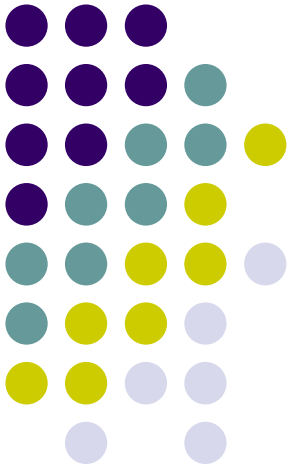
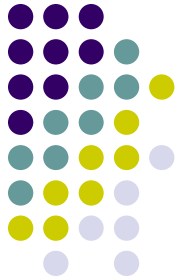
ASN.1:
Introduction

Zdeněk Říha



ASN.1



- Abstract Syntax Notation 1
- notation for describing abstract types and values
- Defined in ITU-T X.680 … X.695
- Used in many file formats, including crypto
  - Public keys, private keys
  - Certificate requests, certificates
  - Digital signatures, padding, encrypted files

ASN.1

- Allows format/storage/transmission of data
  - Compatible among many applications
  - Not dependent on HW platform
    - E.g. little/big endian
  - Not dependent on operating system
- Simple & Structured types
- Multiple encoding rules (methods)

ASN.1 – Types

Type	Tag number (decimal)	Tag number (hexadecimal)
INTEGER	2	02
BIT STRING	3	03
OCTET STRING	4	04
NULL	5	05
OBJECT IDENTIFIER	6	06
SEQUENCE and SEQUENCE OF	16	10
SET and SET OF	17	11
PrintableString	19	13
IA5String	22	16
UTCTime	23	17

ASN.1 – simple types

- Integer
  - signed integer (there's no unsigned integer)
- Bit string
  - The number of bits does not have to be a multiple of 8
- Octet string
  - an arbitrary string of octets
- NULL
  - No data (used in parameters)
- PringtableString, IA5String, UTF8String, …
  - Strings – the sets of characters are various
- UTCTime
  - Time

ASN.1 – OID type

- Object identifier (OID)
  - Sequence of integer components that identify an object
  - Assigned in a hierarchical way
- Example
  - sha-1WithRSAEncryption = 1.2.840.113549.1.1.5
  - iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 5

- 1.2.840.113549.1.1 - PKCS-1
- 1.2.840.113549.1 - PKCS
- 1.2.840.113549 - RSADSI
- 1.2.840 - USA
- 1.2 - ISO member body
- 1 - ISO assigned OIDs
- Top of OID tree

ASN.1 – structured types

- SEQUENCE
  - an ordered collection of one or more types
- SEQUENCE OF
  - an ordered collection of zero or more occurrences of a given type
- SET
  - an unordered collection of one or more types
- SET OF
  - an unordered collection of zero or more occurrences of a given type

ASN.1 Encoding Rules

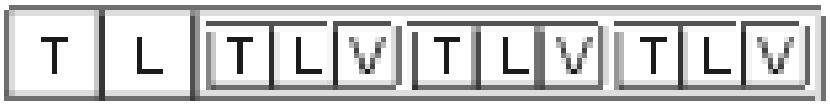
- XML – oriented formats
  - XER (XML Encoding Rules)
- Byte-oriented formats
  - BER (Basic Encoding Rules)
  - CER (Canonical Encoding Rules) – subset of BER
  - **DER (Distinguished Encoding Rules) – subset of BER**
    - **Used for crypto files**
- Bit-oriented formats
  - PER (Packed Encoding Rules)
- Verbose, human readable formats
  - GSER (Generic String Encoding Rules)

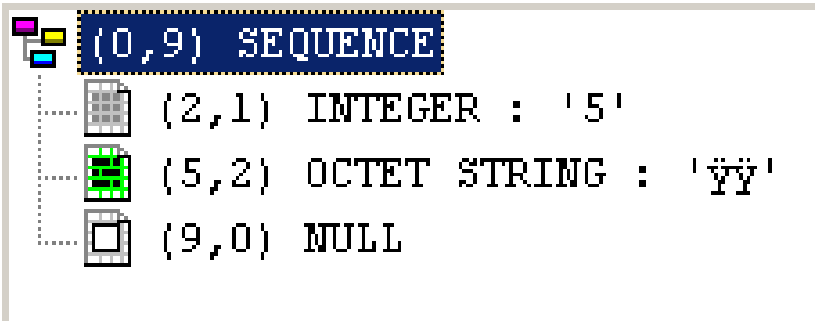
BER encoding

- TLV – Tag Length Value
  - All the data is encoded using a simple TLV format
  - Tag – what kind of data it is
  - Length – the length of the data
  - Value – the data itself
- Example
  - 02 01 05 [hexadecimal values]
    - Tag – Integer
    - Length of data – 1 byte
    - Data: (positive integer) 5

Nested data

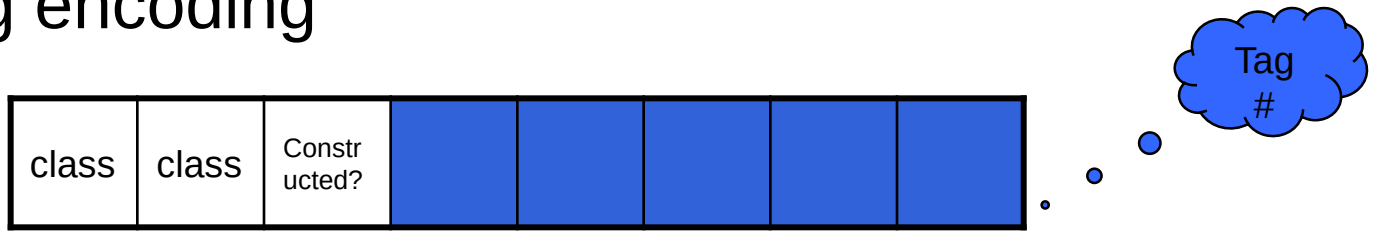
- SEQUENCE is similar to struct/record
- 30 09 02 01 05 04 02 FF FF 05 00
  - 30 09 – sequence of length 9 bytes
  - 02 01 05 – integer 5
  - 04 02 FF FF – octet string FF FF
  - 05 00 – NULL (no data)





BER tags

- Tag encoding

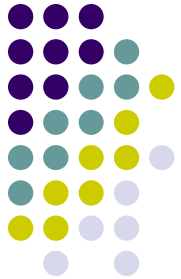


- Class

Class	Bit 8	Bit 7
universal	0	0
application	0	1
context-specific	1	0
private	1	1

- Tag number
  - Bits 1-5
  - If all bits are 1 then the tag continues in the following byte(s)

BER length



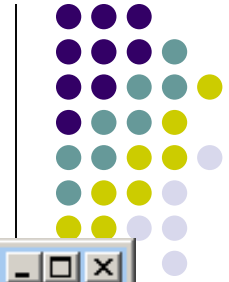
- length >=0 && length <= 127
  - The length is coded directly
    - E.g. ’05’
- Otherwise the bit 8 is set, bits 1-7 code the number of bytes that specify the length
  - E.g. 255 -> ‘81’ ‘FF’
  - E.g. 256 -> ’82’ ‘01’ ‘00’ or also ’83’ ‘00’ ‘01’ ‘00’
    - BER x DER
- ‘80’ is “indefinite” length
  - Not allowed in DER

BER value

- The data itself
- Dependent on data type
  - Integer: signed – e.g. 128 -> ’00 80’
  - Octet string: directly the data
  - Bit string: number of unused bits + padded bit string to a multiple of 8 bits (padding is at the end)
  - UTCTime: string of one of the forms

YYMMDDhhmmZ
YYMMDDhhmm+hh'mm'
YYMMDDhhmm-hh'mm'
YYMMDDhhmmssZ
YYMMDDhhmmss+hh'mm'
YYMMDDhhmmss-hh'mm'

First look at the binary DER file



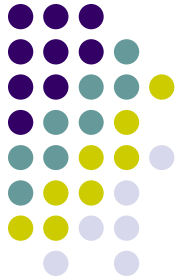
TSL_1.cer (~\Plocha\PKI) - GVIM

Soubor Úpravy Nástroje Syntaxe Buffery Okna Nápověda

```
0,^DĽ0,^C¤ ^C^B^A^B^B^C^W&Ľ0^M^F        *†H†÷^M^A^A^K^E^@0■«1^K0        ^F^CU^D^F^S^BCZ1907
^F^CU^D^C^L0I.CA - Standard Certification Authority, 09/20091-0+^F^CU^D
^L$PrvnĂ- certifikaĂŤnĂ- autorita, a.s.1200^F^CU^D^K^L)I.CA - Provider of Certification Ser
vices0^^^W^M0912211732592^W^M1012211732592Z0■-1^K0    ^F^CU^D^F^S^BCZ1705^F^CU^D^C^L.Ministr
y of the Interior of the Czech Republic1705^F^CU^D
^L.Ministry of the Interior of the Czech Republic1^U0^S^F^CU^D^E^S^LICA - 6139660,^A"0^M^F
     *†H†÷^M^A^A^A^E^@^C,^A^O^@0,^A
^B,^A^A^@‘jHŞÔ5ŤOéO     Ë°/Ź„Ľ×Ą}^T¤ëĹ…Ę’×\šUśĚ■Ě»Ŕ’Ó%č■|f4ÇĽ^^^LôóÄ%-ëIäÛČ^PÎÎ'-Ţ‘?JÇ%]U°”
^_    j]Ęë=Î$}WÁ^âdBĄž2Ú"'Ôr^ZŢ†;uđ^PBRD%^P^OÜp,MŒ^\‹Ž^M^\^ZdýôÁ■n+3■ó6P^UUpw„-$cî■ä·GvôŮQ^R
UkLF¬±g^R^B(/EĎšACşQ,ĚŐH [ÄUÝĎ^G{Ó^WÂÄg0·ä^E^M<wŤËHb\CĽó7^Ků§ÓF^U^]q^GžźGĹv“
»^YĆ^W˜ë.’_“ţ
č■ă^TDÓ^O        ^T^XA^DŮž]¸Ľ}Ň”›  ¸tu^B^C^A^@^AŁ■ű0■ř0^Q^F^CU^]%^D
0^H^F^F^D^@‘7^C^@0^_^F  `†H^A†řB^A^M^D^R^V^P92030300000112730^_^F^CU^]#^D^X0^V■^TÁL8”Ő■†HŮ"
■,ÓÎ^Y^PŰgG‡0^]^F^CU^]^N^D^V^D^TÎv&\şŁJŇ#P[ÄĽą)>[ůĘB0^Z^F^CU^] ^D^S0^Q0^O^F^M+^F^A^D^A■¸H^A
^A<^C^@0^K^F^CU^]^O^D^D^C^B^G■0Y^F^CU^]^_^DR0P0& $ "† http://scrldp1.ica.cz/sica09.crl0& $
"† http://scrldp2.ica.cz/sica09.crl0^M^F       *†H†÷^M^A^A^K^E^@^C,^A^A^@a‡■ó^VrËJŮ@Ě--^SE^
C~Î^\6@ď*^F˜■Ź”%5Ř_^SúyČ'ŐĎIÁěâ^OüýŔ^By^Y^L6D¬6^S'^Lž„ŇŇđv^_ş■7%¸“żg·v wĽĘ^K-§đrź
ę_^Zlxż˘Ţk$%ş^Ear^O& ë5¨ŕ8Nţ■^Yů■»<%^[^W7ÉN■Ô&†“‡×
ř
á^MÍ^XúkËŃ#âă^W^@˜·±4â©TO■˜A4´ř±^DŢ8Ľ¸OŘ*×§âGPoAXj^Azµ©1PŮXôŘ$-:ćÂ^Gü^Xq›K‘<A¦■ď{Ű0Ă8÷iŢ8$@
±^BĽô>?”ő§˜ć-{Mď’úáF‹’Ň^VŇŁë¬■ŘÔ"+Ľí·ł^Cwđ^SW
~
~
                                                                    1,1           Vše
```

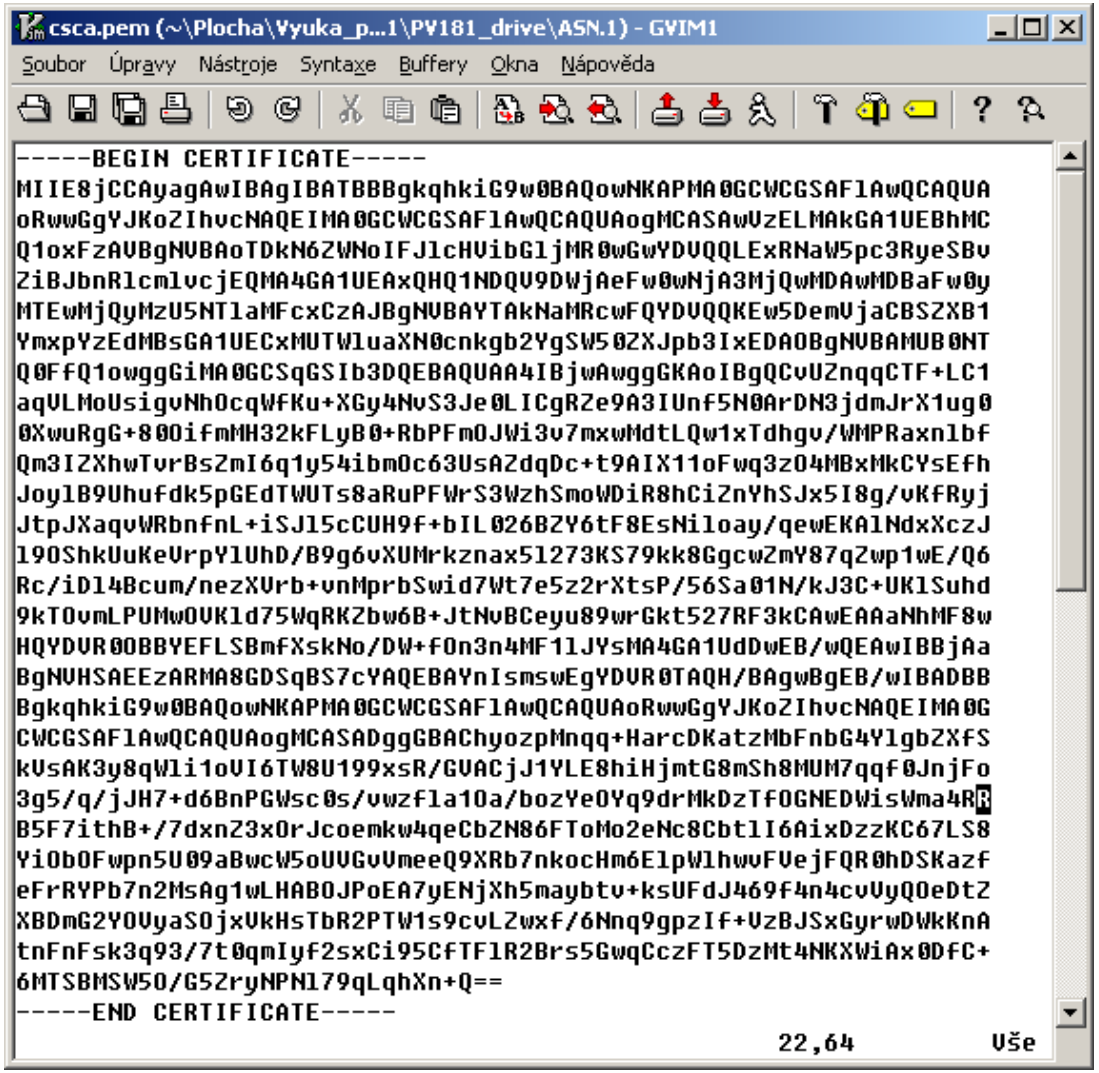
CSCA_CZE.crt

DER vs. PEM



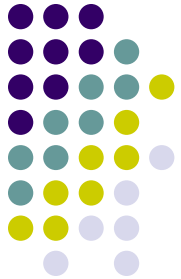
- PEM
  - Privacy Enhanced Mail
- PEM as such not used, but formats still used
- Textual formats
  - Practical for transport channels where full 8bit data can be damaged
- PEM is base64 coded DER enveloped with
  - -----BEGIN SOMETHING-----
  - -----END SOMETHING-----
  - Where SOMETHING is CERTIFICATE/PKCS7/KEY…

Sample PEM file



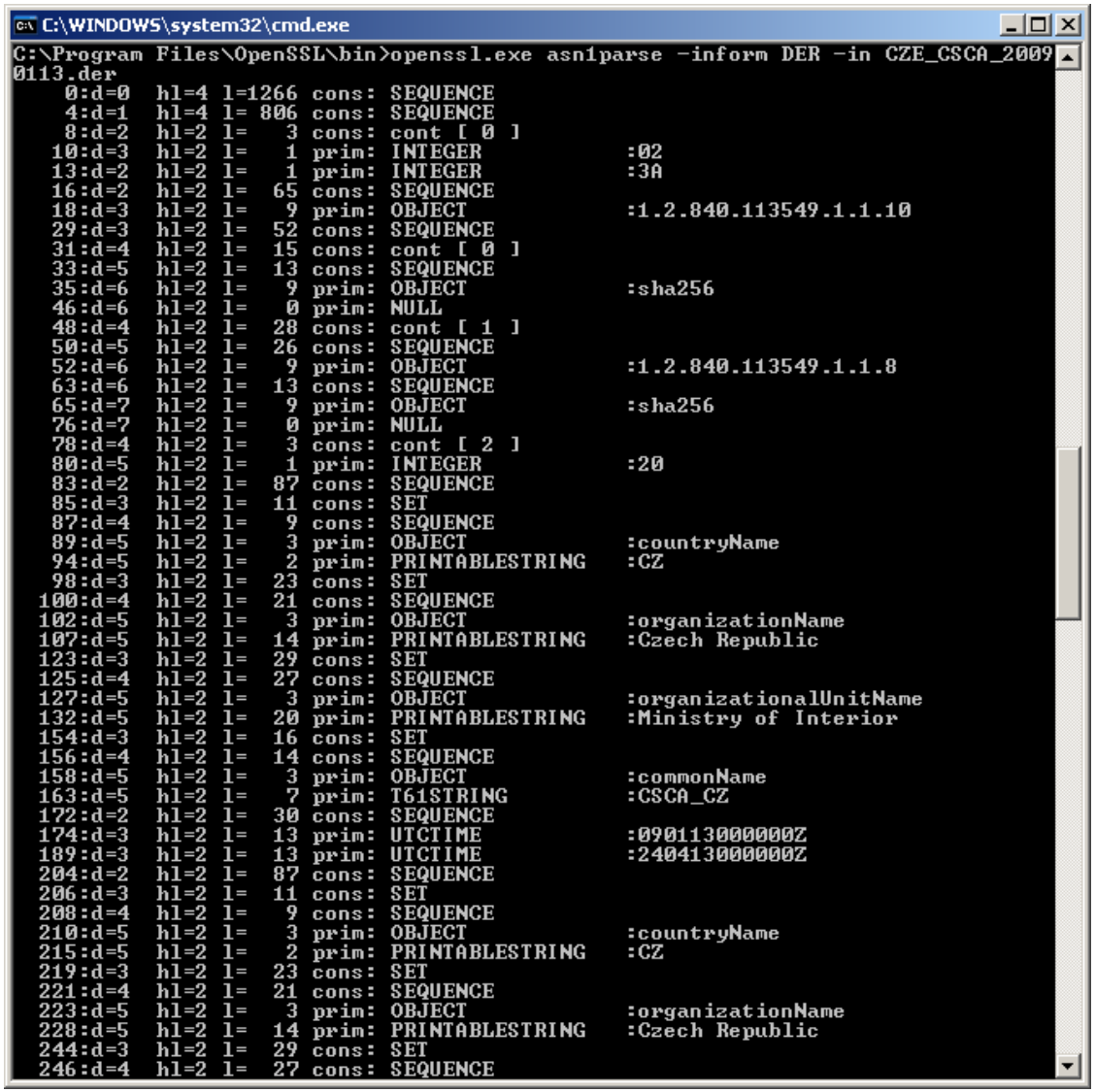
CSCA_CZE.pem

ASN.1 viewers



- Unber (part of asn1c)
- Openssl asn1parse
- ASN.1 Editor
- …

OpenSSL asn1parse



CSCA_CZE.crt

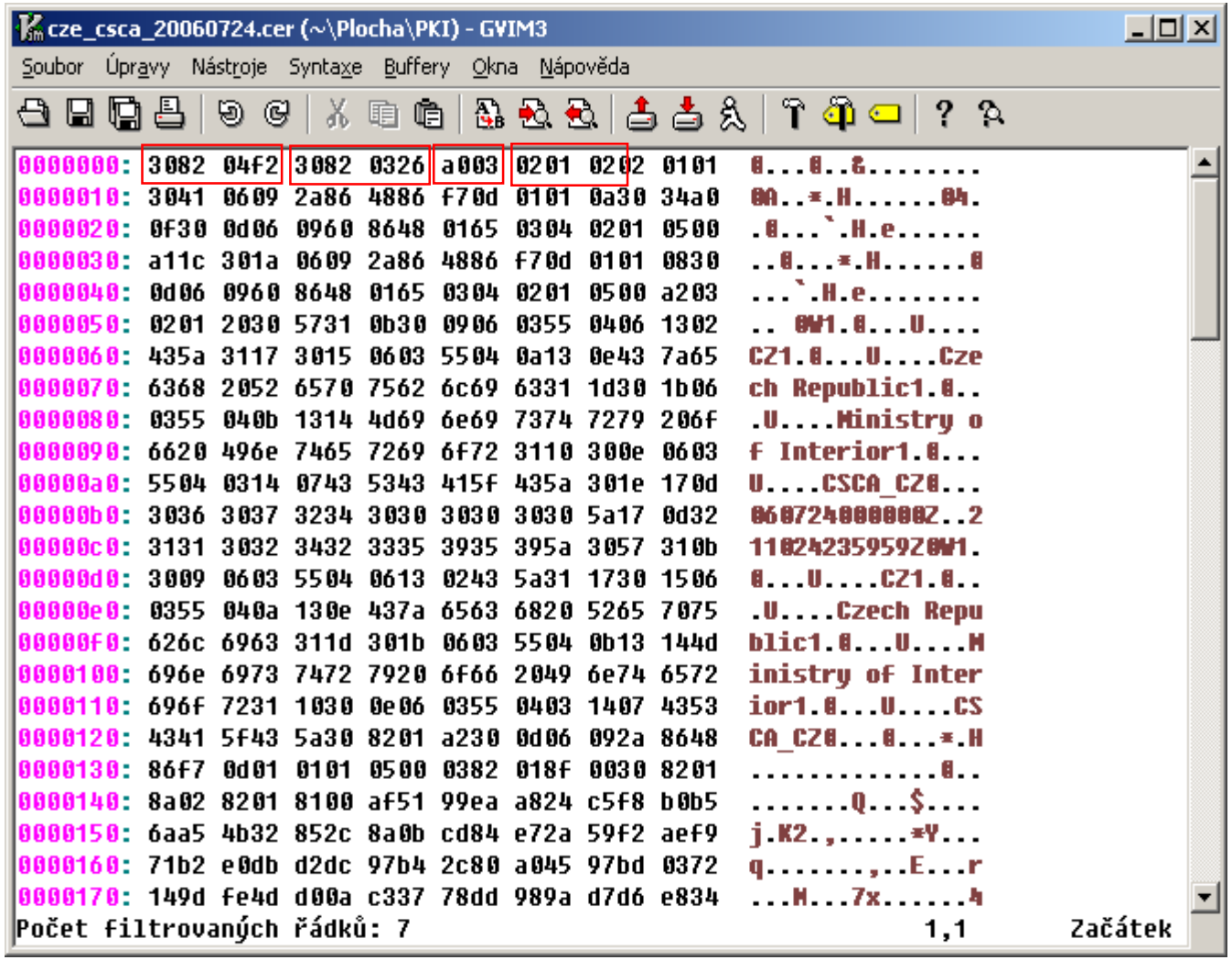
unber

CSCA_CZE.crt

anxur.fi.muni.cz - PuTTY

```
labak:~$ unber /usr/share/doc/dirmngr-1.0.3/examples/extra-certs/bnetza-10r-ocsp-2.crt
<C O="0" T="[UNIVERSAL 16]" TL="4" V="952" A="SEQUENCE">
    <C O="4" T="[UNIVERSAL 16]" TL="4" V="804" A="SEQUENCE">
        <C O="8" T="[0]" TL="2" V="3">
            <P O="10" T="[UNIVERSAL 2]" TL="2" V="1" A="INTEGER" F>2</P>
        </C O="13" T="[0]" L="5">
        <P O="13" T="[UNIVERSAL 2]" TL="2" V="1" A="INTEGER" F>53</P>
        <C O="16" T="[UNIVERSAL 16]" TL="2" V="10" A="SEQUENCE">
            <P O="18" T="[UNIVERSAL 6]" TL="2" V="6" A="OBJECT IDENTIFIER" F>1.3.36.3.3.1.2</P>
            <P O="26" T="[UNIVERSAL 5]" TL="2" V="0" A="NULL"></P>
        </C O="28" T="[UNIVERSAL 16]" A="SEQUENCE" L="12">
        <C O="28" T="[UNIVERSAL 16]" TL="2" V="63" A="SEQUENCE">
            <C O="30" T="[UNIVERSAL 17]" TL="2" V="11" A="SET">
                <C O="32" T="[UNIVERSAL 16]" TL="2" V="9" A="SEQUENCE">
                    <P O="34" T="[UNIVERSAL 6]" TL="2" V="3" A="OBJECT IDENTIFIER" F>2.5.4.6</P>
                    <P O="39" T="[UNIVERSAL 19]" TL="2" V="2" A="PrintableString">DE</P>
                </C O="43" T="[UNIVERSAL 16]" A="SEQUENCE" L="11">
            </C O="43" T="[UNIVERSAL 17]" A="SET" L="13">
            <C O="43" T="[UNIVERSAL 17]" TL="2" V="26" A="SET">
                <C O="45" T="[UNIVERSAL 16]" TL="2" V="24" A="SEQUENCE">
                    <P O="47" T="[UNIVERSAL 6]" TL="2" V="3" A="OBJECT IDENTIFIER" F>2.5.4.10</P>
                    <P O="52" T="[UNIVERSAL 12]" TL="2" V="17" A="UTF8String">Bundesnetzagentur</P>
                </C O="71" T="[UNIVERSAL 16]" A="SEQUENCE" L="26">
            </C O="71" T="[UNIVERSAL 17]" A="SET" L="28">
            <C O="71" T="[UNIVERSAL 17]" TL="2" V="20" A="SET">
                <C O="73" T="[UNIVERSAL 16]" TL="2" V="18" A="SEQUENCE">
                    <P O="75" T="[UNIVERSAL 6]" TL="2" V="3" A="OBJECT IDENTIFIER" F>2.5.4.3</P>
                    <P O="80" T="[UNIVERSAL 12]" TL="2" V="11" A="UTF8String">10R-CA 1:PN</P>
                </C O="93" T="[UNIVERSAL 16]" A="SEQUENCE" L="20">
            </C O="93" T="[UNIVERSAL 17]" A="SET" L="22">
        </C O="93" T="[UNIVERSAL 16]" A="SEQUENCE" L="65">
        <C O="93" T="[UNIVERSAL 16]" TL="2" V="30" A="SEQUENCE">
            <P O="95" T="[UNIVERSAL 23]" TL="2" V="13" A="UTCTime">050804082709Z</P>
            <P O="110" T="[UNIVERSAL 23]" TL="2" V="13" A="UTCTime">071231082349Z</P>
        </C O="125" T="[UNIVERSAL 16]" A="SEQUENCE" L="32">
        <C O="125" T="[UNIVERSAL 16]" TL="2" V="65" A="SEQUENCE">
            <C O="127" T="[UNIVERSAL 17]" TL="2" V="11" A="SET">
                <C O="129" T="[UNIVERSAL 16]" TL="2" V="9" A="SEQUENCE">
                    <P O="131" T="[UNIVERSAL 6]" TL="2" V="3" A="OBJECT IDENTIFIER" F>2.5.4.6</P>
                    <P O="136" T="[UNIVERSAL 19]" TL="2" V="2" A="PrintableString">DE</P>
                </C O="140" T="[UNIVERSAL 16]" A="SEQUENCE" L="11">
            </C O="140" T="[UNIVERSAL 17]" A="SET" L="13">
            <C O="140" T="[UNIVERSAL 17]" TL="2" V="26" A="SET">
                <C O="142" T="[UNIVERSAL 16]" TL="2" V="24" A="SEQUENCE">
                    <P O="144" T="[UNIVERSAL 6]" TL="2" V="3" A="OBJECT IDENTIFIER" F>2.5.4.10</P>
                    <P O="149" T="[UNIVERSAL 12]" TL="2" V="17" A="UTF8String">Bundesnetzagentur</P>
                </C O="168" T="[UNIVERSAL 16]" A="SEQUENCE" L="26">
```

Manual viewing/processing

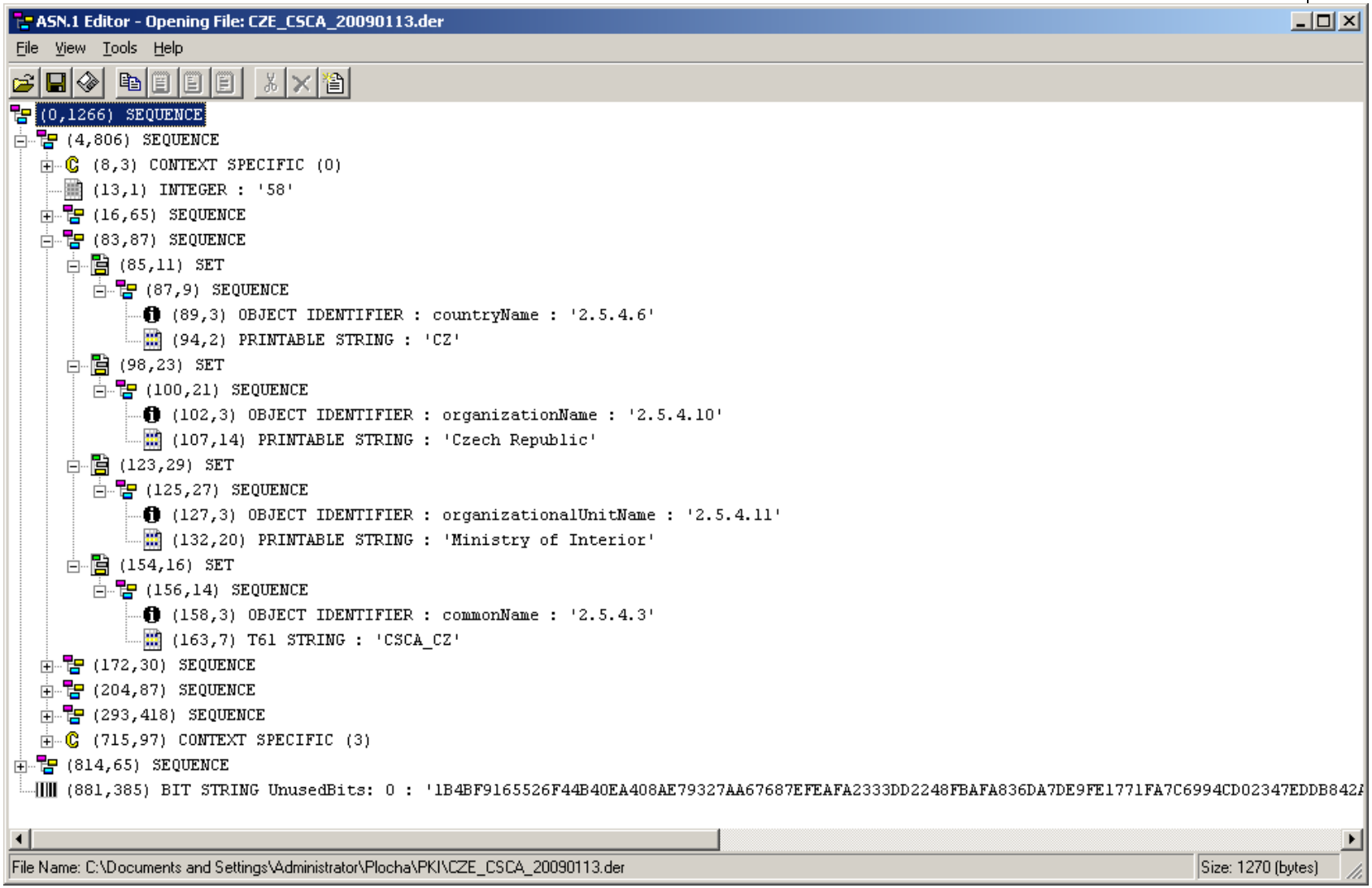


- 30 82 04 f2
  - SEQUENCE
  - length 1266B
- 30 82 03 26
  - SEQUENCE
  - length 806B
- A0 03
  - CONTEXT SPECIFIC 0
  - Length 3B
- 02 01 02
  - INTEGER: 2

CSCA_CZE.crt

ASN.1 Editor

CSCA_CZE.crt



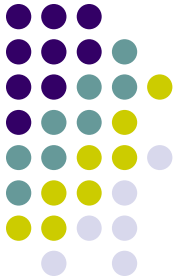
ASN.1 Grammar

- To understand the structure (what is the meaning of particular fields) we need ASN.1 grammar

```
CertificateList  ::=  SEQUENCE  {
     tbsCertList          TBSCertList,
     signatureAlgorithm   AlgorithmIdentifier,
     signatureValue       BIT STRING  }
```

```
TBSCertList  ::=  SEQUENCE  {
     version                 Version OPTIONAL,
                                  -- if present, MUST be v2
     signature               AlgorithmIdentifier,
     issuer                  Name,
     thisUpdate              Time,
     nextUpdate              Time OPTIONAL,
     revokedCertificates     SEQUENCE OF SEQUENCE  {
          userCertificate         CertificateSerialNumber,
          revocationDate          Time,
          crlEntryExtensions      Extensions OPTIONAL
                                        -- if present, MUST be v2
                                  }  OPTIONAL,
     crlExtensions           [0]  EXPLICIT Extensions OPTIONAL
                                        -- if present, MUST be v2
                                  }
```

ASN.1 – RSA keys



```
RSAPublicKey ::= SEQUENCE {
    modulus            INTEGER,  -- n
    publicExponent     INTEGER   -- e
}

--
-- Representation of RSA private key with information for the CRT algorithm.
--
RSAPrivateKey ::= SEQUENCE {
    version            Version,
    modulus            INTEGER,  -- n
    publicExponent     INTEGER,  -- e
    privateExponent    INTEGER,  -- d
    prime1             INTEGER,  -- p
    prime2             INTEGER,  -- q
    exponent1          INTEGER,  -- d mod (p-1)
    exponent2          INTEGER,  -- d mod (q-1)
    coefficient        INTEGER,  -- (inverse of q) mod p
    otherPrimeInfos    OtherPrimeInfos OPTIONAL
}
```

RSA.key

Source:
PKCS#1

ASN.1 – RSA padding

- PKCS#1 v1.5
  - m = 0x00 || 0x01 || 0xFF … 0xFF || 0x00 || T
  - Where T is defined as DER encoding of

```
DigestInfo ::= SEQUENCE {
    digestAlgorithm AlgorithmIdentifier,
    digest OCTET STRING
}
```

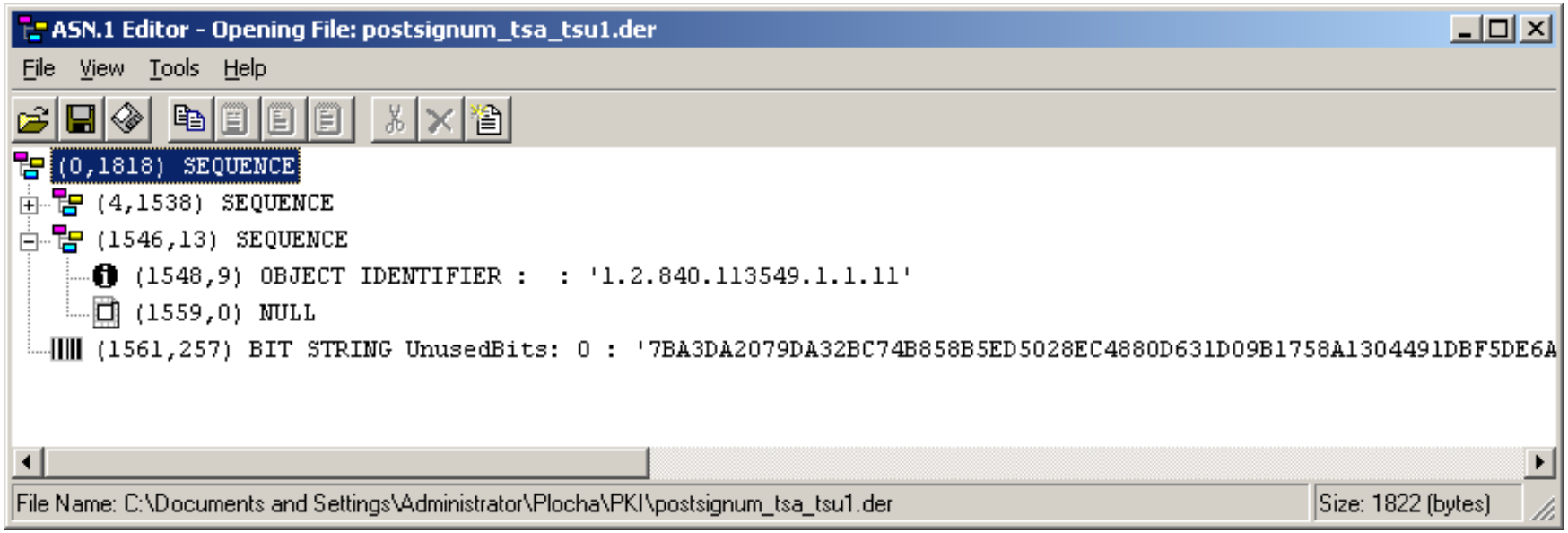
- In practice:

MD2:	(0x)30 20 30 0c 06 08 2a 86 48 86 f7 0d 02 02 05 00 04 10 ‖ H.
MD5:	(0x)30 20 30 0c 06 08 2a 86 48 86 f7 0d 02 05 05 00 04 10 ‖ H.
SHA-1:	(0x)30 21 30 09 06 05 2b 0e 03 02 1a 05 00 04 14 ‖ H.
SHA-256:	(0x)30 31 30 0d 06 09 60 86 48 01 65 03 04 02 01 05 00 04 20 ‖ H.
SHA-384:	(0x)30 41 30 0d 06 09 60 86 48 01 65 03 04 02 02 05 00 04 30 ‖ H.
SHA-512:	(0x)30 51 30 0d 06 09 60 86 48 01 65 03 04 02 03 05 00 04 40 ‖ H.

Source: PKCS#1

ASN.1 – RSA signature

- RSA signature is the number **s** = m^d mod n



TSA.crt

ASN.1 – signature OIDs

RSA Encryption[1]	1.2.840.113549.1.1.1
RSASSA-PKCS1_v15 with SHA1	1.2.840.113549.1.1.5
RSASSA-PSS	1.2.840.113549.1.1.10 (PKCS #1 Version 2.1)
RSASSA-PKCS1_v15 with SHA224	1.2.840.113549.1.1.14
RSASSA-PKCS1_v15 with SHA256	1.2.840.113549.1.1.11
RSASSA-PKCS1_v15 with SHA384	1.2.840.113549.1.1.12
RSASSA-PKCS1_v15 with SHA512	1.2.840.113549.1.1.13

Source:
BSI TR-03105 Part 5.1

ASN.1 – RSA PSS params

```
RSASSA-PSS-params ::= SEQUENCE {
    hashAlgorithm      [0] HashAlgorithm      DEFAULT sha1,
    maskGenAlgorithm   [1] MaskGenAlgorithm   DEFAULT mgf1SHA1,
    saltLength         [2] INTEGER            DEFAULT 20,
    trailerField       [3] TrailerField       DEFAULT trailerFieldBC
}

TrailerField ::= INTEGER { trailerFieldBC(1) }
```

Source: PKCS#1

```
(814,65) SEQUENCE
  (816,9) OBJECT IDENTIFIER :  : '1.2.840.113549.1.1.10'
  (827,52) SEQUENCE
    (829,15) CONTEXT SPECIFIC (0)
      (831,13) SEQUENCE
        (833,9) OBJECT IDENTIFIER :  : '2.16.840.1.101.3.4.2.1'
        (844,0) NULL
    (846,28) CONTEXT SPECIFIC (1)
      (848,26) SEQUENCE
        (850,9) OBJECT IDENTIFIER :  : '1.2.840.113549.1.1.8'
        (861,13) SEQUENCE
          (863,9) OBJECT IDENTIFIER :  : '2.16.840.1.101.3.4.2.1'
          (874,0) NULL
    (876,3) CONTEXT SPECIFIC (2)
      (878,1) INTEGER : '32'
```

RSASSA-PSS

SHA256

MGF1

SHA256

CSCA_CZE.crt

ASN.1 – DSA keys

```
Dss-Parms  ::=  SEQUENCE  {
            p               INTEGER,
            q               INTEGER,
            g               INTEGER  }
```

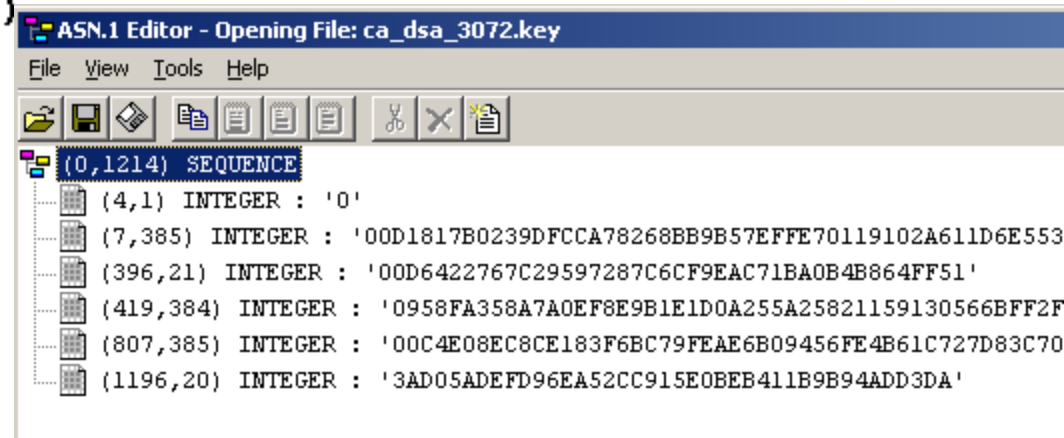
Source:
RFC 5480

```
DSAPublicKey ::= INTEGER -- public key, Y
```

DSAPrivateKey is an INTEGER, usually denoted as X

```
ASN1_SEQUENCE_cb(DSAPrivateKey, dsa_cb) = {
        ASN1_SIMPLE(DSA, version, LONG),
        ASN1_SIMPLE(DSA, p, BIGNUM),
        ASN1_SIMPLE(DSA, q, BIGNUM),
        ASN1_SIMPLE(DSA, g, BIGNUM),
        ASN1_SIMPLE(DSA, pub_key, BIGNUM),
        ASN1_SIMPLE(DSA, priv_key, BIGNUM)
} ASN1_SEQUENCE_END_cb(DSA, DSAPrivateKey)
```

Source:
OpenSSL

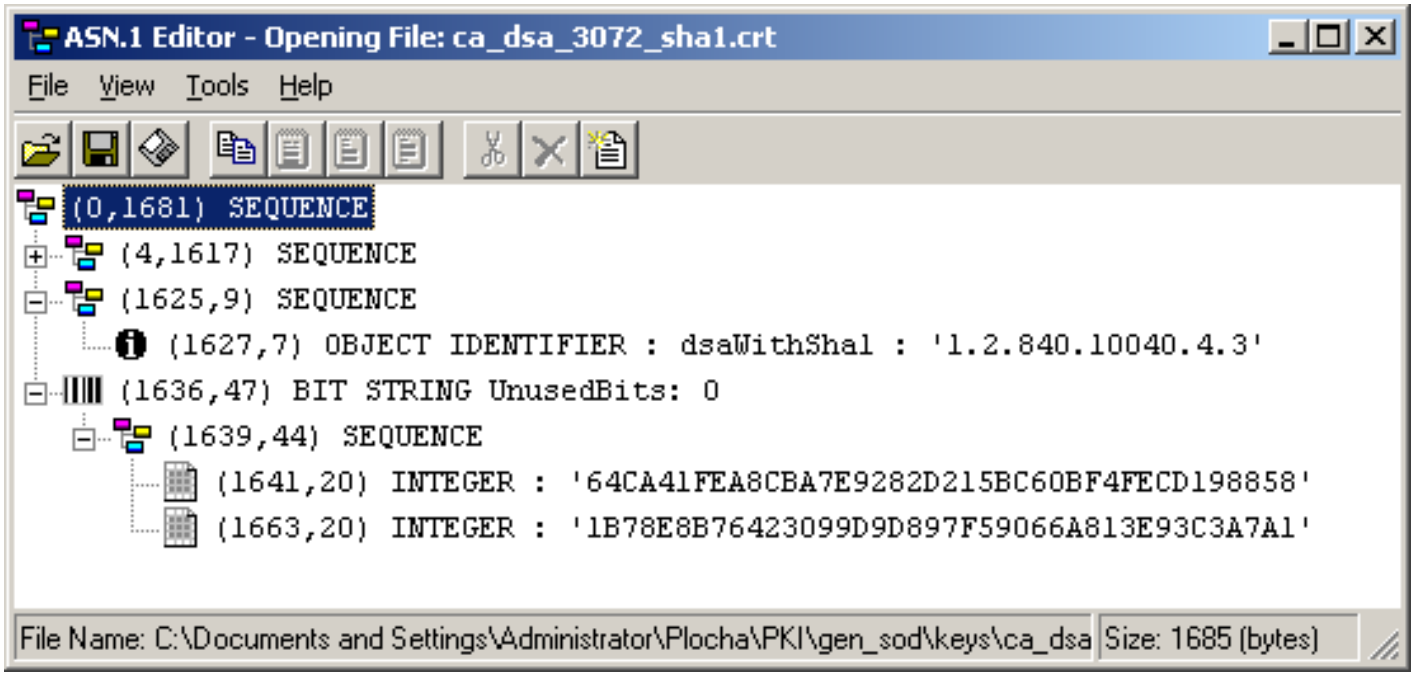


DSA.key

ASN.1 – DSA signature

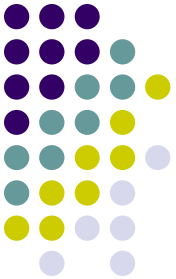
```
Dss-Sig-Value  ::=  SEQUENCE  {
        r       INTEGER,
        s       INTEGER  }
```

Source:
RFC 5480



DSA.crt

ASN.1 – DSA - OIDs



```
-- DSA with SHA-1
-- Parameters are ABSENT

id-dsa-with-sha1 OBJECT IDENTIFIER ::=  {
  iso(1) member-body(2) us(840) x9-57(10040) x9algorithm(4) 3 }

-- DSA with SHA-224
-- Parameters are ABSENT

id-dsa-with-sha224 OBJECT IDENTIFIER  ::=  {
  joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101)
  csor(3) algorithms(4) id-dsa-with-sha2(3) 1 }

-- DSA with SHA-256
-- Parameters are ABSENT

id-dsa-with-sha256 OBJECT IDENTIFIER  ::=  {
  joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101)
  csor(3) algorithms(4) id-dsa-with-sha2(3) 2 }
```

Source: RFC 5480

ASN.1 – ECDSA keys

```
ECParameters ::= SEQUENCE {
    version     INTEGER{ecpVer1(1)} (ecpVer1),
    fieldID     FieldID{{FieldTypes}},
    curve       Curve,
    base        ECPoint,
    order       INTEGER,
    cofactor    INTEGER OPTIONAL,
    ...
}
```

```
Curve ::= SEQUENCE {
    a       FieldElement,
    b       FieldElement,
    seed    BIT STRING OPTIONAL
}
```

```
ECPoint ::= OCTET STRING -- Elliptic curve point
```

```
ECPrivateKey{CURVES:IOSet} ::= SEQUENCE {
    version          INTEGER { ecPrivkeyVer1(1) } ( ecPrivkeyVer1 ),
    privateKey       OCTET STRING,
    parameters   [0] Parameters{{IOSet}} OPTIONAL,
    publicKey    [1] BIT STRING OPTIONAL
}
```

INTEGER

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

ECPoint

Source: RFC 5915

ASN.1 - ECDSA public key

```
(289,437) SEQUENCE
  (293,333) SEQUENCE
    (297,7) OBJECT IDENTIFIER : ecPublicKey : '1.2.840.10045.2.1'
    (306,320) SEQUENCE
      (310,1) INTEGER : '1'
      (313,60) SEQUENCE
        (315,7) OBJECT IDENTIFIER : prime-field : '1.2.840.10045.1.1'
        (324,49) INTEGER : '00FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEFFFFFFFF0000000000000000FFFFFFFF'
      (375,100) SEQUENCE
        (377,48) OCTET STRING : 'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEFFFFFFFF0000000000000000FFFFFFFC'
        (427,48) OCTET STRING : 'B3312FA7E23EE7E4988E056BE3F82D19181D9C6EFE8141120314088F5013875AC656398D8A2ED19D2A85C8EDD3EC2AEF'
      (477,97) OCTET STRING : '04AA87CA22BE8B05378EB1C71EF320AD746E1D3B628BA79B9859F741E082542A385502F25DBF55296C3A545E3872760AB73617DE4
      (576,49) INTEGER : '00FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFC7634D81F4372DDF581A0DB248B0A77AECEC196ACCC52973'
      (627,1) INTEGER : '1'
  (630,98) BIT STRING UnusedBits: 0 : '04DE965639B7C26DA171D4A64605E98BDB637869B273B227A59A31FC436810A2DEB1B340993766AF3B5E752EB36C515192
```

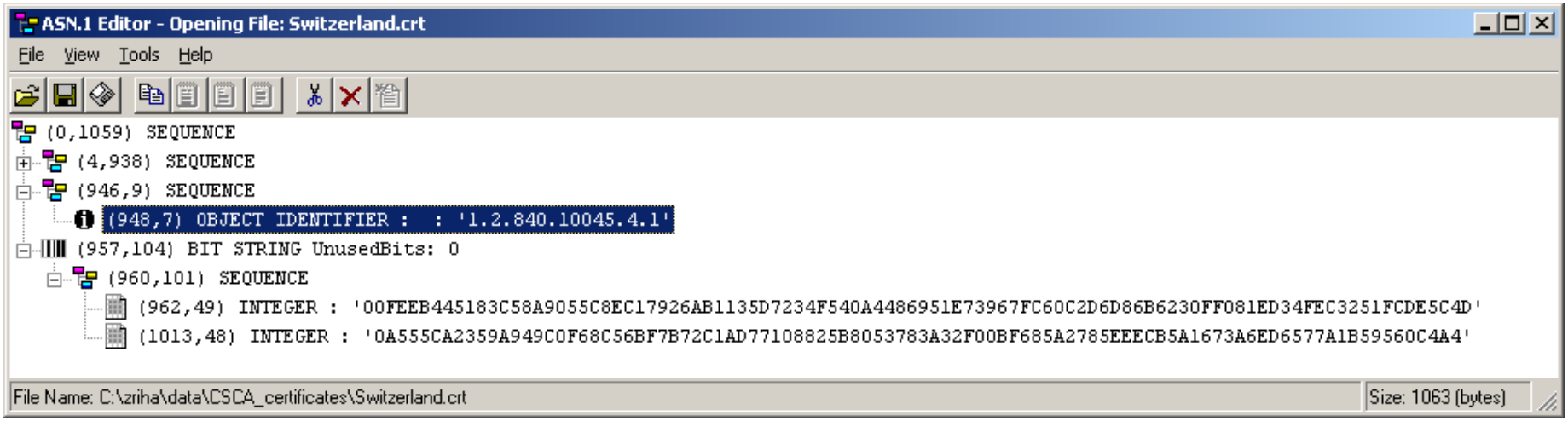
CSCA_Switzerland.crt

ASN.1 – ECDSA signatures

```
ec-signature-value ::= SEQUENCE {
    r     INTEGER,
    s     INTEGER
}
```

Source:
RFC 5480

1.2.840.10045.4.1 - ecdsa-with-SHA1



CSCA_Switzerland.crt

ASN.1 – ECDSA signature OID

ECDSA with SHA1	1.2.840.10045.1 (ANSI X9.62)
ECDSA with SHA1	1.2.840.10045.4.1 (ANSI X9.62)
ECDSA with SHA224	1.2.840.10045.4.3.1 (ANSI X9.62)
ECDSA with SHA256	1.2.840.10045.4.3.2 (ANSI X9.62)
ECDSA with SHA384	1.2.840.10045.4.3.3 (ANSI X9.62)
ECDSA with SHA512	1.2.840.10045.4.3.4 (ANSI X9.62)
ECDSA with SHA1	0.4.0.127.0.7.4.1.1 (BSI)
ECDSA with SHA224	0.4.0.127.0.7.4.1.2 (BSI)
ECDSA with SHA256	0.4.0.127.0.7.4.1.3 (BSI)
ECDSA with SHA384	0.4.0.127.0.7.4.1.4 (BSI)
ECDSA with SHA512	0.4.0.127.0.7.4.1.5 (BSI)

Source:
BSI TR-03105 Part 5.1

ASN.1 - certificates

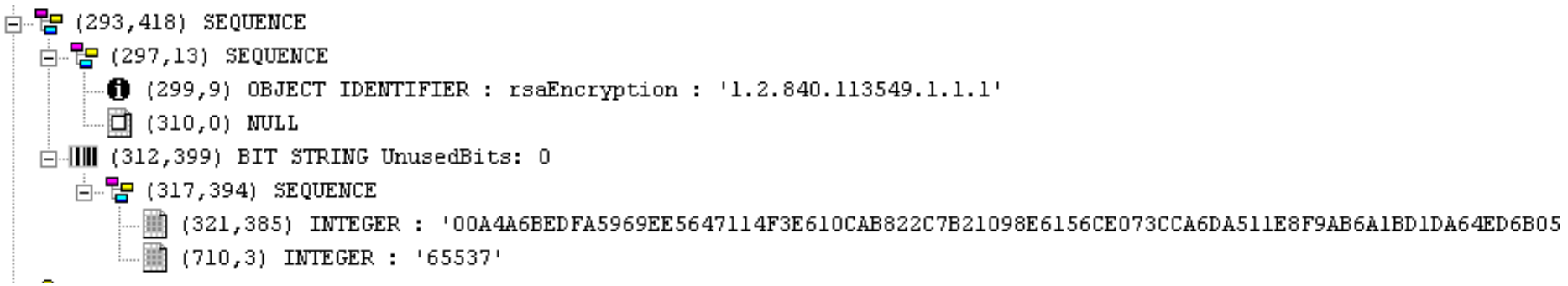
```
Certificate  ::=  SEQUENCE  {
     tbsCertificate       TBSCertificate,
     signatureAlgorithm   AlgorithmIdentifier,
     signatureValue       BIT STRING  }

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 MUST be v2 or v3
     subjectUniqueID [2]  IMPLICIT UniqueIdentifier OPTIONAL,
                          -- If present, version MUST be v2 or v3
     extensions      [3]  EXPLICIT Extensions OPTIONAL
                          -- If present, version MUST be v3
     }

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

Source:
RFC 5280

ASN.1 – certificates - pubkey

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

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

Source:
RFC 5280



CSCA_CZE.crt

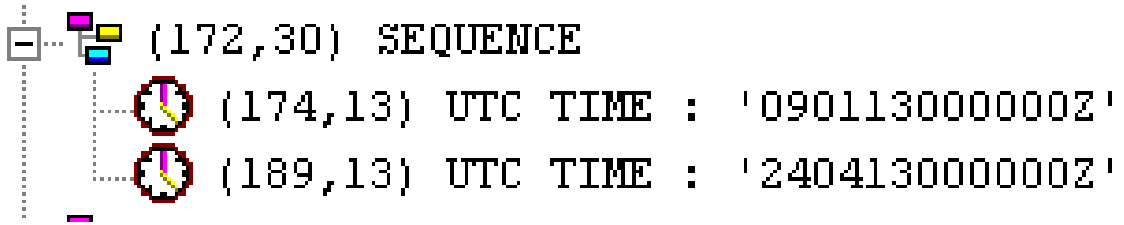
ASN.1 – certificates - times

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

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

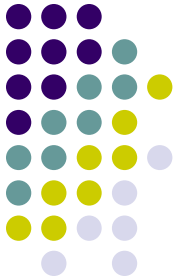
Source:
RFC 5280

- Until 2049: UTCTime
  - YYMMDDHHMMSSZ
- From 2050: GeneralizedTime
  - YYYYMMDDHHMMSSZ

CSCA_CZE.crt



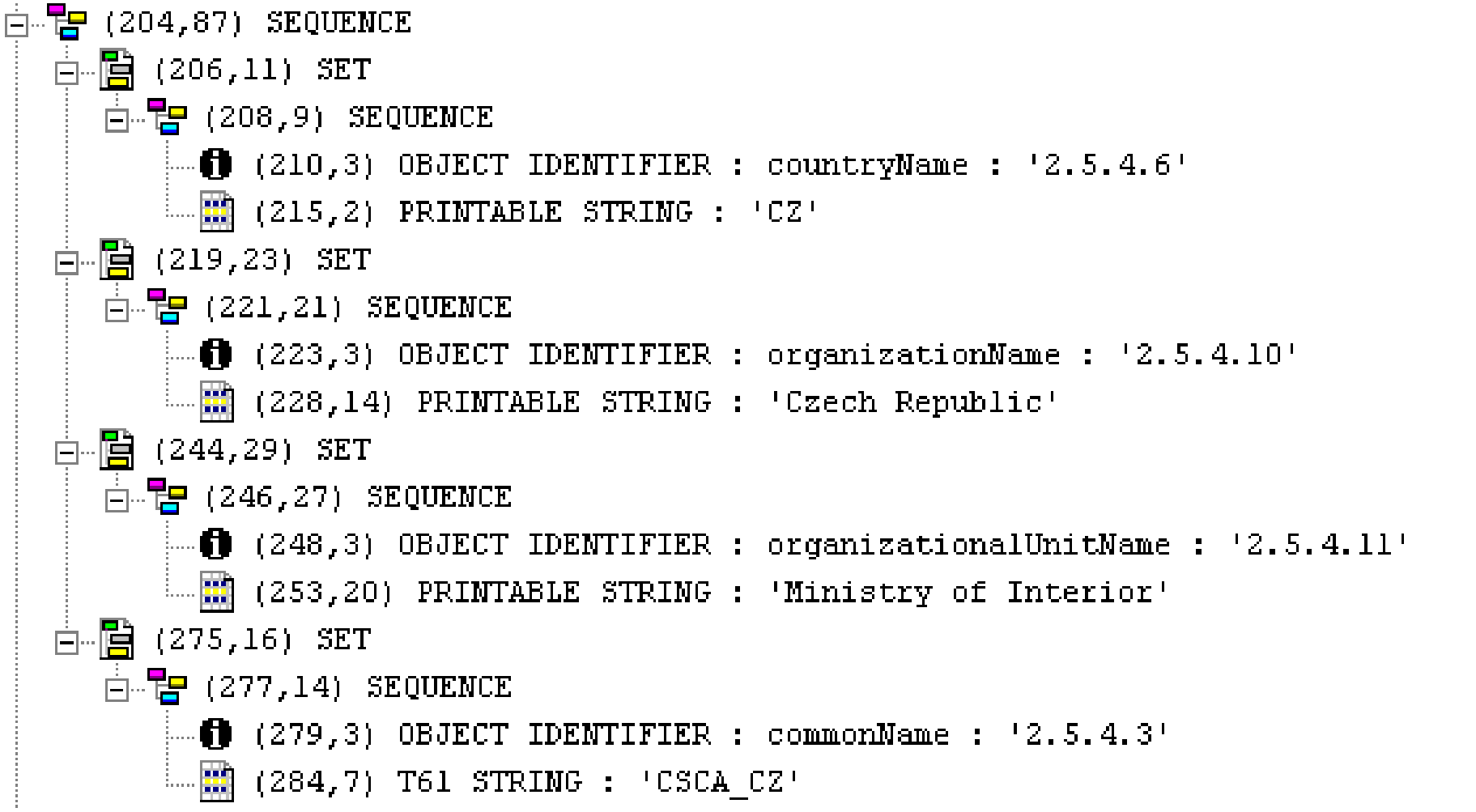
ASN.1 – certificates - names



```
Name ::= CHOICE { -- only one possibility for now --
  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)) }
```

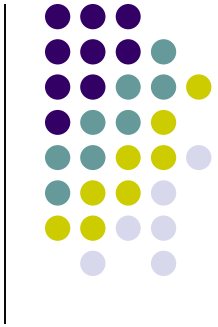
Source:
RFC 5280

ASN.1 – certificate - names



CSCA_CZE.crt

ASN.1 – certificate - names



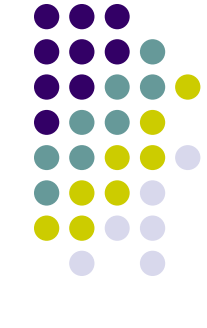
```
commonName ATTRIBUTE  ::=  {
        SUBTYPE OF      name
        WITH SYNTAX     DirectoryString {ub-common-name}
        ID              id-at-commonName }

DirectoryString { INTEGER : maxSize }  ::=  CHOICE  {
        teletexString  TeletexString (SIZE (1..maxSize)),
        printableString     PrintableString (SIZE (1..maxSize)),
        bmpString           BMPString (SIZE (1..maxSize)),
        universalString     UniversalString (SIZE (1..maxSize)),
        uTF8String          UTF8String (SIZE (1..maxSize)) }
```

```
countryName  ATTRIBUTE  ::=  {
        SUBTYPE OF      name
        WITH SYNTAX     CountryName
        SINGLE VALUE    TRUE
        ID              id-at-countryName }

CountryName  ::=  PrintableString (SIZE(2))      --
```

```
-- id-at-objectClass                    OBJECT IDENTIFIER  ::=  {id-at 0}
-- id-at-aliasedEntryName               OBJECT IDENTIFIER  ::=  {id-at 1}
-- id-at-encryptedAliasedEntryName      OBJECT IDENTIFIER  ::=  {id-at 1 2}
id-at-knowledgeInformation              OBJECT IDENTIFIER  ::=  {id-at 2}
id-at-commonName                        OBJECT IDENTIFIER  ::=  {id-at 3}
-- id-at-encryptedCommonName            OBJECT IDENTIFIER  ::=  {id-at 3 2}
id-at-surname                           OBJECT IDENTIFIER  ::=  {id-at 4}
-- id-at-encryptedSurname               OBJECT IDENTIFIER  ::=  {id-at 4 2}
id-at-serialNumber                      OBJECT IDENTIFIER  ::=  {id-at 5}
-- id-at-encryptedSerialNumber          OBJECT IDENTIFIER  ::=  {id-at 5 2}
id-at-countryName                       OBJECT IDENTIFIER  ::=  {id-at 6}
```

Source:
ITU-T X.520

ASN.1 – certificate - names



id-at-localityName	**OBJECT IDENTIFIER ::=**	**{id-at 7}**
-- id-at-encryptedLocalityName	*OBJECT IDENTIFIER ::=*	*{id-at 7 2}*
id-at-collectiveLocalityName	**OBJECT IDENTIFIER ::=**	**{id-at 7 1}**
-- id-at-encryptedCollectiveLocalityName	*OBJECT IDENTIFIER ::=*	*{id-at 7 1 2}*
id-at-stateOrProvinceName	**OBJECT IDENTIFIER ::=**	**{id-at 8}**
-- id-at-encryptedStateOrProvinceName	*OBJECT IDENTIFIER ::=*	*{id-at 8 2}*
id-at-collectiveStateOrProvinceName	**OBJECT IDENTIFIER ::=**	**{id-at 8 1}**
-- id-at-encryptedCollectiveStateOrProvinceName	*OBJECT IDENTIFIER ::=*	*{id-at 8 1 2}*
id-at-streetAddress	**OBJECT IDENTIFIER ::=**	**{id-at 9}**
-- id-at-encryptedStreetAddress	*OBJECT IDENTIFIER ::=*	*{id-at 9 2}*
id-at-collectiveStreetAddress	**OBJECT IDENTIFIER ::=**	**{id-at 9 1}**
-- id-at-encryptedCollectiveStreetAddress	*OBJECT IDENTIFIER ::=*	*{id-at 9 1 2}*
id-at-organizationName	**OBJECT IDENTIFIER ::=**	**{id-at 10}**
-- id-at-encryptedOrganizationName	*OBJECT IDENTIFIER ::=*	*{id-at 10 2}*
id-at-collectiveOrganizationName	**OBJECT IDENTIFIER ::=**	**{id-at 10 1}**
-- id-at-encryptedCollectiveOrganizationName	*OBJECT IDENTIFIER ::=*	*{id-at 10 1 2}*
id-at-organizationalUnitName	**OBJECT IDENTIFIER ::=**	**{id-at 11}**
-- id-at-encryptedOrganizationalUnitName	*OBJECT IDENTIFIER ::=*	*{id-at 11 2}*
id-at-collectiveOrganizationalUnitName	**OBJECT IDENTIFIER ::=**	**{id-at 11 1}**
-- id-at-encryptedCollectiveOrganizationalUnitName	*OBJECT IDENTIFIER ::=*	*{id-at 11 1 2}*
id-at-title	**OBJECT IDENTIFIER ::=**	**{id-at 12}**
-- id-at-encryptedTitle	*OBJECT IDENTIFIER ::=*	*{id-at 12 2}*
id-at-description	**OBJECT IDENTIFIER ::=**	**{id-at 13}**
-- id-at-encryptedDescription	*OBJECT IDENTIFIER ::=*	*{id-at 13 2}*
id-at-searchGuide	**OBJECT IDENTIFIER ::=**	**{id-at 14}**
-- id-at-encryptedSearchGuide	*OBJECT IDENTIFIER ::=*	*{id-at 14 2}*
id-at-businessCategory	**OBJECT IDENTIFIER ::=**	**{id-at 15}**
-- id-at-encryptedBusinessCategory	*OBJECT IDENTIFIER ::=*	*{id-at 15 2}*
id-at-postalAddress	**OBJECT IDENTIFIER ::=**	**{id-at 16}**
-- id-at-encryptedPostalAddress	*OBJECT IDENTIFIER ::=*	*{id-at 16 2}*
id-at-collectivePostalAddress	**OBJECT IDENTIFIER ::=**	**{id-at 16 1}**
-- id-at-encryptedCollectivePostalAddress	*OBJECT IDENTIFIER ::=*	*{id-at 16 1 2}*
id-at-postalCode	**OBJECT IDENTIFIER ::=**	**{id-at 17}**
-- id-at-encryptedPostalCode	*OBJECT IDENTIFIER ::=*	*{id-at 17 2}*
id-at-collectivePostalCode	**OBJECT IDENTIFIER ::=**	**{id-at 17 1}**
-- id-at-encryptedCollectivePostalCode	*OBJECT IDENTIFIER ::=*	*{id-at 17 1 2}*

Source:
ITU-T X.520

Certificate profiles

- For particular areas/purposes there exist certificate profiles which prescribe what kind of attributes will be used in Names
- E.g. for electronic passports ICAO Doc. 9303 states:

The following Attributes SHOULD be used:

- country (country codes SHALL follow the format of two letter country codes, specified in [R16], *ISO/IEC 3166, Codes for the representation of names of countries and their subdivisions — 1997.*).
- organization;
- organizational-unit;
- common name.

Additionally some countries MAY use:

- serial number.

Source:
ICAO Doc. 9303

ASN.1 – certificates – v3

```
UniqueIdentifier  ::=  BIT STRING

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

Extension  ::=  SEQUENCE  {
     extnID      OBJECT IDENTIFIER,
     critical    BOOLEAN DEFAULT FALSE,
     extnValue   OCTET STRING
                 -- contains the DER encoding of an ASN.1 value
                 -- corresponding to the extension type identified
                 -- by extnID
     }
```

Source:
RFC 5280

- Critical x non-critical extensions

ASN.1 – certs – extensions

CSCA_CZE.crt

X509v3 cert extensions

- Authority Key Identifier
  - Identification of the issuing CA
  - Non critical

```
id-ce-authorityKeyIdentifier OBJECT IDENTIFIER ::=  { id-ce 35 }

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

KeyIdentifier ::= OCTET STRING
```

- Similarly “Subject Key Identifier”

Source:
RFC 5280

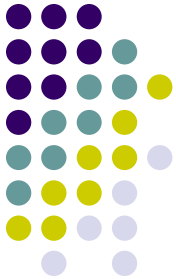
X509v3 cert extensions

- Key Usage
  - Restrictions of the use of the key

```
id-ce-keyUsage OBJECT IDENTIFIER ::=  { id-ce 15 }

 KeyUsage ::= BIT STRING {
      digitalSignature        (0),
      nonRepudiation          (1), -- recent editions of X.509 have
                           -- renamed this bit to contentCommitment
      keyEncipherment         (2),
      dataEncipherment        (3),
      keyAgreement            (4),
      keyCertSign             (5),
      cRLSign                 (6),
      encipherOnly            (7),
      decipherOnly            (8) }
```

Source:
RFC 5280

X509v3 cert extensions

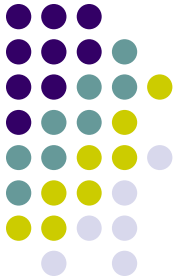


- Extended Key Usage
  - Purposes of the certified key

```
id-ce-extKeyUsage OBJECT IDENTIFIER ::= { id-ce 37 }
ExtKeyUsageSyntax ::= SEQUENCE SIZE (1..MAX) OF KeyPurposeId
KeyPurposeId ::= OBJECT IDENTIFIER
anyExtendedKeyUsage OBJECT IDENTIFIER ::= { id-ce-extKeyUsage 0 }

id-kp OBJECT IDENTIFIER ::= { id-pkix 3 }
id-kp-serverAuth             OBJECT IDENTIFIER ::= { id-kp 1 }
id-kp-clientAuth             OBJECT IDENTIFIER ::= { id-kp 2 }
id-kp-codeSigning             OBJECT IDENTIFIER ::= { id-kp 3 }
id-kp-emailProtection         OBJECT IDENTIFIER ::= { id-kp 4 }
id-kp-timeStamping            OBJECT IDENTIFIER ::= { id-kp 8 }
id-kp-OCSPSigning            OBJECT IDENTIFIER ::= { id-kp 9 }
```

Source:
RFC 5280

X509v3 cert extensions

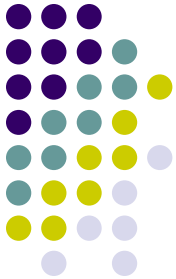


```
id-ce-certificatePolicies OBJECT IDENTIFIER ::=  { id-ce 32 }

anyPolicy OBJECT IDENTIFIER ::= { id-ce-certificatePolicies 0 }

certificatePolicies ::= SEQUENCE SIZE (1..MAX) OF PolicyInformation

PolicyInformation ::= SEQUENCE {
     policyIdentifier   CertPolicyId,
     policyQualifiers   SEQUENCE SIZE (1..MAX) OF
                             PolicyQualifierInfo OPTIONAL }

CertPolicyId ::= OBJECT IDENTIFIER

PolicyQualifierInfo ::= SEQUENCE {
     policyQualifierId  PolicyQualifierId,
     qualifier          ANY DEFINED BY policyQualifierId }

-- policyQualifierIds for Internet policy qualifiers

id-qt          OBJECT IDENTIFIER ::=  { id-pkix 2 }
id-qt-cps      OBJECT IDENTIFIER ::=  { id-qt 1 }
id-qt-unotice  OBJECT IDENTIFIER ::=  { id-qt 2 }

PolicyQualifierId ::= OBJECT IDENTIFIER ( id-qt-cps | id-qt-unotice )

Qualifier ::= CHOICE {
     cPSuri           CPSuri,
     userNotice       UserNotice }

CPSuri ::= IA5String

UserNotice ::= SEQUENCE {
     noticeRef        NoticeReference OPTIONAL,
     explicitText     DisplayText OPTIONAL }

NoticeReference ::= SEQUENCE {
     organization     DisplayText,
     noticeNumbers    SEQUENCE OF INTEGER }

DisplayText ::= CHOICE {
     ia5String        IA5String      (SIZE (1..200)),
     visibleString    VisibleString  (SIZE (1..200)),
     bmpString        BMPString      (SIZE (1..200)),
     utf8String       UTF8String     (SIZE (1..200)) }
```

- Certificate Policies
  - Policy relevant for the issue and use of the certificate
  - Preferably only an OID

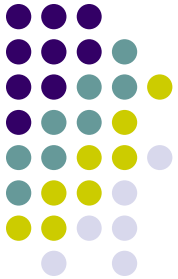
Source:
RFC 5280

X509v3 cert extensions



- Subject Alternative Name
- Issuer Alternative Name
- “Internet style identities”
  - Email
  - DNS name
  - IP address
  - URL
- Must be verified by CA

X509v3 cert extensions



- Basic Constraints
- Is Subject a CA?
- Max. length/depth of the certificate chain/path
  - A pathLenConstraint of zero indicates that no non-self-issued intermediate CA certificates may follow in a valid certification path.

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

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

Source:
RFC 5280

X509v3 cert extensions

- Name Constraints
- Only for CA certificates
- “indicates a name space within which all subject names in subsequent certificates in a certification path MUST be located”

```
id-ce-nameConstraints OBJECT IDENTIFIER ::=  { id-ce 30 }

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

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

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

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

Source:
RFC 5280

X509v3 cert extensions

- Policy Constraints
- Must be critical
- For CA certificates
- Constraints path validation
  - Prohibit policy mapping (or)
  - Require acceptable policy OID in each certificate

```
id-ce-policyConstraints OBJECT IDENTIFIER ::=  { id-ce 36 }

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

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

Source:
RFC 5280

X509v3 cert extensions

- CRL Distribution Points
- How to obtain CRL

```
id-ce-cRLDistributionPoints OBJECT IDENTIFIER ::=  { id-ce 31 }

CRLDistributionPoints ::= 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),
     privilegeWithdrawn      (7),
     aACompromise            (8) }
```

Source:
RFC 5280

ASN.1 – certificate request

```
CertificationRequest ::= SEQUENCE {
 certificationRequestInfo CertificationRequestInfo,
 signatureAlgorithm AlgorithmIdentifier,
 signature BIT STRING
}

CertificationRequestInfo ::= SEQUENCE {
 version INTEGER { v1(0) },
 subject Name,
 subjectPKInfo SubjectPublicKeyInfo,
 attributes [0] Attributes
}

Attributes  ::= SET OF Attribute

Attribute ::= SEQUENCE {
type ATTRIBUTE.&id({IOSet}),
values SET SIZE(1..MAX) OF ATTRIBUTE.&Type({IOSet}{@type})
}
```

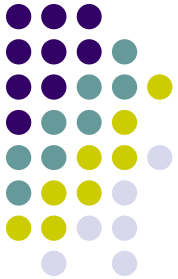
Source:
RFC 5280

ASN.1 - CRL

```
CertificateList  ::=  SEQUENCE  {
     tbsCertList           TBSCertList,
     signatureAlgorithm    AlgorithmIdentifier,
     signatureValue        BIT STRING  }

TBSCertList  ::=  SEQUENCE  {
     version                 Version OPTIONAL,
                                  -- if present, MUST be v2
     signature               AlgorithmIdentifier,
     issuer                  Name,
     thisUpdate              Time,
     nextUpdate              Time OPTIONAL,
     revokedCertificates     SEQUENCE OF SEQUENCE  {
          userCertificate         CertificateSerialNumber,
          revocationDate          Time,
          crlEntryExtensions      Extensions OPTIONAL
                                   -- if present, version MUST be v2
                                }  OPTIONAL,
     crlExtensions           [0]  EXPLICIT Extensions OPTIONAL
                                   -- if present, version MUST be v2
                                }
```

Source:
RFC 5280

ASN.1 – PKCS#7 / CMS



```
ContentInfo ::= SEQUENCE {
  contentType ContentType,
  content [0] EXPLICIT ANY DEFINED BY contentType }

ContentType ::= OBJECT IDENTIFIER

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

DigestAlgorithmIdentifiers ::= SET OF DigestAlgorithmIdentifier

EncapsulatedContentInfo ::= SEQUENCE {
  eContentType ContentType,
  eContent [0] EXPLICIT OCTET STRING OPTIONAL }

SignerInfos ::= SET OF SignerInfo
```

Source:
RFC 5652

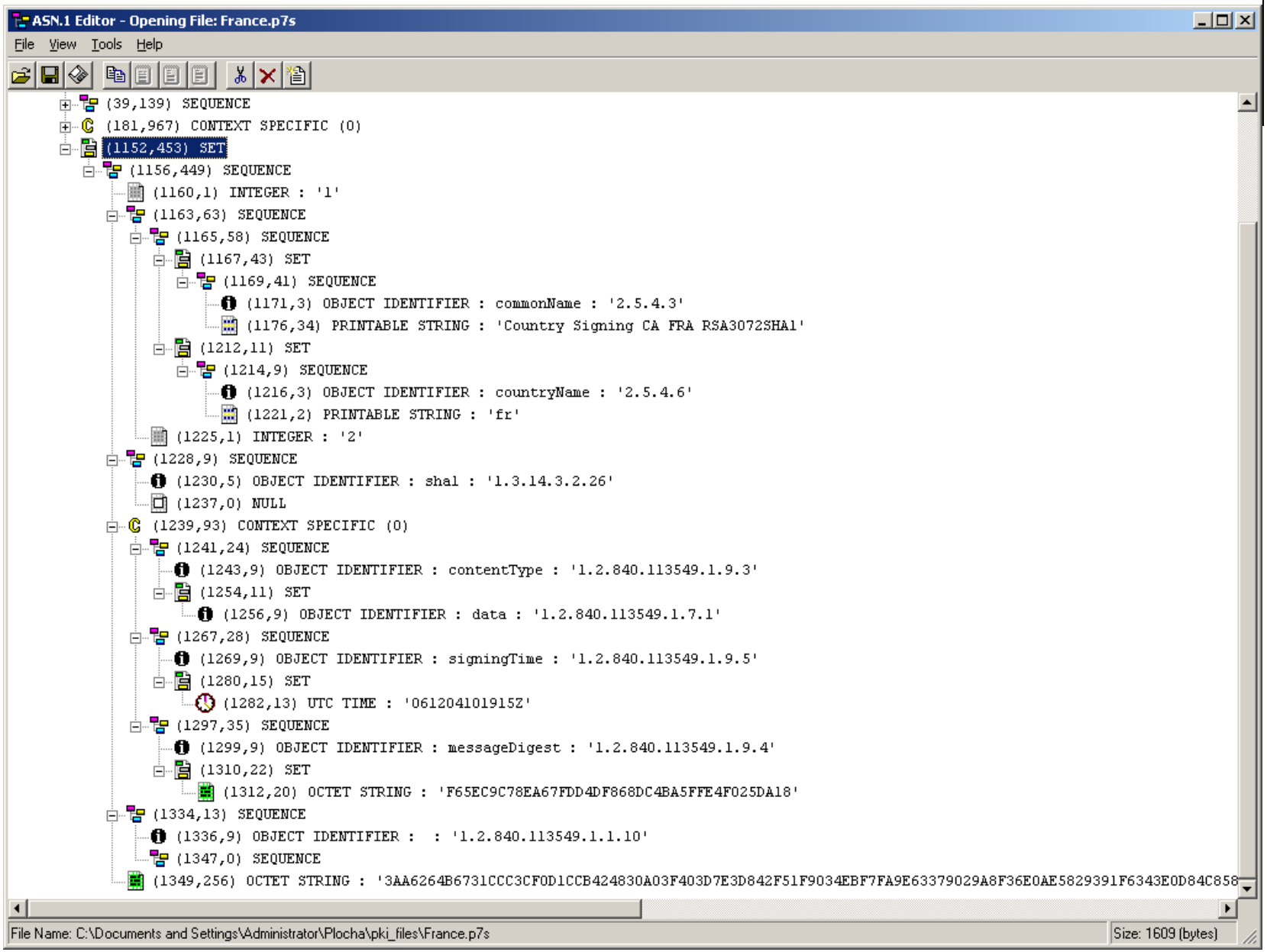
ASN.1 - PKCS#7 / CMS

```
SignerInfo ::= SEQUENCE {
  version CMSVersion,
  sid SignerIdentifier,
  digestAlgorithm DigestAlgorithmIdentifier,
  signedAttrs [0] IMPLICIT SignedAttributes OPTIONAL,
  signatureAlgorithm SignatureAlgorithmIdentifier,
  signature SignatureValue,
  unsignedAttrs [1] IMPLICIT UnsignedAttributes OPTIONAL }

SignerIdentifier ::= CHOICE {
  issuerAndSerialNumber IssuerAndSerialNumber,
  subjectKeyIdentifier [0] SubjectKeyIdentifier }

SignedAttributes ::= SET SIZE (1..MAX) OF Attribute

UnsignedAttributes ::= SET SIZE (1..MAX) OF Attribute

Attribute ::= SEQUENCE {
  attrType OBJECT IDENTIFIER,
  attrValues SET OF AttributeValue }

AttributeValue ::= ANY

SignatureValue ::= OCTET STRING
```

Source:
RFC 5652

PKCS#7 Sample

France.p7s



ASN.1 – PKCS#8

```
-- Private-key information syntax

 PrivateKeyInfo ::= SEQUENCE {
   version Version,
   privateKeyAlgorithm AlgorithmIdentifier,
   privateKey PrivateKey,
   attributes [0] Attributes OPTIONAL }

 Version ::= INTEGER {v1(0)} (v1,...)

 PrivateKey ::= OCTET STRING

 Attributes ::= SET OF Attribute

-- Encrypted private-key information syntax

 EncryptedPrivateKeyInfo ::= SEQUENCE {
     encryptionAlgorithm AlgorithmIdentifier,
     encryptedData EncryptedData
 }

 EncryptedData ::= OCTET STRING
```

Source:
PKCS#8