A-Trust Gesellschaft für Sicherheitssysteme im elektronischen Datenverkehr GmbH.
Landstraßer Hauptstraße 5
Tel.: +43 (1) 713 21 51 – 0
Fax: +43 (1) 713 21 51 – 350
office@a-trust.at
www.a-trust.at

a.trust

Certificate and CRL Specification

Version: 1.6

Date: 2004-09-17

Table of contents

1 Scope of the document

This document primarily aims to describe the structure of the qualified a.trust certificate (a.sign Premium) which is also applicable to non-qualified certificates. If there are important and relevant exceptions to qualified certificates they are stated explicitely (e. g. see section 3) in this document.

2 End entity certificate contents

The basic fields of a certificate are defined in X.509 using ASN.1 syntax as follows:

```
Certificate ::= SEQUENCE {
    tbsCertificate          TBSCertificate,
    signatureAlgorithm      AlgorithmIdentifier,
    signatureValue          BIT STRING }

TBSCertificate ::= SEQUENCE {
    version                 [0] EXPLICIT Version DEFAULT v1,
    serialNumber            CertificateSerialNumber,
    signature               AlgorithmIdentifier,
    issuer                  Name,
    validity                Validity,
    subject                 Name,
    subjectPublicKeyInfo    SubjectPublicKeyInfo,
    issuerUniqueID          [1] IMPLICIT UniqueIdentifier OPTIONAL,
                            -- if present, version shall be v2 or v3
    subjectUniqueID         [2] IMPLICIT UniqueIdentifier OPTIONAL,
                            -- if present, version shall be v2 or v3
    extensions              [3] EXPLICIT Extensions OPTIONAL
                            -- if present, version shall be v3
}
```

2.1 version

X.509 defines Version type as follows:

```
Version ::= INTEGER { v1(0), v2(1), v3(2) }
```

Only version 3 certificates are issued (value for the version integer equals 2).

2.2 serialNumber

X.509 defines CertificateSerialNumber type as follows:

```
CertificateSerialNumber ::= INTEGER
```

All certificates issued by one CA have unique serial numbers (max. 16 bytes). The serial number is allocated automatically by the CA and cannot be used to carry information.

2.3 signature

X.509 defines AlgorithmIdentifier type as follows:

```
AlgorithmIdentifier ::= SEQUENCE {
    algorithm         OBJECT IDENTIFIER,
    parameters        ANY DEFINED BY algorithm OPTIONAL
}
```

The following algorithm is used in all issued certificates:

- sha1WithRSAEncryption { 1, 2, 840, 113549, 1, 1, 5 }

The optional parameters field is set to NULL.

2.4 issuer

X.509 defines the Name type as follows:

```
Name ::= CHOICE { rdnSequence RDNSequence }
RDNSequence ::= SEQUENCE OF RelativeDistinguishedName

RelativeDistinguishedName ::=
    SET SIZE (1..MAX) OF AttributeTypeAndValue

AttributeTypeAndValue ::= SEQUENCE {
    type        AttributeType,
    value       AttributeValue
}
AttributeType ::= OBJECT IDENTIFIER
AttributeValue ::= ANY DEFINED BY AttributeType

DirectoryString ::= CHOICE {
    teletexString           TeletexString (SIZE (1..MAX)),
    printableString         PrintableString (SIZE (1..MAX)),
    universalString         UniversalString (SIZE (1..MAX)),
    utf8String              UTF8String (SIZE (1.. MAX)),
```

```
    bmpString               BMPString (SIZE (1..MAX))
}
```

"The DirectoryString type is defined as a choice of PrintableString, TeletexString, BMPString, UTF8String, and UniversalString. The UTF8String encoding [RFC 2279] is the preferred encoding, and all certificates issued after December 31, 2003 MUST use the UTF8String encoding of DirectoryString." [rfc3280]

The issuer identity is represented by at least the following attributes:

Attribute	OID	Description	Value	ASN.1 type
countryName	{ id-at 6 }	Abbreviation for country	AT	printableString
organizationName	{ id-at 10 }	An informative unique name of the issuing organization	Name of a.trust, see below	utf8String
commonName	{ id-at 3 }	An informative unique (inside organization) name of the CA	a-sign-Premium-Sig-nn, etc.	utf8String
organizationalUnitName	{ id-at 11 }	see commonName	see commonName	utf8String

The field organizationName contains:
"A-Trust Ges. f. Sicherheitssysteme im elektr. Datenverkehr GmbH".

Explanation of commonName: -nn reflects the version of the CA used by a.trust for signing the certificate (e. g. a-sign-Premium-Sig-01).

2.5 validity

X.509 defines the validity type as follows:

```
Validity ::= SEQUENCE {
    notBefore          Time,
    notAfter           Time }

Time ::= CHOICE {
    utcTime                 UTCTime,
    generalizedTime         GeneralizedTime
}
```

Certificate validity dates through year 2049 are coded as UTCTime and validity dates after that as GeneralizedTime. UTCTime values are expressed in Greenwich Mean Time (GMT) and they include seconds as follows: YYMMDDhhmmssZ.

Validity period is set according to the certificate policy. (Normally a validity period of three years will be used).

2.6 subject

The subject field is coded with the same rules as the issuer field.

The subject identity is represented by the following attributes:

Attribute	OID	Description	ASN.1 type
countryName	{ id-at 6 }	Abbreviation for country	printableString
title	{ id-at 12 }	Title of subject	utf8String
surname	{ id-at 4 }	Family name of subject	utf8String
givenName	{ id-at 42 }	First name of subject	utf8String
commonName	{ id-at 3 }	Combination of subject's surname and givenName	utf8String
serialNumber	{ id-at 5 }	Unique identifier of subject	printableString
organizationName	{ id-at 10 }	Optional attribute	utf8String
organizationalUnitName	{ id-at 11 }	Optional attribute	utf8String

The commonName may contain a Pseudonym, which then is indicated as "Pseudonym: Abc Def". In this case the attributes title, surname and givenName will not be present.

The serialNumber attribute contains the Card Identification Number (CIN) which is unique and allows for persistent identity of the card holder, even when the certificate is renewed or the card is replaced. The Card Identification Number consists of 11 digits + 1 check digit (according to Luhn).

2.7 SubjectPublicKeyInfo

2.7.1 RSA-Key

The RSA-algorithm is used for both the qualified signature key certificate and the non-qualified authentication key certificate.

X.509 defines the SubjectPublicKeyInfo type as follows:

```
SubjectPublicKeyInfo ::= SEQUENCE {
    algorithm               AlgorithmIdentifier,
    subjectPublicKey        BIT STRING
}
```

The algorithm identifier is set to the following object identifier:

- rsaEncryption { 1, 2, 840, 113549, 1, 1, 1 }

The value for the subjectPublicKey BIT STRING is the DER-encoding of the ASN.1 type RSAPublicKey defined in PKCS #1 v1.5:

```
RSAPublicKey ::= SEQUENCE {
    modulus                         INTEGER,
    publicExponent                  INTEGER
}
```

2.7.2 ECC-Key

ECC is used for the qualified signature key certificate only.

According to X.509 the public key shall have the following syntax:

```
SubjectPublicKeyInfo ::= SEQUENCE {
algorithm AlgorithmIdentifier {{ECPKAlgorithms}},
subjectPublicKey BIT STRING
}
```

The elliptic curve public key is defined acc. to [X9.62].

```
AlgorithmIdentifier { ALGORITHM:IOSet } ::= SEQUENCE {
          algorithm ALGORITHM.&id({IOSet}),
          parameters ALGORITHM.&Type({IOSet}{@algorithm})
}

ECPKAlgorithms ALGORITHM ::= {
ecPublicKeyType,
...
}
ecPublicKeyType ALGORITHM ::= {
Parameters IDENTIFIED BY id-ecPublicKey
}
ALGORITHM ::= TYPE-IDENTIFIER
```

The object identifier id-publicKeyType represents the tree containing the object identifiers for each public key. It has the following value:

```
id-publicKeyType OBJECT IDENTIFIER ::= { ansi-X9-62 keyType(2) }
```

The object identifier id-ecPublicKey names the public key type defined in [X9.62]. It has the following value:

```
id-ecPublicKey OBJECT IDENTIFIER ::= { id-publicKeyType 1 }
```

The public key Parameters are defined in [X9.62] as a choice of three alternatives. a.trust uses the parameter “namedCurve”.

```
Parameters ::= CHOICE {
ecParameters ECParameters,
namedCurve CURVES.&id({CurveNames}),
implicitlyCA NULL
}

CurveNames CURVES ::= {
... { ID prime192v1 } ...
}

CURVES ::= CLASS {
&id OBJECT IDENTIFIER UNIQUE
}
WITH SYNTAX { ID &id }
```

The object identifier ellipticCurve represents the tree containing the object identifiers for each elliptic curve specified in [X9.62]. It has the following value:

```
ellipticCurve OBJECT IDENTIFIER ::= { ansi-X9-62 curves(3) }
```

The elliptic curve used for a.trust’s certificate is prime192v1.

```
primeCurve OBJECT IDENTIFIER ::= { ellipticCurve prime(1) }

prime192v1 OBJECT IDENTIFIER ::= { primeCurve 1 }
```

Point-to-Octet-String Conversion: Acc. to [X9.62] „an elliptic curve point P = (xp , yp) which is not the point at infinity shall be represented as an octet string in one of the following three forms:
1. compressed form
2. uncompressed form
3. hybrid form.”

a.trust uses the uncompressed form in its certificates, therefore P ist assigned the value 04.

2.8 issuerUniqueIdentifier/ subjectUniqueIdentifier

These fields are not used.

2.9 Extensions

X.509 defines the Extensions type as follows:

```
Extensions ::= SEQUENCE SIZE (1..MAX) OF Extension

Extension ::= SEQUENCE {
     extnId      OBJECT IDENTIFIER,
     critical    BOOLEAN DEFAULT FALSE,
     extnValue   OCTET STRING
}
```

This profile document specifies the extensions used in certificates issued by a.trust in the table below.

Extension	Presence
authorityKeyIdentifier	**mandatory**
subjectKeyIdentifier	**mandatory**
keyUsage	**mandatory**
certificatePolicies	**mandatory**
subjectAltName	**optional**
subjectDirectoryAttributes	**optional**
basicConstraints	**mandatory**
cRLDistributionPoints	**mandatory**
authorityInfoAccess	**mandatory**
qcStatements	**mandatory**
1.2.40.0.10.1.1.1	**optional**

The extensions of this profile are described in more detail below.

2.9.1 authorityKeyIdentifier

X.509 defines authorityKeyIdentifier extension as follows:

```
authorityKeyIdentifier EXTENSION ::= {
    SYNTAX                  AuthorityKeyIdentifier,
    IDENTIFIER BY           id-ce-authorityKeyIdentifier
}

AuthorityKeyIdentifier ::= SEQUENCE {
    keyIdentifier                [0] KeyIdentifier OPTIONAL,
    authorityCertIssuer          [1] GeneralNames OPTIONAL,
    authorityCertSerialNumber    [2] CertificateSerialNumber
                                 OPTIONAL
}
KeyIdentifier ::= OCTET STRING
```

This field is used to identify the public key to be used to verify the signature on this certificate or CRL. It enables distinct keys used by the same CA to be distinguished (e. g., as key updating occurs).

a.trust's certificates contain only the keyIdentifier element.

This is a non-critical extension.

2.9.2 subjectKeyIdentifier

X.509 defines subjectKeyIdentifier extension as follows:

```
subjectKeyIdentifier EXTENSION ::= {
    SYNTAX                  SubjectKeyIdentifier,
    IDENTIFIER BY           id-ce-subjectKeyIdentifier
}

SubjectKeyIdentifier ::= KeyIdentifier
KeyIdentifier ::= OCTET STRING
```

This field is used to identify the public key being certified. It shall contain a key identifier value that shall be unique for each different user key.

According to [rfc3280] the keyIdentifier is composed of a four bit type field with the value 0100 followed by the least significant 60 bits of the SHA-1 hash of the value of the BIT STRING subjectPublicKey (excluding the tag, length, and number of unused bit string bits).

This is a non-critical extension.

2.9.3 subjectAltName

X.509 defines subjectAltName extension as follows:

```
authorityKeyIdentifier EXTENSION ::= {
    SYNTAX            SubjectAltName,
    IDENTIFIER BY     id-ce-subjectAltName
}

SubjectAltName ::= GeneralNames

          GeneralNames ::= SEQUENCE SIZE (1..MAX) OF GeneralName

    GeneralName ::= CHOICE {
          OtherName                       [0]   OtherName,
          rfc822Name                      [1]   IA5String,
          dNSName                         [2]   IA5String,
          x400Address                     [3]   Address,
          directoryName                   [4]   Name,
          ediPartyName                    [5]   EDIPartyName,
          uniformResourceIdentifier       [6]   IA5String,
          iPAddress                       [7]   OCTET STRING,
          registeredID                    [8]   OBJECT IDENTIFIER}
```

The subjectAltName extension may optionally contain the certificate holder's email address using the rfc822Name element.

This is a non-critical extension.

2.9.4 keyUsage

X.509 defines the keyUsage extension as follows:

```
keyUsage EXTENSION ::= {
    SYNTAX                  KeyUsage,
    IDENTIFIER BY           id-ce-keyUsage
}
```

```
KeyUsage ::= BIT STRING {
    digitalSignature  (0),
    nonRepudiation    (1),
    keyEncipherment   (2),
    dataEncipherment  (3),
    keyAgreement      (4),
    keyCertSign       (5),
    cRLSign           (6),
    encipherOnly      (7),
    decipherOnly      (8)
}
```

This field indicates the purpose for which the certified public key is used.

In the signature key certificate, the following key usage bits are set:

- nonRepudiation
  This bit is set when the subject public key is used to verify digital signatures used to provide a non-repudiation service which protects against the signing entity falsely denying some action. This bit may not be combined with setting of any other bits.

- digitalSignature
  This bit is set when the subject public key is used with a digital signature mechanism to support security services other than non-repudiation. Digital signature mechanisms are often used for entity authentication and data origin authentication with integrity.

In the encryption key certificate, the following key usage bits are set:

- digitalSignature
  This bit is set when the subject public key is used with a digital signature mechanism to support security services other than non-repudiation. Digital signature mechanisms are often used for entity authentication and data origin authentication with integrity.

- keyEncipherment
  This bit is set when the subject public key is used for key transport. For example, when an RSA key is to be used for key management, then this bit shall be set.

- dataEncipherment
  This bit is set for enciphering user data but not keys or other security information as above.

This is a critical extension.

2.9.5 certificatePolicies

X.509 defines certificatePolicies extension as follows:

```
certificatePolicies EXTENSION ::= {
    SYNTAX            CertificatePoliciesSyntax,
    IDENTIFIER BY     id-ce-certificatePolicies
}

CertificatePoliciesSyntax ::= SEQUENCE SIZE (1…MAX) OF {
    PolicyInformation
}
PolicyInformation ::= SEQUENCE {
    policyIdentifiers       CertPolicyId,
    policyQualifiers        SEQUENCE SIZE (1…MAX) OF {
    PolicyQualifierInfo OPTIONAL
    }
}
CertPolicyId ::= OBJECT IDENTIFIER
PolicyQualifierInfo ::= SEQUENCE {
    PolicyQualifierId       CERT-POLICY-QUALIFIER.&id
                            ({SupportedPolicyQualifiers}),
    qualifier               CERT-POLICY-QUALIFIER.&Qualifier
                            ({SupportedPolicyQualifiers}
                             {@policyQualifierId}} OPTIONAL
}
```

This field lists one or more policies, recognized by the issuing CA, that apply to the certificate, together with optional qualifier information pertaining to these certificate policies.

This is a non-critical extension.

2.9.6 cRLDistributionPoints

X.509 defines cRLDistributionPoints extension as follows:

```
cRLDistributionPoints EXTENSION ::= {
    SYNTAX  CRLDistPointsSyntax
    IDENTIFIED BY id-ce-cRLDistributionPoints
}

CRLDistPointsSyntax ::= SEQUENCE SIZE (1..MAX) OF DistributionPoint
```

```
DistributionPoint ::= SEQUENCE {
    DistributionPoint       [0]   DistributionPointName OPTIONAL,
    Reasons                 [1]   ReasonFlags OPTIONAL,
    CRLIssuer               [2]   GeneralNames OPTIONAL
}

DistributionPointName ::= CHOICE {
    FullName                [0]   GeneralNames,
    NameRelativeToCRLIssuer [1]   RelativeDistinguishedName
}

ReasonFlags ::= BIT STRING {
    Unused                  (0),
    KeyCompromise           (1),
    CaCompromise            (2),
    AffiliationChanged      (3),
    Superseded              (4),
    CessationOfOperation    (5),
    CertificateHold         (6)
}
```

This field identifies how CRL information is obtained. The distributionPoint element of DistributionPoint SEQUENCE will contain a uniformResourceIdentifier (URI, element [6] of GeneralName CHOICE) pointing to the appropriate CRL for this certificate. Only this option will be used.

Example:

ldap://ldap.a-trust.at/ou=a-sign-premium-Sig-01,o=A-Trust,c=AT?certificaterevocationlist?

This is a non-critical extension.

2.9.7 qcStatements

[rfc3280] defines qcStatements extension as follows:

```
qcStatements  EXTENSION ::= {
          SYNTAX             QCStatements
          IDENTIFIED BY      id-pe-qcStatements }

      id-pe-qcStatements     OBJECT IDENTIFIER ::= { id-pe 3 }

      QCStatements ::= SEQUENCE OF QCStatement
      QCStatement ::= SEQUENCE {
          statementId   QC-STATEMENT.&Id({SupportedStatements}),
```

```
        statementInfo QC-STATEMENT.&Type
        ({SupportedStatements}{@statementId}) OPTIONAL }

    SupportedStatements QC-STATEMENT ::= { qcStatement-1,...}
```

a.trust uses the following statements according to [ETSI-qc] in the qualified certificate:

- Statement claiming that the certificate is issued as a Qualified certificate
- Statement regarding limits on the value of transactions for which the certificate can be used

The statement claiming that the certificate is a Qualified certificate is optional and contains:

- An Identifier of this statement (represented by an OID), stating that the certificate is issued according to the EU-directive, as implemented in the country under which law the issuer is operating.

```
esi4-qcStatement-1 QC-STATEMENT ::= { IDENTIFIED
BY id-etsi-qcs-QcCompliance }

id-etsi-qcs-QcCompliance OBJECT IDENTIFIER ::= { id-etsi-qcs 1 }
```

This is a critical extension.

The statement regarding limits on the value of transactions is optional and contains:

- an identifier of this statement (represented by an OID)
- a monetary value expressing the limit on the value of transactions.

```
esi4-qcStatement-2 QC-STATEMENT ::= { SYNTAX QcEuLimitValue IDENTIFIED
BY id-etsi-qcs-QcLimitValue }

QcEuLimitValue ::= MonetaryValue

MonetaryValue::= SEQUENCE {
currency Iso4217CurrencyCode,
amount INTEGER,
exponent INTEGER}

Iso4217CurrencyCode ::= CHOICE {
alphabetic PrintableString (SIZE 3), -- Recommended
numeric INTEGER (1..999) }
```

```
id-etsi-qcs-QcLimitValue OBJECT IDENTIFIER ::= { id-etsi-qcs 2 }
```

This is a critical extension because: “When this extension is marked critical, this means that all statements included in the extension are regarded as critical.” (see [ETSI-qc]).

2.9.8 authorityInfoAccess

“The authority information access extension indicates how to access CA information and services for the issuer of the certificate in which the extension appears. Information and services may include on-line validation services and CA policy data.” (see [rfc3280]).

```
id-pe-authorityInfoAccess OBJECT IDENTIFIER ::= { id-pe 1 }

    AuthorityInfoAccessSyntax  ::=
          SEQUENCE SIZE (1..MAX) OF AccessDescription

    AccessDescription  ::=  SEQUENCE {
          AccessMethod            OBJECT IDENTIFIER,
          AccessLocation          GeneralName  }

    id-ad OBJECT IDENTIFIER ::= { id-pkix 48 }

    id-ad-caIssuers OBJECT IDENTIFIER ::= { id-ad 2 }

    id-ad-ocsp OBJECT IDENTIFIER ::= { id-ad 1 }
```

The accessLocation specifies the address (URI: "http://..") of the service both for ocsp and caIssuers.

This extension is non-critical.

2.9.9 basicConstraints

```
id-ce-basicConstraints OBJECT IDENTIFIER ::=  { id-ce 19 }

    BasicConstraints ::= SEQUENCE {
          CA                      BOOLEAN DEFAULT FALSE,
          PathLenConstraint       INTEGER (0..MAX) OPTIONAL }
```

In the end user certificate cA is set to false.

This extension is non-critical.

2.9.10 subjectDirectoryAttributes

```
id-ce-subjectDirectoryAttributes OBJECT IDENTIFIER ::=  { id-ce 9 }

SubjectDirectoryAttributes ::= SEQUENCE SIZE (1..MAX) OF Attribute
```

According to [rfc3739] the "subjectDirectoryAttributes extension MAY contain additional attributes associated with the subject, as complement to present information in the subject field and the subject alternative name extension".

End user certificates optionally contain the dateOfBirth attribute.

This extension is non-critical.

2.9.11 Private extension: OID 1.2.40.0.10.1.1.1

This extension provides the information that the holder of the certificate is member of a public authority. Therefore the usage of this extension is restricted to Austrian public authorities only.

```
Extension ::= SEQUENCE {
    extnID = 1.2.40.0.10.1.1.1
    critical = false
    extnValue ::= CHOICE {
          isPublicAuthority alwaysTrue,
                code DirectoryString } }

alwaysTrue BOOLEAN ::= TRUE
```

3 CA certificates

Two types of CA certificates are issued by a.trust:

- a self-signed a.trust root certificate
- subordinate (intermediate) CA certificates signed by the a.trust root

These CA certificates contain the same fields as an ordinary end user certificate with the following exceptions:

- subject equals issuer in the self signed root certificates
- validity for both root and intermediate CA certificate is ten years
- key usages keyCertSign and cRLSign are set in the keyUsage extension in a.trust root and subordinate CA certificates
- basicConstraints extension: the value for the cA element is set to TRUE and the extension is critical
- certificatePolicies extension is not used in the root and the intermediate CA certificates
- cRLDistributionPoints extension is not used in the root certificate
- authorityKeyIdentifier is not used in the root certificate
- authorityInfoAccess is not used in the root certificate
- subjectDirectoryAttributes extension does not appear in the root and the CA certificates
- qcStatements extension is not used in the root and the intermediate CA certificates.

4 CRL

The following types of CRLs can be issued:

1. CRL for user-certificates
2. CRL for intermediate CA-certificates
3. delta-CRL for user-certificates

The CRL-type 1 and 2 will be stored under the DS-attribute certificateRevocationList.

The CRL-type 3 will be stored under the DS-attribute deltaRevocationList.

4.1 Structure of Delta-CRL and CRL

A CRL is always a complete list of all revoked certificates. It is, according to [ISO9594-8], a signed list (CertificateList) with one header record and data records (one to n records of revokedCertificates).

The CRL is defined according to the data structure SIGNED in [ISO9594-8]. The CRL specified at ASN.1 (tbsCertificateList) is DER encoded. Using this input the message digest according to SHA-1 must be calculated. The octet string resulting from the hash operation must be encoded according to BER and signed with the private part of the signature key of the CA. The field signature contains the result.

```
CertificateList  ::=  SEQUENCE  {
     TbsCertList             TBSCertList,
     SignatureAlgorithm      AlgorithmIdentifier,
     signature               BIT STRING  }

AlgorithmIdentifier  ::=  SEQUENCE  {
     Algorithm          OBJECT IDENTIFIER,
     Parameters         ANY DEFINED BY algorithm OPTIONAL}
```

The signature algorithm identifier for signature must be sha-1WithRSASignature (see [PKCS-1] and [rfc3280]).

```
TBSCertList  ::= SEQUENCE {
     version              1(v2),
     signature            AlgorithmIdentifier,
     issuer               DistinguishedName,
     thisUpdate           UTCTime,
```

```
nextUpdate           UTCTime,
revokedCertificates  SEQUENCE OF SEQUENCE {
     userCertificate     CertificateSerialNumber,
     revocationDate      UTCTime,
     crlEntryExtensions  Extensions OPTIONAL } OPTIONAL,
crlExtensions        [0] EXPLICIT Extensions OPTIONAL }
```

version: according to [rfc3280] “this optional field describes the version of the encoded CRL. When extensions are used, as required by this profile, this field MUST be present and MUST specify version 2 (the integer value is 1).”

signature is defined analogous to signature in CertificateList (see above).

issuer, the name of the certificate issuer, must be identical to the name of the CA issuing the certificate to revoke.

thisUpdate is the issuing time of the CRL. Format YYMMDDHHMM00z (UTCTime).

nextUpdate describes the time of the next planned CRL update. Format YYMMDDHHMM00z (UTCTime). nextUpdate is used in order to partly prevent potential manipulations (e.g. delayed CRL issuing) by the CA. At that moment a new CRL must be issued in all cases even if there are no changes to the previous CRL.

userCertificate is the INTEGER formatted certificate number (serialnumber) of the revoked certificate.

revocationDate is the moment of integration of the certificate into the CRL. Format YYMMDDHHMM00Z (UTCTime).

The following CRL extensions are used:

CRL extensions for complete CRLs:

- Authority Key Identifier, non critical
- CRL Number, non critical

Additional CRL extension for delta CRLs:

- Delta CRL Indicator, critical

The following CRL entry extension is used:

- Reason Code, non critical

4.1.1 extension CRLNumber

This non-critical and obligatory extension of the type crlExtensions is the CRL number (INTEGER). With this extension a monotone and rising numbering of the individual CRLs is done. The numbering allows even the issuing of delta-CRLs and CRL updates within the planned update period. So the user can always determine whether between two planned CRLs one or more unplanned updates have occurred or not. The possibility of issuing unplanned CRLs is necessary for the effective usage of suspensions; otherwise the planned update period would have to be very short.

4.1.2 extension deltaCRLIndicator

```
id-ce-deltaCRLIndicator OBJECT IDENTIFIER ::= {2 5 29 27}

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

This critical and, for delta-CRLs obligatory, extension (INTEGER) indicates that

- this CRL is a delta-CRL and
- the delta-CRL refers to the CRL with the CRLNumber deltaCRLIndicator (i. e. deltaCRLIndicator=CRLNumber).

The DS attribute *deltaRevocationList;binary* is MULTI_VALUED, i. e. it may contain several delta-CRLs up to 100 entries. Among these entries there must be the current delta-CRL and the most 99 old delta-CRLs. The whole value of this attribute is the DER encoded description of the following ASN.1 structure (see [ISO9594-2]). This structure is defined as an OCTET STRING:

SET SIZE(1..100) OF deltaRevocationList.

4.1.3 extension authorityKeyIdentifier

Not critical and mandatory, defined in section 2.9.1.

4.1.4 Entry-extension reasonCode

```
id-ce-reasonCode OBJECT IDENTIFIER ::= {2 5 29 21}

reasonCode ::= CRLReason

CRLReason ::= ENUMERATED {
```

```
unspecified             (0),
keyCompromise           (1),
cACompromise            (2),
affiliationChanged      (3),
superseded              (4),
cessationOfOperation    (5),
certificateHold         (6),
removeFromCRL           (8) }
```

The reasonCode is a non-critical CRL entry extension that identifies the reason for the certificate revocation. The type of this extension is crlEntryExtension.

The reasons which can be used are the following:

1. unspecified (0)
2. keyCompromise (1): is only used either on the explicit request of the signatory or if the customer admits the compromising of the signature-PIN
3. caCompromise (2): has to be used only for CA certificates in case of compromisation of the CA signature key
4. affiliationChanged (3): is used if the subject's name or other information in the certificate has been modified
5. certificateHold (6): is used for the suspension of a certificate
6. removeFromCRL (8): is only allowed for delta-CRLs, if an entry must be removed from the CRL (e. g. reversal of suspension) and if the according base-CRL contains an entry for this certificate with the reasonCode=certificateHold.

As far as qualified or non-qualified user certificates are concerned a.trust uses the reason codes 0, 1, 3, 6, 8. In case an intermediate CA-certificate should be compromised a.trust would revoke it using reason code 2.

A suspension of a certificate may either be:

- replaced by a revocation or
- lifted by a reversal of suspension.

5 Reference documents

Ref	Title	Author	Version
[ISO9594-2]	Information Technology - Open Systems Interconnection - The Directory: Models	ISO/IEC	2nd edition 1995-09-15
[ISO9594-8]	Information Technology - Open Systems Interconnection - The Directory: Authentication framework	ISO/IEC	2nd edition 1995-09-15
[X509]	Information Technology - Open Systems Interconnection - The Directory: Authentication framework	ITU-T	4th ed., 2000
[rfc3279]	Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile	W. Polk, R. Housley, L. Bassham	RFC3279 April 2002
[rfc3280]	Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile	R. Housley, W. Ford, W. Polk, D. Solo	RFC3280 April 2002
[rfc2560]	X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP	M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams	RFC2560 Jun 1999
[rfc3739]	Internet X.509 Public Key Infrastructure Qualified Certificates Profile	S. Santesson, M. Nystrom, W. Polk	RFC3739 March 2004
[PKCS-1]	PKCS#1: RSA Encryption Standard	RSA Laboratories	Technical Note Nov 1993
[ETSI-qc]	ETSI TS 101 862 V1.2.1 Qualified certificate profile	ETSI	2001-06
[X9.62]	Working Draft AMERICAN NATIONAL STANDARD X9.62-1998 Public Key Cryptography For The Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)©	American Bankers Association	Sept. 20, 1998