Draft ETSI TS 119 172-3 V0.0.3 (2019-07)

Electronic Signatures and Infrastructures (ESI); Signature Policies; Part 3: ASN.1 format for signature policies

STABLE DRAFT

Send comments ONLY to E-SIGNATURES_COMMENTS@list.etsi.org

by 31 AUGUST 2019

Download the template for comments:

https://docbox.etsi.org/ESI/Open/Latest_Drafts/Template-for-comments.doc

CAUTION: This DRAFT document is provided for information and is for future development work within the ETSI Technical Committee ESI only. ETSI and its Members accept no liability for any further use/implementation of this Specification. Approved and published specifications and reports shall be obtained exclusively via the ETSI Documentation Service at http://www.etsi.org/standards-search

Reference
DTS/ESI-0019172-3

Keywords
ASN.1, electronic signature, e-commerce,
trust services

ETSI

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

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

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

Contents

<PAGE BREAK>

Intellectual Property Rights

Essential patents

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

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

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

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

The present document is part 3 of a multi-part deliverable. Full details of the entire series can be found in part 1 [i.3].

Modal verbs terminology

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

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

Executive summary

TBC

1 Scope

The present document defines an ASN.1 format of machine readable signature policies based on the building blocks and table of contents for human readable signature policy documents as described in ETSI TS 119 172-1 [i.3].

The present document defines elements which can be used to describe signature creation constraints, signature validation constraints and signature augmentation constraints.

For each elment of the machine readable signature policy, the present document references to the semantic described in ETSI TS 119 172-2 [3] and defines the corresponding ASN.1 syntax.

Pure signature applicability rules are out of the scope of the present document.

NOTE: The first version of an ASN.1 format for signature polcies was described in ETSI TR 102 272 [i.1]. That document was written before the framework for standardization of signatures was described. The present document tries to use elements from ETSI TR 102 272 [i.1] where possible to reduce the implementation effort when changing to the present specification.

2 References

2.1 Normative references

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

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

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

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

[1] ETSI EN 219 122-1:"Electronic Signatures and Infrastructures (ESI); CAdES digital signatures; Part 1: Building blocks and CAdES baseline signatures".

[2] ETSI TS 119 612: "Electronic Signatures and Infrastructures (ESI); Trusted Lists".

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

[4] Recommendation ITU-T X.680 (2008): "Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation".

[5] Recommendation ITU-T X.690 (2008): "Information technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)".

[6] IETF RFC 5646: "Tags for Identifying Languages".

[7] IETF RFC 5912 (2010): "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)".

2.2 Informative references

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

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

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

[i.1] ETSI TR 102 272: " Electronic Signatures and Infrastructures (ESI); ASN.1 format for signature policies".

[i.2] ETSI TS 119 102-1 (V1.2.1): "Electronic Signatures and Infrastructures (ESI); Procedures for Creation and Validation of AdES Digital Signatures; Part 1: Creation and Validation".

[i.3] ETSI TS 119 172-1:"Electronic Signatures and Infrastructures (ESI); Signature Policies; Part 1: Building blocks and table of contents for human readable signature policy documents".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in TS 119 172-1 [i.3] and the following] apply:

signature applicability rules: set of rules, applicable to one or more digital signatures, that defines the requirements for determination of whether a signature is fit for a particular business or legal purpose

signature augmentation constraint: criteria used when augmenting a digital signature

signature augmentation policy: set of signature augmentation constraints

signature creation application: application within the signature creation system that creates the AdES digital signature and relies on the signature creation device to create a digital signature value

signature creation constraint: criteria used when creating a digital signature

signature creation policy: set of **signature creation constraints** processed or to be processed by the signature creation application

signature validation application: application that validates a signature against a signature validation policy, and that outputs a status indication (i.e. the signature validation status) and a signature validation report

signature validation policy: set of signature validation constraints processed or to be processed by the signature validation applicaiton

signature validation constraint: technical criteria against which a digital signature can be validated, e.g. as specified in ETSI TS 119 102-1 [i.2]

3.2 Symbols

3.3 Abbreviations

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

BSP Business Scoping parameter

4 ASN.1 syntax for machine processable signature policy document

4.1 Introduction

4.1.1 Technical approach

4.1.2 ASN.1 module

The present document defines two moduls. Annex B.1 describes the elements which were taken over from ETSI TR 102 272 [i.1], Annex B.2 describes the elments first defined in the present document.

4.1.3 ASN.1 encoding

4.1.3.1 DER

Distinguished Encoding Rules (DER) for ASN.1 types shall be as specified in Recommendation ITU-T X.690 [5].

4.1.3.2 BER

If Basic Encoding Rules (BER) are used for some ASN.1 types, it shall be as specified in Recommendation ITU-T X.690 [5].

4.2 The `SignaturePolicy` type

4.2.1 Semantics

The semantic shall be as in clause 4.2.1 of ETSI TS 119 172-2 [3].

4.2.2 Syntax

The `SignaturePolicy` type shall be as defined in annex B.2 and is copied below for information.

```
SignaturePolicy ::= SEQUENCE {
   digest Digest,
   policyComponents PolicyComponents}
```

The element of type `Digest` shall be as specified in clause 4.3.2.

The element of type `PolicyComponents` shall be as specified in clause 4.4.2.

4.3 The `Digest` type

4.3.1 Semantics

The semantic shall be as in clause 4.3.1 of ETSI TS 119 172-2 [3].

4.3.2 Syntax

The `Digest` type shall be as defined in annex B.2 and is copied below for information.

```
Digest ::= OtherHashAlgAndValue
```

The type `OtherHashAlgAndValue` shall be as specified in ETSI EN 319 122-1 [1].

An element of type `OtherHashAlgAndValue` contains a `hashValue` element and a `hashAlgorithm` element. The `hashValue` element shall contain the result of applying the algorithm identified in the corresponding `hashAlgorithm` element on the binaray encoding of the `PolicyComponents` element specified in clause 4.4.2.

EDITOR'S NOTE: Is this clear enough?

4.4 The `PolicyComponents` type

4.4.1 Semantics

The semantic shall be as in clause 4.4.1 of ETSI TS 119 172-2 [3].

4.4.2 Syntax

The `PolicyComponents` type shall be as defined in annex B.2 and is copied below for information.

```
PolicyComponents ::= SEQUENCE {
   geneneralDetails  GeneralDetails,
   otherPolicies [0] OtherPolicies OPTIONAL,
   policyRules PolicyRules }
```

The elment of type `GeneralDetails` shall be as specified in clause 4.5.2.

The elment of type `OtherPolicies` shall be as specified in clause 4.15.2.

The element of type `PolicyRules` shall be as specified in clause 4.16.2.

4.5 The `GeneralDetails` type

4.5.1 Semantics

The semantic shall be as in clause 4.5.1 of ETSI TS 119 172-2 [3].

4.5.2 Syntax

The `GeneralDetails` type shall be as defined in annex B.2 and is copied below for information.

```
GeneralDetails ::= SEQUENCE {
   sigPolicyDetails  SigPolicyDetails,
   authorityDetails AuthorityDetails,
   otherDetails OtherDetails OPTIONAL }
```

The element of type `SigPolicyDetails` shall be as specified in clause 4.6.2.

The element of type `AuthorityDetails` shall be as specified in clause 4.8.2.

The elment of type `OtherDetails` shall be as specified in clause 4.14.2.

4.6 The `SigPolicyDetails` type

4.6.1 Semantics

The semantic shall be as in clause 4.6.1 of ETSI TS 119 172-2 [3].

4.6.2 Syntax

The `SigPolicyDetails` type shall be as defined in annex B.2 and is copied below for information.

```
GeneralDetails ::= SEQUENCE {
   policyIdentifier OBJECT IDENTIFIER,
   policyName InternationalNames,
   distributionPoints [0] DistributionPoints OPTIONAL,
   versionInfo [1] VersionInfo OPTIONAL }

InternationalNames :: = SEQUENCE OF MultiLangText

MultiLangText:: = SEQUENCE {
    lang PrintableString)),
    text UTF8String }

DistributionPoints :: = SEQUENCE (SIZE 1..MAX) OF IA5String
```

The `policyIdentifier` element shall contain an object-identifier that uniquely identifies the signature policy.

The `policyName` element shall contain one or more names of the signature policy. Each name shall be of type `MultiLangText` and qualified with an indication of a language.

The `lang` element shall contain a tag conformant to IETF RFC 5646 [6] and in lower case, that identifies the language in which the content of the `text` element is expressed.

The `distributionPoints` child element shall contain one or more URIs where the signature policy can be reached.

The element of type `VersionInfo` shall be as specified in clause 4.7.2.

4.7 The `VersionInfo` type

4.7.1 Semantics

The semantic shall be as in clause 4.7.1 of ETSI TS 119 172-2 [3].

4.7.2 Syntax

The `VersionInfo` type shall be as defined in annex B.2 and is copied below for information.

```
VersionInfo ::= SEQUENCE {
   thisVersion UTF8String,
   previousVersions PreviousVersions OPTIONAL }

PreviousVersions :: = SEQUENCE OF PreviousVersion

PreviousVersion ::= SEQUENCE {
    version UTF8String,
    distributionPoints  DistributionPoints OPTIONAL }
```

The `thisVersion` elment shall contain the version of the signature policy.

The element of type `PreviousVersions` shall contain versioning information of previous versions of this signature policy.

The `version` element shall contain the version value of the identified previous version.

The `distributionPoints` element shall be defined as in clause 4.6.2 and shall contain the distribution points of the previous version.

4.8 The `AuthorityDetails` type

4.8.1 Semantics

The semantic shall be as in clause 4.8.1 of ETSI TS 119 172-2 [3].

4.8.2 Syntax

The AuthorityDetails type shall be as defined in annex B.2 and is copied below for information.

```
AuthorityDetails ::= SEQUENCE {
    name [0] Name OPTIONAL,
    tradeName [1] TradeName OPTIONAL,
    postalAddresses PostalAddresses,
    electronicAddresses ElectronicAddresses,
    contactPersons [2] ContactPersons OPTIONAL}
```

The element of type Name shall be as specified in clause 4.9.2.

The element of type TradeName shall be as specified in clause 4.10.2.

The element of type PostalAddresses shall be as specified in clause 4.11.2.

The element of type ElectronicAddresses shall be as specified in clause 4.12.2.

The element of type ContactPersons shall be as specified in clause 4.13.2.

4.9 The Name type

4.9.1 Semantics

The semantic shall be as in clause 4.9.1 of ETSI TS 119 172-2 [3].

4.9.2 Syntax

The Name type shall be as defined in annex B.2 and is copied below for information.

```
Name ::= InternationalNames
```

The element of type InternationalNames shall be as specified in clause 4.6.2.

4.10 The TradeName type

4.10.1 Semantics

The semantic shall be as in clause 4.10.1 of ETSI TS 119 172-2 [3].

4.10.2 Syntax

The TradeName type shall be as defined in annex B.2 and is copied below for information.

```
TradeName::= InternationalNames
```

The type InternationalNames shall be as specified in clause 4.6.2.

4.11 The PostalAddresses type

4.11.1 Semantics

The semantic shall be as in clause 4.11.1 of ETSI TS 119 172-2 [3].

4.11.2 Syntax

The PostalAddresses type shall be as defined in annex B.2 and is copied below for information.

```
PostalAddresses ::= SEQUENCE OF PostalAddress
```

The type `PostalAddress` shall be as specified in ETSI EN 319 122-1 [1].

EDITOR'S NOTE: Should we be aligned with the XML / TSL definition or should we use postalAddress as specified in CAdES:

```
PostalAddress ::= SEQUENCE SIZE(1..6) OF DirectoryString{maxSize}
-- maxSize parametrization as specified in X.683
```

4.12 The `ElectronicAddresses` type

4.12.1 Semantics

The semantic shall be as in clause 4.12.1 of ETSI TS 119 172-2 [3].

4.12.2 Syntax

The `ElectronicAddresses` type shall be as defined in annex B.2 and is copied below for information.

```
ElectronicAddresses::= SEQUENCE OF ElectronicAddresse
ElectronicAddresse ::= MultiLangURI
MultiLangURI :: = SEQUENCE {
    lang PrintableString,
    uri IA5String }
```

An element of type `ElectronicAddresses` shall containe one or more elements of type `ElectronicAddresse` and qualified with an indication of a language.

The `lang` element shall contain a tag conformant to IETF RFC 5646 [6] and in lower case, that identifies the language in which the content pointed-to by the URI contained in the `uri` element is expressed.

4.13 The `ContactPersons` type

4.13.1 Semantics

The semantic shall be as in clause 4.13.1 of ETSI TS 119 172-2 [3].

4.13.2 Syntax

The `ContactPersons` type shall be as defined in annex B.2 and is copied below for information.

```
ContactPersons ::= SEQUENCE OF ContactPerson
ContactPerson ::= SEQUENCE {
    Name UTF8String,
    electronicAddresses ElectronicAddresses,
    phoneNumbers SEQUENCE OF PrintableString}
```

An element of type `ElectronicAddresses` shall be as specified in clause 4.12.2.

Each `PrintableString` value in the `phoneNumbers` element shall contain exactly one phone number of the contact peron.

4.14 The `OtherDetails` type

4.14.1 Semantics

The semantic shall be as in clause 4.14.1 of ETSI TS 119 172-2 [3].

4.14.2 Syntax

The `OtherDetails` type shall be as defined in annex B.2 and is copied below for information.

```
OtherDetails ::= SEQUENCE {
    dateOfIssue GeneralizedTime,
    signingPeriod [0] SigningPeriod,
    others [1] SEQUENCE OF OtherDetails}

OtherDetails ::= SEQUENCE {
    otherDetailsId OTHER-DETAILS.&id,
    details OTHER-DETAILS.&Details OPTIONAL
}

OTHER-DETAILS ::= CLASS {
    &id OBJECT IDENTIFIER UNIQUE,
    &Details OPTIONAL }
WITH SYNTAX {
    OTHER-DETAILS-ID &id
    [DETAILS-TYPE & Details] }
```

The elements of type `GeneralizedTime` shall be as specified in Recommendation ITU-T X.680 [4].

The `SigningPeriod` type shall be as defined in annex B.1 and is copied below for information.

```
SigningPeriod ::= SEQUENCE {
    notBefore   GeneralizedTime,
    notAfter    GeneralizedTime OPTIONAL }
```

4.15 The `OtherPolicies` type

4.15.1 Semantics

The semantic shall be as in clause 4.15.1 of ETSI TS 119 172-2 [3].

4.15.2 Syntax

The `OtherPolicies` type shall be as defined in annex B.2 and is copied below for information.

```
OtherPolicies ::= SEQUENCE OF SigPolicyDetails
```

The elements of type `SigPolicyDetails` shall be as specified in clause 4.6.2.

4.16 The `PolicyRules` type

4.16.1 Semantics

The semantic shall be as in clause 4.16.1 of ETSI TS 119 172-2 [3].

4.16.2 Syntax

The `PolicyRules` type shall be as defined in annex B.2 and is copied below for information.

```
PolicyRules  ::= SEQUENCE OF PolicyRuleWithScope

PolicyRuleWithScope ::= SEQUENCE {
    rule   SigPolicyRule,
    scope  SigPolicyScope OPTIONAL}

SigPolicyRule ::= CHOICE {
    commitmentRules            [0]   CommitmentRules,
    basicRule                  [1]   BasicRule }

BasicRule ::= CHOICE {
    dataToBeSignedRules        [0]   DataToBeSignedRules,
    sigToDTBSRelationRules     [1]   SigToDTBSRelationRules,
    dTBSCardinality            [2]   DTBSCardinality,
    sigDTBSRelativePositions   [3]   SigDTBSRelativePositions,
    sigFormatAndLevel          [4]   SigFormatAndLevel,
    augmentationRules          [5]   AugmentationRules,
    signingCertTrustConditions [6]   SigningCertTrustConditions,
    timeEvidLoARules           [7]   TimeEvidLoARules,
    roleTrustCondition         [8]   RoleTrustCondition,
    signatureAttributesRules   [9]   SignatureAttributesRules,
    sCDLoARules                [10]  SCDLoARules,
    cryptoSuitesRules          [11]  CryptoSuitesRules,
    otherRules                 [12]  OtherRules
}


SigPolicyScope  ::= ENUMERATED {
    generation      (0),
    validation      (1),
    Augmentation    (2) }
```

EDITOR'S NOTE: Due to the different ways of defining things in XML and ASN.1 the structure is a bit different. Is this OK like this or are there any better ideas.

The element of type `CommitmentRules` shall be as specified in clause 4.17.2.

The element of type `DataToBeSignedRules` shall be as specified in clause 4.18.2.

The element of type `SigToDTBSRelationRules` shall be as specified in clause 4.19.2.

The element of type `DTBSCardinality` shall be as specified in clause 4.20.2.

The element of type `SigDTBSRelativePositions` shall be as specified in clause 4.21.2.

The element of type `SigFormatAndLevel` shall be as specified in clause 4.22.2.

The element of type `AugmentationRules` shall be as specified in clause 4.23.2.

The element of type `SigningCertTrustConditions` shall be as specified in clause 4.26.2.

The element of type `TimeEvidLoARules` shall be as specified in clause 4.27.2.

The element of type `RoleTrustCondition` shall be as specified in clause 4.28.2.

The element of type `SignatureAttributesRules` shall be as specified in clause 4.29.2.

The element of type `SCDLoARules` shall be as specified in clause 4.30.2.

The element of type `CryptoSuitesRules` shall be as specified in clause 4.31.2.

The element of type `OtherRules` shall be as specified in clause 4.32.2.

4.17 The `CommitmentRules` type

4.17.1 Semantics

The semantic shall be as in clause 4.17 of ETSI TS 119 172-2 [3].

4.17.2 Syntax

The `CommitmentRules` type shall be as defined in annex B.2 and is copied below for information.

```
CommitmentRules ::= SEQUENCE OF CommitmentRule

CommitmentRule ::= SEQUENCE {
    commitmentIdentifier            CommitmentTypeIdentifier
    matchingIndicator               MatchingIndicator,
    basicRules                      SEQUENCE OF BasicRule}

MatchingIndicator ::= ENUMERATED {
    All       (0),
    None      (1),
    atLeastOne(2) }
```

EDITOR'S NOTE: Due to the different ways of defining things in XML and ASN.1 the structure is a bit different. Is this OK like this or are there any better ideas.

The element of type `CommitmentTypeIdentifier` shall be as specified in ETSI EN 319 122 [1].

The element of type `DataToBeSignedRules` shall be as specified in clause 4.18.2.

The element of type `SigToDTBSRelationRules` shall be as specified in clause 4.19.2.

The element of type `DTBSCardinality` shall be as specified in clause 4.20.2.

The element of type `SigDTBSRelativePositions` shall be as specified in clause 4.21.2.

The element of type `SigFormatAndLevel` shall be as specified in clause 4.22.2.

The element of type `AugmentationRules` shall be as specified in clause 4.23.2.

The element of type `SigningCertTrustConditions` shall be as specified in clause 4.26.2.

The element of type `TimeEvidLoARules` shall be as specified in clause 4.27.2.

The element of type `RoleTrustCondition` shall be as specified in clause 4.28.2.

The element of type `SignatureAttributesRules` shall be as specified in clause 4.29.2.

The element of type `SCDLoARules` shall be as specified in clause 4.30.2.

The element of type `CryptoSuitesRules` shall be as specified in clause 4.31.2.

The element of type `OtherRules` shall be as specified in clause 4.32.2.

4.18 The `DataToBeSignedRules` type

4.18.1 Semantics

The semantic shall be as in clause 4.18.1 of ETSI TS 119 172-2 [3].

4.18.2 Syntax

The `DataToBeSignedRules` type shall be as defined in annex B.2 and is copied below for information.

```
DataToBeSignedRules ::= SEQUENCE OF DataToBeSignedRule

DataToBeSignedRule ::= SEQUENCE {
    anyOfMimeType  [0] SEQUENCE OF UTF8String OPTIONAL,
    noneOfMimeType [1] SEQUENCE OF UTF8String OPTIONAL }
```

EDITOR'S NOTE: The structure is not completely aligned with part 2. Might rediscuss the structure.

4.19 The SigToDTBSRelationRules type

4.19.1 Semantics

The semantic shall be as in clause 4.19.1 of ETSI TS 119 172-2 [3].

4.19.2 Syntax

The SigToDTBSRelationRules type shall be as defined in annex B.2 and is copied below for information.

```
SigToDTBSRelationRules::= SEQUENCE {
    dTBSCardinality            DTBSCardinality,
    sigDTBSRelativePosition [0] SigDTBSRelativePosition OPTIONAL,
    sigFormatsAndLevels     [1] SigFormatsAndLevels OPTIONAL }
```

The element of type DTBSCardinality shall be as specified in clause 4.20.2.

The element of type SigDTBSRelativePosition shall be as specified in clause 4.21.2.

The element of type SigFormatsAndLevels shall be as specified in clause 4.22.2.

4.20 The DTBSCardinality type

4.20.1 Semantics

The semantic shall be as in clause 4.20.1 of ETSI TS 119 172-2 [3].

4.20.2 Syntax

The DTBSCardinality type shall be as defined in annex B.2 and is copied below for information.

```
DTBSCardinality::= SEQUENCE {
    maxDTBSNumber [0] MaxDTBSNumber OPTIONAL,
    minDTBSNumber [1] MinDTBSNumber OPTIONAL }

MaxDTBSNumber ::= SEQUENCE {
    dTBSNumber INTEGER,
    maxValueQualifier MaxValueQualifier }

MaxValueQualifier ::= ENUMERATED {
    lessThan       (0),
    lessOrEqualTo  (1),
    equal          (2)  }

MinDTBSNumber ::= SEQUENCE {
    dTBSNumber INTEGER,
    minValueQualifier MinValueQualifier }


MinValueQualifier ::= ENUMERATED {
    higherThan      (0),
    higherOrEqualTo (1),
    equal           (2)  }
```

4.21 The SigDTBSRelativePosition type

4.21.1 Semantics

The semantic shall be as in clause 4.22.1 of ETSI TS 119 172-2 [3].

4.21.2 Syntax

The `SigDTBSRelativePosition` type shall be as defined in annex B.2 and is copied below for information.

```
SigDTBSRelativePosition::= ENUMERATED {
    envelopingSig              (0),
    envelopedSig               (1),
    detachedSig                (2),
    envelopingAndEnvelopedSig  (3),
    envelopingAndDetachedSig   (4),
    envelopedAndDetachedSig    (5),
    aSiC                       (6),
    all                        (7) }
```

4.22 The `SigFormatsAndLevels` type

4.22.1 Semantics

The semantic shall be as in clause 4.22.1 of ETSI TS 119 172-2 [3].

4.22.2 Syntax

The `SigFormatsAndLevels` type shall be as defined in annex B.2 and is copied below for information.

```
SigFormatAndLevels ::= SEQUENCE {
    sigFormats SEQUENCE OF IA5String,
    sigLevels SEQUENCE OF SigLevel }

SigLevel ::= SEQUENCE {
    level IA5String,
    version IA5String }
```

4.23 The `AugmentationRules` type

4.23.1 Semantics

The semantic shall be as in clause 4.23.1 of ETSI TS 119 172-2 [3].

4.23.2 Syntax

The `DTBSCardinality` type shall be as defined in annex B.2 and is copied below for information.

```
AugmentationRules ::= SEQUENCE {
    previousValidationRequired BOOLEAN,
    adESSigLevel AdESSigLevel,
    augQualifier AugmentationQualifier }

AugmentationQualifier ::= ENUMERATED {
    thisLevel     (0),
    minLevel      (1),
    maxLevel      (2)  }
```

The element of type `AdESSigLevel` shall be as specified in clause 4.22.2.

4.24 Types for defining constraints on certificates' trust and certificates revocation status

4.24.1 Introduction

The present clause defines four types:

1) `TrustAnchorsList` which defines the trust anchors.

2) `NameConstraints` which defines constraints on the names of entities.

3) `PolicyConstraints` for defining constraints on certificate policies.

4) `CertificateTrustTrees` which defines constraints on the trust conditions required to certificates.

4.24.2 The `TrustAnchors` type

4.24.2.1 Semantics

The semantic shall be as in clause 4.24.2.1 of ETSI TS 119 172-2 [3].

4.24.2.2 Syntax

The `TrustAnchors` type shall be as defined in annex B.2 and is copied below for information.

```
TrustAnchors ::= SEQUENCE OF TrustAchnor
TrustAnchor ::= CHOICE  {
    Certificate CertAndReliableTime,
    trustedList URIAndReliableTime,
    trustStatusList URIAndReliableTime }

CertAndReliableTime ::= SEQUENCE {
    cert Certificate,
    reliableUntil GeneralizedTime OPTIONAL }

URIAndReliableTime ::= SEQUENCE {
    uri IA5String,
    reliableUntil GeneralizedTime OPTIONAL }
```

The elment of type `Certificate` shall be as specified in IETF RFC 5912 [7].

4.24.3 The `NameConstraints` type

4.24.3.1 Semantics

The semantic shall be as in clause 4.24.3.1 of ETSI TS 119 172-2 [3].

4.24.3.2 Syntax

The `NameConstraints` type shall be as defined in annex B.1 and is copied below for information.

```
NameConstraints ::= SEQUENCE {
          permittedSubtrees       [0]     GeneralSubtrees OPTIONAL,
          excludedSubtrees        [1]     GeneralSubtrees OPTIONAL }

      GeneralSubtrees ::= SEQUENCE SIZE (1..MAX) OF GeneralSubtree

      GeneralSubtree ::= SEQUENCE {
          base                    GeneralName,
          minimum         [0]     BaseDistance DEFAULT 0,
          maximum         [1]     BaseDistance OPTIONAL }

      BaseDistance ::= INTEGER (0..MAX)
```

The element of type `GeneralName` shall be as specified in IETF RFC 5912 [7].

4.24.4 The PolicyConstraints type

4.24.4.1 Semantics

The semantic shall be as in clause 4.24.4.1 of ETSI TS 119 172-2 [3].

4.24.4.2 Syntax

The PolicyConstraints type shall be as defined in annex B.1 and is copied below for information.

```
PolicyConstraints ::= SEQUENCE {
        requireExplicitPolicy           [0] SkipCerts OPTIONAL,
        inhibitPolicyMapping            [1] SkipCerts OPTIONAL }

SkipCerts ::= INTEGER (0..MAX)
```

4.24.5 The CertificateTrustTrees type

4.24.5.1 Semantics

The semantic shall be as in clause 4.24.5.1 of ETSI TS 119 172-2 [3].

4.24.5.2 Syntax

The CertificateTrustTrees and the UseCertPath type shall be as defined in annex B.2 and is copied below for information.

The PathLenConstraint, the AcceptablePolicySet and the CertPolicyId type shall be as defined in annex B.1 and is copied below for information.

```
CertificateTrustTrees ::=   SEQUENCE OF CertificateTrustPoint

CertificateTrustPoint ::= SEQUENCE {
    trustAnchors        TrustAnchors,
    pathLenConstraint   [0] PathLenConstraint   OPTIONAL,
    acceptablePolicySet [1] AcceptablePolicySet OPTIONAL,
    nameConstraints     [2] NameConstraints     OPTIONAL,
    policyConstraints   [3] PolicyConstraints   OPTIONAL,
    userCertPath        [4] UserCertPath        OPTIONAL }


PathLenConstraint    ::=   INTEGER (0..MAX)

AcceptablePolicySet ::= SEQUENCE OF CertPolicyId

CertPolicyId ::= OBJECT IDENTIFIER

UseCertPath ::= CHOICE {
    asInSignature BOOLEAN,
    path SEQUENCE OF Certifciate }
```

The element of type TrustAnchors shall be as specified in clause 4.24.2.2.

The element of type NameConstraints shall be as specified in clause 4.24.3.2.

The element of type PolicyConstraints shall be as specified in clause 4.24.4.2.

4.25 Types for defining constraints on certificates' revocation status

4.25.1 Introduction

The present clause defines twp types:

1) `CertificateRevReq` which defines constraints on the certificate revocation checks procedures.

2) `CertificateRevStatus` which defines constraints on the trust conditions required on the certificates' revocation data.

4.25.2 The `CertificateRevReq` type

4.25.2.1 Semantics

The semantic shall be as in clause 4.25.2.1 of ETSI TS 119 172-2 [3].

4.25.2.2 Syntax

The `CertificateRevReq` type shall be as defined in annex B.2 and is copied below for information.

The `EnuRevReq` type shall be as defined in annex B.1 and is copied below for information.

```
CertificateRevReq ::= SEQUENCE {
    endCertRevReq   EnuRevReq  ,
    caCerts         EnuRevReq }

EnuRevReq  ::= ENUMERATED {
    clrCheck    (0),
    ocspCheck   (1),
    bothCheck   (2),
    eitherCheck (3),
    noCheck     (4),
    other       (5)}
```

4.25.3 The `CertificateRevTrust` type

4.25.3.1 Semantics

The semantic shall be as in clause 4.25.3.1 of ETSI TS 119 172-2 [3].

4.25.3.2 Syntax

The `CertificateRevTrust` type shall be as defined in annex B.2 and is copied below for information.

```
CertificateRevTrust ::= SEQUENCE
    certificateRevReq CertificateRevReq,
    freshness         [0] Freshness OPTIONAL,
    SigCertIssuedByCAKeepsExpiredRevokedCertsInfo [1] BOOLEAN OPTIONAL }

Freshness ::= CHOICE {
    maxDifferenceRevocationAndValidation GeneralizedTime,
    timeAfterSignature                   GeneralizedTime
```

The elment of type `CertificateRevReq` shall be as specified in clause 4.25.2.2.

If not present, the default value for `SigCertIssuedByCAKeepsExpiredRevokedCertsInfo` is false.

4.26 The SigningCertTrustCondition type

4.26.1 Semantics

The semantic shall be as in clause 4.26.1 of ETSI TS 119 172-2 [3].

4.26.2 Syntax

The SigningCertTrustCondition type shall be as defined in annex B.2 and is copied below for information.

```
SigningCertTrustCondition::=
    signerTrustTrees CertificateTrustTrees,
    signerRevTrust   CertificateRevTrust }
```

The elment of type CertificateTrustTrees shall be as specified in clause 4.24.5.2.

The element of type CertificateRevTrust shall be as specified in clause 4.25.3.2.

4.27 The TimeEvidencesRules type

4.27.1 Semantics

The semantic shall be as in clause 4.27.1 of ETSI TS 119 172-2 [3].

4.27.2 Syntax

The TimeEvidLoARules type shall be as defined in annex B.2 and is copied below for information.

The DeltaTime type shall be as defined in annex **Error! Reference source not found.** and is copied below for information.

```
TimeEvidencesRules::= SEQUENCE OF RulesForSetOfEvidences

RulesForSetOfEvidences ::= SEQUENCE {
    evidenceIdentifiers     SEQUENCE OF IA5String,
    levelOfAssurance        IA5String,
    timeStampTrustCondition TimestampTrustCondition OPTIONAL }

TimestampTrustCondition ::= SEQUENCE {
    ttsCertificateTrustTrees    [0]     CertificateTrustTrees   OPTIONAL,
    ttsRevReq                   [1]     CertificateRevReq       OPTIONAL,
    ttsNameConstraints          [2]     NameConstraints         OPTIONAL,
    cautionPeriod               [3]     DeltaTime               OPTIONAL,
    signatureTimestampDelay     [4]     DeltaTime               OPTIONAL }


DeltaTime ::= SEQUENCE {
    deltaSeconds    INTEGER,
    deltaMinutes    INTEGER,
    deltaHours      INTEGER,
    deltaDays       INTEGER }
```

EDITOR'S NOTE: We have CertificateRevReq instead of CertRevReq, thus we cannot really reuse the main type from the TR

The element of type CertificateTrustTrees shall be as specified in clause 4.24.5.2

The element of type CertificateRevReq shall be as specified in clause 4.25.2.2.

The element of type NameConstraints shall be as specified in clause 4.24.3.2.

4.28 The `RoleTrustCondition` type

4.28.1 Semantics

The semantic shall be as in clause 4.28.1 of ETSI TS 119 172-2 [3].

4.28.2 Syntax

The `RoleTrustCondition` type shall be as defined in annex B.2 and is copied below for information.

The `AttributeConstraints` type shall be as defined in annex **Error! Reference source not found.** and is copied below for information.

```
RoleConstraints ::= SEQUENCE {
    noSignerAttributesAllowed BOOLEAN,
    constraintsOnOneSetOfRoles SEQUENCE OF RoleSetConstraints }

RoleSetConstraints ::=
    howCertAttribute             HowCertAttribute,
    attrCertificateTrustTrees    [0] CertificateTrustTrees   OPTIONAL,
    attrRevReq                   [1] CertificateRevTrust     OPTIONAL,
    attributeConstraints         [2] AttributeConstraints    OPTIONAL }

HowCertAttribute ::= ENUMERATED {
    claimedAttribute    (0),
    certifiedAttribtes  (1),
    signedAssertions    (2),
    any                 (3) }

AttributeConstraints ::= SEQUENCE {
    attributeTypeConstraints    [0] AttributeTypeConstraints  OPTIONAL,
    attributeValueConstraints   [1] AttributeValueConstraints OPTIONAL }


AttributeTypeConstraints ::= SEQUENCE OF AttributeType

AttributeValueConstraints ::= SEQUENCE OF AttributeTypeAndValue
```

EDITOR'S NOTE: The TR used the terms "Attribute" instead of Role, part 2 uses "role". What should we use here?

The element of type `CertificateTrustTrees` shall be as specified in clause 4.24.5.2.

The lement of type `CertificateRevTrust` shall be as specified in clause 4.25.3.2

4.29 The `SignatureAttributesRules` type

4.29.1 Semantics

The semantic shall be as in clause 4.29.1 of ETSI TS 119 172-2 [3].

EDITOR'S NOTE: In part 2 we state that the qualifying property shall be identified by an URI. Can we use an OID or should we use an URI?

EDITOR'S NOTE: Change of name between part 2 (QualifyingPropertiesRules) and part 3 (SignatureAttributesRules)

4.29.2 Syntax

The `SignatureAttributesRules` type shall be as defined in annex B.2 and is copied below for information.

```
SignatureAttributesRules ::= SEQUENCE OF LevelAttributesRules

LevelAttributesRules ::= SEQUENCE {
    levelIdentifier    [0] IA5Strig,
```

```
    signedAttributes   [1] SignatureAttributes OPTIONAL,
    unsignedAttributes [2] SignatureAttributes OPTIONAL }

SignatureAttributes ::= SEQUENCE OF CHOICE {
    choice  [0] SEQUENCE OF SignatureAttribute,
    sigAttr [1] SignatureAttribute }

SignatureAttribute ::= SEQUENCE {
    Identifier    OBJECT IDENTIFIER,
    Mandatory     BOOLEAN }
```

4.30 The SCDLoARules type

4.30.1 Semantics

The semantic shall be as in clause 4.30.1 of ETSI TS 119 172-2 [3].

4.30.2 Syntax

The SCDLoARules type shall be as defined in annex B.2 and is copied below for information.

```
SCDLoARules ::= IA5String
```

The SCDDLoARules shall be a URI value indicating the Level of Assurance of the signature creation device.

4.31 The CryptoSuitesRules type

4.31.1 Semantics

The semantic shall be as in clause 4.31.1 of ETSI TS 119 172-2 [3].

4.31.2 Syntax

The CryptoSuitesRules type shall be as defined in annex B.2 and is copied below for information.

```
AlgorithmConstraintSet ::= SEQUENCE {   -- Algorithm constrains on:
    signerAlgorithmConstraints  [0]     AlgorithmConstraints OPTIONAL,
    eeCertAlgorithmConstraints  [1]     AlgorithmConstraints OPTIONAL, certs.
    caCertAlgorithmConstraints  [2]     AlgorithmConstraints OPTIONAL,
    aaCertAlgorithmConstraints  [3]     AlgorithmConstraints OPTIONAL,
    tsaCertAlgorithmConstraints [4]     AlgorithmConstraints OPTIONAL
}

AlgorithmConstraints ::= SEQUENCE OF AlgAndLength

AlgAndLength ::= SEQUENCE {
    algID             OBJECT IDENTIFIER,
    minKeyLength  [0] INTEGER   OPTIONAL,
    minHashLength [1] INTEGER   OPTIONAL,
    other             OtherRules OPTIONAL
}
```

The elment of type OtherRules shall be as defined in clause 4.32.2.

4.32 The OtherRules type

4.32.1 Semantics

The semantic shall be as in clause 4.32.1 of ETSI TS 119 172-2 [3].

4.32.2 Syntax

The `OtherRules` type shall be as defined in annex B.2 and is copied below for information.

```
OtherRules ::= SEQUENCE OF OtherRule
OtherRule ::= SEQUENCE {
        extnID      OBJECT IDENTIFIER,
        extnValue   OCTET STRING  }
```

The `extnID` field shall contain the object identifier for the extension.

The `extnValue` field shall contain the DER (see ITU-T Recommendation X.690 [**Error! Reference source not found.**]) encoded value of the extension. The definition of an extension, as identified by `extnID` shall include a definition of the syntax and semantics of the extension.

Annex A (normative): Signature policy definitions using X.680 ASN.1 syntax

B.1 ASN.1 module containing elements defined in previous versions of the present document

EDITOR'S NOTE 1: For the moment the whole module was copied from the TR. The types which were referenced are marked in green. We still need to remove the values not used.

EDITOR'S NOTE 2: Need to ask editHelp how to change the numbering of B.1 / B.2 to A.1 and A.2.

```
ETS-ElectronicSignaturePolicies-97Syntax { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-mod(0) 8}

DEFINITIONS EXPLICIT TAGS ::=
BEGIN
-- EXPORTS All -

IMPORTS

EDITOR'S NOTE: Updated imports as in CAdES. Not really sure from where to import
AttributeTypeAndValue and AttributeType

-- Internet X.509 Public Key Infrastructure - Certificate and CRL Profile: RFC 2459 or RFC 3280
AttributeTypeAndValue, AttributeType
FROM PKIX1Explicit93 {iso(1) identified-organization(3) dod(6) internet(1) security(5) mechanisms(5)
pkix(7) id-mod(0) id-pkix1-explicit-88(1)}

GeneralNames, GeneralName, PolicyInformation
FROM PKIX1Implicit-2009 {iso(1) identified-organization(3) dod(6) internet(1) security(5)
mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-implicit-02(59)}

Certificate, AlgorithmIdentifier, CertificateList, Name, DirectoryString
FROM PKIX1Explicit-2009 {iso(1) identified-organization(3) dod(6) internet(1) security(5)
mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-explicit-02(51)}

Attribute
FROM CryptographicMessageSyntax-2010 { iso(1) member-body(2) us(840) rsadsi(113549)
pkcs(1) pkcs-9(9) smime(16) modules(0) id-mod-cms-2009(58) }
;
-- The structures may also be imported from :
-- PKIX1Explicit88 { iso(1) identified-organization(3) dod(6) internet(1)
-- security(5) mechanisms(5) pkix(7) id-mod(0) id-pkix1-explicit(18) }


-- S/MIME Object Identifier arcs used in the present document
-- ==========================================================

-- S/MIME  OID arc used in the present document
-- id-smime OBJECT IDENTIFIER ::= { iso(1) member-body(2)
--             us(840) rsadsi(113549) pkcs(1) pkcs-9(9) 16 }

-- S/MIME Arcs
-- id-mod  OBJECT IDENTIFIER ::= { id-smime 0 }
-- modules
-- id-ct   OBJECT IDENTIFIER ::= { id-smime 1 }
-- content types
-- id-aa   OBJECT IDENTIFIER ::= { id-smime 2 }
-- attributes
-- id-spq  OBJECT IDENTIFIER ::= { id-smime 5 }
-- signature policy qualifier
-- id-cti  OBJECT IDENTIFIER ::= { id-smime 6 }
-- commitment type identifier


-- Signature Policy Specification
-- ==============================

SignaturePolicy ::= SEQUENCE {
    signPolicyHashAlg      AlgorithmIdentifier,
    signPolicyInfo         SignPolicyInfo,
```

```
    signPolicyHash         SignPolicyHash     OPTIONAL }

SignPolicyHash ::= OCTET STRING

SignPolicyInfo ::= SEQUENCE {
    signPolicyIdentifier            SignPolicyId,
    dateOfIssue                     GeneralizedTime,
    policyIssuerName                PolicyIssuerName,
    fieldOfApplication              FieldOfApplication,
    signatureValidationPolicy       SignatureValidationPolicy,
    signPolExtensions               SignPolExtensions       OPTIONAL
    }

SignPolicyId ::= OBJECT IDENTIFIER

PolicyIssuerName ::= GeneralNames

FieldOfApplication ::= DirectoryString

SignatureValidationPolicy ::= SEQUENCE {
    signingPeriod          SigningPeriod,
    commonRules            CommonRules,
    commitmentRules        CommitmentRules,
    signPolExtensions      SignPolExtensions        OPTIONAL
     }

SigningPeriod ::= SEQUENCE {
    notBefore   GeneralizedTime,
    notAfter    GeneralizedTime OPTIONAL }

CommonRules  ::= SEQUENCE {
    signerAndVeriferRules          [0]  SignerAndVerifierRules     OPTIONAL,
    signingCertTrustCondition      [1]  SigningCertTrustCondition   OPTIONAL,
    timeStampTrustCondition        [2]  TimestampTrustCondition     OPTIONAL,
    attributeTrustCondition        [3]  AttributeTrustCondition     OPTIONAL,
    algorithmConstraintSet         [4]  AlgorithmConstraintSet      OPTIONAL,
    signPolExtensions              [5]  SignPolExtensions           OPTIONAL
    }

CommitmentRules ::= SEQUENCE OF CommitmentRule

CommitmentRule  ::= SEQUENCE {
    selCommitmentTypes                  SelectedCommitmentTypes,
    signerAndVeriferRules          [0]  SignerAndVerifierRules      OPTIONAL,
    signingCertTrustCondition      [1]  SigningCertTrustCondition   OPTIONAL,
    timeStampTrustCondition        [2]  TimestampTrustCondition     OPTIONAL,
    attributeTrustCondition        [3]  AttributeTrustCondition     OPTIONAL,
    algorithmConstraintSet         [4]  AlgorithmConstraintSet      OPTIONAL,
    signPolExtensions              [5]  SignPolExtensions           OPTIONAL
    }

SelectedCommitmentTypes ::= SEQUENCE OF CHOICE {
    empty                          NULL,
    recognizedCommitmentType     CommitmentType }

CommitmentType ::= SEQUENCE {
    identifier          CommitmentTypeIdentifier,
    fieldOfApplication  [0] FieldOfApplication OPTIONAL,
    semantics           [1] DirectoryString OPTIONAL }

SignerAndVerifierRules ::= SEQUENCE {
    signerRules      SignerRules,
    verifierRules    VerifierRules }

SignerRules ::= SEQUENCE {
    externalSignedData         BOOLEAN  OPTIONAL,
            -- True if signed data is external to CMS structure
            -- False if signed data part of CMS structure
            -- not present if either allowed
    mandatedSignedAttr         CMSAttrs,    -- Mandated CMS signed attributes
    mandatedUnsignedAttr       CMSAttrs,    -- Mandated CMS unsigned attributed
    mandatedCertificateRef     [0] CertRefReq DEFAULT signerOnly,
                -- Mandated Certificate Reference
    mandatedCertificateInfo    [1] CertInfoReq DEFAULT none,
                -- Mandated Certificate Info
    signPolExtensions        [2] SignPolExtensions      OPTIONAL
                          }
```

```
CMSAttrs ::= SEQUENCE OF OBJECT IDENTIFIER

CertRefReq ::= ENUMERATED {
                signerOnly (1),     -- Only reference to signer cert mandated
                fullPath (2)
                        -- References for full cert path up to a trust point required
                                }

CertInfoReq ::= ENUMERATED {
                none (0)    ,           -- No mandatory requirements
                signerOnly (1)  ,       -- Only reference to signer cert mandated
                fullPath (2)
                        -- References for full cert path up to a trust point mandated
        }

VerifierRules ::= SEQUENCE {
        mandatedUnsignedAttr    MandatedUnsignedAttr,
        signPolExtensions       SignPolExtensions       OPTIONAL
        }

MandatedUnsignedAttr ::=  CMSAttrs    -- Mandated CMS unsigned attributed


CertificateTrustTrees ::=   SEQUENCE OF CertificateTrustPoint

CertificateTrustPoint ::= SEQUENCE {
    trustpoint              Certificate,                        -- self-signed certificate
    pathLenConstraint   [0] PathLenConstraint   OPTIONAL,
    acceptablePolicySet [1] AcceptablePolicySet OPTIONAL,  -- If not present "any policy"
    nameConstraints     [2] NameConstraints     OPTIONAL,
    policyConstraints   [3] PolicyConstraints   OPTIONAL }

PathLenConstraint    ::=    INTEGER (0..MAX)

AcceptablePolicySet ::= SEQUENCE OF CertPolicyId

CertPolicyId ::= OBJECT IDENTIFIER

NameConstraints ::= SEQUENCE {
          permittedSubtrees       [0]     GeneralSubtrees OPTIONAL,
          excludedSubtrees        [1]     GeneralSubtrees OPTIONAL }

      GeneralSubtrees ::= SEQUENCE SIZE (1..MAX) OF GeneralSubtree

      GeneralSubtree ::= SEQUENCE {
          base                    GeneralName,
          minimum         [0]     BaseDistance DEFAULT 0,
          maximum         [1]     BaseDistance OPTIONAL }

      BaseDistance ::= INTEGER (0..MAX)

PolicyConstraints ::= SEQUENCE {
       requireExplicitPolicy           [0] SkipCerts OPTIONAL,
       inhibitPolicyMapping            [1] SkipCerts OPTIONAL }

SkipCerts ::= INTEGER (0..MAX)

CertRevReq ::= SEQUENCE {
    endCertRevReq   RevReq,
    caCerts         [0] RevReq
      }

RevReq ::= SEQUENCE  {
    enuRevReq  EnuRevReq,
    exRevReq    SignPolExtensions OPTIONAL}

EnuRevReq  ::= ENUMERATED {
    clrCheck    (0), --Checks shall be made against current CRLs
           -- (or authority revocation lists)
    ocspCheck   (1), -- The revocation status shall be checked
           -- using the Online Certificate Status Protocol (RFC 2450)
    bothCheck   (2),    -- Both CRL and OCSP checks shall be carried out
    eitherCheck (3),    -- At least one of CRL or OCSP checks shall be carried out
    noCheck     (4),    -- no check is mandated
    other       (5)     -- Other mechanism as defined by signature policy extension -- }

SigningCertTrustCondition ::=   SEQUENCE {
     signerTrustTrees             CertificateTrustTrees,
```

```
     signerRevReq                  CertRevReq
}

TimestampTrustCondition ::= SEQUENCE {
    ttsCertificateTrustTrees    [0]     CertificateTrustTrees    OPTIONAL,
    ttsRevReq                   [1]     CertRevReq               OPTIONAL,
    ttsNameConstraints          [2]     NameConstraints          OPTIONAL,
    cautionPeriod               [3]     DeltaTime                OPTIONAL,
    signatureTimestampDelay     [4]     DeltaTime                OPTIONAL }

DeltaTime ::= SEQUENCE {
    deltaSeconds    INTEGER,
    deltaMinutes    INTEGER,
    deltaHours      INTEGER,
    deltaDays       INTEGER }

AttributeTrustCondition ::= SEQUENCE {
    attributeMandated            BOOLEAN,              -- Attribute shall be present
    howCertAttribute             HowCertAttribute,
    attrCertificateTrustTrees   [0] CertificateTrustTrees   OPTIONAL,
    attrRevReq                  [1] CertRevReq              OPTIONAL,
    attributeConstraints        [2] AttributeConstraints   OPTIONAL }


HowCertAttribute ::= ENUMERATED {
    claimedAttribute    (0),
    certifiedAttribtes  (1),
    either              (2) }

AttributeConstraints ::= SEQUENCE {
    attributeTypeConstraints    [0] AttributeTypeConstraints  OPTIONAL,
    attributeValueConstraints   [1] AttributeValueConstraints OPTIONAL }


AttributeTypeConstraints ::= SEQUENCE OF AttributeType

AttributeValueConstraints ::= SEQUENCE OF AttributeTypeAndValue


AlgorithmConstraintSet ::= SEQUENCE {   -- Algorithm constraints on:
signerAlgorithmConstraints  [0]     AlgorithmConstraints OPTIONAL, -- signer
eeCertAlgorithmConstraints  [1]     AlgorithmConstraints OPTIONAL, -- issuer of end entity certs
caCertAlgorithmConstraints  [2]     AlgorithmConstraints OPTIONAL, -- issuer of CA certificates
aaCertAlgorithmConstraints  [3]     AlgorithmConstraints OPTIONAL, -- Attribute Authority
tsaCertAlgorithmConstraints [4]     AlgorithmConstraints OPTIONAL  -- TimeStamping Authority -- }

AlgorithmConstraints ::= SEQUENCE OF AlgAndLength

AlgAndLength ::= SEQUENCE {
    algID           OBJECT IDENTIFIER,
    minKeyLength    INTEGER     OPTIONAL, -- Minimum key length in bits
    other       SignPolExtensions OPTIONAL }

SignPolExtensions ::= SEQUENCE OF SignPolExtn

SignPolExtn ::= SEQUENCE {
        extnID      OBJECT IDENTIFIER,
        extnValue   OCTET STRING  }

END -- ETS- ElectronicSignaturePolicies-97Syntax
```

B.2 ASN.1 module containing elements first defined in this document

EDITOR'S NOTE: Need to update this clause once types are agreed in the main document

```
ETSI-SigPolicy-ASN1 { itu-t(0) identified-organization(4) etsi(0) sigpolicy-asn1(1917202)
id-mod(0) sigpolicy-syntax680(1)}
DEFINITIONS EXPLICIT TAGS ::=
BEGIN
-- EXPORTS All -
IMPORTS
-- Imports from CAdES in ETSI EN 319 122-1
OtherHashAlgAndValue, PostalAddress, CommitmentTypeIdentifier
FROM ETSI-CAdES-ExplicitSyntax97 { itu-t(0) identified-organization(4) etsi(0) cades(19122)
id-mod(0) cades-explicit97(1)}

-- Imports from module defined in annex B.1
SigningPeriod, PathLenConstraint, AcceptablePolicySet, NameConstraints, PolicyConstraints,
EnuRevReq, DeltaTime, AttributeConstraints
FROM ETS-ElectronicSignaturePolicies-97Syntax { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
pkcs-9(9) smime(16) id-mod(0) 8}

-- Imports as defubed ub RFC 5912
Certificate
FROM PKIX1Explicit-2009 {iso(1) identified-organization(3) dod(6) internet(1) security(5)
mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-explicit-02(51)}

-- Definitions of Object Identifier arcs used in the present document
-- ==================================================================

-- Type definitions
-- ====================================


SignaturePolicy ::= SEQUENCE {
    digest Digest,
    policyComponents PolicyComponents}

Digest ::= OtherHashAlgAndValue

PolicyComponents ::= SEQUENCE {
    geneneralDetails  GeneralDetails,
    otherPolicies [0] OtherPolicies OPTIONAL,
    policyRules PolicyRules }

GeneralDetails ::= SEQUENCE {
    sigPolicyDetails  SigPolicyDetails,
    authorityDetails AuthorityDetails,
    otherDetails OtherDetails OPTIONAL }

GeneralDetails ::= SEQUENCE {
    policyIdentifier OBJECT IDENTIFIER,
    policyName InternationalNames,
    distributionPoints [0] DistributionPoints OPTIONAL,
    versionInfo [1] VersionInfo OPTIONAL }

InternationalNames :: = SEQUENCE OF MultiLangText

MultiLangText:: = SEQUENCE {
    lang PrintableString)),
    text UTF8String }

DistributionPoints :: = SEQUENCE (SIZE 1..MAX) OF IA5String

VersionInfo ::= SEQUENCE {
   thisVersion UTF8String,
   previousVersions PreviousVersions OPTIONAL }

PreviousVersions :: = SEQUENCE OF PreviousVersion

PreviousVersion ::= SEQUENCE {
    version UTF8String,
    distributionPoints  DistributionPoints OPTIONAL }

AuthorityDetails ::= SEQUENCE {
```

```
    name [0] Name OPTIONAL,
    tradeName [1] TradeName OPTIONAL,
    postalAddresses PostalAddresses,
    electronicAddresses ElectronicAddresses,
    contactPersons [2] ContactPersons OPTIONAL}

Name ::= InternationalNames

TradeName::= InternationalNames

PostalAddresses ::= SEQUENCE OF PostalAddress
ElectronicAddresses::= SEQUENCE OF ElectronicAddresse
ElectronicAddresse ::= MultiLangURI
MultiLangURI :: = SEQUENCE {
    lang PrintableString,
    uri IA5String }

ContactPersons ::= SEQUENCE OF ContactPerson
ContactPerson ::= SEQUENCE {
    Name UTF8String,
    electronicAddresses ElectronicAddresses,
    phoneNumbers SEQUENCE OF PrintableString}

OtherDetails ::= SEQUENCE {
    dateOfIssue GeneralizedTime,
    signingPeriod [0] SigningPeriod,
    others [1] SEQUENCE OF OtherDetails}

OtherDetails ::= SEQUENCE {
    otherDetailsId OTHER-DETAILS.&id,
    details OTHER-DETAILS.&Details OPTIONAL
}

OTHER-DETAILS ::= CLASS {
    &id OBJECT IDENTIFIER UNIQUE,
    &Details OPTIONAL }
WITH SYNTAX {
    OTHER-DETAILS-ID &id
    [DETAILS-TYPE & Details] }

OtherPolicies ::= SEQUENCE OF SigPolicyDetails

PolicyRules  ::= SEQUENCE OF PolicyRuleWithScope

PolicyRuleWithScope ::= SEQUENCE {
    rule   SigPolicyRule,
    scope  SigPolicyScope OPTIONAL}

SigPolicyRule ::= CHOICE {
    commitmentRules            [0]   CommitmentRules,
    basicRule                  [1]   BasicRule }

BasicRule ::= CHOICE {
    dataToBeSignedRules        [0]   DataToBeSignedRules,
    sigToDTBSRelationRules     [1]   SigToDTBSRelationRules,
    dTBSCardinality            [2]   DTBSCardinality,
    sigDTBSRelativePositions   [3]   SigDTBSRelativePositions,
    sigFormatAndLevel          [4]   SigFormatAndLevel,
    augmentationRules          [5]   AugmentationRules,
    signingCertTrustConditions [6]   SigningCertTrustConditions,
    timeEvidLoARules           [7]   TimeEvidLoARules,
    roleTrustCondition         [8]   RoleTrustCondition,
    signatureAttributesRules   [9]   SignatureAttributesRules,
    sCDLoARules                [10]  SCDLoARules,
    cryptoSuitesRules          [11]  CryptoSuitesRules,
    otherRules                 [12]  OtherRules
}


SigPolicyScope  ::= ENUMERATED {
    generation      (0),
    validation      (1),
    Augmentation    (2) }

CommitmentRules ::= SEQUENCE OF CommitmentRule

CommitmentRule ::= SEQUENCE {
    commitmentIdentifier            CommitmentTypeIdentifier
```

```
    matchingIndicator               MatchingIndicator,
    basicRules                      SEQUENCE OF BasicRule}

MatchingIndicator ::= ENUMERATED {
    All       (0),
    None      (1),
    atLeastOne(2) }

DataToBeSignedRules ::= SEQUENCE OF DataToBeSignedRule

DataToBeSignedRule ::= SEQUENCE {
    anyOfMimeType  [0] SEQUENCE OF UTF8String OPTIONAL,
    noneOfMimeType [1] SEQUENCE OF UTF8String OPTIONAL }

SigToDTBSRelationRules::= SEQUENCE {
    dTBSCardinality             DTBSCardinality,
    sigDTBSRelativePosition [0] SigDTBSRelativePosition OPTIONAL,
    sigFormatsAndLevels     [1] SigFormatsAndLevels OPTIONAL }


DTBSCardinality::= SEQUENCE {
    maxDTBSNumber [0] MaxDTBSNumber OPTIONAL,
    minDTBSNumber [1] MinDTBSNumber OPTIONAL }

MaxDTBSNumber ::= SEQUENCE {
    dTBSNumber INTEGER,
    maxValueQualifier MaxValueQualifier }

MaxValueQualifier ::= ENUMERATED {
    lessThan       (0),
    lessOrEqualTo  (1),
    equal          (2)  }

MinDTBSNumber ::= SEQUENCE {
    dTBSNumber INTEGER,
    minValueQualifier MinValueQualifier }


MinValueQualifier ::= ENUMERATED {
    higherThan      (0),
    higherOrEqualTo (1),
    equal           (2)  }

SigDTBSRelativePosition::= ENUMERATED {
    envelopingSig              (0),
    envelopedSig               (1),
    detachedSig                (2),
    envelopingAndEnvelopedSig  (3),
    envelopingAndDetachedSig   (4),
    envelopedAndDetachedSig    (5),
    aSiC                       (6),
    all                        (7) }

SigFormatAndLevels ::= SEQUENCE {
    sigFormats SEQUENCE OF IA5String,
    sigLevels SEQUENCE OF SigLevel }

SigLevel ::= SEQUENCE {
    level IA5String,
    version IA5String }

AugmentationRules ::= SEQUENCE {
    previousValidationRequired BOOLEAN,
    adESSigLevel AdESSigLevel,
    augQualifier AugmentationQualifier }

AugmentationQualifier ::= ENUMERATED {
    thisLevel     (0),
    minLevel      (1),
    maxLevel      (2)  }

TrustAnchors ::= SEQUENCE OF TrustAchnor
TrustAnchor ::= CHOICE  {
    Certificate CertAndReliableTime,
    trustedList URIAndReliableTime,
    trustStatusList URIAndReliableTime }

CertAndReliableTime ::= SEQUENCE {
```

```
    cert Certificate,
    reliableUntil GeneralizedTime OPTIONAL }

URIAndReliableTime ::= SEQUENCE {
    uri IA5String,
    reliableUntil GeneralizedTime OPTIONAL }

CertificateTrustTrees ::=   SEQUENCE OF CertificateTrustPoint

CertificateTrustPoint ::= SEQUENCE {
    trustAnchors        TrustAnchors,
    pathLenConstraint   [0] PathLenConstraint   OPTIONAL,
    acceptablePolicySet [1] AcceptablePolicySet OPTIONAL,
    nameConstraints     [2] NameConstraints     OPTIONAL,
    policyConstraints   [3] PolicyConstraints   OPTIONAL,
    userCertPath        [4] UserCertPath        OPTIONAL }

UseCertPath ::= CHOICE {
    asInSignature BOOLEAN,
    path SEQUENCE OF Certifciate }

CertificateRevReq ::= SEQUENCE {
    endCertRevReq   EnuRevReq  ,
    caCerts         EnuRevReq }

CertificateRevTrust ::= SEQUENCE
    certificateRevReq CertificateRevReq,
    freshness         [0] Freshness OPTIONAL,
    SigCertIssuedByCAKeepsExpiredRevokedCertsInfo [1] BOOLEAN OPTIONAL }

Freshness ::= CHOICE {
    maxDifferenceRevocationAndValidation GeneralizedTime,
    timeAfterSignature                   GeneralizedTime

SigningCertTrustCondition::=
    signerTrustTrees CertificateTrustTrees,
    signerRevTrust   CertificateRevTrust }

TimeEvidencesRules::= SEQUENCE OF RulesForSetOfEvidences

RulesForSetOfEvidences ::= SEQUENCE {
    evidenceIdentifiers     SEQUENCE OF IA5String,
    levelOfAssurance        IA5String,
    timeStampTrustCondition TimestampTrustCondition OPTIONAL }

TimestampTrustCondition ::= SEQUENCE {
    ttsCertificateTrustTrees    [0]     CertificateTrustTrees   OPTIONAL,
    ttsRevReq                   [1]     CertificateRevReq       OPTIONAL,
    ttsNameConstraints          [2]     NameConstraints         OPTIONAL,
    cautionPeriod               [3]     DeltaTime               OPTIONAL,
    signatureTimestampDelay     [4]     DeltaTime               OPTIONAL }

TimeEvidencesRules::= SEQUENCE OF RulesForSetOfEvidences

RulesForSetOfEvidences ::= SEQUENCE {
    evidenceIdentifiers     SEQUENCE OF IA5String,
    levelOfAssurance        IA5String,
    timeStampTrustCondition TimestampTrustCondition OPTIONAL }

TimestampTrustCondition ::= SEQUENCE {
    ttsCertificateTrustTrees    [0]     CertificateTrustTrees   OPTIONAL,
    ttsRevReq                   [1]     CertificateRevReq       OPTIONAL,
    ttsNameConstraints          [2]     NameConstraints         OPTIONAL,
    cautionPeriod               [3]     DeltaTime               OPTIONAL,
    signatureTimestampDelay     [4]     DeltaTime               OPTIONAL }

SignatureAttributesRules ::= SEQUENCE OF LevelAttributesRules

LevelAttributesRules ::= SEQUENCE {
    levelIdentifier    [0] IA5Strig,
    signedAttributes   [1] SignatureAttributes OPTIONAL,
    unsignedAttributes [2] SignatureAttributes OPTIONAL }

SignatureAttributes ::= SEQUENCE OF CHOICE {
    choice  [0] SEQUENCE OF SignatureAttribute,
    sigAttr [1] SignatureAttribute }

SignatureAttribute ::= SEQUENCE {
```

```
    Identifier    OBJECT IDENTIFIER,
    Mandatory     BOOLEAN }

SCDLoARules ::= IA5String

AlgorithmConstraintSet ::= SEQUENCE {   -- Algorithm constrains on:
    signerAlgorithmConstraints  [0]     AlgorithmConstraints OPTIONAL,
    eeCertAlgorithmConstraints  [1]     AlgorithmConstraints OPTIONAL, certs.
    caCertAlgorithmConstraints  [2]     AlgorithmConstraints OPTIONAL,
    aaCertAlgorithmConstraints  [3]     AlgorithmConstraints OPTIONAL,
    tsaCertAlgorithmConstraints [4]     AlgorithmConstraints OPTIONAL
}

AlgorithmConstraints ::= SEQUENCE OF AlgAndLength

AlgAndLength ::= SEQUENCE {
    algID             OBJECT IDENTIFIER,
    minKeyLength  [0] INTEGER   OPTIONAL,
    minHashLength [1] INTEGER   OPTIONAL,
    other             OtherRules OPTIONAL
}


OtherRules ::= SEQUENCE OF OtherRule
OtherRule ::= SEQUENCE {
        extnID      OBJECT IDENTIFIER,
        extnValue   OCTET STRING  }
END
```

History

Document history		
V0.0.3	2019-07	Stable draft for public availabiliy