
2 **Advanced Electronic Signature**
3 **Profiles of the OASIS Digital**
4 **Signature Service**

5 **2nd Committee Draft, 12 September, 2006** (WD 09)

6 **Document identifier:**

7 oasis-dss-1.0-profiles-AdES-spec-cd-r2

8 **Location:**

9 <http://docs.oasis-open.org/dss/v1.0/>

10 **Editor:**

11 Juan Carlos Cruellas, *individual* <cruellas@ac.upc.es>

12 **Contributors:**

13 Nick Pope, *individual*

14 Ed Shallow, *Universal Post Union*

15 Trevor Perrin, *individual*

16 Konrad Lanz, *Austria Federal Chancellery* <Konrad.Lanz@iaik.tugraz.at>

17 **Abstract:**

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

23 **Status:**

24 This is a **Public Review Draft** produced by the OASIS Digital Signature Service
25 Technical Committee. Comments may be submitted to the TC by any person by
26 clicking on "Send A Comment" on the TC home page at:

27 http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=dss.

28 For information on whether any patents have been disclosed that may be essential to
29 implementing this specification, and any offers of patent licensing terms, please refer
30 to the Intellectual Property Rights section of the Digital Signature Service TC web
31 page at <http://www.oasis-open.org/committees/dss/ipr.php>.

Table of Contents

33	1 INTRODUCTION	5
34	1.1 NOTATION	5
35	1.2 NAMESPACES	5
36	2 OVERVIEW	6
37	3 ADVANCED ELECTRONIC SIGNATURE ABSTRACT PROFILE.....	7
38	3.1 OVERVIEW	7
39	3.2 PROFILE FEATURES	8
40	3.2.1 <i>Scope</i>	8
41	3.2.2 <i>Relationship To Other Profiles</i>	8
42	3.2.3 <i>Signature Object</i>	8
43	3.3 PROFILE OF SIGNING PROTOCOL.....	8
44	3.3.1 <i>Element <SignRequest></i>	9
45	3.3.1.1 <i>Element <OptionalInputs></i>	9
46	3.3.1.1.1 <i>New Optional Inputs</i>	9
47	3.3.1.1.1.1 <i>Optional Input <SignatureForm></i>	9
48	3.3.1.1.2 <i>Optional Inputs already defined in the Core</i>	9
49	3.3.1.1.2.1 <i>Optional Input <SignatureType></i>	9
50	3.3.1.1.2.2 <i>Optional inputs <ClaimedIdentity> and <KeySelector></i>	10
51	3.3.1.1.2.3 <i>Optional Input <SignedProperties></i>	10
52	3.3.1.1.2.3.1 <i>Requesting SigningTime</i>	10
53	3.3.1.1.2.3.2 <i>Requesting CommitmentTypeIndication</i>	11
54	3.3.1.1.2.3.3 <i>Requesting SignatureProductionPlace</i>	11
55	3.3.1.1.2.3.4 <i>Requesting SignerRole</i>	12
56	3.3.1.1.2.3.5 <i>Requesting AllDataObjectsTimeStamp</i>	12
57	3.3.1.1.2.3.6 <i>Requesting DataObjectFormat</i>	12
58	3.3.2 <i>Element <SignResponse></i>	13
59	3.3.2.1 <i>Element <SignatureObject></i>	13
60	3.3.2.2 <i>Optional Outputs</i>	13
61	3.4 PROFILE OF VERIFYING PROTOCOL	13
62	3.4.1 <i>Element <VerifyRequest></i>	13
63	3.4.1.1 <i>Attribute Profile</i>	13
64	3.4.1.2 <i>Element <SignatureObject></i>	13
65	3.4.1.3 <i>Element <OptionalInputs></i>	13
66	3.4.1.3.1 <i>Element <ReturnUpdatedSignature></i>	13
67	3.5 ELEMENT <VERIFYRESPONSE>	14
68	3.5.1.1 <i>Element <OptionalOutputs></i>	14
69	3.5.1.1.1 <i>Optional Output <UpdatedSignature></i>	14
70	4 XML ADVANCED ELECTRONIC SIGNATURES CONCRETE PROFILE	15
71	4.1 OVERVIEW	15
72	4.2 PROFILE FEATURES.....	15
73	4.2.1 <i>Identifier</i>	15
74	4.2.2 <i>Scope</i>	16
75	4.2.3 <i>Relationship To Other Profiles</i>	16
76	4.2.4 <i>Signature Object</i>	16
77	4.2.5 <i>Transport Binding</i>	16
78	4.2.6 <i>Security Binding</i>	16
79	4.3 PROFILE OF SIGNING PROTOCOL.....	16
80	4.3.1 <i>Attribute Profile</i>	16

81	4.3.2	Element <SignRequest>	17
82	4.3.2.1	Element <OptionalInputs>	17
83	4.3.2.1.1	New Optional Inputs	17
84	4.3.2.1.1.1	Element <SignatureForm>	17
85	4.3.2.1.2	Optional Inputs already defined in the Core	17
86	4.3.2.1.2.1	Optional Input <SignatureType>	17
87	4.3.2.1.2.2	Optional inputs < ClaimedIdentity> and <KeySelector>	17
88	4.3.2.1.2.3	Optional Input <SignedProperties>	17
89	4.3.2.1.2.3.1	Requesting SigningTime	17
90	4.3.2.1.2.3.2	Requesting CommitmentTypeIndication	17
91	4.3.2.1.2.3.3	Requesting SignatureProductionPlace	17
92	4.3.2.1.2.3.4	Requesting SignerRole	18
93	4.3.2.1.2.3.5	Requesting AllDataObjectTimeStamp	18
94	4.3.2.1.2.3.6	Requesting DataObjectFormat	18
95	4.3.2.1.2.3.7	Requesting <xades:IndividualDataObjectTimeStamp>	18
96	4.3.3	Element <SignResponse>	19
97	4.3.3.1	Element <SignatureObject>	19
98	4.4	PROFILE OF VERIFYING PROTOCOL	19
99	4.4.1	Element <VerifyRequest>	19
100	4.4.1.1	Attribute Profile	19
101	4.4.1.2	Element <SignatureObject>	19
102	4.4.1.3	Element <OptionalInputs>	20
103	4.4.1.3.1	Optional Output <ReturnUpdatedSignature>	20
104	4.4.2	Element <VerifyResponse>	20
105	4.4.2.1	Element <OptionalOutputs>	20
106	4.4.2.1.1	Optional Output <UpdatedSignature>	20
107	4.5	PROFILE BINDINGS	20
108	4.5.1	Transport Bindings	20
109	4.5.2	Security Bindings	20
110	4.5.2.1	Security Requirements	20
111	4.5.2.2	TLS X.509 Mutual Authentication	20
112	5	CMS-BASED ADVANCED ELECTRONIC SIGNATURE PROFILE	21
113	5.1	OVERVIEW	21
114	5.2	PROFILE FEATURES	22
115	5.2.1	Identifier	22
116	5.2.2	Scope	22
117	5.2.3	Relationship To Other Profiles	22
118	5.2.4	Signature Object	22
119	5.2.5	Transport Binding	22
120	5.2.6	Security Binding	22
121	5.3	PROFILE OF SIGNING PROTOCOL	22
122	5.3.1	Element <SignRequest>	22
123	5.3.1.1	Attribute Profile	23
124	5.3.1.2	Element <OptionalInputs>	23
125	5.3.1.2.1	New Optional Inputs	23
126	5.3.1.2.1.1	Element <SignatureForm>	23
127	5.3.1.2.2	Optional Inputs already defined in the Core	23
128	5.3.1.2.2.1	Element <SignatureType>	23
129	5.3.1.2.2.2	Optional inputs < ClaimedIdentity> / <KeySelector>	23
130	5.3.1.2.2.3	Element <SignedProperties>	23
131	5.3.1.2.2.3.1	Requesting signing-time	23
132	5.3.1.2.2.3.2	Requesting commitment-type-indication	23
133	5.3.1.2.2.3.3	Requesting signer-location	23
134	5.3.1.2.2.3.4	Requesting signer-attributes	24
135	5.3.1.2.2.3.5	Requesting content-time-stamp	24
136	5.3.1.2.2.3.6	Requesting content-hints	24

137	5.3.2	<i>Element <SignResponse></i>	24
138	5.3.2.1	<i>Element <SignatureObject></i>	24
139	5.4	PROFILE OF VERIFYING PROTOCOL	24
140	5.4.1	<i>Element <VerifyRequest></i>	24
141	5.4.1.1	Attribute Profile	24
142	5.4.1.2	<i>Element <OptionalInputs></i>	25
143	5.4.1.2.1	<i>Element <ReturnUpdatedSignature></i>	25
144	5.4.1.3	<i>Element <SignatureObject></i>	25
145	5.4.2	<i>Element <VerifyResponse></i>	25
146	5.4.2.1	<i>Element <OptionalOutputs></i>	25
147	5.4.2.1.1	<i>Element <UpdatedSignature></i>	25
148	5.5	PROFILE BINDINGS	25
149	5.5.1	<i>Transport Bindings</i>	25
150	5.5.2	<i>Security Bindings</i>	25
151	5.5.2.1	Security Requirements	25
152	5.5.2.2	TLS X.509 Mutual Authentication	25
153	6	XML TIMESTAMPS IN XADES SIGNATURES	26
154	6.1	GENERATION AND INCLUSION OF XML TIMESTAMPS	26
155	6.1.1	<i>Profile for XAdES timestamp containers</i>	26
156	6.1.2	<i>XML timestamp within xades:IndividualDataObjectsTimeStamp</i>	27
157	6.1.3	<i>XML timestamp within xades:AllDataObjectsTimeStamp</i>	27
158	6.1.4	<i>XML timestamp within xades:SigAndRefsTimeStamp</i>	27
159	6.1.5	<i>XML timestamp within xades:RefsOnlyTimeStamp</i>	28
160	6.1.6	<i>XML timestamp within xades:ArchiveTimeStamp</i>	28
161	6.2	VERIFICATION OF XML TIMESTAMPS	28
162	6.2.1	<i>Verification of of xades:IndividuallDataObjectsTimeStamp including a XML timestamp</i> 29	
164	6.2.2	<i>Verification of xades:AllDataObjectsTimeStamp including a XML timestamp</i>	29
165	6.2.3	<i>Verification of xades:SigAndRefsTimeStamp including a XML timestamp</i>	30
166	6.2.4	<i>Verification of xades:RefsOnlyTimeStamp including a XML timestamp</i>	31
167	6.2.5	<i>Verification of xades:ArchiveTimeStamp including a XML timestamp</i>	31
168	7	IDENTIFIERS DEFINED IN THIS SPECIFICATION	33
169	7.1	PREDEFINED ADVANCED ELECTRONIC SIGNATURE FORMS IDENTIFIERS	33
170	7.2	RESULT IDENTIFIERS	33
171	8	REFERENCES	35
172	8.1	NORMATIVE	35
173	APPENDIX A. REVISION HISTORY		36
174	APPENDIX B. NOTICES		38
175			

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 Notation

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 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:

- The prefix **dss:** (or no prefix) stands for the DSS core namespace [Core-XSD].
- The prefix **ds:** stands for the W3C XML Signature namespace [XMLSig].
- 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:

- XML encoded signatures based on [XMLSig] such as specified in [XAdES].
- 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:

- SignRequest. This operation supports requests for:
 - Generating predefined advanced signature forms as defined in [XAdES] and [CAAdES].
 - 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.
 - 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.
- SignResponse. This operation supports delivery of:
 - Predefined advanced signature forms as defined in [XAdES] and [CAAdES].
 - 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.
 - 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:

- VerifyRequest. This operation supports requests for:
 - Verifying a predefined advanced signature form.
 - Verifying XML signatures incorporating specific properties whose combination does not fit any predefined XAdES signature form.
 - 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.

- 256 ○ Verifying CMS signatures incorporating specific attributes whose combination
257 does not fit any predefined [CAAdES] signature form.
- 258 ○ Verifying any of the signatures mentioned above PLUS updating them by
259 addition of additional attributes (time-stamps, validation data, etc) leading to a
260 predefined [CAAdES] form.
- 261 ○ Verifying a long-term advanced signature in a certain point of time.
- 262 ○ VerifyResponse. This operation supports delivery of:
 - 263 ○ Advanced signature verification result of signatures mentioned above.
 - 264 ○ Advanced signature verification result PLUS the updated signatures as
265 requested.
- 266 The material for each operation will clearly indicate the lifecycle phase it pertains to.

267 3.2 Profile Features

268 3.2.1 Scope

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

270 3.2.2 Relationship To Other Profiles

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

274 3.2.3 Signature Object

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

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

279 3.3 Profile of Signing Protocol

280 The present profile allows requesting:

- 281 ○ Predefined forms of advanced electronic signatures as defined in [XAdES] and
282 [CAAdES].
- 283 ○ Other forms of signatures based in [XMLSig] or [RFC 3852] defined in other
284 specifications,

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

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

- 290 ○ CAAdES-BES and XAdES-BES. In this form, the signing certificate is secured by the
291 signature itself.

- 292 ○ CAdES-EPES and XAdES-EPES. This form incorporates an explicit identifier of the
293 signature policy that will govern the signature generation and verification.
- 294 ○ CAdES-ES-T and XAdES-T. This form incorporates a trusted time, by means of a
295 time-stamp token or a time-mark.
- 296 ○ CAdES-ES-C and XAdES-C.
- 297 ○ CAdES-ES-X and XAdES-X.
- 298 ○ CAdES-ES-X-L and XAdES-X-L.
- 299 ○ CAdES-ES-A and XAdES-A.

300 In addition, the present profile provides means for requesting incorporation in any of the
301 aforementioned forms any of the signed properties defined in [XAdES] and signed attributes
302 defined in [CAdES].

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

305 3.3.1 Element <SignRequest>

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

307 3.3.1.1 Element <OptionalInputs>

308 3.3.1.1.1 New Optional Inputs

309 3.3.1.1.1.1 Optional Input <SignatureForm>

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

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

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

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

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

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

321 3.3.1.1.2 Optional Inputs already defined in the Core

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

324 3.3.1.1.2.1 Optional Input <SignatureType>

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

326 `urn:ietf:rfc:3275`

327 for requesting XML-based signatures, or

urn:ietf:rfc:3369

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

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

3.3.1.1.2.2 Optional inputs <ClaimedIdentity> and <KeySelector>

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

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

3.3.1.1.2.3 Optional Input <SignedProperties>

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

Signed properties that MAY be requested are:

XAdES	CAAdES
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

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

3.3.1.1.2.3.1 Requesting SigningTime

Value for <Identifier> element:

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

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

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

355 3.3.1.1.2.3.2 Requesting `CommitmentTypeIndication`

356 Value for `<Identifier>` element:

357 **`urn:oasis:names:tc:dss:1.0:profiles:AdES:CommitmentTypeIndication`**

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

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

```
362 <xs:element name="RequestedCommitment">  
363   <xs:complexType>  
364     <xs:choice>  
365       <xs:element ref="xades:CommitmentTypeIndication"/>  
366       <xs:element name="BinaryValue" type="xs:base64Binary"/>  
367     </xs:choice>  
368   </xs:complexType>  
369 </xs:element>
```

370 Element `<xades:CommitmentTypeIndication>` will be present when requesting a XML
371 signature.

372 Element `<BinaryValue>` will be present when requesting an ASN.1 signature. Its contents
373 MUST be the base64 encoding of `commitment-type-indication` ASN.1 attribute defined
374 in [CADES], DER-encoded

375 3.3.1.1.2.3.3 Requesting `SignatureProductionPlace`

376 Value for `<Identifier>` element:

377 **`urn:oasis:names:tc:dss:1.0:profiles:AdES:SignatureProductionPlace`**

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

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

```
382 <xs:element name="RequestedSignatureProductionPlace">  
383   <xs:complexType>  
384     <xs:choice>  
385       <xs:element ref="xades:SignatureProductionPlace"/>  
386       <xs:element name="BinaryValue" type="xs:base64Binary"/>  
387     </xs:choice>  
388   </xs:complexType>  
389 </xs:element>
```

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

392 Element `<BinaryValue>` will be present when requesting an ASN.1 signature. Its contents
393 MUST be the base64 encoding of `signerLocation` ASN.1 attribute defined in [CADES],
394 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.

```
<xs:element name="RequestedDocsFormat" type="DocsFormatType" />

<xs:complexType name="DocsFormatType">
  <xs:sequence>
    <xs:choice>
      <xs:element name="DocFormat" type="DocFormatType"
maxOccurs="unbounded" />
      <xs:element name="BinaryValue" type="xs:base64Binary"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="DocFormatType">
  <xs:complexContent>
    <xs:extension base="DocReferenceType">
      <xs:sequence>
```

```

441         <xs:element ref="xades:DataObjectFormat"/>
442     </xs:sequence>
443 </xs:extension>
444 </xs:complexContent>
445 </xs:complexType>

```

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

447 Element <BinaryValue> will be present when requesting a CMS based signature. Its
 448 contents MUST be the base64 encoding of **content-hints** ASN.1 attribute defined in [RFC
 449 2634] DER-encoded.

450 3.3.2 Element <SignResponse>

451 This clause profiles the `dss:SignResponse` element.

452 3.3.2.1 Element <SignatureObject>

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

454 3.3.2.2 Optional Outputs

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

457 3.4 Profile of Verifying Protocol

458 3.4.1 Element <VerifyRequest>

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

- 460 ○ Requesting verification of advanced signatures.
- 461 ○ Requesting verification of advanced signatures AND update of signatures to other
 462 predefined forms.

463 3.4.1.1 Attribute Profile

464 The value for the `Profile` attribute, indicating the concrete sub-profile of this abstract profile,
 465 MUST be present.

466 3.4.1.2 Element <SignatureObject>

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

468 3.4.1.3 Element <OptionalInputs>

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

471 3.4.1.3.1 Element <ReturnUpdatedSignature>

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

474 The `Type` attribute identifies the advanced signature form requested.

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

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

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

483 3.5 Element <VerifyResponse>

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

485 3.5.1.1 Element <OptionalOutputs>

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

488 3.5.1.1.1 Optional Output <UpdatedSignature>

489 This element SHALL contain a `dss:SignatureObject` element that SHALL NOT contain a
490 `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:

- SignRequest. This operation supports requests for:
 - Generating predefined advanced signature forms as defined in [XAdES].
 - 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.
- SignResponse. This operation supports delivery of:
 - Predefined advanced signature forms as defined in [XAdES].
 - 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:

- VerifyRequest. This operation supports requests for:
 - Verifying a predefined XAdES signature form.
 - Verifying XML signatures incorporating specific properties whose combination does not fit any predefined XAdES signature form.
 - 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.
 - Verifying a long-term advanced signature in a certain point of time.
- VerifyResponse. This operation supports delivery of:
 - Advanced signature verification result of signatures mentioned above.
 - 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:

- 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.
- 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
```

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

596 4.3.2.1.2.3.4 Requesting SignerRole

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

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

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

603 4.3.2.1.2.3.5 Requesting AllDataObjectTimeStamp

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

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

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

608 4.3.2.1.2.3.6 Requesting DataObjectFormat

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

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

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

615 4.3.2.1.2.3.7 Requesting <xades:IndividualDataObjectTimeStamp>

616 Value for <Identifier> element:

617 **urn:oasis:names:tc:dss:1.0:profiles:XAdES:IndividualDataObjectTimeSta**
618 **mp**

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

```
621 <xs:element name="DocsToBeTimeStamped" type="DocReferencesType" />
622
623 <xs:complexType name="DocReferencesType">
624   <xs:sequence>
625     <xs:element name="DocReference" maxOccurs="unbounded"
626       type="DocReferenceType" />
627   </xs:sequence>
628 </xs:complexType>
629
630 <xs:complexType name="DocReferenceType">
631   <xs:attribute name="WhichDocument" type="xs:IDREF"
632     use="required" />
633   <xs:attribute name="RefId" type="xs:string" use="optional" />
634 </xs:complexType>
```

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

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

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

4.3.3 Element <SignResponse>

This section profiles the dss:SignResponse element.

4.3.3.1 Element <SignatureObject>

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

A ds:Signature element containing a XMLSig based signature.

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

4.4 Profile of Verifying Protocol

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

4.4.1 Element <VerifyRequest>

This clause profiles the dss:VerifyRequest element.

4.4.1.1 Attribute Profile

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

4.4.1.2 Element <SignatureObject>

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

4.4.1.3 Element <OptionalInputs>

4.4.1.3.1 Optional Output <ReturnUpdatedSignature>

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

4.4.2 Element <VerifyResponse>

This clause profiles the `dss:VerifyResponse` element.

4.4.2.1 Element <OptionalOutputs>

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

4.4.2.1.1 Optional Output <UpdatedSignature>

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

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

4.5 Profile Bindings

4.5.1 Transport Bindings

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

4.5.2 Security Bindings

4.5.2.1 Security Requirements

This profile MUST use security bindings that:

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

4.5.2.2 TLS X.509 Mutual Authentication

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

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:

- SignRequest. This operation supports requests for:
 - Generating predefined advanced signature forms as defined in [CAAdES].
 - 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.
- SignResponse. This operation supports delivery of:
 - Predefined advanced signature forms as defined in [CAAdES].
 - 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:

- VerifyRequest. This operation supports requests for:
 - Verifying a predefined [CAAdES] signature form.
 - Verifying CMS signatures incorporating specific attributes whose combination does not fit any predefined [CAAdES] signature form.
 - 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.
 - Verifying a long-term advanced signature in a certain point of time.
- VerifyResponse. This operation supports delivery of:
 - Advanced signature verification result of signatures mentioned above.
 - 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:

- Predefined forms of advanced electronic signatures as defined in [CAAdES].
- 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 base64 encoding 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 base64 encoding 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>

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

5.5 Profile Bindings

5.5.1 Transport Bindings

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

5.5.2 Security Bindings

5.5.2.1 Security Requirements

This profile MUST use security bindings that:

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

5.5.2.2 TLS X.509 Mutual Authentication

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

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

886 6.1.2 XML timestamp within `xades:IndividualDataObjectsTimeStamp`

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

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

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

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

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

902 6.1.3 XML timestamp within `xades:AllDataObjectsTimeStamp`

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

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

906 In addition, this timestamp MUST include one `ds:Reference` element without `URI` attribute
907 and the `type` attribute set to the following URI.
908 `http://uri.etsi.org/01903/#AllDataObjectsTimeStamp`

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

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

914 6.1.4 XML timestamp within `xades:SigAndRefsTimeStamp`

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

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

918 In addition, this timestamp MUST include one `ds:Reference` element without `URI` attribute
919 and the `type` attribute set to the following URI.
920 `http://uri.etsi.org/01903/#SigAndRefsTimeStamp`

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

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

924 6.1.5 XML timestamp within xades:RefsOnlyTimeStamp

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

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

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

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

931 It MUST NOT have any ds:Transforms element.

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

934 6.1.6 XML timestamp within xades:ArchiveTimeStamp

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

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

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

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

941 It MUST NOT have any ds:Transforms element.

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

944 6.2 Verification of XML timestamps

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

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

- 950 1. Extract the timestamp token embedded in the incoming signature.
- 951 2. Verify that the verification key and algorithms used conforms to all relevant aspects of the
952 applicable policy. Should this key come within a public certificate, verify that the certificate
953 conforms to all relevant aspects of the applicable policy including algorithm usage, policy
954 OIDs, and time accuracy tolerances.
- 955 3. Verify that the aforementioned verification key is consistent with the
956 ds:SignedInfo/SignatureMethod/@Algorithm attribute value.
- 957 4. Verify the timestamp token signature in accordance with the rules defined in [XMLDSIG].
- 958 5. Verify that the ds:SignedInfo element contains only two ds:Reference elements
- 959 6. Verify that one of the ds:Reference elements has its Type attribute set to
960 "urn:oasis:names:tc:dss:1.0:core:schema:XMLTimeStampToken". Take this one and
961 proceed as indicated below:

- 962 a. Retrieve the referenced data object. Verify that it references a `ds:Object`
963 element, which in turn envelopes a `dss:TSTInfo` element.
- 964 b. Verify that the `dss:TSTInfo` element has a valid layout as per the present
965 specification.
- 966 c. Extract the digest value and associated algorithm from its `<ds:DigestValue>`
967 and `<ds:DigestMethod>` elements respectively.
- 968 d. Recalculate the digest of the retrieved data object as specified by **[XMLDSIG]**
969 with the digest algorithm indicated in `<ds:DigestMethod>`, and compare this
970 result with the contents of `<ds:DigestValue>`.
- 971 Subsequent sub-sections indicate the steps that the server shall perform for completing the
972 verification of each XML timestamp.

973 6.2.1 Verification of `xades:IndividualDataObjectsTimeStamp` 974 including a XML timestamp

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

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

988 Minor Error

989 `urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidIndividualDataObjectsTimeStamp`
990 MUST be used when the cryptographic signature verification
991 succeeds but this timestamp verification fails.

992 6.2.2 Verification of `xades:AllDataObjectsTimeStamp` including a XML 993 timestamp

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

- 997 1. Take the other `ds:Reference` element and proceed to dereference it as indicated
998 below:
- 999 a. Take the first `ds:Reference` element within the XAdES signature's
1000 `ds:SignedInfo` element if and only if the `Type` attribute doesn't have the value
1001 "<http://uri.etsi.org/01903#SignedProperties>".
- 1002 b. Process it according to the reference processing model of XMLDSIG.

- 1003 c. If the result is a node-set, canonicalize it using the algorithm indicated in
 1004 CanonicalizationMethod element of the property, if present. If not, the standard
 1005 canonicalization method as specified by XMLDSIG must be used.
- 1006 d. Concatenate the resulting bytes in an octet stream.
- 1007 e. Repeat steps a) to d) for all the subsequent ds:Reference elements (in their order
 1008 of appearance) within the XAdES signature's ds:SignedInfo element if and
 1009 only if Type attribute has not the value
 1010 "http://uri.etsi.org/01903#SignedProperties".
- 1011 f. Compute the digest of the resulting octet stream using the algorithm indicated in
 1012 the time-stamp token and check if it is the same as the digest present there.
- 1013 2. Check for coherence in the value of the times indicated in the time-stamp tokens. All the
 1014 time instants must be previous to the time when the verification is performed, to the time
 1015 indicated within the SigningTime if present, and to the times indicated within the
 1016 time-stamp tokens enclosed within all the rest of time-stamp container properties except
 1017 IndividualDataObjectsTimeStamp.
- 1018 3. Set the <dss:Result> element as appropriate.
- 1019 Minor Error
 1020 urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidAllData
 1021 ObjectsTimeStamp MUST be used when the cryptographic verification signature succeeds
 1022 but this timestamp verification fails.

1023 6.2.3 Verification of xades:SigAndRefsTimeStamp including a XML 1024 timestamp

- 1025 After completing steps 1 to 5 in section 6.2, the server will perform the steps listed below for
 1026 completing the XML timestamp verification. If any one of them results in failure, then the
 1027 timestamp token SHOULD be rejected.
- 1028 1. Check that those elements that, according to [XAdES] MUST be present for being
 1029 timestamped by this timestamp, are actually present (see [XAdES] section 7.5.1).
- 1030 2. Take the other ds:Reference element and proceed to dereference it as indicated
 1031 below:
- 1032 a. Take the XAdES elements listed in [XAdES] section 7.5.1.1 in the order indicated
 1033 there.
- 1034 b. Canonicalize them and concatenate the resulting bytes in one octet stream. If the
 1035 CanonicalizationMethod element of the property is present, use it for
 1036 canonicalizing. Otherwise, use the standard canonicalization method as specified
 1037 by [XMLSig].
- 1038 c. Compute the digest of the resulting octet stream using the algorithm indicated in
 1039 the time-stamp token and check if it is the same as the digest present there.
- 1040 3. Check that the time indicated by the timestamp is posterior to the one indicated in the
 1041 xades:SigningTime property, and to the times indicated in the timestamps contained
 1042 within xades:AllDataObjectsTimeStamp,
 1043 xades:IndividualDataObjectsTimeStamp or xades:SignatureTimeStamp, if
 1044 present. They must also be previous to the times indicated in the timestamps enclosed by
 1045 any xades:ArchiveTimeStamp present elements
- 1046 Minor Error
 1047 urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidSigAndR

1048 `efsTimestamp` MUST be used when the cryptographic verification signature succeeds but
1049 this timestamp verification fails.

1050 6.2.4 Verification of `xades:RefsOnlyTimeStamp` including a XML 1051 timestamp

1052 After completing steps 1 to 5 in section 6.2, the server will perform the steps listed below for
1053 completing the XML timestamp verification. If any one of them results in failure, then the
1054 timestamp token SHOULD be rejected.

- 1055 1. Check that those elements that, according to [XAdES] MUST be present for being
1056 timestamped by this timestamp, are actually present (see [XAdES] section 7.5.2).
- 1057 2. Take the other `ds:Reference` element and proceed to dereference it as indicated
1058 below:
 - 1059 a. Take the XAdES elements listed in [XAdES] section 7.5.2.1 in the order indicated
1060 there.
 - 1061 b. Canonicalize them and concatenate the resulting bytes in one octet stream. If the
1062 `CanonicalizationMethod` element of the property is present, use it for
1063 canonicalizing. Otherwise, use the standard canonicalization method as specified
1064 by [XMLSig].
 - 1065 c. Compute the digest of the resulting octet stream using the algorithm indicated in
1066 the time-stamp token and check if it is the same as the digest present there.
- 1067 3. Check that the time indicated by the timestamp is posterior to the one indicated in the
1068 `xades:SigningTime` property, and to the times indicated in the timestamps contained
1069 within `xades:AllDataObjectsTimeStamp`,
1070 `xades:IndividualDataObjectsTimeStamp` or `xades:SignatureTimeStamp`, if
1071 present. They must also be previous to the times indicated in the timestamps enclosed by
1072 any `xades:ArchiveTimeStamp` present elements

1073 Minor Error

1074 `urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidRefsOnl`
1075 `yTimeStamp` MUST be used when the cryptographic verification signature succeeds but this
1076 timestamp verification fails.

1077 6.2.5 Verification of `xades:ArchiveTimeStamp` including a XML 1078 timestamp

1079 After completing steps 1 to 5 in section 6.2, the server will perform the steps listed below for
1080 completing the XML timestamp verification. If any one of them results in failure, then the
1081 timestamp token SHOULD be rejected.

- 1082 1. Check that those elements that, according to [XAdES] MUST be present for being
1083 timestamped by this timestamp, are actually present (see [XAdES] section 7.7.1).
- 1084 2. Take the other `ds:Reference` element and proceed to dereference it as indicated
1085 below:
 - 1086 a. Take the XAdES elements listed in [XAdES] section 7.7.1 in the order indicated
1087 there.
 - 1088 b. Canonicalize them and concatenate the resulting bytes in one octet stream. If the
1089 `CanonicalizationMethod` element of the property is present, use it for
1090 canonicalizing. Otherwise, use the standard canonicalization method as specified
1091 by [XMLSig].

1092 c. 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 3. Check that the time indicated by the timestamp is posterior to the one indicated in the
1095 SigningTime property, and to the times indicated in the timestamps contained within
1096 xades:AllDataObjectsTimeStamp, xades:IndividualDataObjectsTimeStamp,
1097 xades:SignatureTimeStamp if present, and xades:RefsOnlyTimeStamp or
1098 xades:SigAndRefsTimeStamp, if present They must also be previous to the times
1099 indicated in the timestamps enclosed by any xades:ArchiveTimeStamp that appear
1100 before the one that is being verified

1101 Minor Error
1102 urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidArchive
1103 TimeStamp MUST be used when the cryptographic verification signature succeeds but this
1104 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

1119 urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidSigAndR
1120 efsTimestamp
1121 urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidRefsOnl
1122 yTimestamp
1123 urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidArchive
1124 Timestamp

8 References

8.1 Normative

- [AdES-XSD] J.C. Cruellas et al. AdES Profile Schema, OASIS, (MONTH/YEAR TBD)
- [CAAdES] CMS Advanced Electronic Signatures. ETSI TS 101 733.
- [Core-XSD] T. Perrin et al. *DSS Schema*. OASIS, (MONTH/YEAR TBD)
- [DSSCore] T. Perrin et al. *Digital Signature Service Core Protocols and Elements*. OASIS, (MONTH/YEAR TBD)
- [RFC 2119] S. Bradner. Key words for use in RFCs to Indicate Requirement Levels. IETF RFC 2396, August 1998.
- <http://www.ietf.org/rfc/rfc2396.txt>.
- [XAdES] XML Advanced Electronic Signatures. ETSI TS 101 903, February 2002 (shortly to be reissued).
- [XML-ns] T. Bray, D. Hollander, A. Layman. *Namespaces in XML*. W3C Recommendation, January 1999.
- <http://www.w3.org/TR/1999/REC-xml-names-19990114>
- [XMLSig] D. Eastlake et al. *XML-Signature Syntax and Processing*. W3C Recommendation, February 2002.
- <http://www.w3.org/TR/1999/REC-xml-names-19990114>
- [RFC 2634] P. Hoffman (ed.). Enhanced Security Services for S/MIME, June 1999.
- [RFC 3852] Cryptographic Message Syntax (CMS). R. Housley. July 2004.
- [Core-XSD]** T. Perrin et al. *DSS Schema*. OASIS, **(MONTH/YEAR TBD)**

Appendix A. Revision History

Rev	Date	By Whom	What
wd-01	2004-03-08	Juan Carlos Cruellas	Initial, incomplete version: SignRequest for predefined forms + optional properties.
wd-02	2004-03-08	Juan Carlos Cruellas	Second version of the initial version: it incorporates SignRequest-SignResponse and VerifyRequest-VerifyResponse. No capability for requesting individually any property. This is still an on-going discussion.
wd-03	2004-06-18	Juan Carlos Cruellas	Third version. Quite a lot of editorial work done. No capability for requesting individually any property. This is still an on-going discussion.
wd-04	2004-08-09	Juan Carlos Cruellas	Fourth version: Suppressed <UpdateSignatureOnly> element. So far: signature forms identified by URI. Not possibility of requesting properties by enumeration. Solved most of editorial issues. Small editorial changes.
wd-05	2004-10-08	Juan Carlos Cruellas	Fifth version: Addition of two concrete sub-profiles: one for XAdES and the other for TS 101733
wd-06	2004-11-09	Juan Carlos Cruellas	Sixth version: Addition of bindings for concrete profiles. Additional changes from comments raised before voting as a CD
wd-07	2006-03-04	Juan Carlos Cruellas	Seventh version: Additional changes for aligning with

Rev	Date	By Whom	What
			latest core version
wd-08	2006-08	Juan Carlos Cruellas	Eight version: Changes for aligning with core WD 46. Inclusion of specific material on how to deal with XML timestamps
wd-09	2006-09	Juan Carlos Cruellas	Ninth version: Change of the URI namespace for being aligned with the one in the Core. Editorial changes

Appendix B. Notices

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

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

Copyright © OASIS Open 2006. *All Rights Reserved.*

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

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

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