NATIONAL SECURITY AUTHORITY

Version 1.0

Interoperability profile intended for Commission Decision 2011/130/EU realization.

1 August 2011

NATIONAL SECURITY AUTHORITY

Information Security and Electronic Signature Department

Budatínska 30, P.O. BOX 16, 850 07 Bratislava 57

http://www.nbusr.sk/

E-mail: podatelna@nbusr.sk

Content

1 Introduction

The main intent of the present document is to provide an overall summary of technical requirements based on implementation of Commission Decision 2011/130/EU and Slovak legislation in order to achieve a detailed view of requirements which must be met to achieve the interoperability.

Verification of Qualified Electronic Signature (hereinafter referred to as QES) validity is realized not only in the time of the qualified certificate or maintenance certificate validity period, but also in later time when e.g. all certificates used for QES verification have expired. According to Commission Decision 2011/130/EU, the cross-border use of advanced electronic signatures supported by a qualified certificate (AdES-qc) is facilitated through Commission Decision 2009/767/EC of 16 October 2009 setting out measures facilitating the use of procedures by electronic means through the 'points of single contact' under Directive 2006/123/EC of the European Parliament and of the Council on services in the internal market (2) which, inter alia, imposes an obligation on Member States to carry out risk assessments before requiring these electronic signatures from service providers and establishes rules for the acceptance by Member States of **advanced electronic signatures based on qualified certificates**, created with a **secure signature creation device (SSCD)** - QES or without **SSCD** - AdES-qc. Only profiles for QES could be specified for interoperability in case of wide variety of expected levels of requirements for any other signatures than QES.

The majority of certification service providers, after certificate expiration, do not continue with providing the information about potential revocation in CRL or OCSP. In order to ensure the **correct verification of QES** also in the time when the certificates are already expired as well as the trusted (root) certificate is already expired and can not be found in the trusted repository of applications, it is required to ensure **a trusted publication of** information containing **the history of** expired **trusted** (root) **certificates** as well as the history of **rules** used in the past which are **verifiable by** the **currently trusted** (root) **certificate**. Publication of such information is most frequently realized by means of a trusted list containing the current and historic data. In order to realize the signing and verification process of the QES validity automatically also in the QES verification with already expired certificates with the least interactions between the signer and verifier, it is necessary to define interoperable automatic procedures and data structures which shall enable such verification and signature attributes where such information will be located e.g. http://www.nbusr.sk/en/electronic-signature/signature-policies/index.html .

2 Scope

The Directive ”Interoperability profile intended for Commission Decision 2011/130/EU realization“ is issued in accordance with Article 10 (j) of the Act No. 215/2002 Coll. on Electronic signature and on the amendment and supplementing of certain acts as amended (hereinafter referred to as the Act) of 15 March 2002. The National Security Authority (hereinafter referred to as the NSA) Decree No.135/2009 Coll. on the format and method of creating the qualified electronic signature, the method of publishing the public key of the National Security Authority, the conditions of validity for the qualified electronic signature, procedure during the verification and verification conditions of the qualified electronic signature, format of the time stamp and method of creating it, requirements on the source of time data and requirements for keeping time stamp documentation (on the creation and verification of the electronic signature and time stamp) together with the Commission Decision 2011/130/EU make the base for creation of this profile.

The purpose of the present Directive is to create a profile which provides the overall summary of technical requirements for subjects providing accredited services, auditors, QES application developers, QES or AdES-qc users according to EU Commission Decision 2011/130/EU in order to determine technical requirements for particular types of QES and AdES-qc to ensure the compatibility and unified environment of the electronic signature in the Slovak Republic with respect to electronic signature environment within the European Union where Member States shall ensure that QES (CWA 14170) on the basis of Article 5 (1) of the European Directive 1999/93/EC [26] meets the following:

Legal effects of electronic signatures
Member States shall ensure that **advanced electronic signatures** which are based on a **qualified certificate** and which are created by a **secure-signature-creation-device (QES)**:

- satisfy the same legal requirements of the signature in relation to data in electronic form in the same manner as a hand-written signature satisfies those requirements in relation to paper-based data;
- are admissible as evidence in legal proceedings.

The trusted list according to CD 2009/767/EC is mainly used as a source of information if the certificate was issued by accredited/supervised provider in EU as a qualified certificate or a qualified certificate based on SSCD.

3 References

References to documents defining used types and methods.

[1] ETSI TS 101 733 Electronic Signature Formats

[2] RFC 3125 Signature Policies | TR 102 272 ASN.1 format for signature policies

[3] RFC 5280 X.509 PKI Certificate and Certificate Revocation List May 2008, https://datatracker.ietf.org/doc/rfc5280/

[4] RFC 3739 Qualified Certificates Profile March 2004

[5] RFC 5126 CMS Advanced Electronic Signatures (CAdES) February 2008

[6] RFC 5652 Cryptographic Message Syntax (CMS) September 2009

[7] RFC 3161 Time-Stamp Protocol (TSP) August 2001

[8] RFC 2560 X.509 PKI Online Certificate Status Protocol June 1999

[9] ISO 19005-1:2005, Document management — Electronic document file format for long-term preservation -- Part 1: Use of PDF 1.4 (PDF/A-1).

[10] EN 14890 Application Interface for Smart Cards used as Secure Signature Creation Devices Part 1: Basic Services; Part 2: Additional Services

[11] RFC 2044 UTF-8, a transformation format of Unicode and ISO 10646 October 1996

[12] ETSI TS 102 231 V3.1.1 Provision of harmonized Trust-service status information

[13] EU Directive 1999/93/EC of the European Parliament and the Council of 13 December 1999 on a Community framework for electronic signatures

[14] ETSI TS 102 176-1 Electronic signatures and infrastructures (ESI); Algorithms and parameters for secure electronic signatures; Part 1: Hash functions and asymmetric algorithms

[15] ETSI TS 101 903 V1.4.1 (2009-06) XML Advanced Electronic Signatures (XAdES)

[16] COMMISSION DECISION 2010/425/EU of 28 July 2010 amending Decision 2009/767/EC as regards the establishment, maintenance and publication of trusted lists of certification service providers supervised/accredited by Member States http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2010:199:0030:0035:EN:PDF

[17] COMMISSION DECISION of 25 February 2011 establishing minimum requirements for the cross-border processing of documents signed electronically by competent authorities under Directive 2006/123/EC of the European Parliament and of the Council on services in the internal market (2011/130/EU), http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2011:053:0066:0072:EN:PDF

[18] ISO 32000-2, Document management — portable document format — Part 2: PDF 2.0

[19] ETSI TS 102 778-3 PDF Advanced Electronic Signature Profiles; Part 3: PAdES Enhanced - PAdES-BES and PAdES-EPES Profiles

[20] ETSI TS 102 778-4 PDF Advanced Electronic Signature Profiles; Part 4: PAdES Long Term - PAdES-LTV Profile

[21] ETSI TS 102 778-5 PDF Advanced Electronic Signature Profiles; Part 5: PAdES for XML Content - Profiles for XAdES signatures

[22] ISO/IEC 10646:2011, Information technology -- Universal Coded Character Set (UCS)

[23] Rec. ITU-T X.509 (08/2008)| ISO/IEC 9594-8:2008 (E), http://www.itu.int/rec/T-REC-X.509-200811-I/en

[24] Rec. ITU-T X.690| ISO/IEC 8825-1 Information technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER), http://www.itu.int/ITU-T/studygroups/com17/languages/

[25] ISO 19005-2:2011 Document management - Electronic document file format for long-term preservation - Part 2: Use of ISO 32000-1 (PDF/A-2)

[26] CORRIGENDA Corrigendum to Commission Decision 2009/767/EC of 16 October 2009 setting out measures facilitating the use of procedures by electronic means through the ‘points of single contact’ under Directive 2006/123/EC of the European Parliament and of the Council on services in the internal market (Official Journal of the European Union L 274 of 20 October 2009) http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2009:299:0018:0054:EN:PDF

4 Abbreviations

AdES	Advanced Electronic Signature
AdES-qc	AdES based on qualified certificates, created **without** SSCD
AID	An application identifier used to address an application in the card
ASCII	American Standard Code for Information Interchange
ASN.1	Abstract Syntax Notation 1
CA	Certification Authority
CAdES	CMS Advanced Electronic Signature
CMS	Cryptographic Message Syntax
CRL	Certificate Revocation List
DER	Distinguished Encoding Rules (for ASN.1)
NSA	National Security Authority of the Slovak Republic
OCSP	Online Certificate Status Protocol
OID	Object Identifier
QC	Qualified Certificate
QES	Qualified Electronic Signature - AdES based on qualified certificates, created **with** SSCD
RID	International Registered Application Provider Identifier (ISO/IEC 7816-5)
SP	Signature Policy
SSCD	Secure-Signature-Creation Device
TL	Trusted list of references on approved SPs and trusted certificates
URL	Uniform Resource Locator
XAdES	XML Advanced Electronic Signature
XML	Extensible Markup Language

5 Certificate profile

A certificate version must be v3 (value 2). A certificate content is defined in ITU-T Rec. X.509 (08/2008)| ISO/IEC 9594-8:2008 (E) [23] and RFC 5280 [3]

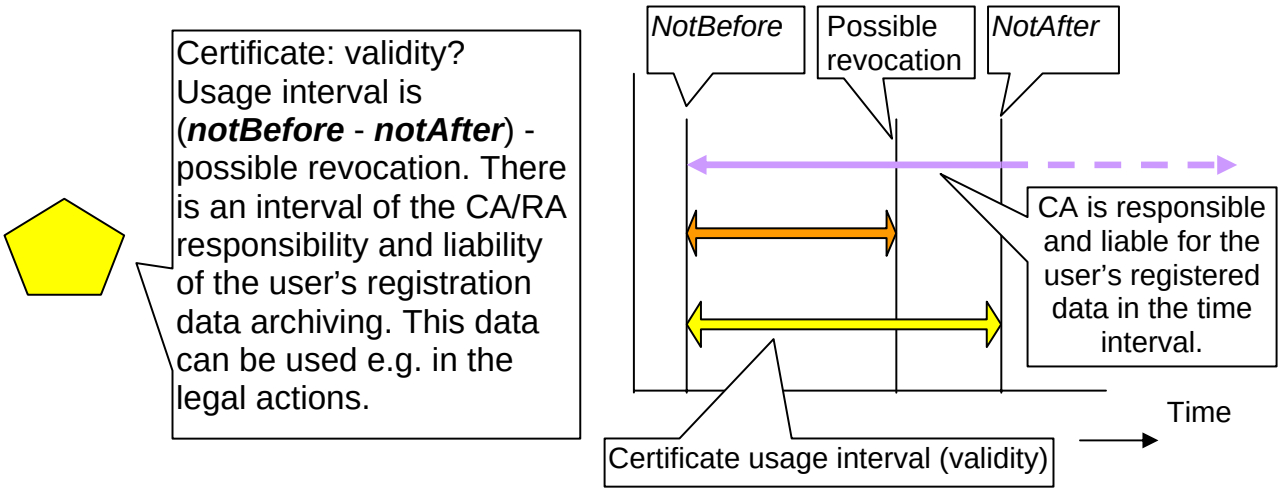


Figure 1: Certificate and certificate usage period

Issuer Name:
It is the name of the certificate issuer (CA). The issuer name must always contain *countryName* where CA has its residence and the name *organizationName*. A field *commonName* is recommended. It can also contain other attributes whose content could cause interoperability problems and for that purpose they are not recommended. The text is non-empty in the field *DirectoryString* and *UTF8String* coding must be used.

***Subject Name*:**
The name of the subject (defined in DN), to whom the certificate is issued, must be unambiguous in CA during the whole CA existence. The same name can be also used in other certificates issued for the same person but it must not be used in certificates issued for the other person.
The reference to physical person's identity of a private key owner, to whom the certificate was issued, is in the field *serialNumber* and the format is "YYYZZ *" where YYY represents the type of identification e.g. "PAS" for identification based on passport number, "IDC" for identification based on identity card number, "PNO" for identification based on personal number. ZZ represents the Country Name according to ISO 3166-1 where YYY identification was registered. The identification replacing * according to YYY is separated with one space " " (char 20) behind YYYZZ. Example is "IDCSK SP989783".
Subject Name must contain *countryName* and *commonName* (If *pseudonym* is used then also *commonName* must contain the word PSEUDONYM appended after or before the given pseudonym). Fields *telephoneNumber*, *pseudonym*, *surname*, *givenName*, *serialNumber*(2 5 4 5), *organizationName*, *organizational-UnitName*, *stateOrProvincename*, *localityName* and *title* are optional. *Rfc822Name* should not be used in *Name* but it is necessary to use an attribute from a certificate extension *subjectAltNames*. The text is non-empty in the field *DirectoryString* and *UTF8String* coding must be used.

Extensions:
A certificate must contain the following extensions: *AuthorityKeyIdentifier*, *SubjectKeyIdentifier*, *KeyUsage*, *BasicConstraints*, *CRLDistributionPoints*, *AuthorityInfoAccess* where:

CRLDistributionPoints must contain HTTP address for direct CRL and subsequent HTTP address can be used for indirect CRL.

AuthorityInfoAccess must contain address of issuer certificates in HTTP address as ".p7c" or ".cer" file where ".p7c" must contain all certificates which are issued for the issuer (CA self-sign and cross-certificates). It can also contain HTTP address for OCSP (Online Certificate Status Protocol). **OCSP protocol is preferred to CRL**. OCSP response **with positive statement** extension is **mandatory for QC** issued by CA accredited **in the Slovak Republic**.

QCStatements must indicate the type of the certificate according to OID *id-etsi-qcs-QcCompliance* or *id-etsi-qcs-QcSSCD* and it is strongly recommended to include also *id-etsi-qcs-QcRetentionPeriod* to inform a relying party about the responsibility of registration data which will be archived and can be made available upon request **beyond the end of the validity period** of the certificate.

If ***CertificatePolicies*** is used then OID of the certificate policy must be connected with the HTTP address *id-qt-cps* containing the CPS document which includes descriptions of how one or more certificate policies are implemented. The usage of *UserNotice* field is not recommended for practical reasons because the current applications are not able to display *UserNotice* text correctly. The extension *CertificatePolicies* can contain more than one certificate policy OID e.g. some OIDs could specify specific rules for the issuer, user and relying party according to which the certificate can be used. According to Slovak legislation such **OIDs** of the certificate policy **identify mandates of** the private **key holder** who is a **physical person**.

The certificate can also contain other extensions and in this case some interoperability problems could occur when applications are not able to process them correctly.

6 CRL profile

CRL version must be v2 (value 1). CRL is defined in ITU-T Rec. X.509 (08/2008)| ISO/IEC 9594-8:2008 (E) [23] and RFC 5280 [3].

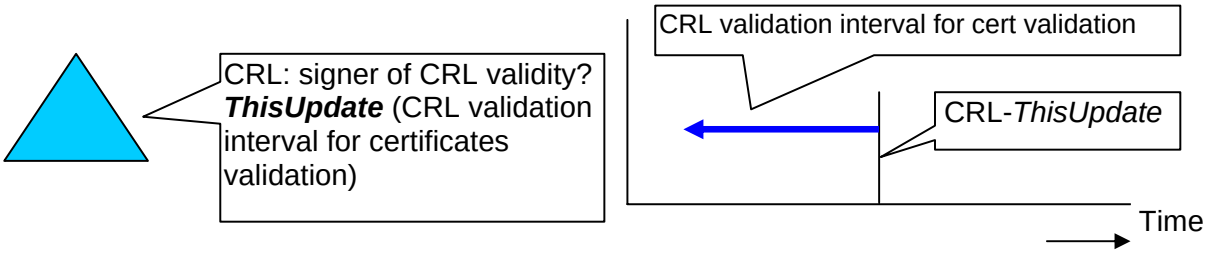


Figure 2: CRL and CRL usage period

Issuer Name:
It is the name of the CRL issuer (CA). The issuer name must be the same as the *Subject Name* of the CRL signer certificate.

***ThisUpdate Time*:**
The revocation information is applicable only in the time period which is before the time of *ThisUpdate* of CRL or OCSP. The field *thisUpdate* in CRL and OCSP is critical in validation to achieve the interoperability.
According to ITU-T Rec. X.509 (08/2008)|ISO/IEC 9594-8:2008 [23] the revocation date in CRL is the value contained in the *revocationDate* component. In CRL *thisUpdate* is the date/time at which CRL was issued.

The *revocationDate* field indicates the date and time at which the certificate was revoked and therefore should be considered as invalid.

- This *revocationDate* **shall not be later** than the *thisUpdate* time of CRL containing this revocation information.
- This *revocationDate* **shall not be earlier** than the *thisUpdate* time of CRL or OCSP (whose status was about this certificate) which does not contain this revocation information.

According to RFC 2560 *thisUpdate* is the time at which the status being indicated is known to be correct. And the Acceptance Requirements of response require: The time at which the status being indicated is known to be correct (*thisUpdate*) is sufficiently recent.
For that reason **in** the signature certificate **validation** the field ***nextUpdate*** **MUST NOT be used** because it is only a hint or wish when some new information could be available but it has no impact on the validity status.
The validation process, where e.g. the timestamp is used, MUST indicate the time from the *thisUpdate* field according to which the validity was determined in validation.
VALIDITY status according to CRL or OCSP where the time from *thisUpdate* field is before a timestamp time is only informative and not stable. For that reason the application in validation where the timestamp is not used must indicate the *thisUpdate* time according to which the validation status is presented (revocation could happen after *thisUpdate* time).

NextUpdate Time (OPTIONAL) field is used as a hint of the latest time for CRL issuance, without any impact on the validity status. CA which plans to finish CRL service shall not include this field in the latest CRL.

UserCertificate CertificateSerialNumber: A serial number of the revoked certificate. When indirect CRL is used then the issuer of the CRL signer certificate must be the same as the issuer of the revoked certificate or the issuer organization of CRL signer certificate is the “global” trust anchor (national root CA).

RevocationDate Time: Date and time of the certificate revocation where the rules defined in CRL *ThisUpdate* paragraph must be used to achieve the interoperability and a unique result.

CRL must contain the following extensions **crlExtensions**:
AuthorityKeyIdentifier and ***CRLNumber*** as a positive sequential number of CRL that is incremented by one during the CRL issuance.

CRL MUST be complete. CRL MUST include revocation information for all reasons (segmenting CRLs by reason code is not permitted) and also must include CA and end user certificates.

If CRL also contains expired certificates then it MUST contain an extension ***expiredCertsOnCRL*** of certificates that expired in the exact time as specified in the extension or after that time.

7 OCSP profile

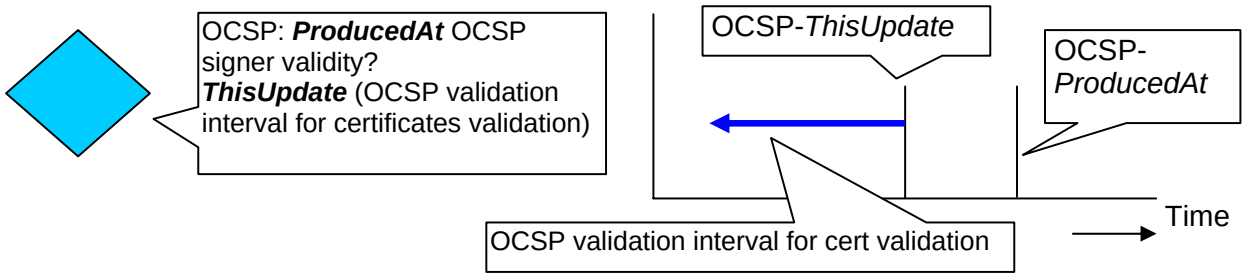


Figure 3: OCSP and OCSP usage period

7.1 OCSP request

OptionalSignature (OPTIONAL RFC 2560): The field *optionalSignature* of *OCSPRequest* must not be required for obtaining the OCSP response.
The field ***requestorName*** must not be required for obtaining the OCSP response.

The fields ***CertID*** are the following:

```
CertID ::= SEQUENCE {
  hashAlgorithm AlgorithmIdentifier,
  issuerNameHash OCTET STRING,     -- Hash of Issuer's DN
  issuerKeyHash OCTET STRING,      -- Hash of Issuer's public key
  serialNumber CertificateSerialNumber }
```

The algorithm must be only from the set of algorithms which are considered to be secure in a particular period. This set of algorithms is published in Algorithms and Parameters for Secure Electronic Signatures ETSI TS 102 176-1 or as a field of signature policy
http://www.nbusr.sk/en/electronic-signature/signature-policies/index.html
SHA256 is recommended for the interoperability.
OCSP request must be in DER coding and OCSP responder must accept OCSP request sent through HTTP or SMTP protocols (without SSL/TSL) and the request must be accepted by OCSP responder without any authentication restrictions.

7.2 OCSP response

The ***BasicOCSPResponse*** (RFC 2560) must contain the ***certs*** field which must include OCSP response signer certificate and the inclusion of the whole certification path is strongly recommended.
Responses *SEQUENCE OF SingleResponse* could contain the status of one or more certificates.

The fields *SingleResponse* are the following:

```
SingleResponse ::= SEQUENCE {
  certID CertID,
  certStatus CertStatus,
  thisUpdate GeneralizedTime,
  nextUpdate [0]EXPLICIT GeneralizedTime OPTIONAL,
  singleExtensions [1]EXPLICIT Extensions OPTIONAL }
CertStatus ::= CHOICE {
  good [0] IMPLICIT NULL,
  revoked [1] IMPLICIT RevokedInfo,
  unknown [2] IMPLICIT UnknownInfo }
RevokedInfo ::= SEQUENCE {
  revocationTime GeneralizedTime,
  revocationReason [0] EXPLICIT CRLReason OPTIONAL }
UnknownInfo ::= NULL
```

ThisUpdate Time: Date and time field where the same rules as defined in CRL *ThisUpdate* paragraph must be used to achieve the interoperability and a unique result.

NextUpdate Time (OPTIONAL) field can be used as a hint of the latest time for OCSP response issuance, without any impact on the validity status. This item is **not present** when the certificate **is expired**. The inclusion of this field is not recommended.

RevocationTime GeneralizedTime: is the same as CRL *RevocationDate* date and time of the certificate revocation where the rules defined in CRL *ThisUpdate* paragraph must be used to achieve the interoperability and a unique result.

SingleExtensions must contain positive statement *CertHash* according to national regulations (Slovak, also e.g. German). *CertHash* is a hash value from the certificate whose status is provided.

Definition of the extension *CertHash* adopted from Common PKI Optional SigG-Profile:

Common PKI Object Identifiers

```
id-commonpki OBJECT IDENTIFIER ::= {1 3 36 8 }
id-commonpki-at OBJECT IDENTIFIER ::= {id-commonpki 3}
id-commonpki-at-certHash OBJECT IDENTIFIER ::= {id-commonpki-at 13}

CertHash ::= SEQUENCE {
  hashAlgorithm AlgorithmIdentifier,
       -- The identifier of the algorithm that has been used
       -- the hash value below.
  certificateHash OCTET STRING
       -- The hash over DER-encoding of the entire PKC or AC (i.e. NOT a hash over tbsCertificate).
}
```

8 Time-stamp profile

Requirements of TSA in paragraph 2.1 from RFC 3161 are extended with the following points:

- the nonce parameter must be supported,
- when the accuracy field is used the validation process must indicate when the expected time is not before the time-stamp time corrected by accuracy interval.

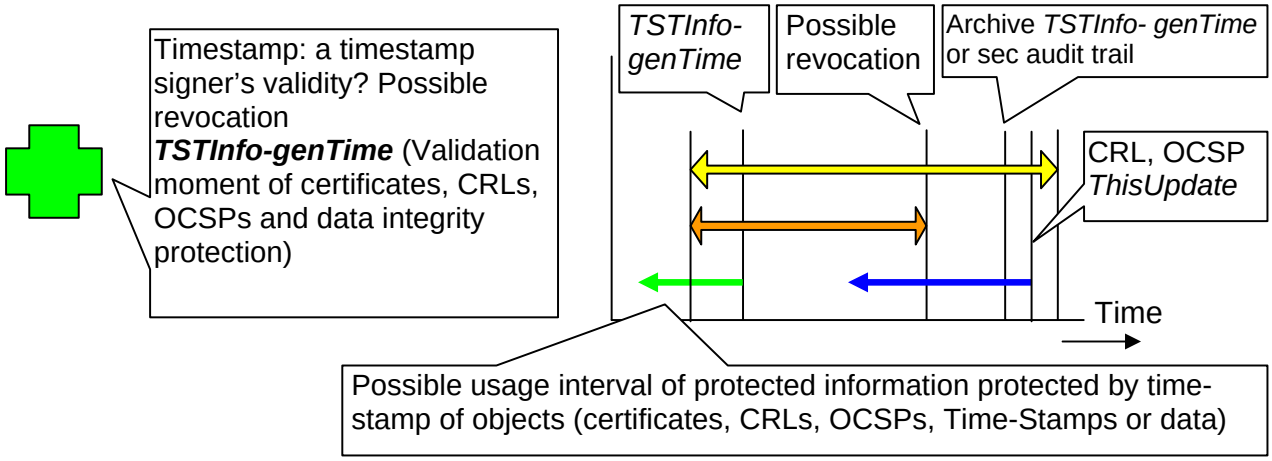


Figure 4: Time-stamp and time-stamp usage period

8.1 Time-stamp request

```
TimeStampReq ::= SEQUENCE  {
   version              INTEGER  { v1(1) },
   messageImprint       MessageImprint,
     --a hash algorithm OID and the hash value of the data to be time-stamped
   reqPolicy            TSAPolicyId              OPTIONAL,
   nonce                INTEGER                  OPTIONAL,
   certReq              BOOLEAN                  DEFAULT FALSE,
   extensions           [0] IMPLICIT Extensions  OPTIONAL  }
```

The presence of *nonce* and *certReq* set to TRUE are strongly recommended.

8.2 Time-stamp response

A signed document *TSTInfo* in CMS signature without *TimeStampResp* envelope must be used as a time-stamp.

```
TimeStampResp ::= SEQUENCE  {
      status                  PKIStatusInfo,
      timeStampToken          TimeStampToken     OPTIONAL  }

TimeStampToken ::= ContentInfo
     -- contentType is id-signedData ([CMS])
     -- content is SignedData ([CMS])

TSTInfo ::= SEQUENCE  {
   version                     INTEGER  { v1(1) },
   policy                      TSAPolicyId,
   messageImprint              MessageImprint,
     -- MUST have the same value as the similar field in TimeStampReq
   serialNumber                INTEGER,
     -- Time-Stamping users MUST be ready to accommodate integers up to 160 bits.
   genTime                     GeneralizedTime,
   accuracy                    Accuracy                 OPTIONAL,
   ordering                    BOOLEAN             DEFAULT FALSE,
   nonce                       INTEGER                  OPTIONAL,
     -- MUST be present if the similar field was present in TimeStampReq.
     -- In that case it MUST have the same value.
   tsa                         [0] GeneralName          OPTIONAL,
   extensions                  [1] IMPLICIT Extensions   OPTIONAL  }
```

The minimum accuracy must be one second. The certificate identifier *ESSCertIDv2* RFC 5816 of the TSA certificate MUST be included, *ESSCertID* is optional. *SignedData-certificates* MUST include the signer certificate and it is strongly recommended to include also the whole certification path up to a recognized "root" or "top-level certification authority".

The time-stamp *SignedData-crls* SHOULD contain CRL or OCSP responses for validation of certificates used before time-stamp time of objects protected by this time-stamp. When OCSP response is used the *SignedData-crls-[1]otherRevInfoFormat* MUST contain OID *id-pkix-ocsp-basic* (`1.3.6.1.5.5.7.48.1.1`) and *SignedData-crls-[1]otherRevInfo* MUST contain *BasicOCSPResponse*. Fields *SignedData-certificates* and *SignedData-crls* are updated by time-stamp users but not necessarily by the time-stamp issuer.

9 Signature Policy profile

The Slovak legislation http://www.nbusr.sk/en/electronic-signature/legislation/index.html in Appendix No. 1 to the NSA Decree No. 135/2009 Coll contains globally applicable restrictions of asymmetric algorithms with or without parameters and of hash algorithms for the products used for the qualified electronic signature creation and validation. Technical descriptions of such restrictions are represented in the machine processable signature policy in ASN.1 DER format on the page http://www.nbusr.sk/en/electronic-signature/signature-policies/index.html. For the reason that legislation defines global restrictions on algorithms, the separation of algorithms into specific sets of algorithms by signature policy is not important from the legal perspective. The separation defined in *AlgorithmConstraintSet* into specific sets by signature policy is important for an organization which requires stronger algorithms for specific purposes than are allowed by legislation. The approved list of signature policies published in *TrustedList.p7s* file must contain only policies with algorithms allowed by legislation for a specific time period:
http://www.nbusr.sk/en/electronic-signature/signature-policies/index.html
http://www.nbusr.sk/ipublisher/files/nbusr.sk/sign_policy/TrustedList.p7m .

Rules defined especially in RFC 3125 are used for automated processing of signatures during signature creation and validation processes.
When the signer has used the explicit signature policy electronic signature identifier then the verifier must use this signature policy to verify the signer expectations. The verifier could also apply its own signature policy which allows automatic verification of rules required by the verifier or systems where the signature is processed.
When the signature contains the reference on the signature policy the signature policy reference must include the following information:

- An unambiguous identifier of the **Algorithm** used to protect the signature policy information.

- A **hash** value of the signature policy information, which must be re-calculated and checked whenever the policy is passed between the issuer and signer/verifier. The hash is calculated over the DER value of the *SignaturePolicy* field without the outer type and length fields, and without the optional *signPolicyHash* field. A **hash** value of the ASN.1 signature policy used in XAdES reference is calculated over the DER value of the *SignaturePolicy* where the optional *signPolicyHash* field must be present

- From the ***SignPolicyInfo*** field the signature policy reference shall contain the **Signature Policy Identifier** (OID) of the signature policy that must uniquely identify the policy, and is specific to a particular version issued on the given date.

The signature policy reference should contain the following optional information:

- A field for the Signature Policy Issuer, which is the body responsible for issuing the Signature Policy. This may be used by the signer or verifier in deciding if a policy is to be trusted, in which case the signer/verifier shall authenticate the origin of the signature policy as coming from the identified issuer.

- A field that can hold information from the field ***fieldOfApplication*** of Signature Policy. This field holds, in general terms, the general legal/contract/application contexts in which the

signature policy is to be used and the specific purposes for which the electronic signature is to be applied.

- URI which contains the web URI or URL reference to the DER encoded signature policy.

The signature creation and validation process where the rules from the signature policy are used for automatic processing are based on the following profile where the *AlgorithmConstraintSet* contains groups of algorithms separated on the basis of responsibility of the creator of objects (the creator of objects must fulfil algorithm constrains in each object created by creator):

```
AlgorithmConstraintSet ::= SEQUENCE {   -- Algorithm constrains 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, -- issuer of Attribute Authority
 tsaCertAlgorithmConstraints [4]    AlgorithmConstraints OPTIONAL  -- issuer of TimeStamping
    -- Authority certificates
}
AlgorithmConstraints ::= SEQUENCE OF AlgAndLength
AlgAndLength ::= SEQUENCE {
    algID           OBJECT IDENTIFIER,
    minKeyLength    INTEGER     OPTIONAL, -- Minimum key length in bits
    other       SignPolExtensions OPTIONAL   }
```

Conditions of *signerAlgorithmConstraints* are applied on signature fields created by signer (AdES and the timestamp signature RFC 3161 and also on fields of these signatures where the used algorithms must be in permitted set of algorithms – they are not fixed on one type of the algorithm by the field type definition e.g. *SigningCertificate*). In particular, conditions apply to attributes which contain the hash value like *MessageImprint* of timestamp or to other attributes in signature like:

```
SignedData.digestAlgorithms,
SignedData.SignerInfo.digestAlgorithm,
SignedData.SignerInfo.signedAttrs.MessageDigest,
SignedData.SignerInfo.signedAttrs.OtherCertID,
SignedData.SignerInfo.signedAttrs.SigningCertificateV2,
SignedData.SignerInfo.signatureAlgorithm,
SignedData.SignerInfo.signature.
```

Conditions of *eeCertAlgorithmConstraints* are applied on the end user certificate signature, CRL signature and OCSP responses signature - used in validation of *eeCert*.
Conditions of *caCertAlgorithmConstraints* are applied on the CA certificate signature, CRL signature and OCSP responses signature - used in validation of *caCert*.
Conditions of *aaCertAlgorithmConstraints* are applied on the Attribute Authority certificate signature, CRL signature and OCSP responses signature - used in validation of *aaCert*.
Conditions of *tsaCertAlgorithmConstraints* are applied on the Time-Stamping Authority certificate signature, CRL signature and OCSP responses signature - used in validation of *tsaCert*.

The signature policy defines rules in the field *signerAlgorithmConstraints* which must be applied by the signer of ES and on the signature time-stamp. Subsequently, the verifier of ES and the signature time-stamp use the attributes *signerAlgorithmConstraints* from the signature policy which was valid at the signing time indicated in the signature time-stamp and its identifier was included in the signed attributes *id-aa-ets-sigPolicyId* by signer.
If the signer has not included the identifier of the signature policy in *id-aa-ets-sigPolicyId* or the archive time-stamp containing the time beyond the time of the signature policy validity indicated in *id-aa-ets-sigPolicyId*, then the verifier will use the signature policy being valid in the time indicated in the validating timestamp (archive timestamp). The signature policy which is used for validation in other time than the signing time (signature timestamp time or time from a secure audit trail) like for archive timestamp validation is called **maintenance** signature policy or simply the **maintenance policy** (cf. ETSI TS 102 176-1 informative Annex H). The management of secure publishing of applicable signature policy and its selection is out of the scope of the present document e.g. defined in http://www.nbusr.sk/en/electronic-signature/signature-policies/index.html .

10 Trusted List and TSL profile

The machine processable forms of trusted lists which can be used in automated public electronic services are defined in Commission Decision 2010/425/EU [16] and [26]. The trusted list according to Commission Decision 2010/425/EU [16] and [26] is used as a trusted source of values used in validation procedures where the **result is the type** of certificate determined according to supervised/accredited system of Member States **as**:

- Not under supervised/accredited system of Member States,
- Qualified,
- Qualified which are based on SSCD.

The trusted list created according to CD 2010/425/EU does not contain enough information for determination of the certificate validity for applications which are not able to use as a trust anchor any X.509 certificate listed in this trusted list which is not a self-signed certificate. For that reason such validation processes must use other representation of 'Service digital identity' (Sdi) (used as a trust anchor) e.g. to create a temporary cross-certificate of Sdi X.509 certificate by a self-signed certificate trusted for validation application and additional source of revocation information and another type of trusted lists which can be used as a trusted source of actual and historical information used in validation processes e.g. actual and historical approved signature policies or CRL/OCSP responses with the status of actual or expired certificates.
Such information is published e.g. in the signed structured TXT document like http://www.nbusr.sk/ipublisher/files/nbusr.sk/sign_policy/TrustedList.p7m on the page http://www.nbusr.sk/en/electronic-signature/signature-policies/index.html where the usage period of the self-signed certificate or signature policy could be shortened.

Trusted lists could contain for one signer certificate more than one issuer certificate e.g. as a cross-certificate what is the case when one CA is accredited by national roots in more than two Member States (ICA in both Czech and Slovak Republic). It is a practical situation which can happen during the signature creation or in signature validation where a possibility of one or more certification paths through a cross-certificate or intermediate self-signed certificate can occur. Applications must be able to choose the correct path or ignore intermediate self-signed certificate. The configuration of the application is according to information from the trusted list and signature policy (If used) able to choose the acceptable root CA certificates but if more than one path is acceptable then it is up to the user to decide which path will be used e.g. the path according to Czech trusted list and Czech legislation or according to Slovak trusted list and Slovak legislation rules while one CSP is included in both trusted lists.

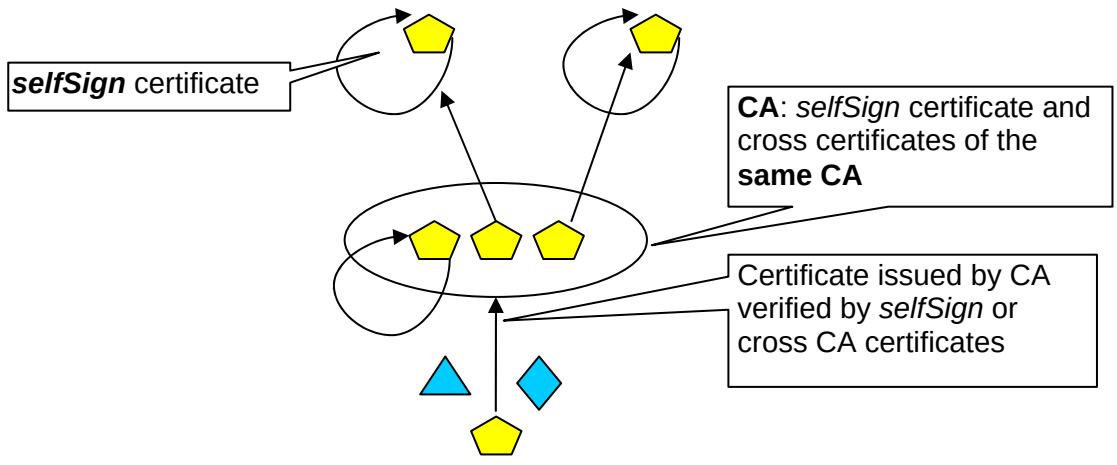


Figure 5: Cross-certificates and 3 possible certification paths with 3 trust anchors (as *selfSign*)

11 CMS AdES profile

QES based on CMS AdES must contain the signed attribute *SigningCertificateV2* defined in RFC 5035. According to this profile it is strongly recommended that *SigningCertificateV2* contains together with the signer certificate reference the references of the whole certification path which must be used for signature validation. The first certificate reference MUST be the signer certificate, the order of other certificate references is not defined and references on attribute certificate or cross-certificates could be included. The inclusion of ordered references only of the one whole certification path is recommended.
CMS advanced electronic signature (CAdES) defined in ETSI TS 101 733 according to this profile must contain certificates of the whole certification path and must be at least in any of the following forms:

- CAdES BES/EPES
- CAdES with signature time-stamp (CAdES-T)
- CAdES with Archival time-stamp (CAdES-A)

By this profile it is strongly recommended that QES based on CAdES signatures contains the signed attribute *ContentHints* with *contentDescription* containing MIME header attribute:

- *Content-Type* and
- optional *Content-Disposition* header field where the parameter *filename* MUST be used and contains the same value as *name* in *Content-Type* when *name* is used.

Content-Type is mandatory. *Content-Disposition* is optional.

The purpose of the MIME *Content-Type* field is to describe the data being signed fully enough that the receiving user agent can pick an appropriate agent or mechanism to present the data to the user, or otherwise deal with the data in an appropriate manner.

The archive timestamp hash calculation is according to CD 2011/130/EU. CD 2011/130/EU contains the following: All attributes of CAdES which are included in the archive timestamp hash calculation (ETSI TS 101 733 V1.8.1 Annex K) MUST be in DER encoding and any other can be in BER encoding to simplify one-pass processing of CAdES.
It means that according to ETSI TS 101 733 V1.8.3 (2011-01) and also V1.8.1 Annex K the "unsignedAttrs [1] IMPLICIT SET SIZE (1..MAX) OF" is not included in the hash calculation and is not DER encoded (it is BER encoded).
For the reason that "unsignedAttrs [1] IMPLICIT SET SIZE (1..MAX) OF" is BER encoded, the unsigned attributes MUST NOT be reordered when at least one archive timestamp is present. The DER encoding (ITU-T X.690| ISO/IEC 8825-1) requires an ordered SET and SET OF.

QES based on CAdES must contain the following signed attributes:
Mandatory:

```
id-contentType OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549)
            pkcs(1) pkcs9(9) 3 }
id-messageDigest OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549)
            pkcs(1) pkcs9(9) 4 }
id-signingTime OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549)
            pkcs(1) pkcs9(9) 5 }
id-aa-signingCertificateV2 OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
        rsadsi(113549) pkcs(1) pkcs9(9) smime(16) id-aa(2) 47 }
```

Conditional:

```
id-aa-ets-sigPolicyId OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549)
            pkcs(1) pkcs9(9) smime(16) id-aa(2) 15 }
id-aa-contentHint OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549)
            pkcs(1) pkcs-9(9) smime(16) id-aa(2) 4}
```

QES based on CAdES-T must contain the following unsigned attributes:
Mandatory:

```
id-aa-signatureTimeStampToken OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
            rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 14}
```

QES based on CAdES-A must contain the following unsigned attributes:
Mandatory:

```
id-aa-signatureTimeStampToken OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
            rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 14}
id-aa-ets-archiveTimestampV2  OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
            rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 48}
```

Conditional:

```
id-aa-ets-certValues OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) rsadsi(113549)
            pkcs(1) pkcs-9(9) smime(16) id-aa(2) 23}
id-aa-ets-revocationValues OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
        rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) id-aa(2) 24}
```

The usage of *id-aa-ets-certValues* and *id-aa-ets-revocationValues* are not recommended. The preferred fields are the *SignedData-certificates* and the *SignedData-crls*.

When the *id-contentType* OID *id-data (1.2.840.113549.1.7.1)* of *SignerInfo - SignedAttributes* (RFC 5652) is not able to identify uniquely the visualization type of signed data then the signed attribute *id-aa-contentHint* (RFC 2634, RFC 5035, RFC 5911) is recommended to be included and contains the field *contentDescription* with the text *Content-Type:* which specifies the MIME type (RFC 2045) of visualization and optional *name* parameter (or *Content-Disposition: attachment; filename*) which contains the proposed file name of signed data in additional processing of signed data e.g. in export and provides a hint of the possible file extension name.

An example of *ContentHints* usage defined in ETSI TS 101 733 V1.8.3 (2011-01) paragraph 5.10.3 content-hints Attribute:

```
Attribute SEQUENCE {
 attrType OBJECT IDENTIFIER 1.2.840.113549.1.9.16.2.4 id-aa-contentHint
 attrValues SET {
  ContentHints SEQUENCE {
   contentDescription UTF8String 'MIME-Version: 1.0
    Content-Type: text/plain; charset=UTF-8; name="Document.txt"
    Content-Disposition: attachment; filename="Document.txt"'
   contentType OBJECT IDENTIFIER 1.2.840.113549.1.7.1 id-data
  }
 }
}
```

12 PDF AdES profile

QES based on PDF AdES must contain the signed attribute *SigningCertificateV2* defined in RFC 5035. According to this profile it is strongly recommended that *SigningCertificateV2* contains together with the signer certificate reference the references of the whole certification path which must be used for signature validation. The first certificate reference MUST be the signer certificate, the order of other certificate references is not defined and references on attribute certificate or cross-certificates could be included. The inclusion of ordered references only of the one whole certification path is recommended.
PDF advanced electronic signature (PAdES) defined in ETSI TS 102 778-3 according to this profile must contain certificates of the whole certification path and must be at least in the following form:

- PAdES BES/EPES
- PAdES with signature time-stamp (PAdES-T)
- PAdES with long term validation data with document time-stamp (PAdES-LTV)

QES based on PDF AdES:

- must be applied on the document encoded according to PDF/A-1/2 (ISO 19005-1/2) [9] or [25] or must be in PDF/A according to rules defined in ISO 32000-2 (profiles ETSI TS 102 778-3/4/5);
- must be created with the *SubFilter* value *ETSI.CAdES.detached*
- the document timestamp which can be used as signature timestamp must be created after CMS signature in PDF document and must contain the *SubFilter* value *ETSI.RFC3161;*
- must be validated with CRL or OCSP where the time from *thisUpdate* field must be after the signing time (signature timestamp or document timestamp following signature in PDF);
- **signature must not have any visual appearance** within the document content (invisible signature type) and must be created according to rules defined in profile ETSI TS 102 778-3 which can be extended with long term validation information of documents (LTV) and document time-stamp.
- PDF document time-stamp applied on the PDF before a validated signature in PDF document is used in the same way as the content-time-stamp in CAdES signature validation.

An example of document timestamp:

```
6 0 obj
    <<
        /Type /DocTimeStamp
        /Filter /Adobe.PPKLite
        /SubFilter /ETSI.RFC3161
        /Contents <00000> % values go here inside of <>
        /ByteRange [0 0 0 0 ] % values go here inside of []
    >>
endobj
```

13 XML AdES profile

QES based on XML AdES must contain the signed element *SigningCertificate*. According to this profile it is strongly recommended that *SigningCertificate* contains together with the signer certificate reference the references of the whole certification path which must be used for signature validation. The first certificate reference MUST be the signer certificate, the order of other certificate references is not defined and references on attribute certificates or cross-certificates could be included. The inclusion of ordered references only of the one whole certification path is recommended.
XML advanced electronic signature (XAdES) defined in ETSI TS 101 903 according to this profile must contain certificates of the whole certification path and must be at least in the following form:

- XAdES BES/EPES
- XAdES with signature time-stamp (XAdES-T)
- XAdES with Archival time-stamp (XAdES-A)

QES based on XAdES signatures must contain the signed element *DataObjectFormat* which must contain visualization type identification in *MimeType* element of data to which the reference element points after all reference transformation were applied in order to achieve a unique visualization. When signed data are intended to be exported from XML signature or the signature is detached then is strongly recommended that *DataObjectFormat* contains at least one <*xades:Description*> element which contains *Content-Type* MIME header field to allow a unique signed data identification for visualization or processing and to provide a hint of the possible file name and extension name in *name* parameter. The element <*xades:Description*> shall be used to indicate the encoding of the data, in accordance with the rules defined in RFC 2045; see an example of structured contents and MIME.

The purpose of the MIME *Content-Type* field is to describe the data being signed fully enough that the receiving user agent can pick an appropriate agent or mechanism to present the data to the user, or otherwise deal with the data in an appropriate manner.

QES based on XAdES must contain the following *SignedProperties*:
Mandatory:
`SigningTime`
`SigningCertificate`
`DataObjectFormat`
Conditional:
`SignaturePolicyIdentifier`

QES based on CAdES-T must contain the following *UnsignedSignatureProperties*:
Mandatory:
`SignatureTimeStamp`

QES based on CAdES-A must contain the following *UnsignedSignatureProperties*:
Mandatory:
`SignatureTimeStamp`
`ArchiveTimeStamp`
Conditional:
`CertificatesValues`
`RevocationValues`

Examples of usage *DataObjectFormat* ETSI TS 101 903 and <*xades:Description*>::

```
<xades:DataObjectFormat ObjectReference="...">
  <xades:Description>
    Content-Type: text/plain; charset=UTF-8; name="Document.txt"
  </xades:Description>
  <xades:MimeType>text/plain</xades:MimeType>
```

```
</xades:DataObjectFormat>
```

14 Signed document profile

The present document specifies profiles of signed documents which are explicitly enumerated e.g. in Commission Decision 2011/130/EU. Any other types of signed documents are not forbidden by this document. It also specifies the rules used to achieve the unique identification of the document type for the signed document visualization components which are critical for the QES usage.

14.1 TXT document

To achieve the interoperability as the basic type of a document signed electronically is TXT document in encoding scheme UTF-8 (UCS Transformation Format — 8-bit RFC 3629, UTF-8 is a multi-byte character encoding for Unicode http://www.unicode.org/charts/). UTF−8 character is backward-compatible with ASCII character (a 7-bit character set).

TXT document is usually used also as a transport format of data encoded in mark-up language (e.g. XML) or script languages. The documents signed by QES must contain all information for unique visualization of interpreted data. It means the QES signature must contain the unique identification of mark-up language type together with the visualization transformation in MIME-TYPE signature signed field of the signed document which was used by signer and must be used by verifier.

14.2 PDF document

PDF document signed by QES (CMS AdES or XML AdES) must be encoded according to PDF/A (ISO 19005-1/2). PDF/A is now the industry standard for archiving the digital documents. PDF document signed by QES (PDF AdES) must be prior to signing in PDF/A(ISO 19005-1/2) format or must be in PDF/A format according to rules defined in ISO 32000-2 (profiles ETSI TS 102 778-3/4/5).

14.3 Signed ZIP container

In situations when one or more documents must be signed together with the information of the document type the ZIP container with the following rules is used.

- The ZIP container contains all documents only in the root directory of ZIP.
- Compression method MUST be DEFLATE, which is described in IETF RFC 1951.
- The MIME type of the document SHALL be included in the *File comment* field of the ZIP central directory in the form “mimetype=” e.g. “mimetype=text/plain; charset=UTF-8”. If the file extension included in the ZIP central directory does not imply a unique type of the document then the "mimetype=" MUST be present in the *File comment* field.

For the reason that ZIP is not specified in easily accessible international standard and only mentioned in ETSI TS 102 918 V1.1.1 (2011-04) the following basic description from the internet sources is provided in this document. A ZIP file is identified by the presence of a central directory located at the end of the file what allows appending of new files. The directory stores a list of names of entries (files or directories) stored in the ZIP file, along with other metadata about the entry and an offset into the ZIP file, pointing to the actual entry data. This allows a file listing of the archive to be performed relatively quickly, as the entire archive does not have to be read to see the list of files. The entries in the ZIP file also include this information for redundancy.
The order of the file entries in the directory does not need to coincide with the order of the file entries in the archive. The number of the file entries must be the same as the number of the file entries in the directory when ZIP is used as the signed container of signed files.
Each entry is introduced by a local header with information about the file such as the comment, file size and file name, followed by optional "Extra" data fields, and then the possibly compressed,

possibly encrypted file data. The "Extra" data fields are the key to the extensibility of the ZIP format. "Extra" fields are not used for ZIP container which is signed according to this profile.

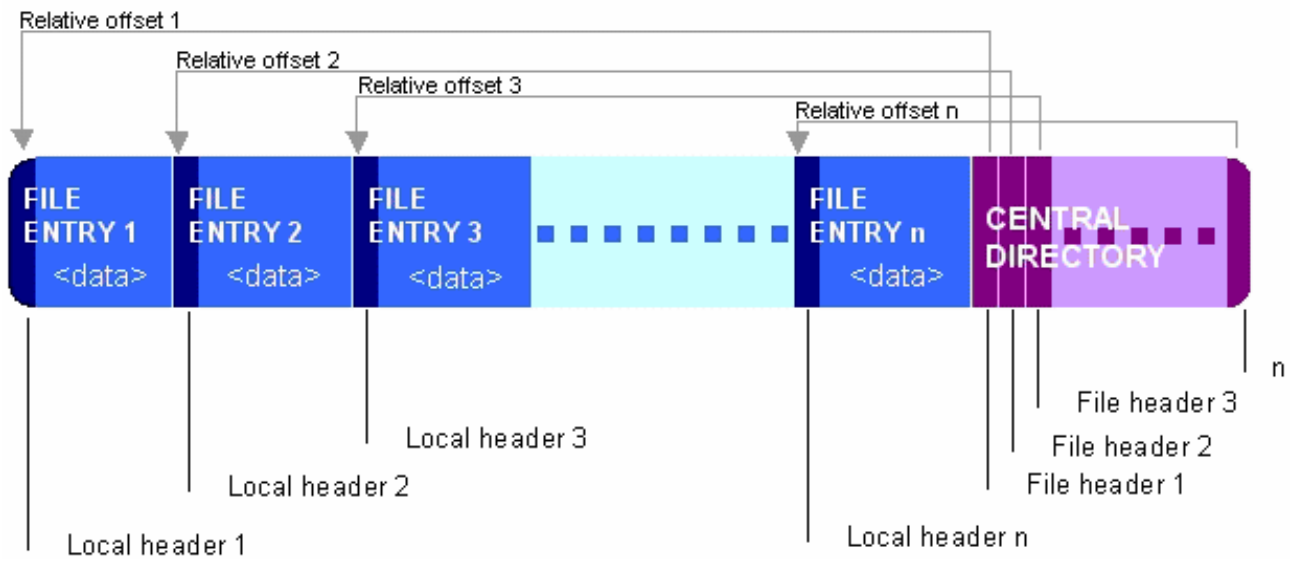


Figure 6: ZIP file structure

File headers

All multi-byte values are little-endian.

ZIP local file header		
Offset	**Bytes**	**Description**
0	4	Local file header signature = 0x04034b50 LSB-MSB
4	2	Version needed to extract (minimum)
6	2	General purpose bit flag
8	2	Compression method
10	2	File last modification time
12	2	File last modification date
14	4	CRC-32
18	4	Compressed size
22	4	Uncompressed size
26	2	File name length (*n*)
28	2	Extra field length (*m*)
30	*n*	File name
30+*n*	*m*	Extra field

The extra field contains a variety of optional data such as OS-specific attributes. It is divided into chunks, each with a 16-bit ID code and a 16-bit length.

This is immediately followed by the compressed data.
If bit 3 (0x08) of the general-purpose flags field is set, then the CRC-32 and file sizes are not known when the header is written. The fields in the local header are filled with zero, and the CRC-32 and size are appended in a 12-byte structure immediately after the compressed data:

ZIP data descriptor		
Offset	**Bytes**	**Description**
0	4	Local file header signature = 0x08074b50
4	4	CRC-32

8	4	Compressed size
12	4	Uncompressed size

The central directory entry is an expanded form of the local header:

ZIP central directory file header		
Offset	**Bytes**	**Description**
0	4	Central directory file header signature = 0x02014b50
4	2	Version made by
6	2	Version needed to extract (minimum)
8	2	General purpose bit flag
10	2	Compression method
12	2	File last modification time
14	2	File last modification date
16	4	CRC-32
20	4	Compressed size
24	4	Uncompressed size
28	2	File name length (*n*)
30	2	Extra field length (*m*)
32	2	File comment length (*k*)
34	2	Disk number where file starts
36	2	Internal file attributes
38	4	External file attributes
42	4	Relative offset of a local file header. This is the number of bytes between the start of the first disk on which the file occurs, and the start of the header. So if the ZIP file is not spanned across multiple disks and the header is to be found at the very beginning of the ZIP file, then these bytes will be 0.
46	*n*	File name
46+*n*	*m*	Extra field
46+*n*+*m*	*k*	File comment

After all entries, the local directory entries comes the end of a central directory record, which marks the end of the ZIP file:

ZIP end of a central directory record		
Offset	**Bytes**	**Description**
0	4	End of a central directory signature = 0x06054b50
4	2	Number of this disk
6	2	Disk where a central directory starts
8	2	Number of central directory records on this disk
10	2	Total number of central directory records
12	4	Size of a central directory (bytes)
16	4	Offset of start of a central directory, relative to start of archive
20	2	ZIP file comment length (*n*)
22	*n*	ZIP file comment

This ordering allows a ZIP file to be created in one pass, but it is usually decompressed by first reading of the central directory at the end.

15 Signature ZIP package profile for detached (external) signatures

To achieve the interoperability, it is necessary to bind a document being signed, its signatures and information necessary for the document signature verification and other documents related to the document being signed into a format whose processing is simple and commonly accessible. ZIP belongs to such commonly accessible formats. For the QES this document provides a profile of ETSI TS 102 918 V1.1.1 (2011-04) specification with additional rules according to Slovak legislation.

The document ETSI TS 102 918 specifies that the extending protection of archived document using techniques such as the archive time-stamp following construction of an ASiC container is not addressed by the ETSI TS 102 918 (2011-04) document and users verifying this container type using a CAdES implementation not aware of ASiC specific rules are warned that any data objects referenced by the signed data may not be checked. Such restrictions are as consequence of "Manifest" http://www.w3.org/TR/xmldsig-core/#sec-o-Manifest technique usage. For that reason **only ASiC-S is intended for interoperability usage**. The format ASiC-E is intended only for specific proprietary implementation of electronic signature usage and for that reason it is not recommended in this profile and must not be used for QES.

It is necessary to determine rules about naming of particular fields in ZIP file for simpler processing and ensuring the compatibility of applications which use or want to use the format mentioned above.

Basic rules:

- One document can be signed with one or more detached signatures (external CMS AdES or detached XML AdES).
- According to TS 102 918 V1.1.1 (2011-04) **only ASiC-S** must be used for QES.
- ZIP compression method MUST be DEFLATE.
- A file signed by detached signatures (external CMS AdES or detached XML AdES) must be stored in the root directory in the ZIP container. When more than one document is signed with one signature then all signed documents are included into the file "**package.zip**" whose format is described in the previous paragraph "Signed ZIP container".
- Detached signatures (CMS AdES or XML AdES), document time-stamp (internal CMS signature of *TSTInfo* document according to RFC 3161) and additional data must be stored into the sub-directory "META-INF".
- A file named "signatures.xml" must be stored into the sub-directory "META-INF" and containing the root element <asic:XAdESignatures> as specified in ETSI TS 102 918 V1.1.1 clause A.4, containing one or more detached ds:Signature elements calculated over the whole data object (a document in root ZIP directory) and conformant to XAdES. For ASiC-S ds:Reference SHALL be used to reference the data object in the container and the rules specified in ETSI TS 102 918 V1.1.1 clause A.5 SHALL apply. In case the referencing attributes (Id, URI or Type) are not present in ds:Reference element then a reference to the signed content is implied. Only canonicalization transformations according to CD 2011/130/EU are allowed in ASiC-S.
- The sub-directory "META-INF" may contain additional sub-directories: "Policy" (a list of used signature policy files) and "Other" (non-specified list of attached files).

File naming rules:

- File extension of signature ZIP package is for practical manipulation ".ZIP" and for automatic processing systems it is RECOMMENDED to use ".asics" (".scs" is allowed for operating systems and/or file systems not allowing more than 3 characters file extensions).

- File extension of internal CMS AdES signatures must be ".P7M".
- File extension of external CMS AdES signatures must be ".P7S".
- File extension of detached XML AdES signatures must be ".XML".
- File extension of document time-stamp (internal CMS signature of *TSTInfo* document according to RFC 3161) must be ".TST" or ".P7M".
- File extension of a certificate must be ".CER".
- File extension of a set of certificates (e.g. self-sign and cross-certificate) must be ".P7C".
- File extension of CRL must be ".CRL".
- File extension of OCSP response must be ".ORS".
- File extension of ASN.1 signature policy must be ".DER".

When detached signatures (CMS AdES or XML AdES) are used the identification of the signed document is realized according to signed information: CMS AdES - *ContentHints* (*contentDescription* contains the MIME header *Content-Type* with *name* parameter) and XML AdES - *DataObjectFormat* (<xades:Description> contains the MIME header *Content-Type* with *name* parameter) where the name attribute or *Content-Disposition* filename attribute contains only the file name of the document located in the root directory of ZIP (without path).

An example: "Content-Type: text/plain; charset=UTF-8; **name**="Document.txt"
Content-Disposition: attachment; **filename**="Document.txt"""

When the document time-stamp is used then the identification of time-stamped document is realized according to hash function and hash value which is located in the signed *TSTInfo* document in *messageImprint* field.

16 AdES creation and long-term validation

The present document defines unique conditions for the AdES creation and validation to achieve the interoperability and the long-term signature validation e.g. in EU.
The present document describes conditions based on TS 101 733 CMS Advanced Electronic Signatures (CAdES). The other AdES formats like TS 102 778 PDF Advanced Electronic Signature (PAdES) as well as TS 101 903 XML Advanced Electronic Signatures (XAdES) can use the same rules adapted according to their signature formats.

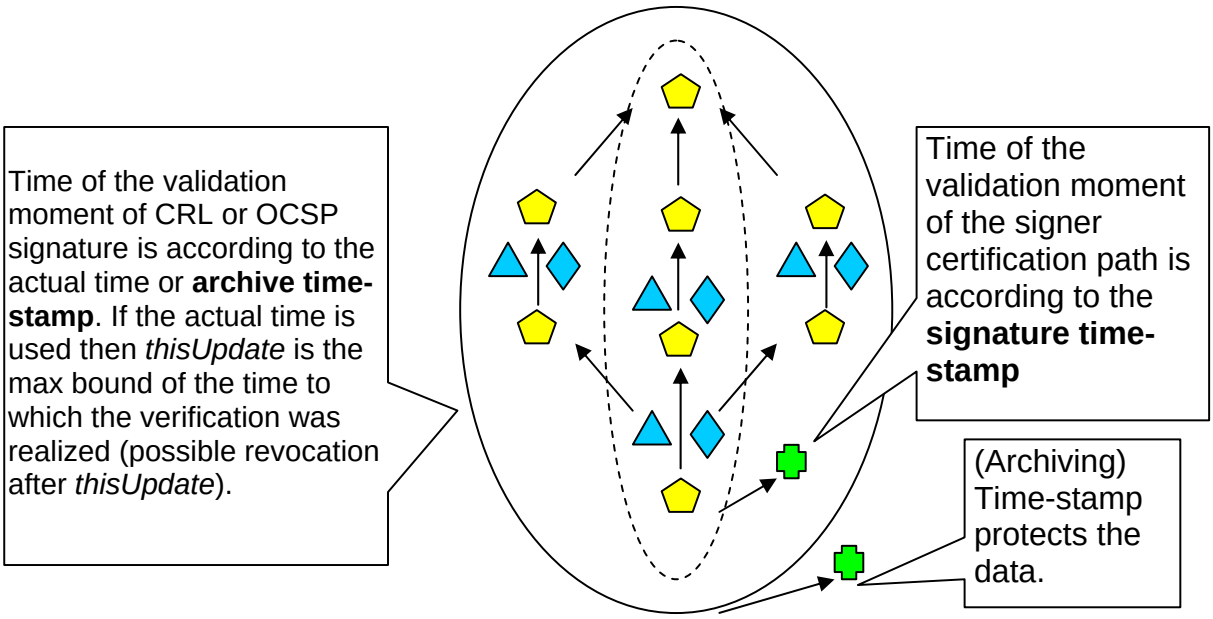


Figure 7: Certificate validated with indirectly issued CRL or OCSP

16.1 Signature creation

The main intent of this document is to achieve the interoperability. For that reason it will use only a limited set of possible signature attributes which are widely used and already interoperable and the attribute Archive time-stamp which is profiled in this document in order to gain the interoperable solution.

16.2 Used attributes

For the signature creation and validation there are used the following CAdES attributes where the following conditions must be met:
The RFC 5652 Cryptographic Message Syntax (CMS) defines:

```
SignedData ::= SEQUENCE {
  version CMSVersion,
  digestAlgorithms DigestAlgorithmIdentifiers,
  encapContentInfo EncapsulatedContentInfo,
  certificates [0] IMPLICIT CertificateSet OPTIONAL,
  crls [1] IMPLICIT RevocationInfoChoices OPTIONAL,
  signerInfos SignerInfos }
```

In RFC 5652 it is specified that the *SignedData-certificates* are sufficient to contain certification paths from a recognized "root" or "top-level certification authority" to all of the signers in the *SignedData-signerInfos* field. There may be more certificates than necessary, and there may be certificates sufficient to contain certification paths from two or more independent top-level certification authorities.
According to this profile the signer certificate MUST be included and also the whole certification path selected by the signer must be included during the signature creation process (CWA 14170),

what means, the signer must perform the initial certificate verification prior to the signature creation (CWA 14170).
In RFC 5652 it is specified that the *SignedData-crls* type gives a set of revocation status information alternatives. It is specified that the set contains information sufficient to determine whether the certificates and attribute certificates with which the set is associated are revoked. However, there MAY be more revocation status information than necessary or there MAY be less revocation status information than necessary. X.509 Certificate revocation lists (CRLs) [23] are the primary source of revocation status information, but OCSP responses are preferred when available. The *OtherRevocationInfoFormat* alternative is provided to support any other revocation information format without further modifications to the CMS. For example, Online Certificate Status Protocol (OCSP) Responses RFC 2560 can be supported using the *OtherRevocationInfoFormat*.
According to this profile the *SignedData-crls* SHOULD contain CRL or OCSP responses for validation of each certificate used in *SignedData-signerInfos* prior to signature archiving by the archiving timestamp or archiving by the secure audit trail. The field *SignedData-crls* MAY also contain CRL or OCSP responses which are not used in validation of signatures included in *SignedData-signerInfos*.

```
RevocationInfoChoices ::= SET OF RevocationInfoChoice
RevocationInfoChoice ::= CHOICE {
  crl CertificateList,
  other [1] IMPLICIT OtherRevocationInfoFormat }
OtherRevocationInfoFormat ::= SEQUENCE {
  otherRevInfoFormat OBJECT IDENTIFIER,
  otherRevInfo ANY DEFINED BY otherRevInfoFormat }
```

According to this profile when OCSP response is used the *SignedData-crls-[1]otherRevInfoFormat* MUST contain OID *id-pkix-ocsp-basic* (`1.3.6.1.5.5.7.48.1.1`) and *SignedData-crls-[1]otherRevInfo* MUST contain *BasicOCSPResponse*.
According to this profile the *BasicOCSPResponse* defined in RFC 2560 MUST contain at least OCSP signer certificate in *BasicOCSPResponse-certs*. The field *ResponderID* SHOULD contain a choice *byName*.

```
BasicOCSPResponse       ::= SEQUENCE {
  tbsResponseData      ResponseData,
   signatureAlgorithm   AlgorithmIdentifier,
   signature            BIT STRING,
   certs                [0] EXPLICIT SEQUENCE OF Certificate OPTIONAL }
ResponseData ::= SEQUENCE {
   version              [0] EXPLICIT Version DEFAULT v1,
   responderID              ResponderID,
   producedAt               GeneralizedTime,
   responses                SEQUENCE OF SingleResponse,
   responseExtensions   [1] EXPLICIT Extensions OPTIONAL }
ResponderID ::= CHOICE {
   byName               [1] Name,
   byKey                [2] KeyHash }
SingleResponse ::= SEQUENCE {
   certID                       CertID,
   certStatus                   CertStatus,
   thisUpdate                   GeneralizedTime,
   nextUpdate         [0]       EXPLICIT GeneralizedTime OPTIONAL,
   singleExtensions   [1]       EXPLICIT Extensions OPTIONAL }
CertStatus ::= CHOICE {
    good        [0]     IMPLICIT NULL,
    revoked     [1]     IMPLICIT RevokedInfo,
    unknown     [2]     IMPLICIT UnknownInfo }
RevokedInfo ::= SEQUENCE {
    revocationTime              GeneralizedTime,
    revocationReason    [0]     EXPLICIT CRLReason OPTIONAL }
UnknownInfo ::= NULL -- this can be replaced with an enumeration
```

The *SingleResponse-singleExtensions* contains the Positive Statement extension *CertHash* defined in the Common PKI Part 4: Operational Protocols and Common PKI and Part 9: SigG-Profile:
Common PKI Private OCSP - Extensions CertHash (Positive Statement)

```
id-commonpki-at-certHash OBJECT IDENTIFIER ::= {1 3 36 8 3 13}
CertHash ::= SEQUENCE {
    hashAlgorithm AlgorithmIdentifier,  -- The identifier of the algorithm that has been used
                                        -- the hash value below.
    certificateHash OCTET STRING }      -- A hash over the DER-encoding of the entire PKC or
                                        -- AC (i.e. NOT a hash over tbsCertificate).)
```

This extension is also used in indirect OCSP response as a positive statement that OCSP responder knows the status of the certificate and also provides the integrity protection of the certificate if the certificate is already expired and algorithms used in the certificate could be weak.
According to this profile the term **Signing Time** is the time of the signed object creation which can be provided by a trusted party, different from the signer, in a trusted verifiable way or is used for validation of such an object in a trusted way. Signing time is usually the signature timestamp time or archive/(PDF document) time-stamp time (from archive/(PDF document) timestamp which protects the verifying object) from the field *TSTInfo-genTime* included in the time-stamp.

Each archive-time-stamp attribute must be included in a new *UnsignedAttributes-Attribute*, it means the SET of *UnsignedAttributes-Attribute-attrValues* MUST contain only **one** archive-time-stamp attribute.

```
UnsignedAttributes ::= SET SIZE (1..MAX) OF Attribute
  Attribute ::= SEQUENCE {
    attrType OBJECT IDENTIFIER,
    attrValues SET OF AttributeValue }
```

The hash value calculation for the time-stamp MUST be according to rules defined in ETSI TS 101 733 V1.8.1 **Annex K Time-stamp hash calculation** and the values used in the hash calculations MUST be used from DER encoded signature without any DER modifications. It means the hash value is calculated over the DER fields directly without the order modification (SET, SET OF), the ASN.1 type or ASN.1 length modification (of DER Type, Length and Value). The field UnsignedAttributes ::= SET SIZE (1..MAX) OF is according to CD 2011/130/EU BER encoded and for that reason MUST not be ordered when at least one archive time-stamp is present.

16.3 Signature creation process

The signature creation application of QES prior to the signature creation MUST allow:

- to chose the signer certificate,
- the application to build the possible certification paths according to rules from the local configuration e.g. signature policy and certificates available in the local certificate store or through references provided in the certificate extension *authorityInfoAccess-caIssuers* where the issuer certificate ".cer" or all issuer cross-certificates ".p7c" are present.
- If more than one certification path is built the signer MUST be able to decide which certification path will be used for the signature validation and all certificates of that path will be included in the field *SignedData-certificates*.

Signing Certificate Reference Attribute MUST contain in the ESS *signing-certificate-v2* at least a reference to the signer certificate. The ESS *signing-certificate-v2* SHOULD contain the references to the whole certification path to protect it from attacks when inappropriate cross-certification could cause that the verifier will not be able to create the certification path as the signer expects.
The signer SHOULD add the signature-time-stamp and the whole certification path for the signature-time-stamp validation into the *time-stamp-SignedData-certificates* in the time-stamp token attribute.

16.4 Signature validation

Conditions which MUST be met:

- If the signature contains only the signer certificate, the signature validation application MUST allow building the possible certification paths according to rules from e.g. the explicit signature policy or if not used explicitly then according to local configuration e.g. the implicit signature policy and certificates available in the local certificate store or through references provided in the certificate extension *authorityInfoAccess-caIssuers* where the issuer certificate ".cer" or all issuer cross-certificates ".p7c" are present.

- If more than one certification path is built the verifier MUST be able to decide which certification path will be used for the signature validation and all certificates of that path will be included in the field *SignedData-certificates*.

Applications must be able to offer a choice of root certificates of certification paths for the signer and verifier when the application is not able to select at least one **correct** certification path with the preferred trust anchor (e.g. national root SK NSA/NBÚ). The following picture of a freeware application http://lockitin.webnode.sk/ is an example where the signer certificate is issued by SNCA2 CA. SNCA2 CA has a self-signed certificate and a cross-certificate issued by KCA NBU SR3 CA. In this case two root certificates can be used in the signature creation or signature validation processes.

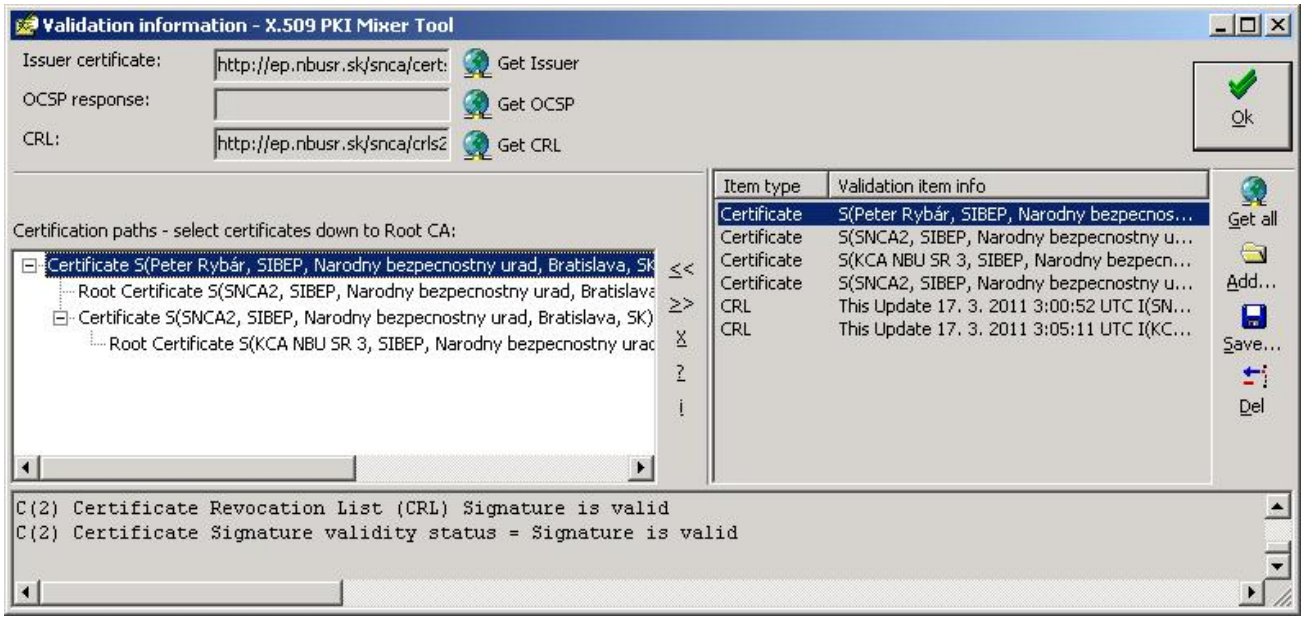


Figure 8: Certification paths with 2 self-signed root certificates and cross-certificate

The Signing Certificate Reference Attribute MUST contain at least a reference to the signer certificate in the signing-certificate. **If the *signing-certificate (-v2)* contains the references** to the certification path then **the verifier MUST use such certificates in the certification path building procedure**.
The verifier SHOULD add the signature-time-stamp if such signature-time-stamp was not added by signer or the signature-time-stamp is not trusted for the verifier and if the whole certification path for the signature-time-stamp validation is not present in the timestamp token then the verifier must add the whole certification path into the *time-stamp-SignedData-certificates* in the time-stamp token attribute.

The verifier, for each certificate in the signer certification path, adds the CRL or OCSP response to the *SignedData-crls* where at least one CRL or OCSP response for each certificate MUST be according to the attribute *thisUpdate* in compliance with the rules described in Annex A and B.

The CRL or OCSP responses are also added in the same way into the signature time-stamp token for the time-stamp certificate validation.

The signature time for the signature validation is determined by the signature timestamp *TSTInfo-genTime* value.

Annex A (informative) Long-term AdES verification

The first key element in AdES long-term verification is the usage of protected information of the signer certificate by the signer signature to prevent it from the substitution attack. This protection by *signingCertificate* also ensures the start point of the signature certificate verification for the verifier in case when non-repudiation is required and also **can be used to protect the whole certification path chosen by signer** if more than one trust anchor is available by cross-certification of CA certificate what can cause incorrect validation if the certification path is created towards the wrong trust anchor (e.g. such trust anchor is not acceptable by signer but the verifier doesn't know about it and has chosen it, like Slovak TSL or Czech TSL where the same ACA is present under different legal requirements). **If** the signer **signature does not protect the signer certificate** or also the whole certification path, then the signature protects only the integrity of the signed document and in this case the signer certificate is used only as a container of the public key which is used for the integrity protection of the signature but **not for non-repudiation** of the signature connection with the **signer identity** contained in the signer certificate or **fake revocation** of the signer certificate if somebody has issued some certificates with the same signer public key (the public key is publicly available and anybody can issue any certificate with such public key in the same way as is issued a cross-certificate for CA). If the same key is used in more than one certificate, then the substitution allows replacing the first signer certificate with another one which could be revoked or expired and therefore it repudiates the validity of the signature and any connection with the signer responsibility. Information in AdES signature is verifiable only while the revocation status of used certificate is provided by CSP under trusted certification path and while any used keys or algorithms are considered as secure. To make the signature verifiable also in later time when this condition is not met, the integrity of all signature elements must be protected by additional secure information which protects the whole signature together with the signed document in verifiable time when all data which are intended to be protected were considered as verifiable. It means **the signer certificate must be protected in the usage period** (e.g. by signature time-stamp) because any **misused situations of the signer key** after the certificate usage period **will not be indicated in CRL** or **OCSP**.

In the long-term signature verification of CRL (OCSP response) signature with the issuer certificate and subsequent verification of the issuer certification path validity by which CRL (OCSP response) is verified, it is necessary to fulfill a rule for the choice of CRL (OCSP response) in accordance with the issuance time of CRL.*thisUpdate* (OCSP.*thisUpdate*).

The rule for the choice of CRL (OCSP response) is following:

> **For the certificate verification, CRL (OCSP response) with the issuance time CRL.*thisUpdate* (OCSP.*thisUpdate*) is chosen in the way that every CRL (OCSP response) of the superior CA is issued after the issuance time CRL.*thisUpdate* (OCSP.*thisUpdate*) of the subordinate CA.**

This rule is necessary to follow in verification of CRL (OCSP response) signature validity that is issued later than the signing time because CA certificate which verifies issued CRL (OCSP response) could have become revoked later after the signing time by hierarchically superior CA for any other reason than is the key compromise. In any case the recommendation is to use the latest CRL which must contain a potential revocation of the certificate. The best way is to use CRL or OCSP response which also contains the potential revocation status of expired certificates as indicated by extensions OCSP.*ArchiveCutoff*, CRL.*expiredCertsOnCRL* or OCSP[certificate].*CertHash*.

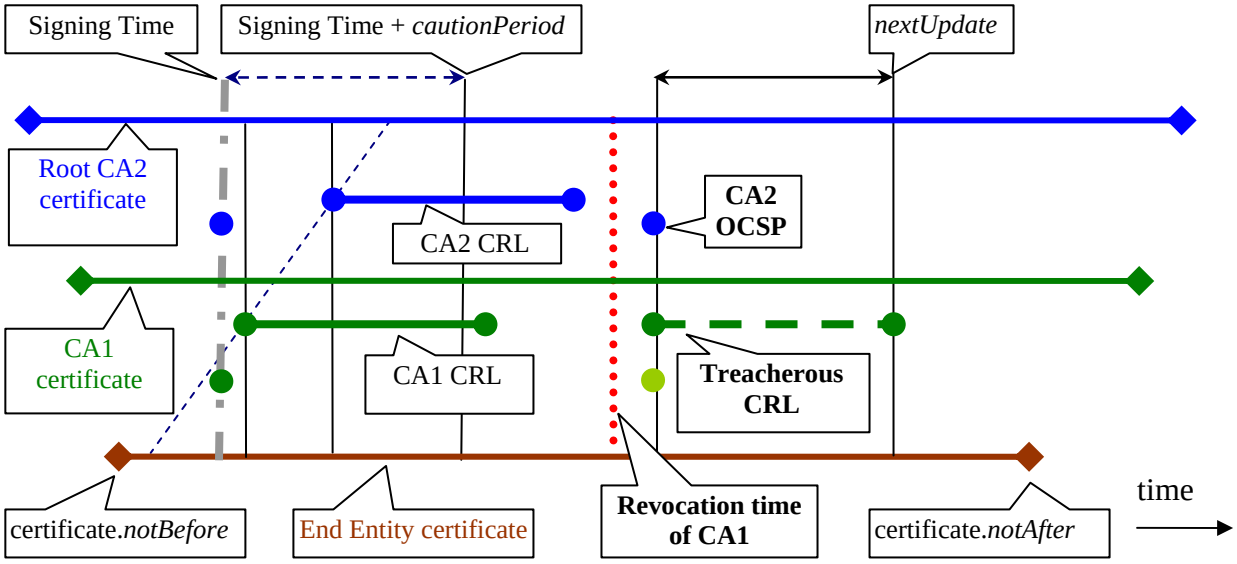


Figure 9: The shift of the time to which the CRL or OCSP is validated in the certification path

Previous rules are not practical and are too complicated because for the long term validation the time chaining of CRL and OCSP responses are critical and periodical archive timestamp protection and CRL or OCSP responses collection is too expensive for the correct management.

Presently in EU there are used two solutions which are acceptable secure, simple and more inexpensive then the previous ones:

- Trusted Service List (TSL) published according to COMMISSION DECISION 2010/425/EU [16], where the trust anchors are published in the form of X.509 certificate (contains issuer DN name, public key and parameters), validation conditions and attributes securely protected by TSL signer. End user certificates usually a qualified certificate, time-stamp certificate and indirect OCSP response signer certificate are directly validated with trust anchors from TSL. In this case there is not a problem with the time chaining and certification path building (the path contains only one certificate – the signer) and in this case each signature must contain the **whole certification path** where **the trust anchor** certificate is **also included** (**inclusion of the trust** anchor **is critical** to allow a selection of appropriate TSL in situation when one CA is included in more than one TSL). Signatures of National Trusted Lists are verified to a currently trusted certificate which is published in a trusted way according to national rules and also in EU Commission List Of The Lists (LOTL)
  http://ec.europa.eu/information_society/policy/esignature/eu_legislation/trusted_lists/ .

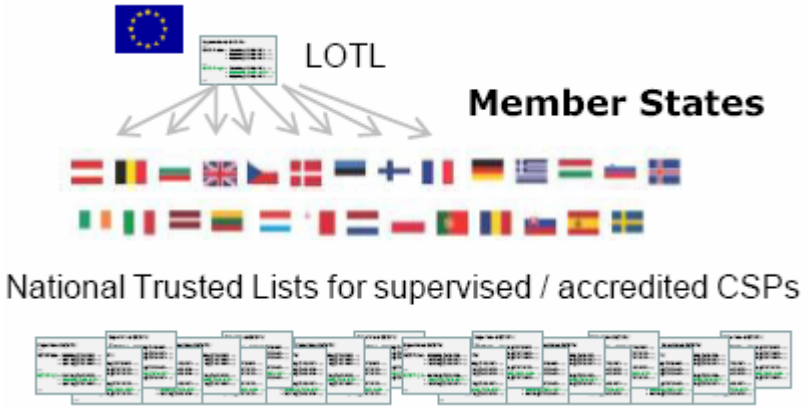


Figure 10: LOTL

Records about accredited/supervised providers in the list contain even the information about expired certificates which were considered as trust anchors in the past and which can be used for verification of data being protected by e.g. archive time-stamp (ATS).

- The same model as previous or hierarchical X.509 where the **CRL or OCSP response is not included** in signature for the reason that **CA** is able to **provide Indirect Positive OCSP** which provides the status of the expired certificates **for the period while the signature will be used**. The OCSP response with the positive statement is issued with actually non-expired certificate for validation and each signature **must contain the whole certification path** also **with the trust anchor** in X.509 certificate form.

In view of the fact that the information like the time-stamp, which protects the integrity in the certificate usage time, also contains certificates (CA certificates, time-stamp certificates, indirect CRL or OCSP signer certificates) and algorithms which must be verified later, this information must be also protected in the time period in which this information was usable and valid by additional recursive information which protects the whole data together with used CRLs or OCSP responses. The secure information which protects the signature and the signed document is usually represented as an archiving timestamp or a timestamp of records in secure archiving (audit) trail, where one timestamp protects collections of records with secure references which contain hash values of signatures and signed data.

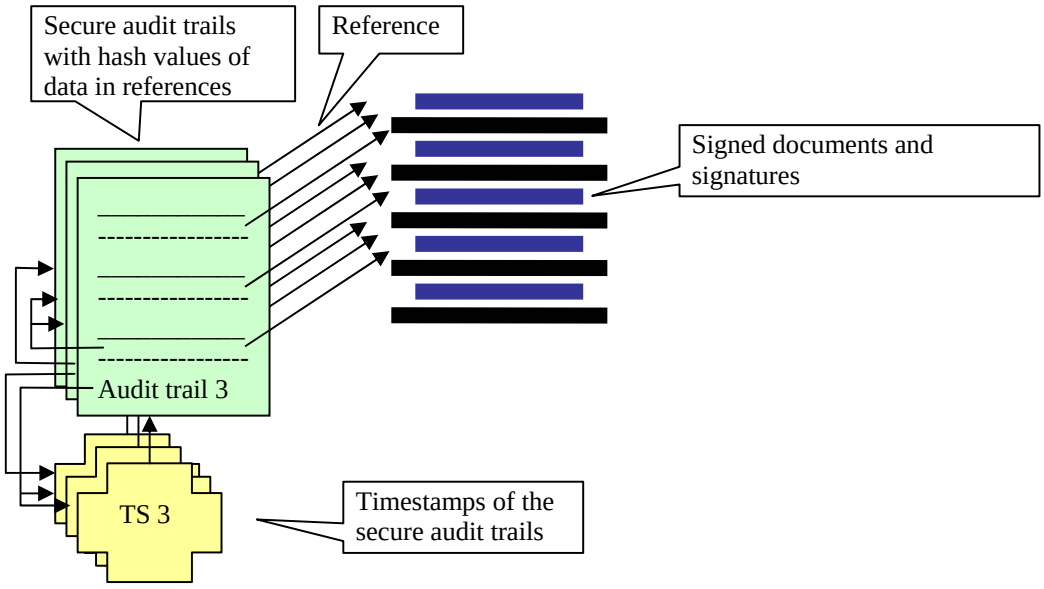


Figure 11: Time-stamped secure audit trails with hash values of data in references

AdES ETSI ESI and AdES CEN CWA documents propose the use of experimental unsigned attributes (elements) like *CompleteCertificateReferences, CompleteRevocationReferences, ESCTimeStampToken and TimestampedCertsCRLs* which were not able to solve many practical situations in the long term validation. For that reason the usage of these unsigned attributes (elements) is not recommended and is used only for a backward compatibility during validation of old signatures. New application MUST NOT use these attributes (elements). The useless problematic and expensive complexities of their usage are described in the following picture. Complete references are not protected by signature and are only as a hint and can be replaced with other values.

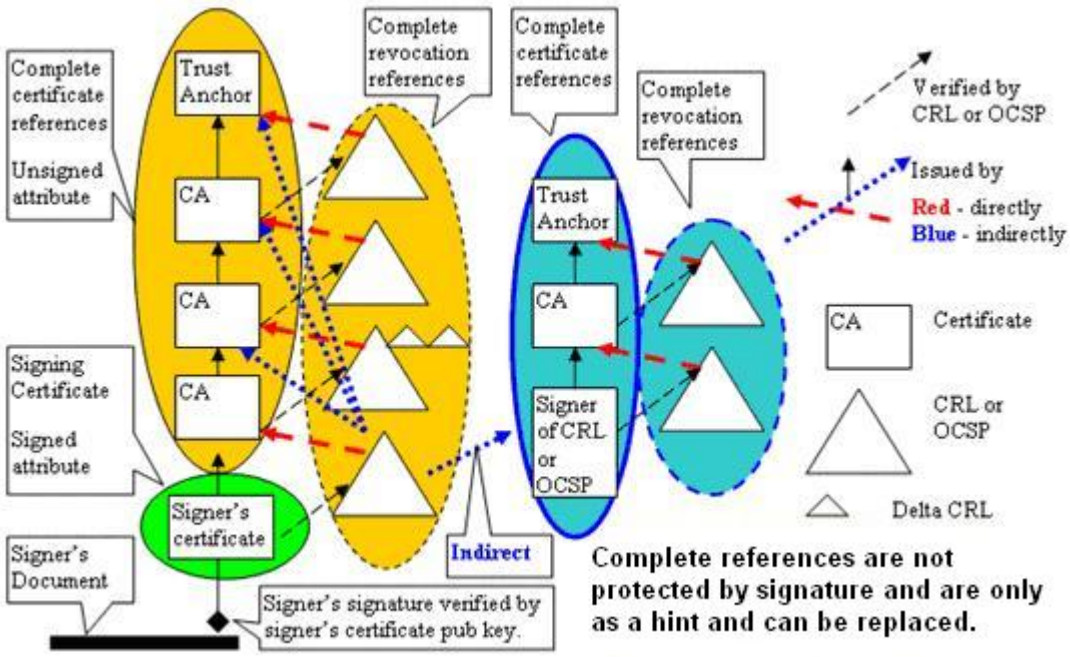


Figure 12: AdES protected by recursive archiving timestamp (archive time-stamp with AdES references on certificates and references on OCSPs and CRLs)

The figure 12 describes AdES as a container of information which is protected by recursive archiving timestamps.

Archiving of information for signature certificate verification by AdES-A form is useful in systems in which the signatures are not frequently generated and therefore the system is able to schedule the automatic process of archive time stamping (ATS) of signature information or previous ATS which contains only the whole certification path of certificates prior to expiration and algorithms which are secure but the CRL or OCSP for validations of protected TS are included in the following ATS which protects the previous ATS**. It means *thisUpdate* of CRL and OCSP response for validation of ATS1 is greater than the time from ATS2 which protects the ATS1 and CRL or OCSP response is included in ATS2. The validation process starts from the newest ATS and collects all CRLs and OCSPs which are also used for the previous ATS validation if suitable for validation**.

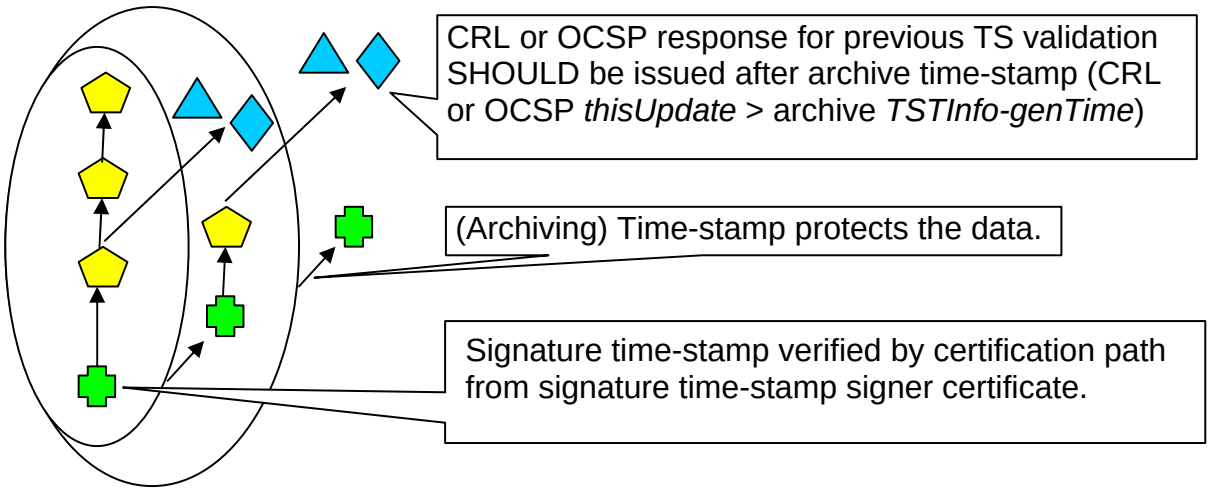


Figure 13: Validation to the time of archive time-stamp which protects the validated data

The signature time for the time-stamp signature validation is the most actual time or when time-stamp is protected with the following ATS than the signature time is after the ATS *TSTInfo-*

genTime value which allows determining the appropriate CRL and OCSP responses in compliance with the rules described in Annex B.
Before the archive-time-stamp is added the whole signer certification path MUST be included in the *SignedData-certificates* and also for each certificate of the document signer certification path the CRL or OCSP response SHOULD be included. The signature time-stamp token MUST contain the whole certification path of the time-stamp signer certificate.
After the verifier adds the archive-time-stamp the archive-time-stamp token SHOULD contain the whole certification path for the archive-time-stamp signature certificate validation and the CRL or OCSP response for validation of the previous time-stamp according to the attribute (CRL or OCSP)-*thisUpdate* where (CRL or OCSP)-*thisUpdate* is issued later (is >) than is the time from the actual archive-time-stamp-*TSTInfo-genTime*.
A new archive-time-stamp MUST be added periodically prior to expiration of certificates used in latest archive-time-stamp.

Such solutions are also problematic in the time when issuing CA is going to be expired and for that reason the usage of Indirect Positive OCSP responses which provide the status of already expired certificates is recommended.

Large systems which contain many signatures are not able to handle the archive re-time stamping of each individual AdES signatures because the scheduling system of each element in signatures is complicated and expensive. Large systems use the archive timestamp as an independent timestamp response which timestamps a document containing secure references to the signed document, signatures with complete certification path and revocation information. Re-time stamping in large systems is realized by a new timestamp of the document where the document contains all previous secure references created with actually secure hash algorithms and also contains secure references to the previous document of previous references with the previous archive timestamp and the actual archive timestamp contains all information (CRL or OCSP responses) for verification of the previous archive timestamp.

The verification process is performed recursively in backward order. Firstly the last archiving timestamp with the most actual revocation information is verified. Then the verification of the previous archiving time-stamp or AdES is realized to the time to which it was time-stamped by already verified archive time-stamp and also with the algorithm which was considered as secure as in the time of already verified archiving timestamp. The verification of the algorithm if the algorithm was secure in the past in the time of archiving is realized e.g. by information from signature policy which was valid in that time and is present in e.g. trusted list of history of signature policies. The long term verification must also have the trusted list of history of trust anchors which were used for the certification path verification in the past. History of trust anchors and signature policies could be, due to simplified use, joined to one trusted list which also contains the history signed with actually verifiable (not expired) certificate and under actually trust anchor in the certification path.

Annex B (informative) Verification of the certificate validity

ITU-T X.509 (08/2008) | ISO/IEC 9594-8:2008 [23], IETF RFC 5280 and IETF RFC 2560 define formats of the certificate for advanced electronic signatures and describe the rules for the certificate verification of certificates which are validated in the time of their usage, like the encryption certificate in the time when it was intended to be used for encryption. The process of the signature certificate validation is realized in the present time when we verify the certificate status being in the past. The unique rules for the signature certificate validation of the event created in the past are specified in this annex.

B.1 Verification based on OCSP

This clause provides certificate validity conditions for verification through OCSP. The OCSP response must contain the potential revocation which if happens must be only in the usage period of the certificate from certificate.*notBefore* to certificate.*notAfter*. This information about the status could be also provided after the expiration of the certificate certificate.*notAfter*. Thus, OCSP can contain *ArchiveCutoff* with the value which is smaller than the certificate expiration time or can contain a positive statement in the form of *CertHash* in the extension of the OCSP response about the fact that OCSP responder knows the certificate and its status.

Table B.1

1. **if** (certificate.*notBefore* **<** *OCSP*[certificate].*thisUpdate*) **and**
   (((OCSP.*ArchiveCutoff* <= certificate.*notAfter*) **and** (0 < OCSP.*ArchiveCutoff*)) **or**
   ((*OCSP*[certificate].*thisUpdate* **<=** certificate.*notAfter*) **and** (0 = OCSP.*ArchiveCutoff*)) **or**
   (*OCSP*[certificate].*CertHash* = certificate.*CertHash)*) **then**
2. **if** *OCSP*[certificate].*CertStatus* = *good* **then**
3. **if** (SigningTime **+** *cautionPeriod*) <= *OCSP*[certificate].*thisUpdate* **then**
   VALID
4. **else**
   INCOMPLETE VERIFICATION: obtaining a new OCSP response
5. **else**
   if *OCSP*[certificate].*CertStatus* = *revoked* **then**
   if SigningTime < *OCSP*[certificate].*revocationTime* **then**
   VALID
6. **else**
   INVALID
7. **else**
   INCOMPLETE AUTOMATIC VERIFICATION: OCSP does not know the current certificate status because *OCSP*[certificate].*CertStatus* = *unknown* It is necessary to obtain OCSP from another address or to verify through CRL.
8. **else**
   INCOMPLETE AUTOMATIC VERIFICATION: request to CA for CRL or OCSP issued in the period of the certificate validity + a period of time during which the record about the certificate revocation in CRL or OCSP is still present.

Where:

- OCSP.ArchiveCutoff - if ArchiveCutoff is not in the OCSP response, then its value is 0, otherwise the value stored in ArchiveCutoff is defined in accordance with RFC 2560.
- OCSP[certificate].*CertHash* is the hash of the certificate whose status OCSP returns (Common PKI private extensions). If this extension is found in OCSP, then the extension creates the positive information about the fact that OCSP knows the certificate and the status of the verified certificate.
- Certificate.*CertHash* is the hash of the certificate whose validity is verified.
- OCSP.*producedAt* is the time of the OCSP issuance.
- *OCSP*[certificate].*thisUpdate* is the time before which the correct information about the certificate status was known.

- *OCSP*[certificate].*nextUpdate* is the auxiliary time at or before which a newer information about the status of the certificate will be available. Responders MUST not include *nextUpdate* if the certificate is expired.
- Certificate.*notBefore* is the time of the beginning of verified certificate validity.
- Certificate.*notAfter* is the time after which the certificate will be expired.
- OCSP[certificate].*revocationTime* is the time of the certificate revocation.
- OCSP[certificate].*CertStatus* is the certificate status in OCSP which can have only 3 values.

Explanations to the following conditions:

1) OCSP is issued in the time of the certificate validity + a period of time during which the record about the certificate revocation for OCSP is known even after the certificate expiration.
2) The certificate was not revoked; it is not in OCSP.
3) The certificate status in OCSP is known after the signing time.
4) The certificate status in OCSP is not known after the signing time. It is necessary to ask for a new OCSP.
5) The certificate was revoked after the signing time, thus it is valid.
6) The certificate is revoked in OCSP prior to the signing time.
7) OCSP is not able to determine the certificate status, it is necessary to try other OCSP or CRL.
8) It is necessary to obtain OCSP or CRL issued in the time when the certificate has not been expired yet + a period of time during which the certificate status is still known in OCSP or CRL.

B.2 Verification based on CRL

This clause provides certificate validity conditions for verification through CRL. CRL must contain the potential revocation which if happens must be only in the usage period of the certificate from certificate.*notBefore* to certificate.*notAfter*. If the information about the certificate revocation is available in CRL even after the certificate expiration, then CRL must contain an extension *expiredCertsOnCRL* with the value that is smaller than the certificate expiration time certificate.*notAfter*.

Table B.2

1. **if** (certificate.*notBefore* **<** *CRL.thisUpdate*) **and**
   (((CRL.*expiredCertsOnCRL* **<=** certificate.*notAfter*) **and** (0 < CRL.*expiredCertsOnCRL*)) **or**
   ((*CRL.thisUpdate* **<=** certificate.*notAfter*) **and** (0 = CRL.*expiredCertsOnCRL*))) **then**
2. **if** certificate **is not in** *CRL* **then**
3. **if** (SigningTime **+** *cautionPeriod*) <= *CRL.thisUpdate* **then**
   VALID
4. **else**
   INCOMPLETE VERIFICATION: waiting for a new CRL
5. **else**
   if SigningTime < *CRL*[certificate].*revocationDate* **then**
   VALID
6. **else**
   INVALID
7. **else**
   INCOMPLETE AUTOMATIC VERIFICATION: request to CA for CRL issued in the time of the certificate validity + a period of time during which the entry about the certificate revocation in CRL is still present.

Where:

- If CRL.*expiredCertsOnCRL* is not present in the CRL extension, then its value is 0, otherwise the value is according to ITU-T X.509 (08/2008) [23].

- *CRL.thisUpdate* is the time before which the correct information about the certificate status was known.
- Certificate.*notBefore* is the time of the beginning of verified certificate validity.
- Certificate.*notAfter* is the time after which the certificate is expired.
- *CRL*[certificate].*revocationDate* is the date of the certificate revocation in CRL.

Explanations to the following conditions:

1) CRL is issued in the time of the certificate validity + a period of time during which the record about the certificate revocation is known in CRL even after the certificate expiration.
2) The certificate was not revoked; it is not in CRL.
3) The certificate status in CRL is known after the signing time.
4) CRL is not issued after the signing time, and it is necessary to wait for a new CRL.
5) The certificate was revoked after the signing time, thus it is valid.
6) The certificate is revoked prior to the signing time in CRL.
7) It is necessary to obtain CRL issued in the time when the certificate has not been expired yet + a period of time during which the certificate status is still known in CRL.

Annex C (informative) Smart Card Interface Profile

QES creation and validation application must be able to communicate with the SSCD (smart card) where the private signing key together with the signer qualified certificate is stored. SSCD usually contains the whole certification path used and included into the signature during the signature creation process. In the validation application the smart card is usually used as a trusted store of the trusted root certificates, which are read from the protected part of smart card according to information from Cryptographic Information Application (ISO/IEC 7816-15: 2004, PKCS#15).

In order to achieve the interoperability the standard EN 14890-1:2008 “Application Interface for smart cards used as Secure Signature Creation Devices - Part 1: Basic services” defines the functional and security requirements for a smart card intended to be used as a Secure Signature Creation Device according to the terms of the European Directive on Electronic Signature 1999/93 whereby a card compliant to the standard shall be able to produce a 'Qualified electronic signature' that fulfils the requirements of Article 5.1 of the Electronic Signature Directive and therefore can be considered as an equivalent to hand-written signatures. That standard additionally provides generic Identification, Authentication and Digital Signature (IAS) services and thereby contains all additional cryptographic services.
That standard will enable the development of interoperable cards issued by any card industry sector.

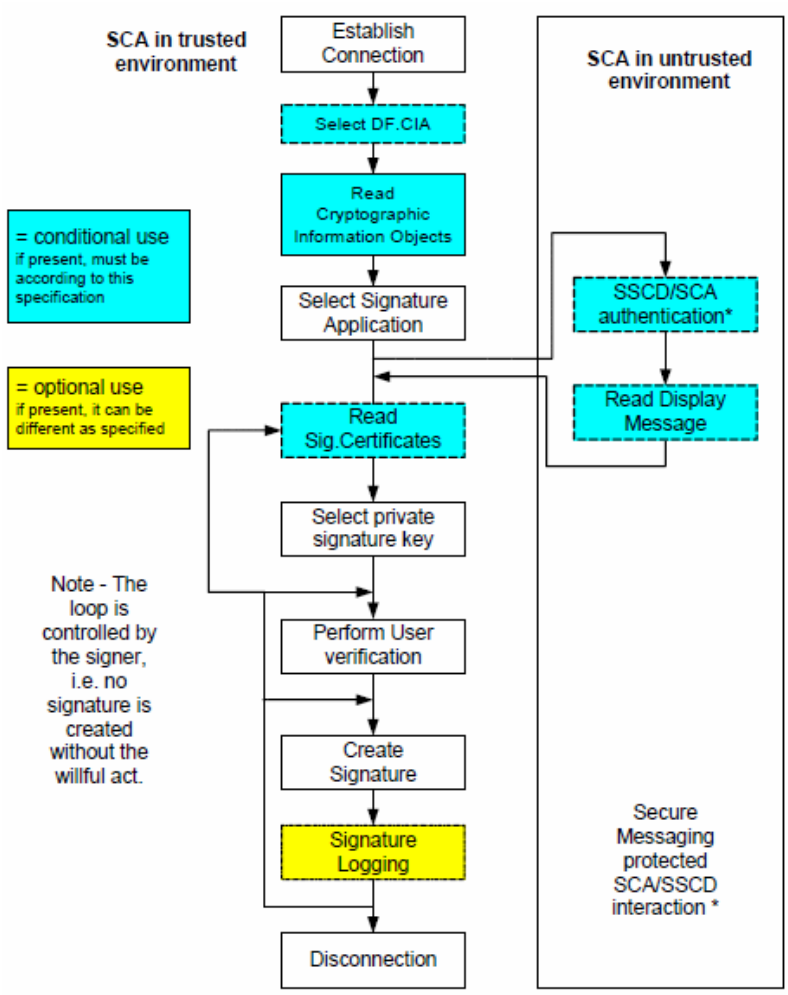
QES applications are able due to EN 14890-1:2008 to communicate directly with the cards through APDU (Application protocol data unit) or use certified application interface like PKCS#11. In this case the libraries which implement PKCS#11 interface and internally implement the functionality defined in EN 14890-1:2008 represent a key element for the interoperable and secure solution (according to certification of all QES creation and validation application components).

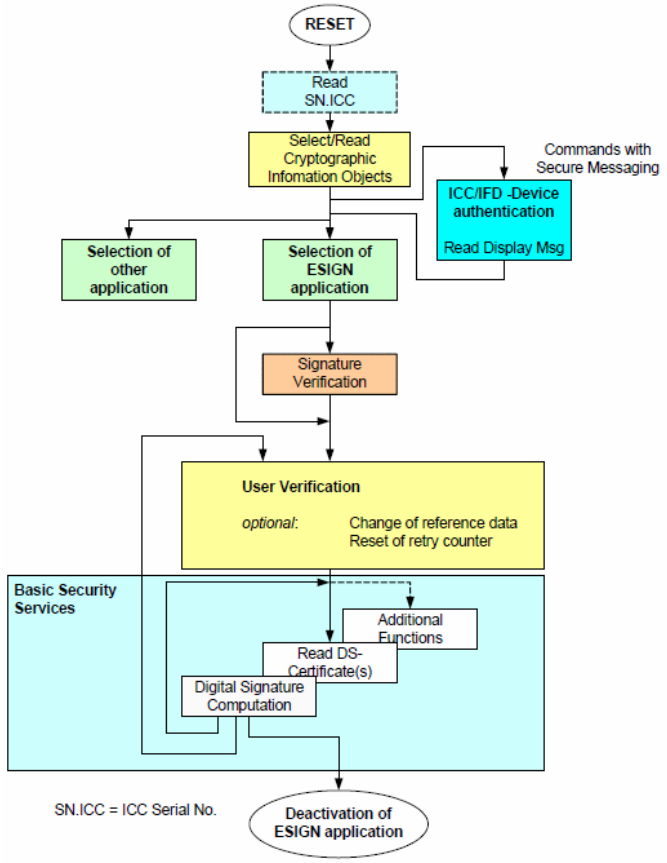
The following diagrams from EN 14890-1:2008 demonstrate a typical application flow when an ESIGN application is selected by its Application Identifier (AID). The RID (Registered Application Provider Identifier) is registered by the ISO registration authority and has the following value:
A0 00 00 01 67.

The entire AID has the following value
AID = A0 00 00 01 67 || “ESIGN” = A0 00 00 01 67 45 53 49 47 4E
With Category = ‘A.....’ (international)
PIX (Proprietary Application Identifier Extension)= ‘ESIGN’ = 5 bytes

When the application is to be offered internationally, then it needs an international AID. This consists of a Registered Application Provider Identifier (RID), which identifies Application Providers offering international IC card applications, and is issued by the ISO/IEC 7816-5 Registration Authority (TDC Services A/S). This is followed by a Proprietary Application Identifier Extension (PIX) which enables the Application Provider to differentiate between the different international IC card applications offered.

Each Application Provider is only allowed one RID per ISO/IEC 7816-5. It is up to the Application Provider to use the PIX to identify its different applications. If the organization already has an Issuer Identification Number (IIN), issued according to ISO/IEC 7812, organization may use this IIN instead of getting an RID per ISO/IEC 7816-5. In this case, AID will consist of IIN followed by ‘FF’ and a Proprietary Application Identifier Extension (PIX) as described above.


SCA in trusted environment
Establish Connection
SCA in untrusted environment
Select DF.CIA
Read Cryptographic Information Objects
= conditional use
if present, must be according to this specification
Select Signature Application
SSCD/SCA authentication*
= optional use
if present, it can be different as specified
Read Display Message
Read Sig.Certificates
Select private signature key
Note - The loop is controlled by the signer, i.e. no signature is created without the willful act.
Perform User verification
Create Signature
Signature Logging
Secure Messaging protected SCA/SSCD interaction *
Disconnection
* only if level of confidence is not achieved with organisational means


RESET
Read SN.ICC
Select/Read Cryptographic Infomation Objects
Commands with Secure Messaging
ICC/IFD -Device authentication
Read Display Msg
Selection of other application
Selection of ESIGN application
Signature Verification
User Verification
optional:
Change of reference data
Reset of retry counter
Basic Security Services
Additional Functions
Read DS-Certificate(s)
Digital Signature Computation
SN.ICC = ICC Serial No.
Deactivation of ESIGN application

Annex D (informative) Bibliography

Basic documents of the Slovak Republic legislation for electronic signature
http://www.nbusr.sk/en/electronic-signature/legislation/index.html
Qualified electronic signature formats
http://www.nbusr.sk/en/electronic-signature/approved-formats/index.html
Signature Policies for QES
http://www.nbusr.sk/en/electronic-signature/signature-policies/index.html

- IETF RFC 4158 "Internet X.509 Public Key Infrastructure: Certification Path Building"

NOTE: Available at http://www.rfc-archive.org/getrfc.php?rfc=4158

- IETF RFC 5217 "Multi-Domain PKI Interoperability" July 2008

NOTE: Available at http://www.rfc-archive.org/getrfc.php?rfc=5217

- IETF RFC 4853 (2007): "Cryptographic Message Syntax (CMS) Multiple Signer Clarification"
- IETF RFC 3447 (2003): "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1"
- ISO/IEC 8825-1:1998, Information technology — ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)
- IETF RFC 3279 (2002): "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile"
- IETF RFC 4055 (2005): "Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure - Certificate and Certificate Revocation List (CRL) Profile"
- IETF RFC 3281 (2002): "An Internet Attribute Certificate profile for Authorization"
- IETF RFC 3370 (2002): "Cryptographic Message Syntax (CMS) Algorithms"
- ETSI TS 102 176-1: Electronic Signatures and Infrastructures (ESI); Algorithms and Parameters for Secure Electronic Signatures; Part 1: Hash functions and asymmetric algorithms
- ETSI TR 102 038: "TC Security - Electronic Signatures and Infrastructures (ESI); XML format for signature policies"
- ETSI TS 102 778-3: Electronic Signatures and Infrastructures (ESI); PDF Advanced Electronic Signature Profiles; Part 3: PAdES Enhanced - PAdES-BES and PAdES-EPES Profiles
- ETSI TS 102 778-4: Electronic Signatures and Infrastructures (ESI); PDF Advanced Electronic Signature Profiles; Part 4: PAdES Long Term - PAdES-LTV Profile
- ETSI TS 102 778-5: Electronic Signatures and Infrastructures (ESI); PDF Advanced Electronic Signature Profiles; Part 5: PAdES for XML Content - Profiles for XAdES signatures
- ETSI TS 101 861: "Time stamping profile"
- EN 14890-2:2008: "Application Interface for smart cards used as Secure Signature Creation Devices - Part 2: Additional Services"
- ETSI TS 101 456: "Electronic Signatures and Infrastructures (ESI); Policy requirements for certification authorities issuing qualified certificates"
- ETSI TS 102 042: "Electronic Signatures and Infrastructures (ESI); Policy requirements for certification authorities issuing public key certificates"
- CWA 14167-1: "Security Requirements for Trustworthy Systems Managing Certificates for Electronic Signatures - Part 1: System Security Requirements"

- CWA 14167-2: "Security Requirements for Trustworthy Systems Managing Certificates for Electronic Signatures - Part 2: Cryptographic module for CSP Signing Operations with Backup - Protection Profile"
- CWA 14167-3: "Security Requirements for Trustworthy Systems Managing Certificates for Electronic Signatures - Part 3: Cryptographic module for CSP key generation services - Protection profile (CMCKG-PP)"
- CWA 14167-4: "Security Requirements for Trustworthy Systems Managing Certificates for Electronic Signatures - Part 4: Cryptographic module for CSP signing operations - Protection profile - CMCSO PP"
- W3C Recommendation (10 June 2008): "XML Signature Syntax and Processing (Second Edition)"

NOTE: Available at http://www.w3.org/TR/xmldsig-core/

- CWA 14169: "Secure Signature-Creation Devices "EAL 4+"
- NIST X.509 path validation test suite

NOTE: Available at http://csrc.nist.gov/pki/testing/x509paths.html
http://csrc.nist.gov/pki/testing/pathdiscovery.html

- Object Identifier (OID) Repository: ITU-T X.660 & X.670 Recommendation series (or ISO/IEC 9834 series of International Standards)

NOTE: Available at http://www.oid-info.com/

- FESA – Forum of European Supervisory Authorities,

NOTE: Available at http://www.fesa.rtr.at

- OID tree structure,

NOTE: Available at http://www.darmstadt.gmd.de/secude/Doc/htm/oidgraph.htm

- Common ISIS-MTT Specification for interoperable PKI applications. Version 1.1. 16 March 2004
- Internet Draft "X.509 Public Key Infrastructure: Additional Algorithms and Identifiers for DSA and ECDSA"

NOTE: Available at http://tools.ietf.org/html/draft-ietf-pkix-sha2-dsa-ecdsa-05

- Internet Draft "X.509 Public Key Infrastructure Time-Stamp Protocol (TSP) "

NOTE: Available at http://tools.ietf.org/html/draft-ietf-pkix-rfc3161bis-01

- TeleTrusT Deutschland e. V., "OID-Liste",

NOTE: Available at http://www.teletrust.de/index.php?id=171

European Commission http://ec.europa.eu/

- Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures

NOTE: Available at
http://eur-lex.europa.eu/smartapi/cgi/sga_doc?smartapi!celexapi!prod!CELEXnumdoc&lg=EN&numdoc=31999L0093&model=guichett

- IDABC stands for Interoperable Delivery of European eGovernment Services to public Administrations, Businesses and Citizens. - eSignature Agenda & Presentations

NOTE: Available at http://ec.europa.eu/idabc/en/document/7312

- European Network and Information Security Agency (ENISA)

NOTE: Available at http://www.enisa.europa.eu/

- PKIX Status Pages http://tools.ietf.org/wg/pkix/

Annex E History

Version	Date of issuing	Note	Editor
Version 1.0 No. 917/2011/IBEP/OEP-002	1 August 2011	First edition	Peter Rybár, NSA