BALTIC WPKI FORUM STANDARD **BDOC-1.0:2008**

BDOC: FORMAT FOR DIGITAL SIGNATURES

Revisions and Changes

Version	Date	Changes
0.61	25.12.2007	First public release
0.7	07.03.2008	Element <SigningTime> is made optional Element < SignedDataObjectProperties> is made optional BDOC container format specification is significantly improved: • location and form of signatures is specified • versioning information of container format and XAdES profile is provided
0.9	12.05.2008	"Note on hash algorithm" is introduced in section 5.1
1.0	23.05.2008	Official release

Table of Contents

0. Introduction

The European Directive on a community framework for Electronic Signatures defines an electronic signature as: "data in electronic form which is attached to or logically associated with other electronic data and which serves as a method of authentication".

The present document is intended to cover electronic signatures for various types of transactions, including business transactions (e.g. purchase requisition, contract, and invoice applications). Thus the present document can be used for any transaction between an individual and a company, between two companies, between an individual and a governmental body, etc. The present document is independent of any environment. It can be used with different signature creation devices e.g. smart cards, GSM SIM cards, special programs for electronic signatures, etc.

The ETSI standard TS 101 903 [1] (hereinafter: XAdES) defines formats for advanced electronic signatures that remain valid over long periods, are compliant with the European Directive and incorporate additional useful information in common use cases (like indication of the role or resolution of the signatory). XAdES is XML-based and therefore suitable for the current ICT environment.

The present document:

- specifies profiles of XAdES by narrowing down choices of elements and value types in the standard;
- defines sets of XAdES elements for long-time validity of XAdES signature;
- specifies container format for embedding signed files and signatures.

For the further reference, term BDOC is used thorough the text to denote both XAdES profile and container format.

1. Scope

The present document defines XML formats for advanced electronic signatures that remain valid over long periods, are compliant with the European Directive and incorporate additional useful information in common uses cases. This includes evidence as to its validity even if the signer or verifying party later attempts to deny (repudiates) the validity of the signature.

The present document builds on the following standards:

- ETSI TS 101 903 v1.3.2 – XML Advanced Electronic Signatures (XAdES) [1];
- ITU-T Recommendation X.509 [2];
- RFC 3261 – PKIX Time-Stamp protocol [3];
- RFC 2560 – Online Certificate Status Protocol [4];
- Packaging conventions, part of OpenDocument [5] standard.

For a complete list of references, see section 2.

Section 5 defines basic profile of the BDOC format. This profile contains just signature without any validation data.

Section 6 defines two profiles of the BDOC format with validation data providing for “replacement of handwritten signature”.

Section 7 discusses and defines means for achieving long-time validity of the electronic signatures.

Section 8 specifies container format for embedding signed files and signatures into one data unit.

2. References

[1] ETSI 101 903 V1.3.2 - XML Advanced Electronic Signatures (XAdES)
[2] ITU-T Recommendation X.509: "Information technology - Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks"
[3] IETF RFC 3261: "Internet X.509 Public Key Infrastructure Time-Stamp protocol"
[4] RFC 2560: “X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP”
[5] ISO/IEC 26300:2006 Information technology -- Open Document Format for Office Applications (OpenDocument) v1.0
[6] IETF RFC 3275: “XML-Signature Syntax and Processing”
[7] ETSI TS 102 023 V1.2.1 - Policy requirements for time-stamping authorities

3. Definitions and Abbreviations

See Clause 4 of XAdES[1] for basic definitions and abbreviations.

BDOC – a profile of XAdES and container packaging rules

4. Overview

Whereas XAdES has been around for several years and there are a number of implementations of this standard around, they remain incompatible. The reasons are the following:

- XAdES allows for myriad of options. Implementations of XAdES usually do not support every non-mandatory building block or element which results in incompatibility of XAdES signatures;
- Use of XAdES optional building blocks heavily depends on security requirements of application and PKI services provided. As those requirements and set of services tend to vary, corresponding XAdES profiles do as well.
- XAdES specifies just a signature format allowing the source data (to be signed) be anywhere and referenced by URI. In practice it is often required source data and signatures to be bound together in a single data unit ("container" or "file"). Implementers have free choice here which results in incompatibility of *digitally signed files*.

This specification solves abovementioned problems by:

- defining subset of XAdES elements and parameters – "BDOC profile [of XAdES]";
- defining requirement profiles for PKI, time-stamping and certificate validation services and corresponding XAdES building blocks;
- defining container format for embedding source data and signatures – "BDOC file format".

The rest of the document relies entirely on XAdES [1] standard document and therefore is not self-consistent. The reader shall use XAdES standard as a basis and follow references and profiling notes given in this document. The OpenDocument [5] standard shall be consulted for packaging rules defined in section 8 of this document.

5. BDOC Basic Profile

The BDOC Basic Profile is an XML structure containing a single cryptographic signature over the well-defined set of data. It does not contain any validation data for full signature validation such as timestamps or certificate validity confirmations. It just forms basis for other forms of BDOC described in next chapter.

The BDOC Basic profile bases on Basic Electronic Signature – XAdES-BES – defined in clause 4.4.1 of XAdES [1].

Following notions will be used in further text for defining requirements for usage of the elements:

Notion	***Creating application***	***Processing application***
M (Mandatory)	Must produce this element	Must process this element
C (Critical)	May produce this element	Must process this element if present
O (Optional)	May produce this element	May process this element if present
N/A	Element is not used	Element is not used

Next, we profile `xades:Cert` element for use in both in BDOC Basic Profile and BDOC extended forms described in next chapter.

5.1. Cert element

The `xades:Cert` element follows the definition of clause 7.2.2 of XAdES and is type of `CertIDListType`. Optional parameter `URI` is not used.

Parameter `Algorithm` in `DigestMethod` element shall have value
`http://www.w3.org/2000/09/xmldsig#sha1`

Note on hash algorithm. SHA-256 or better is strongly recommended instead of SHA-1 when creating BDOC documents. However, due to technical difficulties it is not always possible to use anything else besides SHA-1. As a consequence, the BDOC-compatible application must be capable of handling SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512 algorithms in verification and make best efforts to create BDOC with SHA-256 or better.

In summary, the `Cert` structure contains the following:

```
Cert
  CertDigest
    DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
    DigestValue               -- hash value of the certificate
  IssuerSerial
```

```
X509IssuerName              -- Distinguished Name parsed
                            -- according to RFC2253
X509SerialNumber            -- serial number of certificate
```

5.2. Definition

The BDOC Basic Profile structure consists of:

- `ds:SignedInfo` block containing references (with hash values) to data objects to be signed
- `ds:SignatureValue` element containing the actual signature value
- `ds:KeyInfo` structure containing signer's certificate

Following restrictions apply for elements of `ds:SignedInfo` block:

Element		***Parameter***	***Comment***
Signature	M	Id	For example "S0"
SignedInfo	M		
CanonicalizationMethod	M	Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"	
SignatureMethod	M	Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"	See "note on hash algorithm" in section 5.1
Reference	M		See below

The `ds:SignedInfo` block contains two or more `ds:Reference` structures pointing to data objects to be signed:

- one reference to every original data object to be signed. In this case element `ds:Reference` must have parameter `URI` referring to the data object (e.g. "`#D0`"). Note, that URI can point to *external* data object as well.
- just one reference to `SignedProperties` data block described below. In that case element `ds:Reference` must have parameter `Type=`http://uri.etsi.org/01903/#SignedProperties and parameter `URI` referring to the SignedProperties block (e.g. "`#S0-SignedProperties`")

Element `Transforms` in `ds:Reference` structure is not created but must be treated as *Critical* if present. Element `ds:DigestMethod` shall have value

`http://www.w3.org/2000/09/xmldsig#sha1`

See "note on hash algorithm" in section 5.1.

This specification does not mandate a separate `ds:Reference` element for `ds:KeyInfo` as the element `SigningCertificate` is mandatory by this specification and thus the signing certificate is secured by the signature.

Only signer's certificate shall be present in `ds:KeyInfo` element. The following profiles usage of the `ds:KeyInfo` element as of clause 4.4 of XMLDSIG [6]:

```
KeyInfo
  KeyValue
    RSAKeyValue
      Modulus               -- value of modulus
      Exponent              -- value of exponent
  X509Data
    X509Certificate         -- certificate in BASE64 encoding
```

XAdES[1] defines a mechanism for encapsulating additional parameters into signature by using `ds:Object` method. The `SignedProperties` element referenced in the `ds:SignedInfo` block is encapsulated in the `ds:Object` element as described in clause 6.2.1 of XAdES.

The following profiles use of the qualified properties of the BDOC Basic Profile:

Element		***XAdES clause***	***Comment***
QualifyingProperties	M	6.2	Must have parameter `Target` pointing to `ds:Signature` element (e.g. "`#S0`")
SignedProperties	M	6.2.1	Must have parameter `Id` (e.g. "`S0-SignedProperties`")
SignedSignatureProperties	M	6.2.3	
SigningTime	O	7.2.1	Zulu time zone shall be used
SigningCertificate	M	7.2.2	See section 5.1 of current document
SignaturePolicyIdentifier	N/A	7.2.3	
SignatureProductionPlace	C	7.2.7	
SignerRole	C	7.2.8	The `ClaimedRoles` element is allowed, `CertifiedRoles` is not supported.
SignedDataObjectProperties	O	6.2.4	
DataObjectFormat	C	7.2.5	As a rule, this element is not created. When present, the application shall ensure data objects are shown to the user in format claimed here
CommitmentTypeIndication	N/A	7.2.6	Use ClaimedRoles element for indication of commitment or resolution in free text.
AllDataObjectsTimeStamp	N/A	7.2.9	
IndividualDataObjectsTimeStamp	N/A	7.2.10	
UnsignedProperties	M	6.2.2	See Section 6
UnsignedSignatureProperties	M	6.2.5	See Section 6
CounterSignature	N/A	7.2.4	

6. Qualified BDOC signatures

BDOC Basic Profile does not contain necessary elements to verify whether the signer's certificate was valid at the (claimed) time of signing. XAdES Basic Profile form may be used in internal systems where there are other (undocumented) means for dealing with the certificate validity issue.

The validation data shall be obtained as soon as possible after creation of the XAdES-BES signature. Two scenarios are applicable here:

- in desktop environment – the signer's application itself shall obtain validation data as soon as possible after signature creation;
- in web environment – the server-side application shall obtain validation data as soon as possible after receiving signature from the user.

As the signature creation is an off-line act and timing of thereof cannot be identified in trusted manner, this specification relies on notion that "signature creation time" shall be derived from the time information of validation and/or time-stamping services. In other words, signature shall not be considered *complete* or *valid* without validation data from external services.

This BDOC specification defines two methods for creating electronic signatures equal to "handwritten signature". Both profiles are compliant to XAdES-X-L (see XAdES[1] Annex B.2) form and are providing means for including certificate validity and trusted time-stamp data with the signature:

- **time-marking**. In this scenario, the OCSP responder shall follow specific service requirements described in section 6.1. In this case, additional time-stamp service is not required
- **time-stamping**. This shall be used in case OCSP is not replacing need for additional trusted time-stamps from external Time-Stamping Authority. Refer to XAdES[1] clause 4.4.3.1.

Software implementations complying with current BDOC specification must support both abovementioned methods.

Both supported forms are using the following profile of XAdES elements from "C" and "L" blocks. Usage of time-stamps ("T" and "X" blocks) differ.

Element		***XAdES clause***	***Comment***
SignatureTimeStamp		7.3	See paragraphs 6.1 and 6.2 below
CompleteCertificateRefs	M	7.4.1	
CertRefs	M	7.4.1	Contains pointer to CA certificate in form defined in clause 5.1 of current document
CompleteRevocationRefs	M	7.4.2	Only references to OCSP information are supported
OCSPRef	M	7.4.2	
OCSPIdentifier	M	7.4.2	Parameter `URI` is required pointing to encapsulated OCSP response (e.g. "`#N0`")
ResponderID	M	7.4.2	Distinguished Name parsed according to RFC2253
ProducedAt	M	7.4.2	Zulu timezone shall be used
DigestAndValue	M	7.4.2	At least SHA256 from 01.01.2009
AttributeCertificateRefs	N/A	7.4.3	
AttributeRevocationRefs	N/A	7.4.4	
SigAndRefsTimeStamp		7.5.1	See paragraphs 6.1 and 6.2 below
RefsOnlyTimeStamp	N/A	7.5.2	
CertificateValues	M	7.6.1	`EncapsulatedX509Certificate` is supported only. Must contain OCSP responder certificate and signer's CA certificate. In case time-stamping in used (as defined in 6.2), the TSA certificate shall also be included here.
RevocationValues	M	7.6.2	`OCSPValues` and `EncapsulatedOCSPValue` elements are supported only
AttrAuthoritiesCertValues	N/A	7.6.3	
AttributeRevocationValues	N/A	7.6.4	
ArchiveTimeStamp		7.7	See paragraph 7 below
UnsignedDataObjectProperties	N/A	6.2.6	

6.1. BDOC with time-marks

As the XAdES specification defines, “a time-mark provided by a Trusted Service would have similar effect to the time-stamp property but in this case no property is added to the electronic signature as it is the responsibility of the TSP to provide evidence of a time mark when required to do so.”

This BDOC specification defines a mechanism for time-marking through using OCSP [4] protocol. Immediately after signature creation, the digital signing application shall obtain a certificate validity confirmation using OCSP protocol. Hash value of the signature shall be present in the “nonce” field of the OCSP request. The OCSP responder shall return this “nonce” value within the signed response.

This mechanism solves the complexity of time-stamping and certificate validity relations in one step by combining these two services. The OCSP response described above shall be treated as a validity confirmation telling “at the time I saw this signature, corresponding certificate was valid”, digitally signed by the service. As a result no additional timestamps are required and elements `SignatureTimeStamp` and `SigAndRefsTimeStamp` are not used.

OCSP service shall be “real-time” reflecting current validity of the certificate (not based on CRL). The service shall follow principles documented in ETSI standard “Policy requirements for time-stamping authorities” [7].

The value of `ProducedAt` in the OCSP response shall be treated as a time of signature creation.

6.2. BDOC with time-stamps

BDOC profile with time-stamps is used in case OCSP service does not correspond to requirements described in paragraph 6.1. In this case, additional time-stamps are required in order to fix time of certificate validity information.

This is accomplished by including two time-stamp elements - `SignatureTimeStamp` and `SigAndRefsTimeStamp` to the signature structure. The first time-stamp is obtained as soon as possible after signature creation and the latter is obtained after receiving OCSP response.

This specification does not set verification policy requirements regarding acceptable time intervals between different signature elements (claimed signing time in `SigningTime`, time in `SignatureTimeStamp`, time in `ProducedAt` field OCSP response, time in `SigAndRefsTimeStamp`).

Time-stamps in XAdES are defined in clause 7.1.4 of XAdES[1] allowing utterly flexible implementations. This BDOC specification profiles time-stamps as follows:

- Only IETF-style (RCF3161) time-stamps are supported i.e. `EncapsulatedTimeStamp` is supported only
- Time-stamping must use *explicit* mechanism for referring to data objects to be time-stamped using `Include` element specified in clause 7.1.4.3.1 of XAdES[1]. This means that `ReferenceInfo` method is not used.
- In the `Include` element, `URI` is supported, `referencedData` is not supported.
- Attribute `Id` is mandatory

Those rules apply to all time-stamps used in this specification.

Time value in the element `SigAndRefsTimeStamp` shall be treated as a time of signature creation.

7. Mechanisms for long-time validity

Qualified BDOC signature specified in last section is secure enough provided that cryptographic algorithms used are unbreakable, key lengths are sufficient and private keys of CSP (the CA and OCSP key) remain under control of the service provider.

Fast advances in computing suggest that key lengths and algorithms used today may not be secure enough in the future. There is also always a (theoretical) possibility that private key of some PKI service can get corrupted.

Additional measures are needed to protect electronic signatures from threats like those. This document describes two mechanisms for maintaining long-time validity of electronic signatures:

- **Logging:** service providing external evidence of certificate validity at the time of signing creates and maintains log containing issued responses
- **Re-time-stamping:** the whole material of the signature is periodically re-time-stamped

The first option does not require any end-user activity or additional functionality from BDOC-compliant system and therefore is preferred method. From the other hand the logging puts additional requirements to the service provider which may not be followed. In order to fully secure end-user and give him some additional independence of service provider, the re-time-stamping mechanism shall also be supported.

7.1. Logging

This mechanism builds on purpose of preserving evidence that “the signer’s certificate was valid at the time of signing” for a long time.

Depending on the Qualified BDOC method used, corresponding services must create log of all issued responses:

- in case of time-marking (section 6.1), the OCSP service
- in case of time-stamping (section 6.2), the time-stamping service

The log entry shall be created **before** issuing a response. If the log entry creation fails, error response shall be produced by the service. This principle ensures existence of the every response given.

The service provider shall provide a public interface for verification of existence of certain response in the log.

For further fortification of the logging mechanism, additional security measures may be implemented:

- Cryptographic linking: every log record is dependent on the previous. This can be accomplished by creating a hash chain making every log record dependent on all others. This prevents from threats of deleting or interjecting falsified log records.
- Periodical publication of latest log record in printed media. This mechanism provides for a kind of non-repudiation property for the service provider – it takes away all possibilities to forge the log after publishing as the published log record represents the whole log. Of course, it is usable only when cryptographic linking is used.

Extreme care shall be taken on proper maintenance and back-up of the logs.

7.2. Re-time-stamping

This mechanism builds on notion "let's secure what may be weak". Successive time-stamps protect the whole material against vulnerable hashing algorithms or the breaking of the cryptographic material or algorithms.

It should be noted, that time-stamping is commonly a user-initiated act. In case digitally signed files are all scattered over user's computers (or even on external carriers), it might be tricky (or even impossible) to make users to time-stamp existing documents at proper time. However, re-time-stamping may be usable in case digitally signed files are stored in some central repository.

The BDOC re-time-stamping profile follows clause 7.7 of XAdES[1]. "Not distributed case" is supported only as defined in 7.7.1 of XAdES[1]. The `ArchiveTimeStamp` must be profiled according to paragraph 6.2 of this specification.

8. Container Format

This section describes container format for packaging signed original files and signatures into one data unit (file). In other words, it defines what “digitally signed file” is.

The BDOC file format bases on OpenDocument[5] standard, the packaging convention is described in section 17 of the standard. OpenDocument foresees ZIP container with file structure inside. The following OpenDocument requirements must be followed in BDOC file format:

1. **mimetype file**. The file “mimetype” shall be present and formed as described in section 17.4 of the OpenDocument[5] standard. The content must be:

```
application/vnd.bdoc-[C_Version]
```

The `[C_Version]` reflects version of BDOC packaging convention (described in this Section) and by current document shall have a value 1.0.

2. **manifest file**. The file “manifest.xml” shall be present in directory META-INF/ and contain list of all directories and files with their types present in the container as described in section 17.7 of the OpenDocument[5] standard.

Root element shall be the same type as in the “mimetype” file.

Signature files shall have MIME type reflecting signature profile version and profile:

```
signature/bdoc-[S_Version]/[Profile]
```

The `[S_Version]` reflects version of BDOC profile of XAdES (described in Sections 5..7) and by current document shall have value 1.0.

Allowed profiles for the value in `[Profile]` are:

BES – Basic BDOC signature (Section 5)
TM – Qualified BDOC signature with time-marks (Section 6.1)
TS – Qualified BDOC signature with time-stamps (Section 6.2)
TM-A – Qualified BDOC signature with time-marks and archive time-stamp(s)
TS-A – Qualified BDOC signature with time-stamps and archive time-stamp(s)

Digital signatures are stored in the META-INF folder, one file per signature. The names of these files shall contain the term "signature".

Other features of OpenDocument packaging convention are not supported.

As a rule, all files in BDOC container are signed except file “mimetype” and files in the META-INF directory. However, signed objects are explicitly referred by `<Reference>` elements in the signatures (which may even to refer to external objects) and therefore BDOC-compatible applications shall not base on this foresaid rule.

The file extension of BDOC file format is “.bdoc”, MIME-type is “application/bdoc”.

Appendix: Sample BDOC

The following sample BDOC file contains single embedded data file, one signature and has created with time-stamps.

1. BDOC file structure

```
document.doc
mimetype
META-INF/manifest.xml
META-INF/signature1.xml
```

2. Content of file “mimetype”

```
application/vnd.bdoc-1.0
```

3. Content of file “META-INF/manifest.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE manifest:manifest PUBLIC "-//OpenOffice.org//DTD
Manifest 1.0//EN" "Manifest.dtd">
<manifest:manifest
xmlns:manifest="urn:oasis:names:tc:opendocument:xmlns:manifest:1.
0">

<manifest:file-entry manifest:media-type="application/vnd.bdoc-
1.0" manifest:full-path="/" />
<manifest:file-entry manifest:media-type="application/msword"
     manifest:full-path="document.doc" />
<manifest:file-entry manifest:media-type="signature/bdoc-1.0/TS"
     manifest:full-path="META-INF/signature1.xml" />
</manifest:manifest>
```

4. Content of file “META-INF/signature1.xml”

```
<?xml version="1.0" encoding="UTF-8"?>
<Signature Id="S0" xmlns="http://www.w3.org/2000/09/xmldsig#">
 <SignedInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
  <CanonicalizationMethod
Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
20010315"></CanonicalizationMethod>
  <SignatureMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-
sha1"></SignatureMethod>
  <Reference URI="/document.doc">
   <DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></DigestMethod
>
```

```
   <DigestValue>3TioJCtSJ+Fb9c5ECs++QlJW3xE=</DigestValue>
  </Reference>
  <Reference Type="http://uri.etsi.org/01903/#SignedProperties"
URI="#S0-SignedProperties">
   <DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></DigestMethod
>
   <DigestValue>hAsmLsq50o6di4he1sirCa/7tbQ=</DigestValue>
 </Reference>
 </SignedInfo>
 <SignatureValue Id="S0-SIG">

sgpQHbuCY55+f/clXcOrX3f3kBOcCj5stLwtVlRqAEgVFPjKuH3fjmZQspCC28cO
  ...
  EL+hFK0EOEEd4MNYUTPc9aMfy/QPYp15Qs1Uy6ddpcU=
 </SignatureValue>
 <KeyInfo>
  <KeyValue>
   <RSAKeyValue>

<Modulus>uVyoN5mbzWAKNOKIvXp1lnMEWThhf/gVblyjwiTquaAhyGeVGO7A+u0F
4HZtjCKj

6P0TxEtTvwQZPuPvM8aaMg3QXmSo+owDj7johtTZIaQ7oqCxYTIo7gLhC7gS/yaO
     Bf288Vy0CuEr5Da8bWvZRwVDmaDRmNDZsN6dnvK2mbM=</Modulus>
    <Exponent>ngqL</Exponent>
   </RSAKeyValue>
  </KeyValue>
  <X509Data>
   <X509Certificate>

MIID+jCCAuKgAwIBAgIERVl1yDANBgkqhkiG9w0BAQUFADB8MRgwFgYJKoZIhvcN
    ...
    5Xs6XA1pN4sO7uW7dhM=</X509Certificate>
  </X509Data>
 </KeyInfo>
 <Object>
  <QualifyingProperties xmlns="http://uri.etsi.org/01903/v1.3.2#"
Target="#S0">
   <SignedProperties Id="S0-SignedProperties">
    <SignedSignatureProperties>
     <SigningTime>2007-12-25T14:06:01Z</SigningTime>
     <SigningCertificate>
      <Cert>
       <CertDigest>
        <DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></DigestMethod
>
        <DigestValue>bEfrE3Lr0rbGroosrjEfx8klDCA=</DigestValue>
       </CertDigest>
       <IssuerSerial>
```

```
        <X509IssuerName
xmlns="http://www.w3.org/2000/09/xmldsig#">emailAddress=pki@sk.ee
,C=EE,O=AS Sertifitseerimiskeskus,OU=ESTEID,SN=1,CN=ESTEID-
SK</X509IssuerName>
        <X509SerialNumber
xmlns="http://www.w3.org/2000/09/xmldsig#">1163490760</X509Serial
Number>
       </IssuerSerial>
      </Cert>
     </SigningCertificate>
     <SignatureProductionPlace>
      <City>Tallinn</City>
      <StateOrProvince>Harju</StateOrProvince>
      <PostalCode>10122</PostalCode>
      <CountryName>Estonia</CountryName>
     </SignatureProductionPlace>
     <SignerRole>
      <ClaimedRoles>
       <ClaimedRole>Agreed</ClaimedRole>
      </ClaimedRoles>
     </SignerRole>
    </SignedSignatureProperties>
    <SignedDataObjectProperties>
    </SignedDataObjectProperties>
   </SignedProperties>
   <UnsignedProperties
xmlns="http://uri.etsi.org/01903/v1.3.2#">
    <UnsignedSignatureProperties>
     <SignatureTimeStamp Id="S0-T0">
      <Include URI="#S0-SIG"></Include>
      <EncapsulatedTimeStamp>
MIIMYzADAgEAMIIMWgYJKoZIhvcNAQcCoIIMSzCCDEcCAQMxCzAJBgUrDgMCGgUA
       ...
       ZSQAy4ewaA==
      </EncapsulatedTimeStamp>
     </SignatureTimeStamp>
     <CompleteCertificateRefs>
      <CertRefs>
       <Cert>
        <CertDigest>
         <DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
         <DigestValue>gkBzQxlGGptYR4pniNVJEGsgDio=</DigestValue>
        </CertDigest>
        <IssuerSerial>
         <X509IssuerName
xmlns="http://www.w3.org/2000/09/xmldsig#">C=EE,O=AS
Sertifitseerimiskeskus,CN=Juur-
SK,emailAddress=pki@sk.ee</X509IssuerName>
         <X509SerialNumber
xmlns="http://www.w3.org/2000/09/xmldsig#">1110287047</X509Serial
Number>
```

```
        </IssuerSerial>
       </Cert>
      </CertRefs>
     </CompleteCertificateRefs>
     <CompleteRevocationRefs>
      <OCSPRefs>
       <OCSPRef>
        <OCSPIdentifier URI="#N0">
         <ResponderID>
          <ByName>C=EE,O=ESTEID,OU=OCSP,CN=ESTEID-SK 2007 OCSP
RESPONDER,emailAddress=pki@sk.ee</ByName>
         </ResponderID>
         <ProducedAt>2007-12-25T14:06:09Z</ProducedAt>
        </OCSPIdentifier>
        <DigestAlgAndValue>
         <DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></DigestMethod
>
         <DigestValue>yjUvWvsB+llg0n+Tl3ccmCo/geQ=</DigestValue>
        </DigestAlgAndValue>
       </OCSPRef>
      </OCSPRefs>
     </CompleteRevocationRefs>
     <SigAndRefsTimeStamp Id="S0-T1">
      <Include URI="#S0-SIG"></Include>
      <Include URI="#S0-T0"></Include>
      <Include URI="#S0-CERTREFS"></Include>
      <Include URI="#S0-REVOCREFS"></Include>
      <EncapsulatedTimeStamp>
MIIMZDADAgEAMIIMWwYJKoZIhvcNAQcCoIIMTDCCDEgCAQMxCzAJBgUrDgMCGgUA
       ...
       ooaIpdYL8Ds=
      </EncapsulatedTimeStamp>
     </SigAndRefsTimeStamp>
     <CertificateValues>
      <EncapsulatedX509Certificate Id="S0-CA-CERT">
MIIDPDCCAiSgAwIBAgIEQi2iwTANBgkqhkiG9w0BAQUFADB8MRgwFgYJKoZIhvcN
       ...
       EWyMVkNnZooWHIjLpNucQA==
      </EncapsulatedX509Certificate>
      <EncapsulatedX509Certificate Id="S0-RESPONDER_CERT">
MIIEITCCAwmgAwIBAgIBDDANBgkqhkiG9w0BAQUFADCBgDELMAkGA1UEBhMCSUUx
       ...
       EWyMVkNnZooWHIjLpNucQA==
      </EncapsulatedX509Certificate>
      <EncapsulatedX509Certificate Id="S0-TSA_CERT">
MIIDKDCCAhCgAwIBAgIEQi1s4zANBgkqhkiG9w0BAQUFADBfMQswCQYDVQQGEwJF
       ...
       EWyMVkNnZooWHIjLpNucQA==
      </EncapsulatedX509Certificate>
     </CertificateValues>
     <RevocationValues>
```

```
      <OCSPValues>
       <EncapsulatedOCSPValue Id="N0">
MIIBtgoBAKCCAa8wggGrBgkrBgEFBQcwAQEEggGcMIIBmDCCAQGhcTBvMQswCQYD
        ...
        knf8XDhdklVD0w==
       </EncapsulatedOCSPValue>
      </OCSPValues>
     </RevocationValues>
    </UnsignedSignatureProperties>
   </UnsignedProperties>
  </QualifyingProperties>
 </Object>
</Signature>
```