

Advanced Electronic Signature Profiles of the OASIS Digital Signature Service Version 1.0

OASIS Standard

11 April 2007

Specification URIs:

This Version:

<http://docs.oasis-open.org/dss/v1.0/oasis-dss-profiles-AdES-v1.0-os.html>

<http://docs.oasis-open.org/dss/v1.0/oasis-dss-profiles-AdES-v1.0-os.pdf>

<http://docs.oasis-open.org/dss/v1.0/oasis-dss-profiles-AdES-v1.0-os-doc>

Latest Version:

<http://docs.oasis-open.org/dss/v1.0/oasis-dss-profiles-AdES-v1.0-os.html>

<http://docs.oasis-open.org/dss/v1.0/oasis-dss-profiles-AdES-v1.0-os.pdf>

<http://docs.oasis-open.org/dss/v1.0/oasis-dss-profiles-AdES-v1.0-os-doc>

Technical Committee:

OASIS Digital Signature Services TC

Chair(s):

Nick Pope, Thales eSecurity

Juan Carlos Cruellas, Centre d'aplicacions avançades d'Internet (UPC)

Editor(s):

Juan Carlos Cruellas, Centre d'aplicacions avançades d'Internet (UPC)

Related work:

This specification is related to:

- oasis-dss-core-spec-v1.0-os

Abstract:

This document defines one abstract profile of the OASIS DSS protocols for the purpose of creating and verifying XML or CMS based Advanced Electronic Signatures. It also defines two concrete sub-profiles: one for creating and verifying XML Advanced Electronic Signatures and the other for creating and verifying CMS based Advanced Electronic Signatures.

Status:

This document was last revised or approved by the membership of OASIS on the above date. The level of approval is also listed above. Check the current location noted above for possible later revisions of this document. This document is updated periodically on no particular schedule.

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical

40 Committee by using the “Send A Comment” button on the Technical Committee’s
41 web page at <http://www.oasis-open.org/committees/dss/>.
42 For information on whether any patents have been disclosed that may be essential to
43 implementing this specification, and any offers of patent licensing terms, please refer
44 to the Intellectual Property Rights section of the Technical Committee web page
45 (<http://www.oasis-open.org/committees/dss/ipr.php>).
46 The non-normative errata page for this specification is located at [http://www.oasis-](http://www.oasis-open.org/committees/dss/)
47 [open.org/committees/dss/](http://www.oasis-open.org/committees/dss/).

48 Notices

49 OASIS takes no position regarding the validity or scope of any intellectual property or other
50 rights that might be claimed to pertain to the implementation or use of the technology
51 described in this document or the extent to which any license under such rights might or might
52 not be available; neither does it represent that it has made any effort to identify any such
53 rights. Information on OASIS's procedures with respect to rights in OASIS specifications can
54 be found at the OASIS website. Copies of claims of rights made available for publication and
55 any assurances of licenses to be made available, or the result of an attempt made to obtain a
56 general license or permission for the use of such proprietary rights by implementors or users
57 of this specification, can be obtained from the OASIS Executive Director.

58 OASIS invites any interested party to bring to its attention any copyrights, patents or patent
59 applications, or other proprietary rights which may cover technology that may be required to
60 implement this specification. Please address the information to the OASIS Executive Director.

61 Copyright © OASIS® 1993–2007. All Rights Reserved.

62 This document and translations of it may be copied and furnished to others, and derivative
63 works that comment on or otherwise explain it or assist in its implementation may be
64 prepared, copied, published and distributed, in whole or in part, without restriction of any kind,
65 provided that the above copyright notice and this paragraph are included on all such copies
66 and derivative works. However, this document itself may not be modified in any way, such as
67 by removing the copyright notice or references to OASIS, except as needed for the purpose
68 of developing OASIS specifications, in which case the procedures for copyrights defined in
69 the OASIS Intellectual Property Rights document must be followed, or as required to translate
70 it into languages other than English.

71 The limited permissions granted above are perpetual and will not be revoked by OASIS or its
72 successors or assigns.

73 This document and the information contained herein is provided on an "AS IS" basis and
74 OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT
75 LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL
76 NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY
77 OR FITNESS FOR A PARTICULAR PURPOSE.

78 The names "OASIS" are trademarks of OASIS, the owner and developer of this specification,
79 and should be used only to refer to the organization and its official outputs. OASIS welcomes
80 reference to, and implementation and use of, specifications, while reserving the right to
81 enforce its marks against misleading uses. Please see [http://www.oasis-](http://www.oasis-open.org/who/trademark.php)
82 [open.org/who/trademark.php](http://www.oasis-open.org/who/trademark.php) for above guidance.

Table of Contents

84	1	Introduction.....	7
85	1.1	Terminology	7
86	1.2	Normative References	7
87	1.3	Non-Normative References.....	8
88	1.4	Namespaces	8
89	2	Overview.....	9
90	3	Advanced Electronic Signature abstract profile	10
91	3.1	Overview	10
92	3.2	Profile Features.....	11
93	3.2.1	Scope	11
94	3.2.2	Relationship To Other Profiles.....	11
95	3.2.3	Signature Object	11
96	3.3	Profile of Signing Protocol.....	11
97	3.3.1	Element <SignRequest>	12
98	3.3.1.1	Element <OptionalInputs>	12
99	3.3.1.1.1	New Optional Inputs	12
100	3.3.1.1.1.1	Optional Input <SignatureForm>.....	12
101	3.3.1.1.2	Optional Inputs already defined in the Core	12
102	3.3.1.1.2.1	Optional Input <SignatureType>	12
103	3.3.1.1.2.2	Optional inputs <ClaimedIdentity> and <KeySelector>	12
104	3.3.1.1.2.3	Optional Input <SignedProperties>	12
105	3.3.1.1.2.3.1	Requesting SigningTime	13
106	3.3.1.1.2.3.2	Requesting CommitmentTypeIndication	13
107	3.3.1.1.2.3.3	Requesting SignatureProductionPlace	13
108	3.3.1.1.2.3.4	Requesting SignerRole	14
109	3.3.1.1.2.3.5	Requesting AllDataObjectsTimeStamp	14
110	3.3.1.1.2.3.6	Requesting DataObjectFormat.....	14
111	3.3.2	Element <SignResponse>	15
112	3.3.2.1	Element <SignatureObject>	15
113	3.3.2.2	Optional Outputs.....	15
114	3.4	Profile of Verifying Protocol.....	15
115	3.4.1	Element <VerifyRequest>	15
116	3.4.1.1	Attribute Profile	15
117	3.4.1.2	Element <SignatureObject>	15
118	3.4.1.3	Element <OptionalInputs>	16
119	3.4.1.3.1	Element <ReturnUpdatedSignature>	16
120	3.5	Element <VerifyResponse>	16
121	3.5.1.1	Element <OptionalOutputs>	16
122	3.5.1.1.1	Optional Output <UpdatedSignature>	16
123	4	XML Advanced Electronic Signatures concrete Profile.....	17
124	4.1	Overview	17
125	4.2	Profile features	17
126	4.2.1	Identifier	17
127	4.2.2	Scope	17
128	4.2.3	Relationship To Other Profiles.....	18
129	4.2.4	Signature Object	18
130	4.2.5	Transport Binding.....	18
131	4.2.6	Security Binding	18
132	4.3	Profile of Signing Protocol.....	18
133	4.3.1	Attribute Profile	18
134	4.3.2	Element <SignRequest>	18
135	4.3.2.1	Element <OptionalInputs>	18
136	4.3.2.1.1	New Optional Inputs	18

137	4.3.2.1.1.1	Element <SignatureForm>	18
138	4.3.2.1.2	Optional Inputs already defined in the Core	19
139	4.3.2.1.2.1	Optional Input <SignatureType>	19
140	4.3.2.1.2.2	Optional inputs < ClaimedIdentity> and <KeySelector>	19
141	4.3.2.1.2.3	Optional Input <SignedProperties>	19
142	4.3.2.1.2.3.1	Requesting SigningTime	19
143	4.3.2.1.2.3.2	Requesting CommitmentTypeIndication	19
144	4.3.2.1.2.3.3	Requesting SignatureProductionPlace	19
145	4.3.2.1.2.3.4	Requesting SignerRole	19
146	4.3.2.1.2.3.5	Requesting AllDataObjectTimeStamp	19
147	4.3.2.1.2.3.6	Requesting DataObjectFormat	20
148	4.3.2.1.2.3.7	Requesting <xades:IndividualDataObjectTimeStamp>	20
149	4.3.3	Element <SignResponse>	20
150	4.3.3.1	Element <SignatureObject>	20
151	4.4	Profile of Verifying Protocol	21
152	4.4.1	Element <VerifyRequest>	21
153	4.4.1.1	Attribute Profile	21
154	4.4.1.2	Element <SignatureObject>	21
155	4.4.1.3	Element <OptionalInputs>	21
156	4.4.1.3.1	Optional Output <ReturnUpdatedSignature>	21
157	4.4.2	Element <VerifyResponse>	21
158	4.4.2.1	Element <OptionalOutputs>	21
159	4.4.2.1.1	Optional Output <UpdatedSignature>	21
160	4.5	Profile Bindings	21
161	4.5.1	Transport Bindings	21
162	4.5.2	Security Bindings	21
163	4.5.2.1	Security Requirements	21
164	4.5.2.2	TLS X.509 Mutual Authentication	22
165	5	CMS-based Advanced Electronic Signature profile	23
166	5.1	Overview	23
167	5.2	Profile features	23
168	5.2.1	Identifier	23
169	5.2.2	Scope	23
170	5.2.3	Relationship To Other Profiles	24
171	5.2.4	Signature Object	24
172	5.2.5	Transport Binding	24
173	5.2.6	Security Binding	24
174	5.3	Profile of Signing Protocol	24
175	5.3.1	Element <SignRequest>	24
176	5.3.1.1	Attribute Profile	24
177	5.3.1.2	Element <OptionalInputs>	24
178	5.3.1.2.1	New Optional Inputs	24
179	5.3.1.2.1.1	Element <SignatureForm>	24
180	5.3.1.2.2	Optional Inputs already defined in the Core	24
181	5.3.1.2.2.1	Element <SignatureType>	25
182	5.3.1.2.2.2	Optional inputs < ClaimedIdentity> / <KeySelector>	25
183	5.3.1.2.2.3	Element <SignedProperties>	25
184	5.3.1.2.2.3.1	Requesting signing-time	25
185	5.3.1.2.2.3.2	Requesting commitment-type-indication	25
186	5.3.1.2.2.3.3	Requesting signer-location	25
187	5.3.1.2.2.3.4	Requesting signer-attributes	25
188	5.3.1.2.2.3.5	Requesting content-time-stamp	25
189	5.3.1.2.2.3.6	Requesting content-hints	26
190	5.3.2	Element <SignResponse>	26
191	5.3.2.1	Element <SignatureObject>	26
192	5.4	Profile of Verifying Protocol	26
193	5.4.1	Element <VerifyRequest>	26
194	5.4.1.1	Attribute Profile	26
195	5.4.1.2	Element <OptionalInputs>	26

196	5.4.1.2.1 Element <ReturnUpdatedSignature>	26
197	5.4.1.3 Element <SignatureObject>	26
198	5.4.2 Element <VerifyResponse>	26
199	5.4.2.1 Element <OptionalOutputs>	26
200	5.4.2.1.1 Element <UpdatedSignature>	26
201	5.5 Profile Bindings	27
202	5.5.1 Transport Bindings	27
203	5.5.2 Security Bindings	27
204	5.5.2.1 Security Requirements	27
205	5.5.2.2 TLS X.509 Mutual Authentication	27
206	6 XML timestamps in XAdES signatures	28
207	6.1 Generation and inclusion of XML timestamps	28
208	6.1.1 Profile for XAdES timestamp containers	28
209	6.1.2 XML timestamp within xades:IndividualDataObjectsTimeStamp	29
210	6.1.3 XML timestamp within xades:AllDataObjectsTimeStamp	29
211	6.1.4 XML timestamp within xades:SigAndRefsTimeStamp	29
212	6.1.5 XML timestamp within xades:RefsOnlyTimeStamp	30
213	6.1.6 XML timestamp within xades:ArchiveTimeStamp	30
214	6.2 Verification of XML timestamps	30
215	6.2.1 Verification of of xades:IndividualDataObjectsTimeStamp including a XML	
216	timestamp	31
217	6.2.2 Verification of xades:AllDataObjectsTimeStamp including a XML timestamp...	31
218	6.2.3 Verification of xades:SigAndRefsTimeStamp including a XML timestamp	32
219	6.2.4 Verification of xades:RefsOnlyTimeStamp including a XML timestamp	32
220	6.2.5 Verification of xades:ArchiveTimeStamp including a XML timestamp	33
221	7 Identifiers defined in this specification	34
222	7.1 Predefined advanced electronic signature forms identifiers	34
223	7.2 Result Identifiers	34
224	A. Acknowledgements	36
225		

1 Introduction

The DSS signing and verifying protocols are defined in [DSSCore]. As defined in that document, the DSS protocols have a fair degree of flexibility and extensibility. This document defines an abstract profile for the use of the DSS protocols for creating and verifying XML and CMS-based Advanced Electronic Signatures as defined in [XAdES] and [CAdES]. This document also defines two concrete profiles derived from the abstract one: one for creating and verifying XAdES signatures and the other for creating and verifying CAdES signatures.

1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in IETF RFC 2119 [RFC 2119]. These keywords are capitalized when used to unambiguously specify requirements over protocol features and behavior that affect the interoperability and security of implementations. When these words are not capitalized, they are meant in their natural-language sense.

This specification uses the following typographical conventions in text: `<ns:Element>`, **Attribute**, **Datatype**, **OtherCode**.

1.2 Normative References

- [AdES-XSD] J. C. Cruellas et al. AdES Profile Schema, OASIS, February 2007.
- [CAdES] CMS Advanced Electronic Signatures. ETSI TS 101 733, January 2007.
- [Core-XSD] S. Drees et al. *DSS Schema*. OASIS, February 2007).
- [DSSCore] S. Drees et al. *Digital Signature Service Core Protocols and Elements*. OASIS, February 2007.
- [RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.
- [RFC 2634] . Hoffman (ed.). Enhanced Security Services for S/MIME <http://www.ietf.org/rfc/rfc2634.txt>, , IETF RFC 2634 June 1999
- [RFC 3852] R. Housley. Cryptographic Message Syntax (CMS) , IETF RFC 3852, July 2004.
- [XAdES] Advanced Electronic Signatures. ETSI TS 101 733. March 2006.
- [XML-ns] T. Bray, D. Hollander, A. Layman. *Namespaces in XML*. <http://www.w3.org/TR/1999/REC-xml-names-19990114>, W3C Recommendation, January 1999.
- [XMLSig] D. Eastlake et al. *XML-Signature Syntax and Processing*. <http://www.w3.org/TR/1999/REC-xml-names-19990114>, W3C Recommendation, February 2002.

1.3 Non-Normative References

1.4 Namespaces

The structures described in this specification are contained in the schema file [AdES-XSD]. All schema listings in the current document are excerpts from the schema file. In the case of a disagreement between the schema file and this document, the schema file takes precedence.

This schema is associated with the following XML namespace:

urn:oasis:names:tc:dss:1.0:profiles:AdES:schema#

Conventional XML namespace prefixes are used in this document:

- o The prefix **dss:** (or no prefix) stands for the DSS core namespace [Core-XSD].
- o The prefix **ds:** stands for the W3C XML Signature namespace [XMLSig].
- o The prefix **xades:** stands for ETSI XML Advanced Electronic Signatures (XAdES) document [XAdES].

Applications MAY use different namespace prefixes, and MAY use whatever namespace defaulting/scoping conventions they desire, as long as they are compliant with the Namespaces in XML specification [XML-ns].

2 Overview

This document defines three profiles of the protocols specified in: “Digital Signature Services Core Protocol and Elements” [DSSCore].

The first one is an abstract profile defining messages for supporting the lifecycle of advanced electronic signatures. Both, XML and CMS-based advanced electronic signatures are supported by this profile.

One concrete profile, derived from the aforementioned abstract profile, gives support to the lifecycle of XML advanced electronic signatures as specified in [XAdES].

A second concrete profile, also derived from the abstract one, gives support to the lifecycle of CMS-based advanced electronic signatures as specified in [CAdES].

Implementations should implement one of the concrete profiles (or both) in order to request generation or validation of advanced electronic signatures in one of the two formats (or both).

3 Advanced Electronic Signature abstract profile

3.1 Overview

This abstract profile supports operations within each phase of the lifecycle of two types of advanced electronic signature:

- o XML encoded signatures based on [XMLSig] such as specified in [XAdES].
- o Binary encoded signatures based on [RFC 3852] such as specified in [CAAdES].

Henceforward, the document will use the term advanced signature when dealing with issues that affect to both types of signatures. The document will use XAdES or CAAdES signatures when dealing with issues that affect one or the other but not both of them.

For the generation of advanced signatures, the following operations apply:

- o SignRequest. This operation supports requests for:
 - o Generating predefined advanced signature forms as defined in [XAdES] and [CAAdES].
 - o Generating XML signatures incorporating specific signed/unsigned properties whose combination does not fit any predefined XAdES signature form. In such cases, the form MUST have been defined in a proprietary specification and MUST be identified by one URI.
 - o Generating CMS signatures incorporating specific signed/unsigned attributes whose combination does not fit any predefined [CAAdES] signature form. In such cases, the form MUST have been defined in a proprietary specification and MUST be identified by one URI.
- o SignResponse. This operation supports delivery of:
 - o Predefined advanced signature forms as defined in [XAdES] and [CAAdES].
 - o XML signatures with specific properties whose combination does not fit any predefined XAdES signature form. In such cases, the form MUST have been defined in some other specification and MUST be identified by one URI.
 - o CMS signatures incorporating specific signed attributes whose combination does not fit any predefined [CAAdES] signature form. In such cases, the form MUST have been defined in some other specification and MUST be identified by one URI.

For advanced signature verification (and updating) the following operations apply:

- o VerifyRequest. This operation supports requests for:
 - o Verifying a predefined advanced signature form.
 - o Verifying XML signatures incorporating specific properties whose combination does not fit any predefined XAdES signature form.
 - o Verifying any of the signatures mentioned above PLUS updating them by addition of additional properties (time-stamps, validation data, etc) leading to a predefined XAdES form.
 - o Verifying CMS signatures incorporating specific attributes whose combination does not fit any predefined [CAAdES] signature form.
 - o Verifying any of the signatures mentioned above PLUS updating them by addition of additional attributes (time-stamps, validation data, etc) leading to a predefined [CAAdES] form.

- 341 o Verifying a long-term advanced signature in a certain point of time.
- 342 o VerifyResponse. This operation supports delivery of:
- 343 o Advanced signature verification result of signatures mentioned above.
- 344 o Advanced signature verification result PLUS the updated signatures as
- 345 requested.
- 346 The material for each operation will clearly indicate the lifecycle phase it pertains to.

347 **3.2 Profile Features**

348 **3.2.1 Scope**

349 This document profiles the DSS signing and verifying protocols defined in [DSSCore].

350 **3.2.2 Relationship To Other Profiles**

351 The profile in this document is based on the [DSSCore]. The profile in this document may not
352 be directly implemented. It is further profiled by the two concrete profiles also defined in
353 sections 4 and 5.

354 **3.2.3 Signature Object**

355 This profile supports the creation and verification of advanced signatures as defined in
356 [XAdES] and [CAAdES].

357 This profile also supports update of advanced signatures by addition of unsigned properties
358 (time-stamps and different types of validation data), as specified in [XAdES] and [CAAdES].

359 **3.3 Profile of Signing Protocol**

360 The present profile allows requesting:

- 361 o Predefined forms of advanced electronic signatures as defined in [XAdES] and
- 362 [CAAdES].
- 363 o Other forms of signatures based in [XMLSig] or [RFC 3852] defined in other
- 364 specifications,

365 In both cases, the specific requested form will be identified by an URI.

366 According to this profile, the following predefined advanced signature forms defined in
367 [XAdES] and [CAAdES] MAY be requested (those forms whose name begin by XAdES- are
368 forms names for XAdES signatures; those ones whose name begin by CAAdES are names for
369 CAAdES signatures):

- 370 o CAAdES-BES and XAdES-BES. In this form, the signing certificate is secured by the
- 371 signature itself.
- 372 o CAAdES-EPES and XAdES-EPES. This form incorporates an explicit identifier of the
- 373 signature policy that will govern the signature generation and verification.
- 374 o CAAdES-ES-T and XAdES-T. This form incorporates a trusted time, by means of a
- 375 time-stamp token or a time-mark.
- 376 o CAAdES-ES-C and XAdES-C.
- 377 o CAAdES-ES-X and XAdES-X.
- 378 o CAAdES-ES-X-L and XAdES-X-L.
- 379 o CAAdES-ES-A and XAdES-A.

380 In addition, the present profile provides means for requesting incorporation in any of the
381 aforementioned forms any of the signed properties defined in [XAdES] and signed attributes
382 defined in [CAAdES].

383 Other electronic signature forms based in [XMLSig] or [RFC 3852], defined elsewhere, MAY
384 also be requested using the mechanisms defined in this profile.

385 **3.3.1 Element <SignRequest>**

386 This clause profiles the `dss:SignRequest` element.

387 **3.3.1.1 Element <OptionalInputs>**

388 **3.3.1.1.1 New Optional Inputs**

389 **3.3.1.1.1.1 Optional Input <SignatureForm>**

390 The form of signature required MAY be indicated using the following new optional input

391 `<xs:element name="SignatureForm" type="xs:anyURI"/>`

392 If not present the signature form SHALL be implied by the selected <SignaturePolicy> or
393 the signature policy applied by the server.

394 Section 7.1 of this abstract profile defines a set of URIs identifying the predefined advanced
395 electronic signature forms specified in [CAAdES] and [XAdES].

396 Should other standard or proprietary specification define new signature forms and their
397 corresponding URIs, concrete sub-profiles of this abstract profile could be defined for giving
398 support to their verification and update.

399 Should a form identified by an URI, admit different properties combinations, the server will
400 produce a specific combination depending on its policy or configuration settings.

401 **3.3.1.1.2 Optional Inputs already defined in the Core**

402 None of the optional inputs specified in the [DSS Core] are precluded in this abstract profile. It
403 only constrains some of them and specifies additional optional inputs.

404 **3.3.1.1.2.1 Optional Input <SignatureType>**

405 This element is OPTIONAL. If present, <SignatureType> SHALL be either:

406 `urn:ietf:rfc:3275`

407 for requesting XML-based signatures, or

408 `urn:ietf:rfc:3369`

409 for requesting CMS-based signatures, as defined in 7.1 of [DSS Core].

410 If not present the signature type SHALL be implied by the selected <SignaturePolicy> or
411 the signature policy applied by the server.

412 **3.3.1.1.2.2 Optional inputs <ClaimedIdentity> and <KeySelector>**

413 As forms defined in [XAdES] and [CAAdES] require that the signing certificate is protected by
414 the signature, the server MUST gain access to that certificate.

415 <dss:ClaimedIdentity> or <dss:KeySelector> optional inputs MAY be present. If
416 they are not present, the server may use means not specified in this profile to identify the
417 signer's key and gain access to its certificate.

418 **3.3.1.1.2.3 Optional Input <SignedProperties>**

419 The requester MAY request to the server the addition of optional signed properties using the
420 <dss:SignedProperties> element's <dss:Property> child profiled as indicated in
421 clauses below. First names correspond to the one given by XAdES to the signed properties.
422 Second ones correspond to the names given by CAAdES to the signed attributes.

423 Signed properties that MAY be requested are:

XAdES	CAdES
SigningTime	signing-time
CommitmentTypeIndication	commitment-type-indication
SignerRole	signer-attributes
SignatureProductionPlace	signer-location
DataObjectFormat	content-hints
AllDataObjectsTimeStamp	content-time-stamp
IndividualDataObjectsTimeStamp	No equivalent signed attribute

424

425 Next sub-sections show how a client should request each of the aforementioned properties-
426 attributes. The type of signature requested (XAdES or CAdES) will determine whether a
427 XAdES property or a CAdES attribute is generated by the server.

428 **3.3.1.1.2.3.1 Requesting SigningTime**

429 Value for <Identifier> element:

430 **urn:oasis:names:tc:dss:1.0:profiles:AdES:SigningTime**

431 If the client does not request such property, the server still MAY generate and include this
432 property depending on its policy.

433 No content is required for Value element, since the actual contents of the property will be
434 generated by the server when required.

435 **3.3.1.1.2.3.2 Requesting CommitmentTypeIndication**

436 Value for <Identifier> element:

437 **urn:oasis:names:tc:dss:1.0:profiles:AdES:CommitmentTypeIndication**

438 If the client does not request such property, the server still MAY generate and include it with
439 values that depend on server's policy.

440 The client MAY request the generation and inclusion of this signed property. In such cases
441 the <Value> element MUST have the following content:

```
442 <xs:element name="RequestedCommitment">
443   <xs:complexType>
444     <xs:choice>
445       <xs:element ref="xades:CommitmentTypeIndication"/>
446       <xs:element name="BinaryValue" type="xs:base64Binary"/>
447     </xs:choice>
448   </xs:complexType>
449 </xs:element>
```

450 Element <xades:CommitmentTypeIndication> will be present when requesting a XML
451 signature.

452 Element <BinaryValue> will be present when requesting an ASN.1 signature. Its contents
453 MUST be the base64 encoding of **commitment-type-indication** ASN.1 attribute defined
454 in [CAdES], DER-encoded

455 **3.3.1.1.2.3.3 Requesting SignatureProductionPlace**

456 Value for <Identifier> element:

urn:oasis:names:tc:dss:1.0:profiles:AdES:SignatureProductionPlace

The client MAY request a certain value for this property. Nevertheless, this value MAY be ignored by the server depending on its own policy, and the property be set to another value.

For requesting a value for this property, the <Value> element MUST have the following content:

```
<xs:element name="RequestedSignatureProductionPlace">
  <xs:complexType>
    <xs:choice>
      <xs:element ref="xades:SignatureProductionPlace"/>
      <xs:element name="BinaryValue" type="xs:base64Binary"/>
    </xs:choice>
  </xs:complexType>
</xs:element>
```

Element <xades:SignatureProductionPlace> will be present when requesting a XML signature.

Element <BinaryValue> will be present when requesting an ASN.1 signature. Its contents MUST be the base64 encoding of **SignerLocation** ASN.1 attribute defined in [CAvES], DER-encoded.

3.3.1.1.2.3.4 Requesting SignerRole

Value for <Identifier> element:

urn:oasis:names:tc:dss:1.0:profiles:AdES:SignerRole

When the client requests the generation and inclusion of this signed property the <Value> element MUST have the following content:

```
<xs:element name="RequestedSignerRole">
  <xs:complexType>
    <xs:choice>
      <xs:element ref="xades:SignerRole"/>
      <xs:element name="BinaryValue" type="xs:base64Binary"/>
    </xs:choice>
  </xs:complexType>
</xs:element>
```

Element <xades:SignerRole> will be present when requesting a XML signature.

Element <BinaryValue> will be present when requesting a ASN.1 signature. Its contents MUST be the base64 encoding of **signer-attributes** ASN.1 attribute defined in [CAvES], DER-encoded.

3.3.1.1.2.3.5 Requesting AllDataObjectsTimeStamp

This element will be added for requesting the generation and inclusion of a time-stamp token on (all) the data object(s) to be signed.

Value for <Identifier> element:

urn:oasis:names:tc:dss:1.0:profiles:AdES:AllDataObjectsTimeStamp

No content is required for <Value> element, since the actual contents of the property will be generated by the server when required.

3.3.1.1.2.3.6 Requesting DataObjectFormat

Value for Identifier element:

urn:oasis:names:tc:dss:1.0:profiles:AdES:DataObjectFormat

When the client requests the generation and inclusion of this signed property the <Value> element MUST have the following content.

```

505 <xs:element name="RequestedDocsFormat" type="DocsFormatType"/>
506
507 <xs:complexType name="DocsFormatType">
508   <xs:sequence>
509     <xs:choice>
510       <xs:element name="DocFormat" type="DocFormatType"
511 maxOccurs="unbounded"/>
512       <xs:element name="BinaryValue" type="xs:base64Binary"/>
513     </xs:choice>
514   </xs:sequence>
515 </xs:complexType>
516
517 <xs:complexType name="DocFormatType">
518   <xs:complexContent>
519     <xs:extension base="DocReferenceType">
520       <xs:sequence>
521         <xs:element ref="xades:DataObjectFormat"/>
522       </xs:sequence>
523     </xs:extension>
524   </xs:complexContent>
525 </xs:complexType>

```

526 Elements <DocFormat> will be present when requesting an XML based signature.

527 Element <BinaryValue> will be present when requesting a CMS based signature. Its
528 contents MUST be the base64 encoding of **content-hints** ASN.1 attribute defined in [RFC
529 2634] DER-encoded.

530 3.3.2 Element <SignResponse>

531 This clause profiles the dss:SignResponse element.

532 3.3.2.1 Element <SignatureObject>

533 This element **SHALL NOT** contain a dss:TimeStamp element as a child.

534 3.3.2.2 Optional Outputs

535 None of the optional outputs specified in the [DSS Core] are neither precluded nor further
536 profiled in this abstract profile.

537 3.4 Profile of Verifying Protocol

538 3.4.1 Element <VerifyRequest>

539 This clause specifies the profile for the contents of the dss:VerifyRequest when used for:

- 540 o Requesting verification of advanced signatures.
- 541 o Requesting verification of advanced signatures AND update of signatures to other
- 542 predefined forms.

543 3.4.1.1 Attribute Profile

544 The value for the Profile attribute, indicating the concrete sub-profile of this abstract profile,
545 MUST be present.

546 3.4.1.2 Element <SignatureObject>

547 This element **SHALL NOT** contain a dss:TimeStamp element as a child.

3.4.1.3 Element <OptionalInputs>

None of the optional inputs specified in the [DSS Core] are precluded in this abstract profile. It only constrains some of them and specifies additional optional inputs.

3.4.1.3.1 Element <ReturnUpdatedSignature>

This element MUST be present when the client requests verification of a signature and update to a predefined form of advanced signature.

The Type attribute identifies the advanced signature form requested.

Acceptable predefined values for this attribute are the URIs specified in table 1 corresponding to the following forms predefined in [CAAdES] and [XAdES]: XAdES-T/CAAdES -T, XAdES-C/CAAdES-C, XAdES-X/CAAdES-X, XAdES-X-L/CAAdES-X-L, XAdES-A/CAAdES-A.

Should other standard or proprietary specification define new signature forms and their corresponding URIs, concrete sub-profiles of this abstract profile could be defined for giving support to their verification and update.

When the requested form allows for different contents, the server MUST decide the specific contents of the updated signature delivered, according to its configuration and settings.

3.5 Element <VerifyResponse>

This clause profiles the `dss:VerifyResponse` element.

3.5.1.1 Element <OptionalOutputs>

None of the optional inputs specified in the [DSS Core] are precluded in this abstract profile. It only constrains some of them.

3.5.1.1.1 Optional Output <UpdatedSignature>

This element SHALL contain a `dss:SignatureObject` element that SHALL NOT contain a `dss:TimeStamp` element as a child.

4 XML Advanced Electronic Signatures concrete Profile

4.1 Overview

This concrete profile supports operations within each phase of the lifecycle of XML Advanced Electronic Signature based on [XMLSig] such as specified in [XAdES]. It will then provide all the features related to XAdES signatures that are specified in the abstract profile defined in section 3.

For the generation of XAdES signatures, the following operations apply:

- o SignRequest. This operation supports requests for:
 - o Generating predefined advanced signature forms as defined in [XAdES].
 - o Generating XML signatures incorporating specific signed/unsigned properties whose combination does not fit any predefined XAdES signature form. In such cases, the form MUST have been defined in a proprietary specification and MUST be identified by one URI.
- o SignResponse. This operation supports delivery of:
 - o Predefined advanced signature forms as defined in [XAdES].
 - o XML signatures with specific properties whose combination does not fit any predefined XAdES signature form. In such cases, the form MUST have been defined in a proprietary specification and MUST be identified by one URI.

For verification [and updating] of XAdES signatures the following operations apply:

- o VerifyRequest. This operation supports requests for:
 - o Verifying a predefined XAdES signature form.
 - o Verifying XML signatures incorporating specific properties whose combination does not fit any predefined XAdES signature form.
 - o Verifying any of the signatures mentioned above PLUS updating them by adding unsigned properties (time-stamps, validation data, etc) leading to a predefined XAdES form.
 - o Verifying a long-term advanced signature in a certain point of time.
- o VerifyResponse. This operation supports delivery of:
 - o Advanced signature verification result of signatures mentioned above.
 - o Advanced signature verification result PLUS the updated signatures as requested.

4.2 Profile features

4.2.1 Identifier

urn:oasis:names:tc:dss:1.0:profiles:XAdES.

4.2.2 Scope

This document profiles the DSS abstract profile defined in section 3 of the present document.

4.2.3 Relationship To Other Profiles

The profile in this section is based on the abstract profile for Advanced Electronic Signatures defined in section 3.

4.2.4 Signature Object

This profile supports the creation and verification of XML advanced signatures as defined in [XAdES].

This profile also supports verification and update of advanced signatures by addition of unsigned properties (time-stamps and different types of validation data), as specified in [XAdES]

4.2.5 Transport Binding

This profile does not specify or constrain the transport binding.

4.2.6 Security Binding

This profile does not specify or constrain the security binding.

4.3 Profile of Signing Protocol

The present profile allows requesting:

- o Predefined forms of advanced electronic signatures as defined in [XAdES]. A server aligned with this profile SHALL generate XAdES signatures with direct incorporation of qualifying properties as defined in [XAdES] section 6.3.

- o Other forms of signatures based in [XMLSig] defined in other specifications,

In both cases, the specific requested form will be identified by an URI.

According to this profile, the following predefined advanced signature forms defined in [XAdES] MAY be requested: XAdES-BES, XAdES-EPES, XAdES-T, XAdES-C, XAdES-X, XAdES-X-L., and XAdES-A.

In addition, the present profile provides means for requesting incorporation in any of the aforementioned forms any of the following properties: `SigningTime`, `CommitmentTypeIndication`, `SignatureProductionPlace`, `SignerRole`, `IndividualDataObjectTimeStamp`, `AllDataObjectTimeStamp` and `DataObjectFormat`.

Other electronic signature forms based in [XMLSig] defined elsewhere MAY also be requested using the mechanisms defined in this profile.

4.3.1 Attribute Profile

`urn:oasis:names:tc:dss:1.0:profiles:XAdES.`

4.3.2 Element <SignRequest>

This clause profiles the `dss:SignRequest` element.

4.3.2.1 Element <OptionalInputs>

4.3.2.1.1 New Optional Inputs

4.3.2.1.1.1 Element <SignatureForm>

Usage of these elements is according to what is stated in section 3.3.1.1.1.1.

4.3.2.1.2 Optional Inputs already defined in the Core

None of the optional inputs specified in the [DSS Core] are precluded in this abstract profile. It only constrains some of them and specifies additional optional inputs.

4.3.2.1.2.1 Optional Input <SignatureType>

This element is MANDATORY. Its value MUST be:

urn:ietf:rfc:3275

4.3.2.1.2.2 Optional inputs < ClaimedIdentity> and <KeySelector>

Usage of these elements is according to what is stated in section 3.3.1.1.2.2.

4.3.2.1.2.3 Optional Input <SignedProperties>

4.3.2.1.2.3.1 Requesting SigningTime

Clients MAY use the URI defined in 3.3.1.1.2.3.1 or alternatively they MAY also use the following one:

urn:oasis:names:tc:dss:1.0:profiles:XAdES:SigningTime

Usage of these elements is according to what is stated in section 3.3.1.1.2.3.1.

4.3.2.1.2.3.2 Requesting CommitmentTypeIndication

Clients MAY use the URI defined in 3.3.1.1.2.3.2 or alternatively they MAY also use the following one:

urn:oasis:names:tc:dss:1.0:profiles:XAdES:CommitmentTypeIndication

When this optional input is present, the <Value> element MUST contain a <RequestedCommitment> element as defined in section 3.3.1.1.2.3.2 with the <xades:CommitmentTypeIndication>.

4.3.2.1.2.3.3 Requesting SignatureProductionPlace

Clients MAY use the URI defined in 3.3.1.1.2.3.3 or alternatively they MAY also use the following one:

urn:oasis:names:tc:dss:1.0:profiles:XAdES:SignatureProductionPlace

When this optional input is present, the <Value> element MUST contain a <RequestedSignatureProductionPlace> element as defined in section 3.3.1.1.2.3.3 with the <xades:SignatureProductionPlace>.

4.3.2.1.2.3.4 Requesting SignerRole

Clients MAY use the URI defined in 3.3.1.1.2.3.4 or alternatively they MAY also use the following one:

urn:oasis:names:tc:dss:1.0:profiles:XAdES:SignerRole

When this optional input is present, the <Value> element MUST contain a <RequestedSignerRole> element as defined in section 3.3.1.1.2.3.4 with the <xades:SignerRole> child.

4.3.2.1.2.3.5 Requesting AllDataObjectTimeStamp

Clients MAY use the URI defined in 3.3.1.1.2.3.5 or alternatively they MAY also use the following one:

urn:oasis:names:tc:dss:1.0:profiles:XAdES:AllDataObjectsTimeStamp

687 Usage of these elements is according to what is stated in section 3.3.1.1.2.3.5.

688 4.3.2.1.2.3.6 Requesting DataObjectFormat

689 Clients MAY use the URI defined in 3.3.1.1.2.3.6 or alternatively they MAY also use the
690 following one:

691 **urn:oasis:names:tc:dss:1.0:profiles:XAdES:AllDataObjectsTimeStamp**

692 When this optional input is present, the <Value> element MUST contain a
693 <RequestedDocsFormat> element as defined in section 3.3.1.1.2.3.6 with one or more
694 <DocFormat> children.

695 4.3.2.1.2.3.7 Requesting <xades:IndividualDataObjectTimeStamp>

696 Value for <Identifier> element:

697 **urn:oasis:names:tc:dss:1.0:profiles:XAdES:IndividualDataObjectTimeSta**
698 **mp**

699 In this case, the content of <Value> element will be the element
700 <DocsToBeTimeStamped>, defined as shown below.

```
701 <xs:element name="DocsToBeTimeStamped" type="DocReferencesType"/>
702
703 <xs:complexType name="DocReferencesType">
704   <xs:sequence>
705     <xs:element name="DocReference" maxOccurs="unbounded"
706       type="DocReferenceType"/>
707   </xs:sequence>
708 </xs:complexType>
709
710 <xs:complexType name="DocReferenceType">
711   <xs:attribute name="WhichDocument" type="xs:IDREF"
712     use="required"/>
713   <xs:attribute name="RefId" type="xs:string" use="optional"/>
714 </xs:complexType>
```

715 WhichDocument attribute contains the reference to the document whose time-stamp is
716 requested (see attribute ID in [CoreDSS] section 2.4.1). Should the client request the
717 generation of several ds:Reference element for this document (using
718 dss:SignedReferences optional input), the server SHALL timestamp all the data objects
719 referenced by these ds:Reference elements. Under these conditions, each
720 dss:SignedReference element MUST have its RefId attribute set to a not empty value.

721 [XAdES] mandates that <ds:Reference> elements corresponding to signed data objects
722 that have been individually time-stamped before being signed, must include an **Id** attribute.
723 [XAdES] also mandates <xades:IndividualDataObjectsTimeStamp> element to use
724 this **Id** attribute to indicate what signed documents have actually been time-stamped before
725 signing. See [XAdES] <xades:TimeStampType> and
726 <xades:IndividualDataObjectsTimeStamp> definitions for more details.

727 The client MAY request a value for the <ds:Reference> element's Id attribute using the
728 RefId optional attribute if a <dss:SignedReference> forcing a value for such an attribute
729 is not present in the request. If the request does not specify a value for this attribute, then the
730 server will automatically generate it.

731 4.3.3 Element <SignResponse>

732 This section profiles the dss:SignResponse element.

733 4.3.3.1 Element <SignatureObject>

734 The content of this element MUST be one of the following:

735 A ds:Signature element containing a XMLSig based signature.

736 A `dss:SignaturePtr` pointing to the XMLSig based signature embedded in an output
737 document.

738 **4.4 Profile of Verifying Protocol**

739 A server verifying XAdES signatures SHOULD follow the recommendations made by the
740 XAdES standard it aligns to with respect on how to verify the signed and unsigned properties
741 (version XAdES v1.3.2 includes an informative annex on this topic).

742 **4.4.1 Element <VerifyRequest>**

743 This clause profiles the `dss:VerifyRequest` element.

744 **4.4.1.1 Attribute Profile**

745 `urn:oasis:names:tc:dss:1.0:profiles:XAdES`.

746 **4.4.1.2 Element <SignatureObject>**

747 This element SHALL NOT contain a `dss:TimeStamp` element as a child.

748 **4.4.1.3 Element <OptionalInputs>**

749 **4.4.1.3.1 Optional Output <ReturnUpdatedSignature>**

750 Usage of these elements is according to what is stated in section 3.4.1.3.1.

751 **4.4.2 Element <VerifyResponse>**

752 This clause profiles the `dss:VerifyResponse` element.

753 **4.4.2.1 Element <OptionalOutputs>**

754 None of the optional inputs specified in the [DSS Core] are precluded in this profile. It only
755 constrains some of them.

756 **4.4.2.1.1 Optional Output <UpdatedSignature>**

757 The content of the `dss:UpdatedSignature` will be a `dss:SignatureObject` element
758 with one of the following contents:

- 759 o A `ds:Signature` containing a XMLSig based signature.
- 760 o A `dss:SignaturePtr` pointing to the XMLSig based signature embedded in one of
761 the inputdocuments.

762 **4.5 Profile Bindings**

763 **4.5.1 Transport Bindings**

764 Messages transported in this profile MAY be transported by the HTTP POST Transport
765 Binding and the SOAP 1.2 Transport Binding defined in [DSSCore].

766 **4.5.2 Security Bindings**

767 **4.5.2.1 Security Requirements**

768 This profile MUST use security bindings that:

- 769 o Authenticates the requester to the DSS server

- 770 o Authenticates the DSS server to the DSS client
- 771 o Protects the integrity of a request, response and the association of response to the
- 772 request.
- 773 o Optionally, protects the confidentiality of a request and response.
- 774 o The following MAY be used to meet these requirements.

775 **4.5.2.2 TLS X.509 Mutual Authentication**

776 This profile is secured using the TLS X.509 Mutual Authentication Binding defined in
777 [DSSCore].

778

5 CMS-based Advanced Electronic Signature profile

5.1 Overview

This concrete profile supports operations within each phase of the lifecycle of CMS based Advanced Electronic Signature based on [RFC 3852] such as specified in [CAAdES]. It will then provide all the features related to CAAdES signatures that are specified in the abstract profile defined in section 3.

For the generation of CAAdES signatures, the following operations apply:

- o SignRequest. This operation supports requests for:
 - o Generating predefined advanced signature forms as defined in [CAAdES].
 - o Generating CMS signatures incorporating specific signed/unsigned attributes whose combination does not fit any predefined [CAAdES] signature forms. In such cases, the form MUST have been defined in a proprietary specification and MUST be identified by one URI.
- o SignResponse. This operation supports delivery of:
 - o Predefined advanced signature forms as defined in [CAAdES].
 - o CMS signatures incorporating specific signed attributes whose combination does not fit any predefined [CAAdES] signature forms. In such cases, the form MUST have been defined in a proprietary specification and MUST be identified by one URI.

For verification [and updating] of signatures as specified in [CAAdES] the following operations apply:

- o VerifyRequest. This operation supports requests for:
 - o Verifying a predefined [CAAdES] signature form.
 - o Verifying CMS signatures incorporating specific attributes whose combination does not fit any predefined [CAAdES] signature form.
 - o Verifying any of the signatures mentioned above PLUS updating them by addition of additional attributes (time-stamps, validation data, etc) leading to a predefined [CAAdES] form.
 - o Verifying a long-term advanced signature in a certain point of time.
- o VerifyResponse. This operation supports delivery of:
 - o Advanced signature verification result of signatures mentioned above.
 - o Advanced signature verification result PLUS the updated signatures as requested.

5.2 Profile features

5.2.1 Identifier

urn:oasis:names:tc:dss:1.0:profiles:CAAdES.

5.2.2 Scope

This document profiles the DSS abstract profile defined in section 3 of the present document.

5.2.3 Relationship To Other Profiles

The profile in this document is based on the abstract profile for Advanced Electronic Signatures defined in section 3.

5.2.4 Signature Object

This profile supports the creation and verification of CMS based advanced signatures as defined in [CAAdES].

This profile also supports verification and update of advanced signatures by addition of unsigned properties (time-stamps and different types of validation data), as specified in [CAAdES]

5.2.5 Transport Binding

This profile does not specify or constrain the transport binding.

5.2.6 Security Binding

This profile does not specify or constrain the security binding.

5.3 Profile of Signing Protocol

The present profile allows requesting:

- o Predefined forms of advanced electronic signatures as defined in [CAAdES].
- o Other forms of signatures based in [RFC 3852] defined in other specifications,

In both cases, the specific requested form will be identified by an URI.

According to this profile, the following predefined advanced signature forms defined in [CAAdES] MAY be requested: CAAdES-BES, CAAdES-EPES, CAAdES-T, CAAdES-C, CAAdES-X, CAAdES-X-L, and CAAdES-A

In addition, the present profile provides means for requesting incorporation in any of the aforementioned forms any of the following attributes: **signing-time**, **commitment-type-indication**, **signer-attributes**, **signer-location**, **content-hints**, and **content-time-stamp**

Other electronic signature forms based in [RFC 3852], defined elsewhere, MAY also be requested using the mechanisms defined in this profile.

5.3.1 Element <SignRequest>

This clause profiles the `dss:SignRequest` element.

5.3.1.1 Attribute Profile

`urn:oasis:names:tc:dss:1.0:profiles:CAAdES.`

5.3.1.2 Element <OptionalInputs>

5.3.1.2.1 New Optional Inputs

5.3.1.2.1.1 Element <SignatureForm>

Usage of these elements is according to what is stated in 3.3.1.1.1.1.

5.3.1.2.2 Optional Inputs already defined in the Core

None of the optional inputs specified in the [DSS Core] are precluded in this abstract profile. It only constrains some of them and specifies additional optional inputs.

5.3.1.2.2.1 Element <SignatureType>

This element is MANDATORY. Its value MUST be:

urn:ietf:rfc:3369

5.3.1.2.2.2 Optional inputs <ClaimedIdentity> / <KeySelector>

Usage of these elements is according to what is stated in section 3.3.1.1.2.2.

5.3.1.2.2.3 Element <SignedProperties>

This section profiles section 3.3.1.1.2.3.

5.3.1.2.2.3.1 Requesting signing-time

Clients MAY use the URI defined in 3.3.1.1.2.3.1 or alternatively they MAY also use the following one:

urn:oasis:names:tc:dss:1.0:profiles:CAAdES:signing-time

Usage of these elements is according to what is stated in section 3.3.1.1.2.3.1.

5.3.1.2.2.3.2 Requesting commitment-type-indication

Clients MAY use the URI defined in 3.3.1.1.2.3.2 or alternatively they MAY also use the following one:

urn:oasis:names:tc:dss:1.0:profiles:CAAdES:commitment-type-indication

When this optional input is present, the <Value> element MUST contain a <RequestedCommitment> element as defined in section 3.3.1.1.2.3.2 with the <BinaryValue> child containing the base64encoding of **commitment-type-indication** ASN.1 attribute as specified in [CAAdES], DER-encoded.

5.3.1.2.2.3.3 Requesting signer-location

Clients MAY use the URI defined in 3.3.1.1.2.3.3 or alternatively they MAY also use the following one:

urn:oasis:names:tc:dss:1.0:profiles:CAAdES:signer-location

When this optional input is present, the <Value> element MUST contain a <RequestedSignatureProductionPlace> element as defined in section 3.3.1.1.2.3.3 with the <BinaryValue> child containing the base64encoding of **signer-location** ASN.1 attribute as specified in [CAAdES], DER-encoded.

5.3.1.2.2.3.4 Requesting signer-attributes

Clients MAY use the URI defined in 3.3.1.1.2.3.4 or alternatively they MAY also use the following one:

urn:oasis:names:tc:dss:1.0:profiles:CAAdES:signer-attributes

When this optional input is present, the <Value> element MUST contain a <RequestedSignerRole> element as defined in section 3.3.1.1.2.3.4 with the <BinaryValue> child containing the base64encoding of **signer-attributes** ASN.1 attribute as specified in [CAAdES], DER-encoded.

5.3.1.2.2.3.5 Requesting content-time-stamp

Clients MAY use the URI defined in 3.3.1.1.2.3.5 or alternatively they MAY also use the following one:

urn:oasis:names:tc:dss:1.0:profiles:CAAdES:content-time-stamp

Usage of these elements is according to what is stated in section 3.3.1.1.2.3.5

5.3.1.2.2.3.6 Requesting content-hints

Clients MAY use the URI defined in 3.3.1.1.2.3.6 or alternatively they MAY also use the following one:

urn:oasis:names:tc:dss:1.0:profiles:CAAdES:content-hints

When this optional input is present, the <Value> element MUST contain a <RequestedDocsFormat> element as defined in section 3.3.1.1.2.3.6 with the <BinaryValue> child containing the base64 encoding of **content-hints** ASN.1 attribute as specified in [CAAdES], DER-encoded.

5.3.2 Element <SignResponse>

This section profiles the dss:SignResponse element.

5.3.2.1 Element <SignatureObject>

The dss:SignatureObject MUST contain the dss:Base64Signature child with a CMS based signature base-64 encoded.

5.4 Profile of Verifying Protocol

5.4.1 Element <VerifyRequest>

This clause profiles the dss:VerifyRequest element.

5.4.1.1 Attribute Profile

urn:oasis:names:tc:dss:1.0:profiles:CAAdES.

5.4.1.2 Element <OptionalInputs>

5.4.1.2.1 Element <ReturnUpdatedSignature>

Usage of these elements is according to what is stated in section 3.4.1.3.1.

5.4.1.3 Element <SignatureObject>

The dss:SignatureObject element MUST contain the dss:Base64Signature child with a CMS based signature base64 encoded.

5.4.2 Element <VerifyResponse>

This clause profiles the dss:VerifyResponse element.

5.4.2.1 Element <OptionalOutputs>

Usage of these elements is according to what is stated in section 3.5.1.1.

5.4.2.1.1 Element <UpdatedSignature>

- o The content of the dss:UpdatedSignature will be a dss:SignatureObject element with a dss:Base64Signature element with the CMS based signature base64 encoded.

930 **5.5 Profile Bindings**

931 **5.5.1 Transport Bindings**

932 Messages transported in this profile MAY be transported by the HTTP POST Transport
933 Binding and the SOAP 1.2 Transport Binding defined in [DSSCore].

934 **5.5.2 Security Bindings**

935 **5.5.2.1 Security Requirements**

936 This profile MUST use security bindings that:

- 937 o Authenticates the requester to the DSS server
- 938 o Authenticates the DSS server to the DSS client
- 939 o Protects the integrity of a request, response and the association of response to the
940 request.
- 941 o Optionally, protects the confidentiality of a request and response.
- 942 o The following MAY be used to meet these requirements.

943 **5.5.2.2 TLS X.509 Mutual Authentication**

944 This profile is secured using the TLS X.509 Mutual Authentication Binding defined in
945 [DSSCore].

6 XML timestamps in XAdES signatures

XAdES specification [XAdES] defines a placeholder for incorporating XML timestamps within XAdES signatures. As at the time [XAdES] was written no XML timestamps had been specified, no details on their structure and management were included.

The current section provides rules for including XML timestamps into XAdES signatures. For the rest of the present document a XML timestamp is a `dss:Timestamp` element as defined in [DSSCore] section 5.1, incorporating a `ds:Signature` element profiled as indicated in [DSSCore] section 5.1.1.

6.1 Generation and inclusion of XML timestamps

6.1.1 Profile for XAdES timestamp containers

[XAdES] defines the following timestamps containers:

`xades:IndividualDataObjectTimeStamp`, `xades:AllDataObjectTimeStamp`,
`xades:SignatureTimeStamp`, `xades:RefsOnlyTimeStamp`,
`xades:SigAndRefsTimeStamp` and `xades:ArchiveTimeStamp`.

XAdES timestamp containers MAY include more than one XML timestamp.

XAdES timestamp containers including XML timestamps will not use the explicit referencing mechanism (the `xades:Include` element) defined in [XAdES] section 7.1.4.3.1.

The current document defines the structure of XML timestamps that timestamp more than one item in XAdES signatures i.e., all the timestamps defined in XAdES except the signature timestamp, which has already been profiled in [DSSCore] section 3.5.2.2.

6.1.2 XML timestamp within `xades:IndividualDataObjectsTimeStamp`

This timestamp will be included within `xades:IndividualDataObjectsTimeStamp`'s `xades:XMLTimeStamp` child.

This timestamp must be compliant with the profile defined in [DSSCore] section 5.1.1.

In addition, this timestamp MUST include within its `ds:SignedInfo` one or more `ds:Reference` elements that will be built as indicate below.

1. Take all the XAdES signature's `ds:Reference` referencing those data objects designated by `dss:DocsToBeTimestamped`.
2. For each one proceed as indicated below:
 - a. Generate a copy.
 - b. Suppress the `Id` attribute of the copy if present.
 - c. Set the `type` attribute of the copy to the following URI:
`http://uri.etsi.org/01903/#IndividualDataObjectsTimeStamp`.
 - d. Add the copy to the timestamp's `ds:SignedInfo`.

Applications compliant with the present profile MUST dereference all the `ds:Reference` elements within XML timestamp's `ds:SignedInfo` as indicated in [XMLSig]

6.1.3 XML timestamp within `xades:AllDataObjectsTimeStamp`

This timestamp will be included within `xades:AllDataObjectsTimeStamp`'s `xades:XMLTimeStamp` child.

This timestamp must be compliant with the profile defined in [DSSCore] section 5.1.1.

In addition, this timestamp MUST include one `ds:Reference` element without `URI` attribute and the `type` attribute set to the following URI.

`http://uri.etsi.org/01903/#AllDataObjectsTimeStamp`

It MUST NOT have any `ds:Transforms` element.

Applications compliant with the present profile MUST dereference this element by processing, as indicated in [XAdES] section 7.2.9 steps 1 to 3, all the `ds:Reference` elements in XAdES' `ds:SignedInfo`, except the one referencing the `xades:SignedProperties` element.

6.1.4 XML timestamp within `xades:SigAndRefsTimeStamp`

This timestamp will be included within `xades:SigAndRefsTimeStamp`'s `xades:XMLTimeStamp` child.

This timestamp must be compliant with the profile defined in [DSSCore] section 5.1.1.

In addition, this timestamp MUST include one `ds:Reference` element without `URI` attribute and the `type` attribute set to the following URI.

`http://uri.etsi.org/01903/#SigAndRefsTimeStamp`

It MUST NOT have any `ds:Transforms` element.

Applications compliant with the present profile MUST dereference this element by taking the data objects listed in [XAdES] section 7.5.1.1 and process them as indicated there.

6.1.5 XML timestamp within xades:RefsOnlyTimeStamp

This timestamp will be included within xades:RefsOnlyTimeStamp's xades:XMLTimeStamp child.

This timestamp must be compliant with the profile defined in [DSSCore] section 5.1.1.

In addition, this timestamp MUST include one ds:Reference element without URI attribute and the type attribute set to the following URI.

<http://uri.etsi.org/01903/#RefsOnlyTimeStamp>

It MUST NOT have any ds:Transforms element.

Applications compliant with the present profile MUST dereference this element by the data objects listed in [XAdES] section 7.5.2.1 and process them as indicated there.

6.1.6 XML timestamp within xades:ArchiveTimeStamp

This timestamp will be included within xades:ArchiveTimeStamp's xades:XMLTimeStamp child.

This timestamp must be compliant with the profile defined in [DSSCore] section 5.1.1.

In addition, this timestamp MUST include one ds:Reference element without URI attribute and the type attribute set to the following URI.

<http://uri.etsi.org/01903/#ArchiveTimeStamp>

It MUST NOT have any ds:Transforms element.

Applications compliant with the present profile MUST dereference this element by taking the data objects listed in [XAdES] section 7.7.1 and process them as indicated there.

6.2 Verification of XML timestamps

This section specifies the steps to be performed by a server for verifying the XML timestamps present in a XAdES signature.

The steps that the server shall perform for initiating the verification of each XML timestamp within the corresponding container are listed in order below (if any one of them results in failure, then the timestamp token SHOULD be rejected).

1. Extract the timestamp token embedded in the incoming signature.
2. Verify that the verification key and algorithms used conforms to all relevant aspects of the applicable policy. Should this key come within a public certificate, verify that the certificate conforms to all relevant aspects of the applicable policy including algorithm usage, policy OIDs, and time accuracy tolerances.
3. Verify that the aforementioned verification key is consistent with the ds:SignedInfo/SignatureMethod/@Algorithm attribute value.
4. Verify the timestamp token signature in accordance with the rules defined in [XMLDSIG].
5. Verify that the ds:SignedInfo element contains only two ds:Reference elements
6. Verify that one of the ds:Reference elements has its Type attribute set to "urn:oasis:names:tc:dss:1.0:core:schema:XMLTimeStampToken". Take this one and proceed as indicated below:
 - a. Retrieve the referenced data object. Verify that it references a ds:Object element, which in turn envelopes a dss:TSTInfo element.
 - b. Verify that the dss:TSTInfo element has a valid layout as per the present specification.
 - c. Extract the digest value and associated algorithm from its <ds:DigestValue> and <ds:DigestMethod> elements respectively.

1049 d. Recalculate the digest of the retrieved data object as specified by [XMLDSIG]
1050 with the digest algorithm indicated in <ds:DigestMethod>, and compare this
1051 result with the contents of <ds:DigestValue>.

1052 Subsequent sub-sections indicate the steps that the server shall perform for completing the
1053 verification of each XML timestamp.

1054 **6.2.1 Verification of of xades:IndividualDataObjectsTimeStamp** 1055 **including a XML timestamp**

1056 After completing steps 1 to 5 in section 6.2., the server will perform the tasks detailed below
1057 for completing the XML timestamp verification. If any one of them results in failure, then the
1058 timestamp token SHOULD be rejected. For each of the remaining ds:Reference proceed
1059 as indicated below:

- 1060 1. Check that it has been built from one of the ds:Reference elements within XAdES
1061 signature applying the changes mentioned in section 6.1.2
- 1062 2. Dereference and validate it according to the rules stated in [XMLSig].
- 1063 3. Check for coherence in the value of the times indicated in the time-stamp tokens. All the
1064 time instants must be previous to the time when the verification is performed, to the time
1065 indicated within the SigningTime if present, and to the times indicated within the
1066 time-stamp tokens enclosed within all the rest of time-stamp container properties except
1067 other IndividualDataObjectsTimeStamp.
- 1068 4. Set the <dss:Result> element as appropriate.

1069 Minor Error
1070 urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidIndivid
1071 ualDataObjectsTimeStamp MUST be used when the cryptographic signature verification
1072 succeeds but this timestamp verification fails.

1073 **6.2.2 Verification of xades:AllDataObjectsTimeStamp including a** 1074 **XML timestamp**

1075 After completing steps 1 to 5 in section 6.2., the server will perform the steps listed below for
1076 completing the XML timestamp verification. If any one of them results in failure, then the
1077 timestamp token SHOULD be rejected.

- 1078 1. Take the other ds:Reference element and proceed to dereference it as indicated
1079 below:
 - 1080 a. Take the first ds:Reference element within the XAdES signature's
1081 ds:SignedInfo element if and only if the Type attribute doesn't have the value
1082 "<http://uri.etsi.org/01903#SignedProperties>".
 - 1083 b. Process it according to the reference processing model of XMLDSIG.
 - 1084 c. If the result is a node-set, canonicalize it using the algorithm indicated in
1085 CanonicalizationMethod element of the property, if present. If not, the standard
1086 canonicalization method as specified by XMLDSIG must be used.
 - 1087 d. Concatenate the resulting bytes in an octet stream.
 - 1088 e. Repeat steps a) to d) for all the subsequent ds:Reference elements (in their order
1089 of appearance) within the XAdES signature's ds:SignedInfo element if and
1090 only if Type attribute has not the value
1091 "<http://uri.etsi.org/01903#SignedProperties>".
 - 1092 f. Compute the digest of the resulting octet stream using the algorithm indicated in
1093 the time-stamp token and check if it is the same as the digest present there.
- 1094 2. Check for coherence in the value of the times indicated in the time-stamp tokens. All the
1095 time instants must be previous to the time when the verification is performed, to the time
1096 indicated within the SigningTime if present, and to the times indicated within the
1097 time-stamp tokens enclosed within all the rest of time-stamp container properties except
1098 IndividualDataObjectsTimeStamp.

1099 3. Set the <dss:Result> element as appropriate.
1100 Minor Error
1101 urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidAllData
1102 ObjectsTimestamp MUST be used when the cryptographic verification signature succeeds
1103 but this timestamp verification fails.

1104 **6.2.3 Verification of xades:SigAndRefsTimeStamp including a XML** 1105 **timestamp**

1106 After completing steps 1 to 5 in section 6.2, the server will perform the steps listed below for
1107 completing the XML timestamp verification. If any one of them results in failure, then the
1108 timestamp token SHOULD be rejected.

- 1109 1. Check that those elements that, according to [XAdES] MUST be present for being
1110 timestamped by this timestamp, are actually present (see [XAdES] section 7.5.1).
- 1111 2. Take the other ds:Reference element and proceed to dereference it as indicated
1112 below:
 - 1113 a. Take the XAdES elements listed in [XAdES] section 7.5.1.1 in the order indicated
1114 there.
 - 1115 b. Canonicalize them and concatenate the resulting bytes in one octet stream. If the
1116 CanonicalizationMethod element of the property is present, use it for
1117 canonicalizing. Otherwise, use the standard canonicalization method as specified
1118 by [XMLSig].
 - 1119 c. Compute the digest of the resulting octet stream using the algorithm indicated in
1120 the time-stamp token and check if it is the same as the digest present there.
- 1121 3. Check that the time indicated by the timestamp is posterior to the one indicated in the
1122 xades:SigningTime property, and to the times indicated in the timestamps contained
1123 within xades:AllDataObjectsTimeStamp,
1124 xades:IndividualDataObjectsTimeStamp OR xades:SignatureTimeStamp, if
1125 present. They must also be previous to the times indicated in the timestamps enclosed by
1126 any xades:ArchiveTimeStamp present elements

1127 Minor Error
1128 urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidSigAndR
1129 efsTimestamp MUST be used when the cryptographic verification signature succeeds but
1130 this timestamp verification fails.

1131 **6.2.4 Verification of xades:RefsOnlyTimeStamp including a XML** 1132 **timestamp**

1133 After completing steps 1 to 5 in section 6.2, the server will perform the steps listed below for
1134 completing the XML timestamp verification. If any one of them results in failure, then the
1135 timestamp token SHOULD be rejected.

- 1136 1. Check that those elements that, according to [XAdES] MUST be present for being
1137 timestamped by this timestamp, are actually present (see [XAdES] section 7.5.2).
- 1138 2. Take the other ds:Reference element and proceed to dereference it as indicated
1139 below:
 - 1140 a. Take the XAdES elements listed in [XAdES] section 7.5.2.1 in the order indicated
1141 there.
 - 1142 b. Canonicalize them and concatenate the resulting bytes in one octet stream. If the
1143 CanonicalizationMethod element of the property is present, use it for
1144 canonicalizing. Otherwise, use the standard canonicalization method as specified
1145 by [XMLSig].
 - 1146 c. Compute the digest of the resulting octet stream using the algorithm indicated in
1147 the time-stamp token and check if it is the same as the digest present there.

1148 3. Check that the time indicated by the timestamp is posterior to the one indicated in the
1149 `xades:SigningTime` property, and to the times indicated in the timestamps contained
1150 within `xades:AllDataObjectsTimeStamp`,
1151 `xades:IndividualDataObjectsTimeStamp` or `xades:SignatureTimeStamp`, if
1152 present. They must also be previous to the times indicated in the timestamps enclosed by
1153 any `xades:ArchiveTimeStamp` present elements

1154 Minor Error
1155 `urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidRefsOnlyTimeStamp` MUST be used when the cryptographic verification signature succeeds but this
1156 timestamp verification fails.
1157

1158 **6.2.5 Verification of `xades:ArchiveTimeStamp` including a XML** 1159 **timestamp**

1160 After completing steps 1 to 5 in section 6.2, the server will perform the steps listed below for
1161 completing the XML timestamp verification. If any one of them results in failure, then the
1162 timestamp token SHOULD be rejected.

- 1163 1. Check that those elements that, according to [XAdES] MUST be present for being
1164 timestamped by this timestamp, are actually present (see [XAdES] section 7.7.1).
- 1165 2. Take the other `ds:Reference` element and proceed to dereference it as indicated
1166 below:
 - 1167 a. Take the XAdES elements listed in [XAdES] section 7.7.1 in the order indicated
1168 there.
 - 1169 b. Canonicalize them and concatenate the resulting bytes in one octet stream. If the
1170 `CanonicalizationMethod` element of the property is present, use it for
1171 canonicalizing. Otherwise, use the standard canonicalization method as specified
1172 by [XMLSig].
 - 1173 c. Compute the digest of the resulting octet stream using the algorithm indicated in
1174 the time-stamp token and check if it is the same as the digest present there.
- 1175 3. Check that the time indicated by the timestamp is posterior to the one indicated in the
1176 `SigningTime` property, and to the times indicated in the timestamps contained within
1177 `xades:AllDataObjectsTimeStamp`, `xades:IndividualDataObjectsTimeStamp`,
1178 `xades:SignatureTimeStamp` if present, and `xades:RefsOnlyTimeStamp` or
1179 `xades:SigAndRefsTimeStamp`, if present They must also be previous to the times
1180 indicated in the timestamps enclosed by any `xades:ArchiveTimeStamp` that appear
1181 before the one that is being verified

1182 Minor Error
1183 `urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidArchive`
1184 `TimeStamp` MUST be used when the cryptographic verification signature succeeds but this
1185 timestamp verification fails.

7 Identifiers defined in this specification

7.1 Predefined advanced electronic signature forms identifiers

The table below shows the URIs for standard forms of advanced electronic signature:

Advanced signature FORM	URI
XAdES-BES CAdES-BES	urn:oasis:names:tc:dss:1.0:profiles:AdES:forms:BES
XAdES-EPES CAdES-EPES	urn:oasis:names:tc:dss:1.0:profiles:AdES:forms:EPES
XAdES-T CAdES-ES-T	urn:oasis:names:tc:dss:1.0:profiles:AdES:forms:ES-T
XAdES-C CAdES-ES-C	urn:oasis:names:tc:dss:1.0:profiles:AdES:forms:ES-C
XAdES-X CAdES-ES-X	urn:oasis:names:tc:dss:1.0:profiles:AdES:forms:ES-X
XAdES-X-L CAdES--X-L	urn:oasis:names:tc:dss:1.0:profiles:AdES:forms:ES-X-L
XAdES-A CAdES-X-A	urn:oasis:names:tc:dss:1.0:profiles:AdES:forms:ES-A

Table 1.

7.2 Result Identifiers

This profile defines the <ResultMinor> values listed below. All of them indicate that the cryptographic verification of the signature succeeded, and that the verification of the indicated timestamp failed.

urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidIndividualDataObjectsTimestamp

urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidAllDataObjectsTimestamp

urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidSigAndRefsTimestamp

urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidRefsOnlyTimestamp

1205 urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidArchive
1206 Timestamp

A. Acknowledgements

1207

1208 The following individuals have participated in the creation of this specification and are
1209 gratefully acknowledged:

1210 **Participants:**

1211 Nick Pope, Thales eSecurity
1212 Ed Shallow, Universal Post Union
1213 Trevor Perrin, individual

1214