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

Committee Specification

13 February 2007

Specification URIs:

This Version:

http://docs.oasis-open.org/dss/v1.0/oasis-dss-profiles-AdES-cs-v1.0-r1.html

http://docs.oasis-open.org/dss/v1.0/oasis-dss-profiles-AdES-cs-v1.0-r1.pdf

Latest Version:

http://docs.oasis-open.org/dss/v1.0/oasis-dss-profiles-AdES-cs-v1.0-r1.html

http://docs.oasis-open.org/dss/v1.0/oasis-dss-profiles-AdES-cs-v1.0-r1.pdf

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-cs-v1.0-r1

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 OASIS Digital Signature Services TC 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 Committee by using the "Send A Comment" button on the Technical Committee's web page at http://www.oasis-open.org/committees/dss.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (http://www.oasis-open.org/committees/dss/ipr.php.

The non-normative errata page for this specification is located at http://www.oasis-open.org/committees/dss.

Notices

Table of Contents

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:

- 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:

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

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 CAdES 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 [CAdES].
    - 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 [CAdES] 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 [CAdES].
    - 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 [CAdES] 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.

    - Verifying CMS signatures incorporating specific attributes whose combination does not fit any predefined [CAdES] 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 [CAdES] 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.

The material for each operation will clearly indicate the lifecycle phase it pertains to.

3.2 Profile Features

3.2.1 Scope

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

3.2.2 Relationship To Other Profiles

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

3.2.3 Signature Object

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

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

3.3 Profile of Signing Protocol

The present profile allows requesting:

- Predefined forms of advanced electronic signatures as defined in [XAdES] and [CAdES].
- Other forms of signatures based in [XMLSig] or [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 [XAdES] and [CAdES] MAY be requested (those forms whose name begin by XAdES- are forms names for XAdES signatures; those ones whose name begin by CAdES are names for CAdES signatures):

- CAdES-BES and XAdES-BES. In this form, the signing certificate is secured by the signature itself.

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

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

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

3.3.1 Element <SignRequest>

This clause profiles the `dss:SignRequest` element.

3.3.1.1 Element <OptionalInputs>

3.3.1.1.1 New Optional Inputs

3.3.1.1.1.1 Optional Input <SignatureForm>

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

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

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

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

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.

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

3.3.1.1.2 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.

3.3.1.1.2.1 Optional Input <SignatureType>

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

```
urn:ietf:rfc:3275
```

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 [CAdES] 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 CAdES to the signed attributes.

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

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

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

3.3.1.1.2.3.2 Requesting CommitmentTypeIndication

Value for <`Identifier`> element:

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

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

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

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

Element <`xades:CommitmentTypeIndication`> 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 **`commitment-type-indication`** ASN.1 attribute defined in [CAdES], DER-encoded

3.3.1.1.2.3.3 Requesting SignatureProductionPlace

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 [CAdES], 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 [CAdES], 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>
```

```
                <xs:element ref="xades:DataObjectFormat"/>
            </xs: sequence >
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

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

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

3.3.2 Element <SignResponse>

This clause profiles the dss:SignResponse element.

3.3.2.1 Element <SignatureObject>

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

3.3.2.2 Optional Outputs

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

3.4 Profile of Verifying Protocol

3.4.1 Element <VerifyRequest>

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

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

3.4.1.1 Attribute Profile

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

3.4.1.2 Element <SignatureObject>

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 [CAdES] and [XAdES]: XAdES-T/CAdES -T, XAdES-C/CAdES-C, XAdES-X/CAdES-X,XAdES-X-L/CAdES-X-L, XAdES-A/CAdES-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 vaule 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:CommitmentTypeIndicatio
n
```

When this optional input is present, the `<Value>` element MUST contain a `<RequestedCommitment>` element as defined in section in 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:SignatureProductionPlac
e
```

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
```

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

4.3.2.1.2.3.6 Requesting DataObjectFormat

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:XAdES:AllDataObjectsTimeStamp
```

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 one or more `<DocFormat>` children.

4.3.2.1.2.3.7 Requesting <xades:IndividualDataObjectTimeStamp>

Value for `<Identifier>` element:

```
urn:oasis:names:tc:dss:1.0:profiles:XAdES:IndividualDataObjectTimeSta
mp
```

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

```
<xs:element name="DocsToBeTimeStamped" type="DocReferencesType"/>

<xs:complexType name="DocReferencesType">
   <xs:sequence>
      <xs:element name="DocReference" maxOccurs="unbounded"
         type="DocReferenceType"/>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="DocReferenceType">
   <xs:attribute name="WhichDocument" type="xs:IDREF"
      use="required"/>
   <xs:attribute name="RefId" type="xs:string" use="optional"/>
</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:

- o A `ds:Signature` containing a XMLSig based signature.
- o 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:

- o Authenticates the requester to the DSS server
- o Authenticates the DSS server to the DSS client
- o Protects the integrity or a request, response and the association of response to the request.
- o Optionally, protects the confidentiality of a request and response.
- o 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 [CAdES]. It will then provide all the features related to CAdES signatures that are specified in the abstract profile defined in section 3.

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

- SignRequest. This operation supports requests for:
    - Generating predefined advanced signature forms as defined in [CAdES].
    - Generating CMS signatures incorporating specific signed/unsigned attributes whose combination does not fit any predefined [CAdES] 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 [CAdES].
    - CMS signatures incorporating specific signed attributes whose combination does not fit any predefined [CAdES] 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 [CAdES] the following operations apply:

- VerifyRequest. This operation supports requests for:
    - Verifying a predefined [CAdES] signature form.
    - Verifying CMS signatures incorporating specific attributes whose combination does not fit any predefined [CAdES] 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 [CAdES] 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:CAdES.`

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 [CAdES].

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 [CAdES]

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 [CAdES].
- 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 [CAdES] MAY be requested: CAdES-BES, CAdES-EPES, CAdES-T, CAdES-C, CAdES-X, CAdES-X-L, and CAdES-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:CAdES.
```

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:CAdES: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:CAdES: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 [CAdES], 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:CAdES: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 [CAdES], 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:CAdES: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 [CAdES], 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:CAdES: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:CAdES: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 [CAdES], 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:CAdES.

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.

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:

- o Authenticates the requester to the DSS server
- o Authenticates the DSS server to the DSS client
- o Protects the integrity or a request, response and the association of response to the request.
- o Optionally, protects the confidentiality of a request and response.
- o 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.

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 timestmp's ds: `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.
    d. Recalculate the digest of the retrieved data object as specified by **[XMLDSIG]** with the digest algorithm indicated in `<ds:DigestMethod>`, and compare this result with the contents of `<ds:DigestValue>`.

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

6.2.1 Verification of of xades:IndividuallDataObjectsTimeStamp including a XML timestamp

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

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

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

6.2.2 Verification of xades:AllDataObjectsTimeStamp including a XML timestamp

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

1. Take the other `ds:Reference` element and proceed to dereference it as indicated below:
    a. Take the first `ds:Reference` element within the XAdES signature's `ds:SignedInfo` element if and only if the `Type` attribute doesn"t have the value "http://uri.etsi.org/01903#SignedProperties".

b. Process it according to the reference processing model of XMLDSIG.

c. If the result is a node-set, canonicalize it using the algorithm indicated in CanonicalizationMethod element of the property, if present. If not, the standard canonicalization method as specified by XMLDSIG must be used.

d. Concatenate the resulting bytes in an octet stream.

e. Repeat steps a) to d) for all the subsequent ds:Reference elements (in their order of appearance) within the XAdES signature's `ds:SignedInfo` element if and only if Type attribute has not the value "http://uri.etsi.org/01903#SignedProperties".

f. Compute the digest of the resulting octet stream using the algorithm indicated in the time-stamp token and check if it is the same as the digest present there.

2. Check for coherence in the value of the times indicated in the time-stamp tokens. All the time instants must be previous to the time when the verification is performed, to the time indicated within the SigningTime if present, and to the times indicated within the time-stamp tokens enclosed within all the rest of time-stamp container properties except IndividualDataObjectsTimeStamp.
3. Set the `<dss:Result>` element as appropriate.

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

6.2.3 Verification of xades:SigAndRefsTimeStamp including a XML timestamp

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

1. Check that those elements that, according to [XAdES] MUST be present for being timestamped by this timestamp, are actually present (see [XAdES] section 7.5.1).
2. Take the other `ds:Reference` element and proceed to dereference it as indicated below:

   a. Take the XAdES elements listed in [XAdES] section 7.5.1.1 in the order indicated there.

   b. Canonicalize them and concatenate the resulting bytes in one octet stream. If the `CanonicalizationMethod` element of the property is present, use it for canonicalizing. Otherwise, use the standard canonicalization method as specified by [XMLSig].

   c. Compute the digest of the resulting octet stream using the algorithm indicated in the time-stamp token and check if it is the same as the digest present there.
3. Check that the time indicated by the timestamp is posterior to the one indicated in the `xades:SigningTime` property, and to the times indicated in the timestamps contained within `xades:AllDataObjectsTimeStamp`, `xades:IndividualDataObjectsTimeStamp` or `xades:SignatureTimeStamp`, if present. They must also be previous to the times indicated in the timestamps enclosed by any `xades:ArchiveTimeStamp` present elements

Minor Error
`urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidSigAndR`

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

6.2.4 Verification of xades:RefsOnlyTimeStamp including a XML timestamp

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

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

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

6.2.5 Verification of xades:ArchiveTimeStamp including a XML timestamp

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

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

    c. Compute the digest of the resulting octet stream using the algorithm indicated in the time-stamp token and check if it is the same as the digest present there.

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

Minor Error `urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidArchiveTimestamp` MUST be used when the cryptographic verification signature succeeds but this 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`

`urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidArchiveTimestamp`

A. Acknowledgements

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

Participants:

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