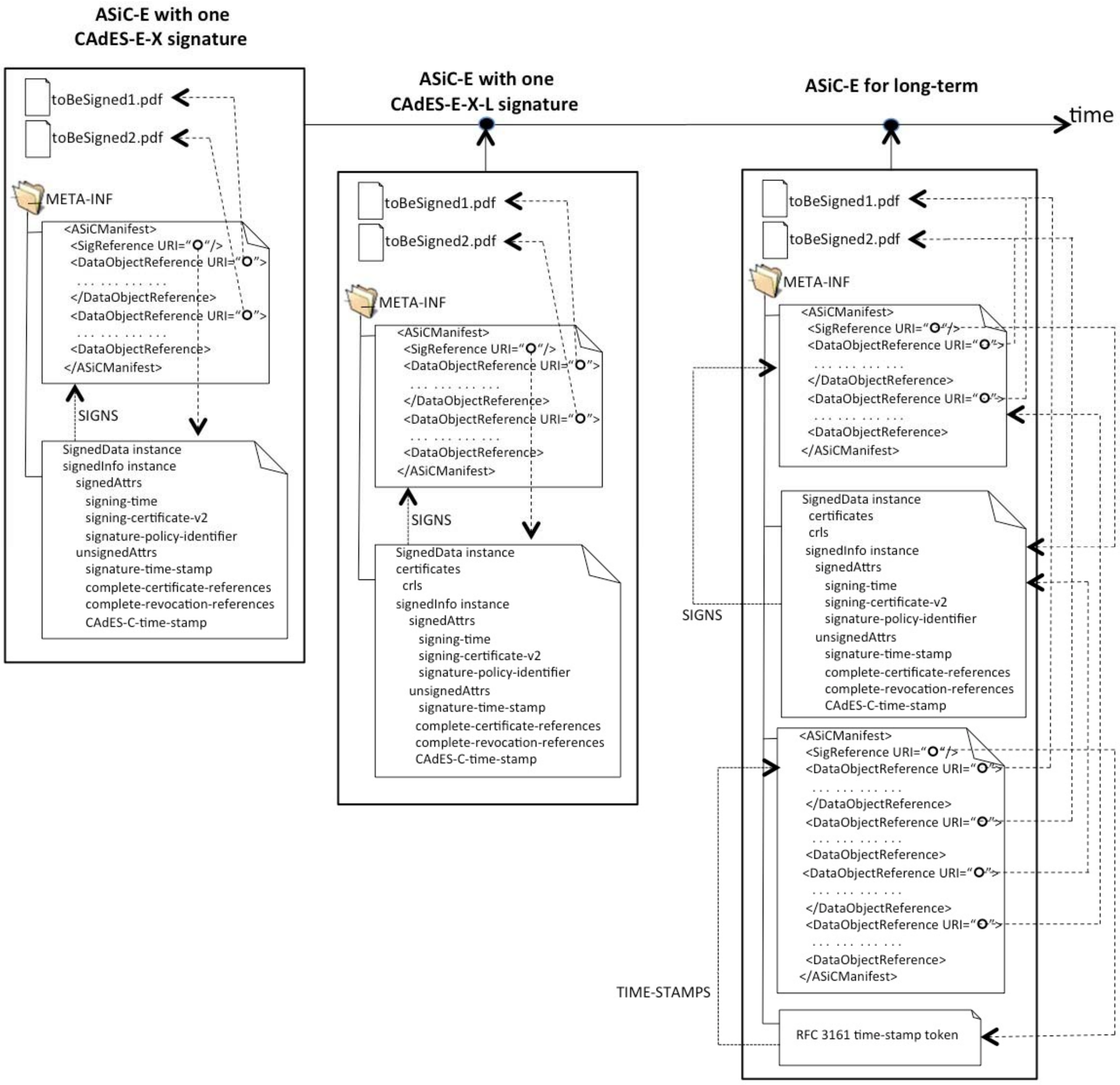


Figure 21: Transitions for an ASiC-E with one CAdES-E-X to an ASiC-E with one CAdES-E-X-L signature and to an ASiC-E with availability and integrity of validation material

8.12 Selecting proper Signature Creation Devices

It is out of the scope of the present document to provide guidance on devices for digital signature creation.

Instead, implementers should read CEN TR 419 200 [i.27]. This is another guidance document, which specifically addresses area 2 ("Signature Creation and Other Related Devices") of the Rationalized Framework [i.1].

Implementers will find in that document material that will guide them in the usage of the different types of documents within that area (Policy and Security Requirements, Technical Specifications, and Conformity Assessment) for selecting the signature creation device most suitable for the targeted business processes.

8.13 Selecting proper cryptographic suites

It is out of the scope of the present document to provide guidance on cryptographic suites.

Instead, implementers should read ETSI TR 119 300 [i.28]. This is another guidance document, which specifically addresses area 3 ("Cryptographic Suites") of the Rationalized Framework [i.1].

At the time of writing the present document, this area contains only two documents, namely: the aforementioned ETSI TR 119 300 [i.28], and ETSI TS 119 312 [i.29].

ETSI TS 119 312 [i.29] defines a number of different cryptographic suites for secure digital signatures. Implementers will find in ETSI TR 119 300 [i.28] material that will guide in the selection of cryptographic suites for the requirements identified within the targeted business processes.

8.14 Signature generation, augmentation and validation applications

8.14.1 Introduction

When dealing with the technicalities of implementing (or selecting) applications for generating, augmenting and/or validating digital signatures, implementers should carefully read the following documents present within area 1 of the Rationalized Framework [i.1].

1) CEN EN 419 111 [i.12]: "Protection Profiles for Signature Creation & Validation Applications".

2) ETSI EN 319 102-1 [i.10]: "Procedures for Signature Creation and Validation".

3) ETSI TS 119 101 [i.11]: "Security requirements for signature creation applications and signature validation applications".

Clauses 8.14.2, 8.14.3 and 8.14.4 provide details on these documents.

8.14.2 Selecting the suitable Protection Profile

CEN EN 419 111 ([i.12], [i.13], [i.14], [i.15] and [i.16]) is a multi-part document, which in its introduction defines the security requirements for Signature Creation and Signature Validation Applications.

Implementers of a Signature Creation Application should carefully read CEN EN 419 111-2 [i.13] that specifies the core protection profile for a signature creation validation (whose Target of Evaluation is software running on an operating system and a Signature Creation Platform hardware), and CEN EN 419 111-3 [i.14], which defines extensions to the core protection profile for a variety of situations.

Implementers of a Signature Validation Application should carefully read CEN EN 419 111-4 [i.15] that specifies the core protection profile for a signature validation application (whose Target of Evaluation is software running on an operating system and a Signature Validation Platform hardware), and CEN EN 419 111-5 [i.16], which defines extensions to the core protection profile for a variety of situations.

Implementers, after reading these documents should select the Protection Profile(s) that their tools should be compliant with for properly fulfilling the requirements imposed by the targeted business processes.

8.14.3 Implementing the signature generation and augmentation processes

With regards to the process of generating and augmenting a digital signature, ETSI EN 319 102-1 [i.10] specifies procedures for creating and augmenting digital signatures standardized by ETSI in a format-agnostic way. It introduces general principles, objects and functions relevant when creating and augmenting signatures. It also defines the general classes of digital signatures mentioned in clause 8.11.6.2 of the present document with increasing longevity. It is based on the use of public key cryptography to produce such signatures, which are supported by public key certificates. ETSI TS 119 101 [i.11] provides security requirements for applications for signature generation applications.

Implementers will find within ETSI EN 319 102-1 [i.10] a model for the signature creation environment, which includes the signature creation system (SCS hereinafter), formed by the SCA and the SCDev. They will also find an information model for signature creation. Implementers should ensure that their implementations actually provide the functionality specified as mandatory within this document. However, the distribution of such functionality can be done among a set of components that is different from the set identified within ETSI EN 319 102 [i.10].

Implementers will find in ETSI TS 119 101 [i.11] security requirements for SCA, including, among others, requirements on data content type, on the creation attributes/properties to be incorporated to the signature, on timing and sequence, on signature invocation, on signer's authentication, on preparation of the data to be signed (DTBS) and its representation (DTBSR), on the SCDev, and on the SCDev/SCA interface.

8.14.4 Implementing the signature validation process

With regards to the process of validating a digital signature, ETSI EN 319 102-1 [i.10] specifies procedures for establishing whether a digital signature standardized by ETSI is technically valid and is the reference for implementing a Signature Validation Application (SVA). ETSI TS 119 101 [i.11] provides security requirements for signature validation applications.

ETSI EN 319 102-1 [i.10] defines an algorithm to validate digital signatures, with special consideration on signature validation of digital signatures where certificates may have expired or been revoked or even the usage period of algorithms have been exceeded. The algorithm takes advantage of security measures that have been applied (using the augmentation techniques mentioned in the present document) by the different entities that act on the signatures during their lifecycle (e.g. signer or previous verifiers that can have augmented the initial signatures) and ensures that such signatures still can be validated. Although the process is presented as an algorithm, implementers are not supposed nor recommended to implement it as described. However, any implementation claiming conformance will provide the same results as the algorithm would provide.

ETSI EN 319 102-1 [i.10] contextualizes the operation of a SVA as follows:

1) The SVA is called by the Driving Application (DA), to which it has to return the results of the validation process, in the form of a validation report. This validation report will be standardized in ETSI EN 319 102-2 [i.10], which at the time the present document is written has not been yet produced. ETSI EN 319 102-1 [i.10] specifies a minimum set of pieces of information to be included within this report, including the overall result, which can be TOTAL-PASSED, TOTAL-FAILED and INDETERMINATE. INDETERMINATE means that the results of the performed checks do not allow to ascertain the signature to be TOTAL-PASSED or TOTAL-FAILED, and also that this capability of ascertaining the signature to be TOTAL-PASSED or TOTAL-FAILED might change or not, depending of the cause of the INDETERMINATE result, as some of these causes would disappear if the validation application could gain access to certain additional information; under these circumstances a new validation would result either in TOTAL-PASSED or TOTAL-FAILED.

2) The algorithm takes as inputs the digital signature to be validated and a set of constraints coming from different sources whose fulfilment the SVA ascertains during the validation process. A constraint, according to that document, is any abstract formulation of rules, ranges and computation results whose fulfilment is assessed during the validation of the signature. These validation constraints can be defined in different ways:

   - Using formal policy specifications. An example of such situations is signature policy files containing the signature policy validation expressed in ASN.1 or XML syntaxes as specified in ETSI TS 119 172-2 [i.18] and ETSI TS 119 172-3 [i.19], which at the moment of writing the present document were not yet produced.

   - Defined explicitly in system specific control data: e.g. in conventional configuration-files like property or in-files or stored in a registry or database. Or

   - Implicitly by the implementation itself.

   Additionally, the DA can provide constraints to the SVA via parameters implied by the application or the user.

3) Finally, ETSI TS 119 102 [i.10] proposes the contents of the validation report (although without proposing any specific format). This report contains:

   - a result code, indicating the major result of the validation procedure (VALID, INVALID, INDETERMINATE);

- a result sub-code, indicating the reasons for the major result; and
- a set of associated validation report data, specific for each sub-code.

The algorithm specified by ETSI EN 319 102 [i.10]:

1) Identifies basic building blocks in charge of:

    - Identifying the signer's certificate.
    - Initializing the validation context, i.e. initializing the validation constraints and parameters to be used during the validation process.
    - Validating X.509 certificate. The process defined for this block builds on the Certification Path Validation, as specified in IETF RFC 5280 [i.35].
    - Cryptographically verifying digital signature.
    - Validating the acceptance of the signature, i.e. performing any additional required validation on the attributes (properties) of the signature.

    As stated before, the validation process is presented as an algorithm that suitably makes use of the aforementioned building blocks.

2) Defines the steps required for performing the so-called Basic Validation, i.e. the process required for performing a short-term signature validation, adequate for basic signatures (like the ones within CRLs, OCSP responses, etc.), which as mentioned above include B-B level in baseline signatures, and also E-BES, and E-EPES levels.

3) Defines the steps required for performing the Validation of time-stamp tokens, which builds on the aforementioned Basic Validation algorithm by adding an additional step of data extraction, consisting in returning relevant data items from the time-stamp token itself (like the generation time, the message imprint, etc.), which can be used in the process of validating higher levels of ETSI digital signatures, where these time-stamp tokens are present.

4) Defines the steps required for performing the validation of signatures with trusted time indication, i.e. E-T forms, which builds on the Basic Validation and the Validation of time-stamp tokens, and the validation with signatures for long term availability of validation material, adequate for validating (C/X)AdES-E-C, (C/X)AdES-E-X, (C/X)AdES-E-XL, (C/X)AdES-E-X-Long, and a subset of PAdES-LTV signatures.

5) Defines the steps required for performing the Validation of signatures for long term availability and integrity validation data, adequate for validating (C/X)AdES-E-A and PAdES-LTV that incorporate DocumentTimeStamp dictionary(ies) time-stamping already present signatures. The algorithms are built on the concept of Proof Of Existence (POE) and a set of additional building blocks, listed below:

    - Proof Of Existence (POE) of an object, is an evidence that proves that this object (a certificate, a CRL, signature value, hash value, etc.) existed at a specific date/time in the past. There are several ways of generating such a type of POEs: time-stamping an object in certain time provides a POE of that object time afterwards; but also electronic notaries, archival services or other services can provide this type of POEs.
    - Past Certificate Validation process. This is a process that validates a certificate at a date/time that can be in the past. This can be needed in the verification of a long-lived signature, which can include expired certificates for instance.
    - POE extraction, a process that derives POEs from a given time-stamp token within the digital signature.
    - X.509 Certificate path validation constraints, Additional Chain Constraints, Additional Revocation Constraints, Additional Time-Stamp Trust Constraints, Constraints on X.509 Certificate meta-data, and Cryptographic Constraints.

9 Signature creation and validation catalysing toolkit

9.1 Introduction

Implementers should also be aware of the existence of a holistic toolkit that they can use for assessing the conformance of their implementations to referenced standards. This toolkit aims at further supporting and accelerating of the deployment of interoperable digital signatures across Europe.

Clauses 9.2, 9.3 and 9.4 provide an overview of the elements that integrate the package.

9.2 Technical Specifications

The first element of the aforementioned toolkit is a set of ETSI Technical Specifications for testing conformance and interoperability of applications with regards to the implementation of signature formats and of signature policies as listed below:

1) ETSI TS 119 124 [i.22]: "CAdES Testing Conformance and Interoperability".

2) ETSI TS 119 134 [i.23]: "XAdES Testing Conformance and Interoperability".

3) ETSI TS 119 144 [i.24]: "PAdES Testing Conformance and Interoperability".

4) ETSI TS 119 164 [i.25]: "ASiC Testing Conformance and Interoperability".

ETSI TS 119 124 [i.22], ETSI TS 119 134 [i.23], ETSI TS 119 144 [i.24] and ETSI TS 119 164 [i.25] address each of the ETSI digital signature formats and the ASiC package. All of them have 4 parts. In all of them, implementers will find the following contents:

1) Parts 1 provide an overview and the structure of the multi-part document.

2) Parts 2 and 3 specify test suites for testing interoperability. They include test cases aiming at ascertaining that different implementations generating and validating digital signatures standardized by ETSI and ASiC containers are able to interoperate, i.e. that the signatures/containers generated by one implementation are properly validated by the others. The test suites defined within these documents address those aspects that have relevance for achieving interoperability. They also include different types of test cases:

   - Positive cross-validation test cases. These test cases require to an implementation to generate a valid CAdES, PAdES, or XADES digital signature or ASiC container according to a detailed specification of its contents. Other implementations aiming at testing interoperability with the first one should try to validate this signature/container. A VALID result means that implementations successfully interoperate with regarding to the aspects tested.

   - Positive cross-validation, augmentation and arbitration test cases. These test cases require the participation of at least 3 different implementations and works as follows: implementation A generates a valid CAdES, PAdES, or XAdES signature or ASiC container according to a detailed specification of its contents. Implementation B, acting as relying party, validates this signature and augments it to a more evolved level, also according to the specifications of the test case. Finally, a third implementation C, acting as a purported arbitrator, validates the augmented signature. These test cases serve for testing how implementations behave in situations where signatures are augmented and these augmented signatures are in turn validated by entities that are neither the signer, nor the one that firstly validated the signature and after augmented it.

   - Negative test cases. These test cases specify signatures for which the validation process cannot end with the VALID result, according to ETSI EN 319 102 [i.10]. They aim at ascertaining that implementations actually correctly deal with signatures or containers that cannot be considered as technically valid due to a number of reasons, and in consequence, do not generate false positive results.

   These test suites are built taking into account not only the specifications on the formats, but also on the signature validation process specified within ETSI EN 319 102 [i.10]. This, among other things, require the presence of different PKIs of different degree of complexity, ranging from a very simple one (where all the certificates, certificate status data, and time-stamps appertain to the same hierarchy of CAs), to complex combinations of PKIs that try to be close to real situations.

For all the formats, parts 2 specify test suites for testing interoperability on baseline signatures/containers, while parts 3 of the document specify test suites for the corresponding extended/additional signatures/extended containers.

3) Parts 4 and 5 define complete sets of test assertions that aim at ascertaining each and every of the requirements specified by CAdES, PAdES, XAdES and ASiC. In consequence, if a CAdES, PAdES, XAdES signature or an ASiC container passes all the assertions specified within Part 4 it can be claimed that the signatures are baseline signatures compliant with ETSI EN 319 122-1 [i.2], ETSI EN 319 132-1 [i.4], or ETSI EN 319 142-1 [i.6], and the containers are baseline containers compliant with ETSI EN 319 162-1 [i.8]. Similarly, if it passes all the assertions specified within Part 5, it can be claimed that a signature is an CAdES or XAdES extended signature compliant with 319 122-2 [i.2], or ETSI EN 319 132-2 [i.5] respectively, or a PAdES signature compliant with ETSI EN 319 142-2 [i.7], or an ASiC additional container compliant with ETSI EN 319 162-2 [i.9], respectively.

CEN EN 419 103 [i.21] specifies general requirements for testing interoperability and conformance conformance of signature creation and applications.

9.3 Conformance testing software tools

The second element of the catalysing toolkit is a set of software tools, freely available, that test conformance of CAdES, PAdES, XAdES signatures, and ASiC containers against their corresponding core and baseline and extended/additional specifications.

In its definitive version, each software tool performs the whole set of test assertions specified in the corresponding part of ETSI TS 119 124 [i.22], ETSI TS 119 134 [i.23], ETSI TS 119 144 [i.24] and ETSI TS 119 164 [i.25]. The output of the tools does not only provide details on each assertion tested and its corresponding result, but also on the different components of the signature/container, focussing specifically on certificates and time-stamp tokens. Additionally, they provide useful trace information on computations that experience has proved to be source of interoperability problems: they provide, for instance, the trace of the contributions that have to be made for building the input to the computation of the message imprints for the different time-stamp tokens types that appear within a signature. This has proved to be of great usefulness for implementers, as helps them to identify within their applications the sources of specific problems when dealing with such computations, and facilitates a unified reading and understanding of the corresponding specification.

These tools are freely available through the ETSI Signature Conformance checkers webpage (http://signatures-conformance-checker.etsi.org/pub/index.shtml).

9.4 Interoperability test events

The third element of the catalysing toolkit is the ETSI CTI Portal for Digital Signatures. This is an online portal that provides full support to the conduction of remote interoperability test events on signature creation and validation. Using the facilities provided by this portal, the participants in the event do not need to travel to a certain place and meet face to face for a certain number of days, devoting all the working hours to actually perform interoperability tests. Instead, they can organize their time in their own premises, working asynchronously, and meeting remotely at specific dates and times while the event is alive (the experience proves that a duration of 3 weeks is suitable for this kind of events). The portal contains all the information that the participants require for conducting their tests, namely:

1) The interoperability test suites. Participants find at the portal a complete and detailed specification of each test case.

2) Repository of signatures generated by each participant, suitably structured.

3) Repository of validation reports coming from each participant, suitably structured.

4) Global interoperability matrix, automatically updated each time that a participant uploads a new validation report at the portal.

5) Per participant interoperability matrixes, which reports to each participant the results obtained by the others after they have tried to validate each of her signatures.

6) Documentation explaining how to conduct while participating in the events, i.e. the steps to be performed by each participant, and how they have to interact with the portal for uploading signatures/containers/reports and downloading other participants' signatures/containers.

7) The conformance testing tools described above, allowing them to not only test interoperability with other implementations but also test conformance of their own tools against the corresponding specification.

The experience proves that implementers find at this kind of events a place where:

1) To ascertain the conformance of their own tools against the reference specification.

2) To ascertain the degree of interoperability of their tools with other tools in the market.

3) To identify conformance and/or interoperability problems within their own tools.

4) To discuss with other relevant players in the field about specific issues within the standards. This includes:

   - Identify errors within the standards, discuss potential solutions and recommend one of them to the standardization body in charge of the specification.

   - Identify ambiguities within the standard that lead to different interpretations (and in consequence, to lack of interoperability), build consensus on a unique interpretation, and raise recommendations for fixing them to the standardization body in charge of the specification.

   - Discuss with other participants about what would be suitable in a potential evolution of the standard (e.g. addition of new functionality), and raise the corresponding request to the standardization body in charge of the specification.

10 Evaluation processes

While implementing a signature creation, augmentation and/or validation application, implementers should be aware that very likely they can be requested that they pass an evaluation process that ensures that the application:

1) Generates signatures compliant with the selected formats, forms and levels.

2) Complies with the requirements defined within ETSI EN 319 102 [i.10] with regards to the procedures for generating, augmenting, and/or validating digital signatures.

3) Is compliant with the selected Protection Profiles.

4) Is compliant, along with the environment where it is used, with the policy requirements specified within ETSI TS 119 101 [i.11].

Implementers are suggested to read CEN EN 419 103 [i.21] for a deep understanding of the evaluation processes their applications can need to face.

11 Corollary: the process within the context of the Standardization Framework

As a corollary of this guide, this clause summarizes the existing relationships between each of the phases within the proposed process for implementing digital signatures in electronic business and the existing documents within the area 1 (Signature Creation and Validation) of Standardization Framework.

Figure 22 graphically shows these relationships.

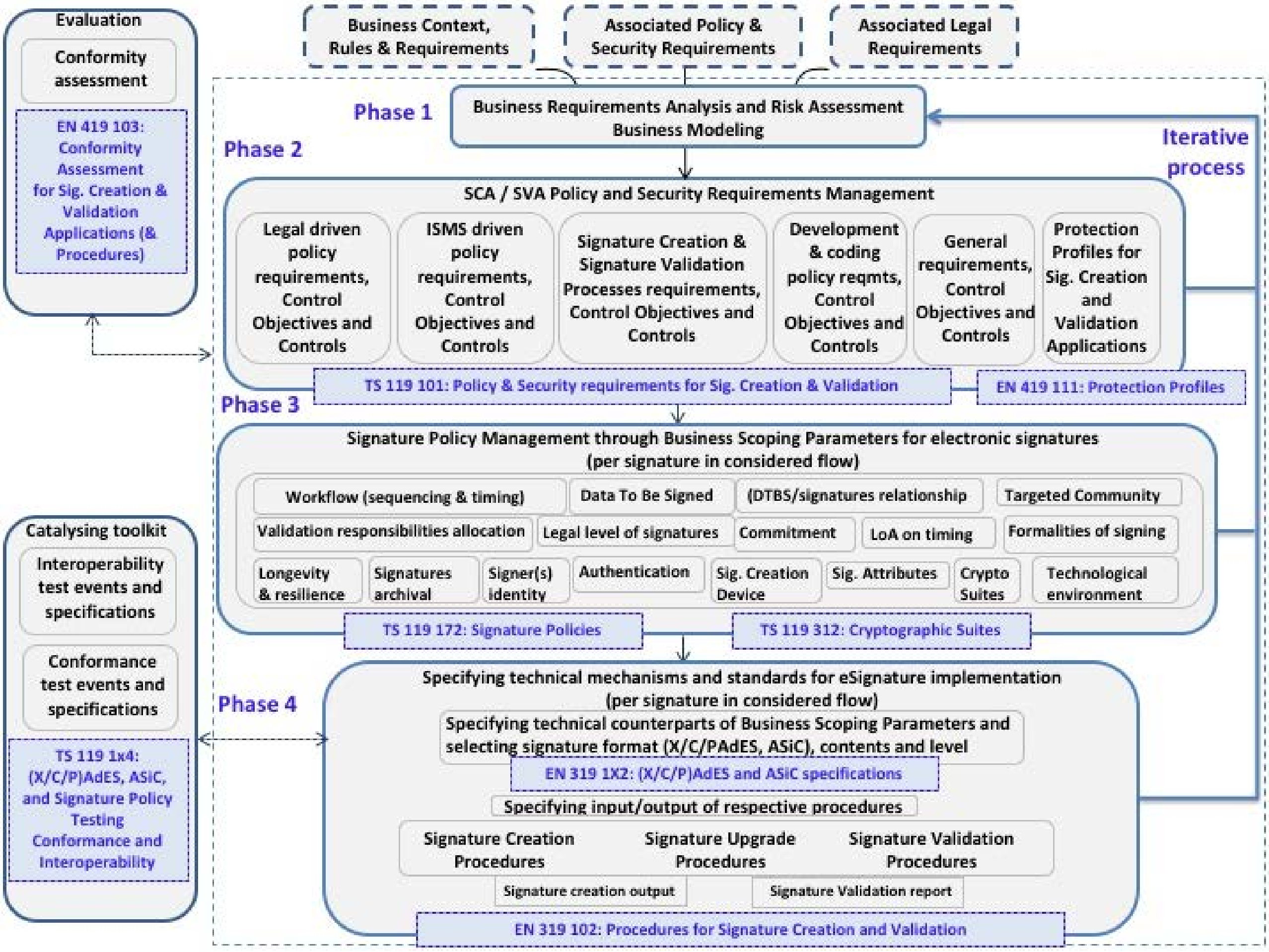

Evaluation
Conformity assessment
EN 419 103: Conformity Assessment for Sig. Creation & Validation Applications (& Procedures)
Business Context, Rules & Requirements
Associated Policy & Security Requirements
Associated Legal Requirements
Phase 1
Business Requirements Analysis and Risk Assessment Business Modeling
Iterative process
Phase 2
SCA / SVA Policy and Security Requirements Management
Legal driven policy requirements, Control Objectives and Controls
ISMS driven policy requirements, Control Objectives and Controls
Signature Creation & Signature Validation Processes requirements, Control Objectives and Controls
Development & coding policy reqmts, Control Objectives and Controls
General requirements, Control Objectives and Controls
Protection Profiles for Sig. Creation and Validation Applications
TS 119 101: Policy & Security requirements for Sig. Creation & Validation
EN 419 111: Protection Profiles
Phase 3
Signature Policy Management through Business Scoping Parameters for electronic signatures (per signature in considered flow)
Workflow (sequencing & timing)
Data To Be Signed
(DTBS/signatures relationship
Targeted Community
Validation responsibilities allocation
Legal level of signatures
Commitment
LoA on timing
Formalities of signing
Longevity & resilience
Signatures archival
Signer(s) identity
Authentication
Sig. Creation Device
Sig. Attributes
Crypto Suites
Technological environment
TS 119 172: Signature Policies
TS 119 312: Cryptographic Suites
Catalysing toolkit
Interoperability test events and specifications
Conformance test events and specifications
TS 119 1x4: (X/C/P)AdES, ASiC, and Signature Policy Testing Conformance and Interoperability
Phase 4
Specifying technical mechanisms and standards for eSignature implementation (per signature in considered flow)
Specifying technical counterparts of Business Scoping Parameters and selecting signature format (X/C/PAdES, ASiC), contents and level
EN 319 1X2: (X/C/P)AdES and ASiC specifications
Specifying input/output of respective procedures
Signature Creation Procedures
Signature Upgrade Procedures
Signature Validation Procedures
Signature creation output
Signature Validation report
EN 319 102: Procedures for Signature Creation and Validation

Figure 22: Relationship between process' tasks and documents within the area 1 of the Rationalized Framework

Annex A: Securing signed detached objects in XAdES signatures in the long term

A.1 Introduction

XML signatures and XAdES signatures allow signing objects in two ways, namely: using `ds:Reference` children of `ds:SignedInfo` element, and using `ds:Reference` children of signed `ds:Manifest` elements.

XAdES signatures specified in ETSI EN 319 132-1 [i.4] and ETSI EN 319 132-2 [i.5] can secure signed detached objects in the long term regardless the way they have been signed.

The present annex describes the techniques used for securing signed detached objects in the long term.

A.2 Securing detached objects signed with `ds:SignedInfo`

ETSI EN 319 132-1 [i.4] requires that when an application wants to augment a XAdES signature by incorporating one `xadesv141:ArchiveTimeStamp`, the input to the message imprint computation concatenates, among others, the result of processing each `ds:Reference` child within `ds:SignedInfo` as specified in the reference processing model in XMLDSIG [i.37], clause 4.4.3.2. This model obliges to retrieve the detached data object pointed by the `URI` attribute of the `ds:Reference` element and apply to it the transforms indicated within this element.

If the digest algorithm (for instance Dig1) used in one of these `ds:Reference` for computing the digest value of a detached object is known to be suffering problems in a near future, a new `xadesv141:ArchiveTimeStamp` can be generated with a different digest algorithm (for instance Dig2) that is not weak. If the weak algorithm (Dig1) is eventually broken, and the original signed detached object is replaced by a fake detached object whose digest value according to Dig1 algorithm is the same as the digest value of the original one computed with Dig1 algorithm, when checking the message imprint of the last `xadesv141:ArchiveTimeStamp` the fake detached object would be retrieved, and would contribute to the message imprint computation input. The digest with Dig2 algorithm would result in an error, which is the expected behaviour.

Consequently, incorporation of `xadesv141:ArchiveTimeStamp` ensures that detached objects, signed by `ds:Reference` children within `ds:SignedInfo` are secured even if the digest algorithm within the `ds:Reference` element is broken.

A.3 Detached objects signed with signed `ds:Manifest`

A.3.1 The initial situation

Figure A.1 shows a XAdES signature whose `ds:SignedInfo` signs the XAdES signed properties and one `ds:Manifest` element. The `ds:Reference` children of `ds:Manifest`, refer to two detached objects, which means that the XAdES signature signs these two detached objects through a signed `ds:Manifest`.

Under these circumstances, if time after the signature generation, a `xadesv141:ArchiveTimeStamp` is requested, the message imprint computation input would be built as follows:

```
IN_MI = CanonIfReq(REF^-1 (SignedInfo.Reference[1]))    // Canon. SignedProperties el.
    | CanonIfReq(REF^-1 (SignedInfo.Reference[2]))    // Canonicalized Manifest element
    | Canon(SignedInfo)
    | Canon(SignatureValue) | Canon(KeyInfo)
    | Canon(UnsignedSignatureProperties[1])
    | Canon(UnsignedSignatureProperties[2]) |…
    | Canon(UnsignedSignatureProperties[lastExistingAtTimeStampingTime])
    | Canon(Object[2])                         // Canonicalized Object containing Manifest
```

The expression above uses the following notation:

REF^{-1}(SignedInfo.Reference[1]) stands for the process of completely processing the `ds:Reference` element between round brackets (the first `ds:Reference` child found within `ds:SignedInfo` in this case) as specified by XMLDSIG [i.37], clause 4.4.3.2. This process includes retrieval of the data object pointed by the URI, and the application of any transformation indicated within the optional `ds:Transforms` element.

Canon() means: "Canonicalization result of what appears between round brackets".

CanonIfReq() means: "Canonicalization result of what appears between round brackets, If Required". In the expression above this is applied to a REF^{-1}() for indicating that if the result of processing a `ds:Reference` is an octet stream, then there is no need to apply canonicalization; however, if this result is a XML node set, then a canonicalization will be applied for getting an octet stream.

UnsignedSignatureProperties [lastExistingAtTimeStampingTime] means "the last child of `xades:UnsignedSignatureProperties` found when the archive time-stamp is going to be requested.

It can be seen that in this case, **the signed detached data objects themselves do not contribute to the message imprint computation input of the time-stamp token. Instead their digest values, as present within the signed ds:Manifest contribute to the message imprint computation input.**

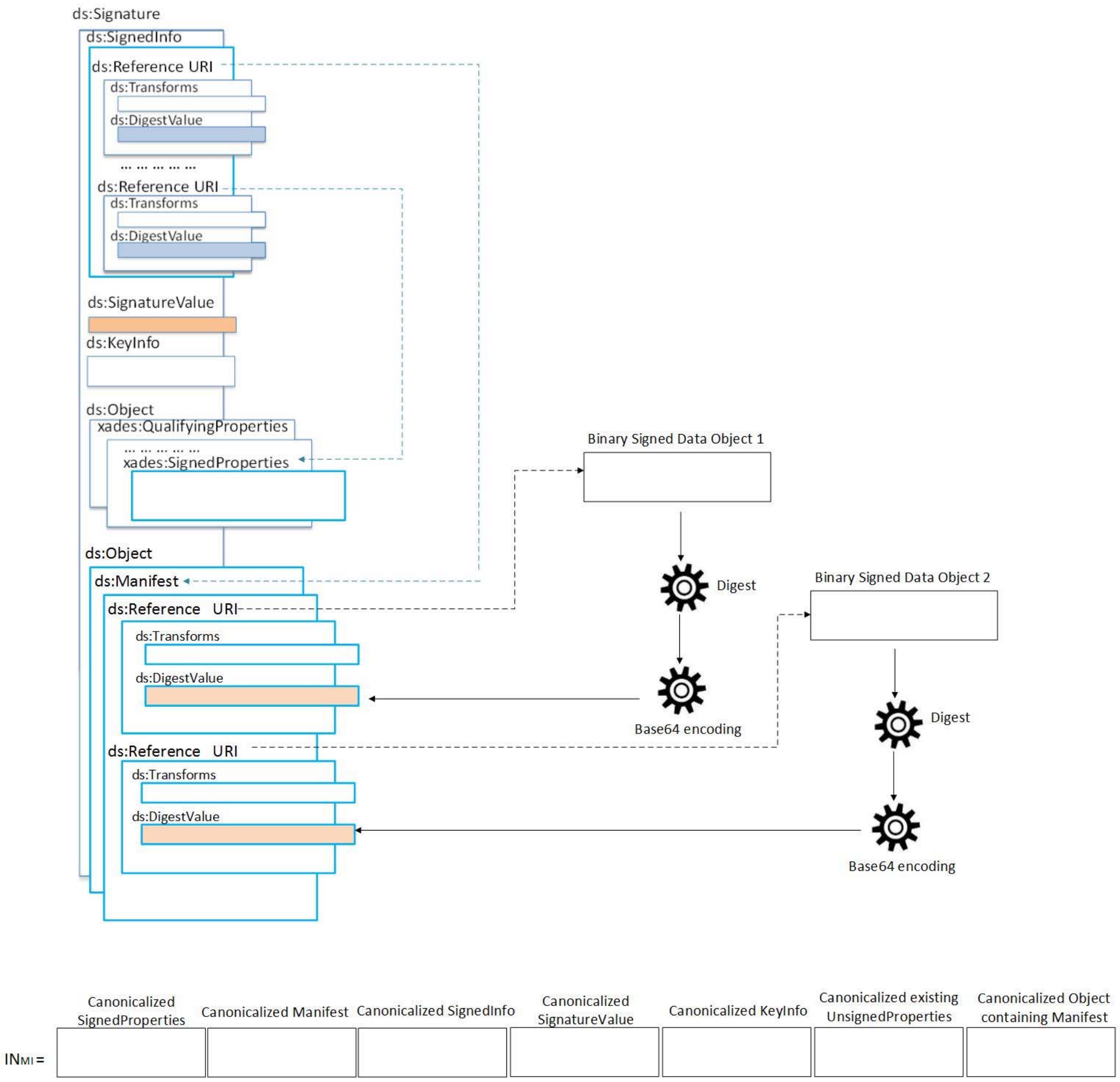
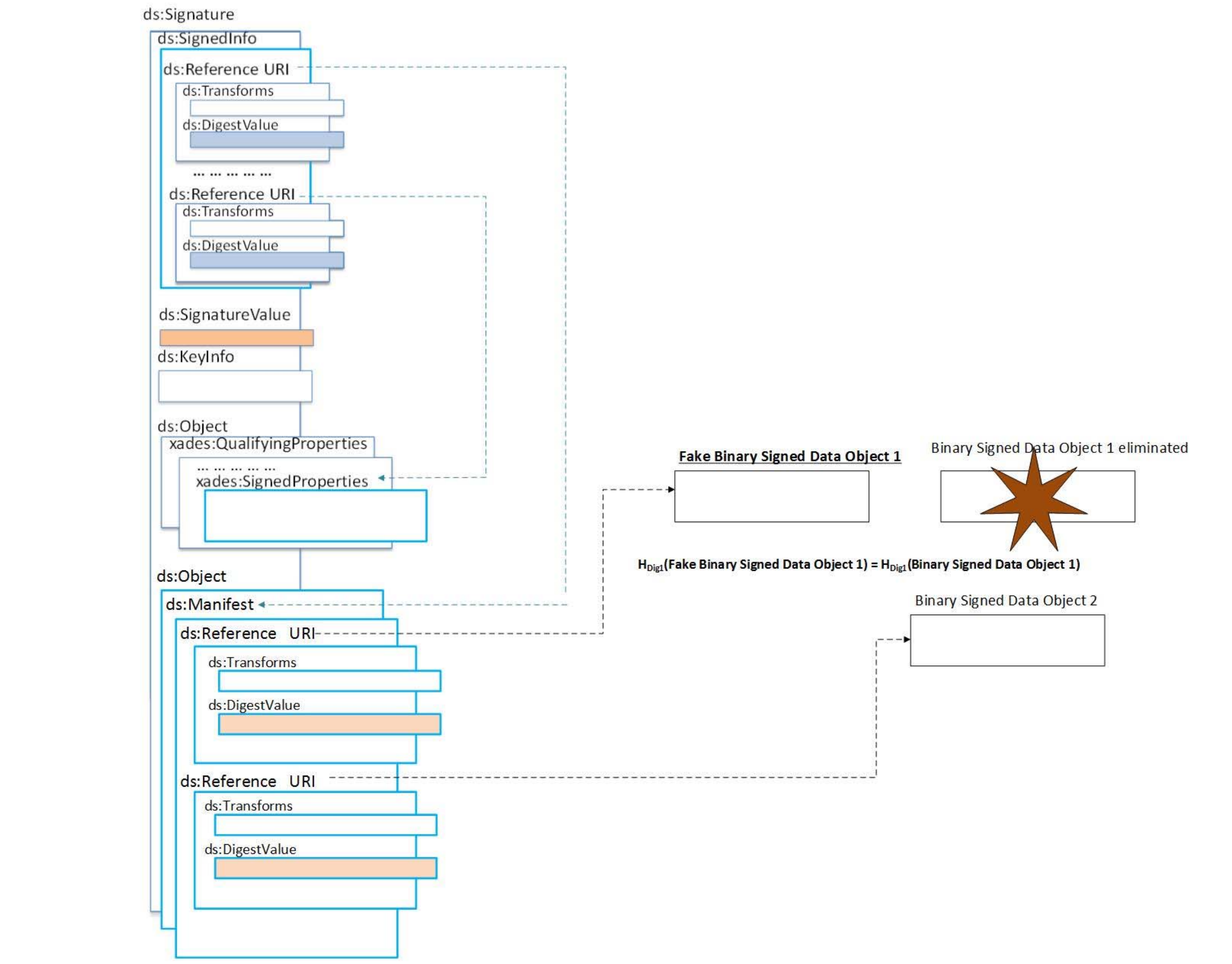


Figure A.1: A XAdES signature signing detached objects using signed ds:Manifest

A.3.2 The problem: detached data objects signed through ds:Manifest and their resilience to digest algorithm break

Figure A.2 shows a potential attack to signatures built as shown in Figure A.1. Such an attack would remain unnoticed if only `xadesv141:ArchiveTimeStamp` qualifying property is used for augmenting the signature.



INMI computed by party that validates the signature

	Canonicalized SignedProperties	Canonicalized Manifest	Canonicalized SignedInfo	Canonicalized SignatureValue	Canonicalized KeyInfo	Canonicalized existing UnsignedProperties	Canonicalized Object containing Manifest
INMI =							

The validation of the signature would actually succeed, as all the contributions to the message imprint computation remain unchanged. However, one of the remote signed data objects has been changed!!. The algorithm for building the message imprint computation of xadesv141: ArchiveTimeStamp does not counter the threat of a break in the digest algorithm used for remote data object indirectly signed by a ds:Manifest, as the original document may be changed by another one whose digest is the same as the original

Figure A.2: A potential attack when the digest algorithm is broken: substitution of one of the external indirectly signed data objects

It could happen that the digest algorithm (Dig1) used for computing one of the signed data objects (let us say Signed Data Object 1), becomes insecure, and that somebody is able to find a Fake Signed Data Object such as:

- H_{Dig1}(Signed Data Object 1) = H_{Dig1} (Fake Signed Data Object 1)

Where H_{Dig1}(O) stands for compute the hash of the data object O using algorithm Dig1.

It could also happen that somebody gains access to the repository containing the Signed Data Object 1 and replaces it with Fake Signed Data Object 1.

This would lead to the situation shown by Figure A.2. The first `ds:Reference` within `ds:Manifest` is now referencing the Fake Binary Signed Data Object 1. A relying party validating the XAdES-A signature at an instant after the Fake Binary Signed Data Object 1 replaced the Signed Data Object 1, **would fail in noticing such a replacement.**

Under these circumstances, the verification of the value of ds:SignatureValue would succeed, as none of the contents referenced by the contents of ds:SignedInfo (namely the SignedProperties and the ds:Manifest element) have changed. Additionally, the validation of the xadesv141:ArchiveTimeStamp would also be successful, as the same message imprint computation input would be built by the relying party, because NONE of the values of the indirectly signed objects through a signed ds:Manifest has contributed to build the message imprint computation. The relying party would build the following input to the message imprint of the xadesv141:ArchiveTimeStamp:

```
INMI = CanonIfReq(REF-1 (SignedInfo.Reference[1]))  // Canon. SignedProperties el.
    | CanonIfReq(REF-1 (SignedInfo.Reference[2]))  // Canonicalized Manifest element
    | Canon(SignedInfo)
    | Canon(SignatureValue) | Canon(KeyInfo)
    | Canon(UnsignedSignatureProperties[1])
    | Canon(UnsignedSignatureProperties[2]) |…
    | Canon(UnsignedSignatureProperties[lastExistingAtTimeStampingTime])
    | Canon(Object[2])                            // Canonicalized
```

This would lead to the same message imprint present in the time-stamp token encapsulated by the xadesv141:ArchiveTimeStamp property, and the attack would have succeeded.

Additionally, any check of the digest values present within the signed ds:Manifest would also succeed as the Fake Signed Data Object 1 has the same digest value, when computed with Dig1 algorithm, than the Signed Data Object 1.

Under these circumstances the relying party would not notice any problem in the validation of this XAdES-A signature.

A.3.3 The solution: xadesv141:RenewedDigests element

The present clause details how the usage of xadesv141:RenewedDigests unsigned property counters the attack described in clause A.3.2.

As Figure A.3 shows, before the algorithm Dig1 is broken, a new xadesv141:RenewedDigests unsigned property is incorporated into the XAdES signature. This property encapsulates one xadesv141:RenewedDigest child element per each detached signed data object signed through ds:Manifest whose digest had been computed with algorithm Dig1. After that, the signature is augmented by incorporation of a new xadesv141:ArchiveTimeStamp unsigned property.

The content of each xadesv141:RenewedDigest will be the base-64 encoding of the digest value computed according algorithm Dig2 on the aforementioned detached signed data objects.

Figure A.3 shows that after the incorporation of xadesv141:RenewedDigests unsigned property, a new archive time-stamp is requested and encapsulated into a new xadesv141:ArchiveTimeStamp unsigned property.

Figure A.3 also shows the message imprint computation input for the time-stamp token encapsulated by this new xadesv141:ArchiveTimeStamp unsigned property. This input concatenates the contents of any xadesv141:RenewedDigests unsigned property already incorporated into the XAdES signature at the moment of building the this message imprint computation input. The aforementioned xadesv141:RenewedDigests unsigned property includes the digest values of the detached signed data objects computed with Dig2, which is not broken at that point in time. This means that if the algorithm Dig1 is broken, and after that one of the original signed detached data objects is substituted by a fake detached object whose digest according to algorithm Dig1 is the same as the digest of the original signed detached object, the relying party still could be aware of the substitution of the original signed detached object because the digest values of the fake and the original signed detached objects are different when they are computed according algorithm Dig2.

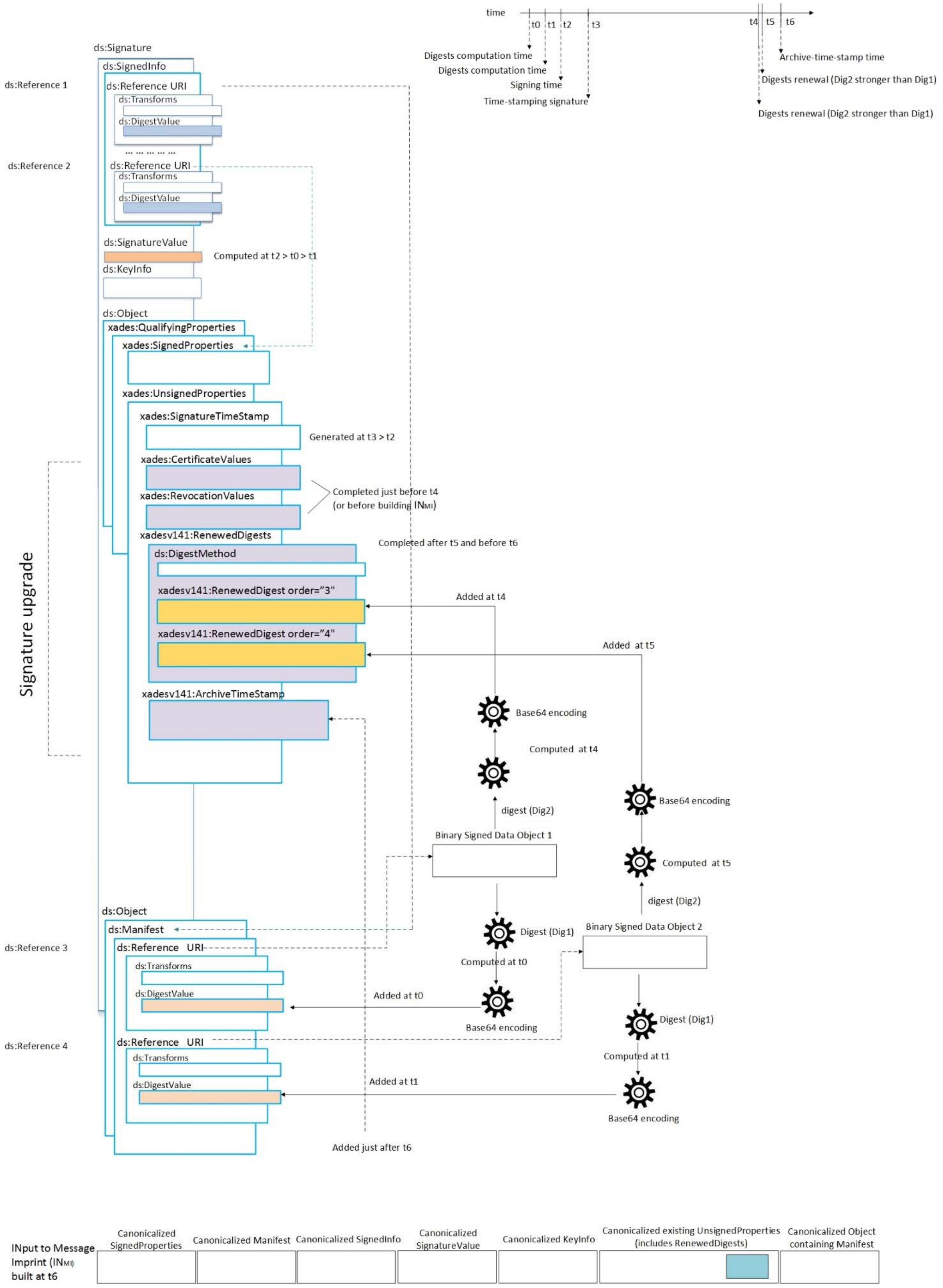


Figure A.3: Using xadesv141:RenewedDigests for countering signed detached objects substitution attack

In Figure A.3, if in t7 after t6, algorithm Dig1 is broken and an attacker succeeds in replacing Signed Data Object 1 by Fake Data Object 1 such as:

- H_{Dig1}(Signed Data Object 1) = H_{dig1} (Fake Data Object 1);

a relying party would still be able to the substitution, because:

- H_{Dig2}(Signed Data Object 1) != H_{Dig2} (Fake Data Object 1);

and this would make the relying party aware that Signed Data Object 1 had been changed since the time t6 when the `xadesv141:RenewedDigests` unsigned property was created.

A relying party would detect this type of substitutions performing the following steps for each `xadesv141:RenewedDigests` qualifying property found within the XAdES signature while validating it:

- For each `xadesv141:RenewedDigest` child element of the `xadesv141:RenewedDigests` qualifying property DO:

  a) Find the `ds:Reference` element within the suitable `ds:Manifest` as indicated by the value of the attribute `Order`.

  b) Process the aforementioned `ds:Reference` element following the reference processing model specified in XMLDSIG [i.37], clause 4.4.3.2.

  c) If the result is a XML node set, canonicalize it.

  d) Compute the digest value of the result in the previous step using the algorithm indicated within the `ds:DigestMethod` child element of `xadesv141:RenewedDigests` element. If the computed digest value is different from the digest value indicated in the `xadesv141:RenewedDigest` being processed, then annotate that there is a problem with the signed detached data object referenced by the `ds:Reference` within the signed `ds:Manifest` for further notification.

Annex B: Bibliography

- CROBIES WP 5-1: "Guidelines and guidance for cross-border and interoperable implementation of electronic signatures. WP 5-1".

- Directive 2006/123/EC of the European Parliament and of the Council of 12 December 2006 on services in the internal market.

History

Document history		
V1.1.1	March 2016	Publication