New ASN.1 Modules for PKIX
draft-hoffman-pkix-new-asn1-00.txt

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on May 11, 2008.

Copyright Notice

Copyright (C) The IETF Trust (2007).

Abstract

The PKIX certificate format, and many associated formats, are expressed using ASN.1. The current ASN.1 modules conform to the 1988 version of ASN.1. This document updates those ASN.1 modules to conform to the 2002 version of ASN.1. There are no bits-on-the-wire changes to any of the formats; this is simply a change to the syntax.
Table of Contents

1. Introduction ................................................. 3
   1.1. Issues .................................................. 3
       1.1.1. More Modules To Be Added ........................... 3
       1.1.2. Algorithm Structure ................................. 4
       1.1.3. Module OIDs Changing ............................... 4
2. ASN.1 Module for RFC 2560 ............................... 4
3. ASN.1 Module for RFC 2986 ............................... 7
4. ASN.1 Module for RFC 3279 ............................... 8
5. ASN.1 Module for RFC 3280 (Explicit) ................... 14
6. ASN.1 Module for RFC 3280 (Implicit) ..................... 26
7. ASN.1 Module for RFC 3281 ............................... 35
8. ASN.1 Module for RFC 3852 (Attribute Certificate v1) ..... 40
9. ASN.1 Module for RFC 4210 ............................... 41
10. ASN.1 Module for RFC 4211 ............................... 51
11. ASN.1 Module for RFC-to-be, SCVP ....................... 57
12. Security Considerations ................................... 66
13. Normative References ..................................... 66
Authors’ Addresses ............................................. 67
Intellectual Property and Copyright Statements ............... 68
1. Introduction

Some developers would like the IETF to use the latest version of ASN.1 in its standards. Most of the RFCs that relate to security protocols still use ASN.1 from the 1988 standard, which has been deprecated. This is particularly true for the standards that relate to PKIX, CMS, and S/MIME.

This document updates the following RFCs to use ASN.1 modules that conform to the 2002 version of ASN.1 [ASN1-2002]. Note that not all the modules are updated; some are included to simply make the set compete.

- RFC 2560, PKIX Online Certificate Status Protocol (OCSP) [RFC2560]
- RFC 2986, PKCS #10 certificate request [RFC2986]
- RFC 3279, PKIX algorithms and identifier [RFC3279]
- RFC 3280, PKIX certificate and CRL profile [RFC3280] (both the implicit and explicit modules)
- RFC 3281, PKIX attribute certificates, version 2 [RFC3281]
- RFC 3852, contains PKIX attribute certificates, version 1 [RFC3852]
- RFC 4210, PKIX CMP (Certificate Management Protocol) [RFC4210]
- RFC 4211, PKIX CRMF (Certificate Request Message Format) [RFC4211]
- RFC-to-be, PKIX SCVP (Server-based Certificate Validation Protocol) [SCVP]

Note that some of the modules in this document get some of their definitions from places different than the modules in the original RFCs. The idea is that these modules, when combined with the modules in [NEW-CMS-SMIME] can stand on their own and do not need to import definitions from anywhere else.

1.1. Issues

This section will be removed before final publication.

1.1.1. More Modules To Be Added

There are many modules from standards-track RFCs that are not listed in this document or the companion document on CMS and S/MIME. We
will discuss with the two communities which modules are appropriate for the two documents. We will also consider making "super-modules", individual modules which might update multiple RFCs at one time. We may also add objects to some of the modules.

1.1.2. Algorithm Structure

Algorithms are currently not defined here. We need to discuss what structure we want for algorithm objects. Currently, we just do "parameter, OID", but we could add more. Because we don’t know what the final structure is, the object sets in the various modules are commented out. We will fix this before finishing this project.

1.1.3. Module OIDs Changing

The OIDs given in the modules in this version of the document are the same as the OIDs from the original modules, even though some of the modules have changed syntax. That is clearly incorrect. In a later version of this document, we will change the OIDs for every changed module.

2. ASN.1 Module for RFC 2560

OCSP

{iso(1) identified-organization(3) dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) id-mod-ocsp(14)}

DEFINITIONS EXPLICIT TAGS ::= BEGIN

IMPORTS

AuthorityInfoAccessSyntax, GeneralName FROM PKIX1Implicit88
{iso(1) identified-organization(3) dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) 19} -- Change module number

Name, CertificateSerialNumber, Extensions, id-kp, id-ad-ocsp, Certificate, AlgorithmIdentifier FROM PKIX1Explicit88
{iso(1) identified-organization(3) dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) 18};

CRLReason ::= INTEGER

OCSPRequest ::= SEQUENCE {
  tbsRequest TBSRequest,
...}
optionalSignature   [0]     EXPLICIT Signature OPTIONAL } 

TBSRequest ::= SEQUENCE { 
version [0] EXPLICIT Version DEFAULT v1, 
requestorName [1] EXPLICIT GeneralName OPTIONAL, 
requestList     SEQUENCE OF Request, 
requestExtensions [2] EXPLICIT Extensions OPTIONAL } 

Signature ::= SEQUENCE { 
signatureAlgorithm AlgorithmIdentifier, 
signature            BIT STRING, 
certs            [0] EXPLICIT SEQUENCE OF Certificate OPTIONAL } 

Version ::= INTEGER { v1(0) } 

Request ::= SEQUENCE { 
reqCert                    CertID, 
singleRequestExtensions    [0] EXPLICIT Extensions OPTIONAL } 

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

OCSPResponse ::= SEQUENCE { 
responseStatus         OCSPResponseStatus, 
responseBytes          [0] EXPLICIT ResponseBytes OPTIONAL } 

OCSPResponseStatus ::= ENUMERATED { 
successful            (0), --Response has valid confirmations 
malformedRequest      (1), --Illegal confirmation request 
internalError         (2), --Internal error in issuer 
tryLater              (3), --Try again later 
-- (4) is not used 
sigRequired           (5), --Must sign the request 
unauthorized          (6)  --Request unauthorized } 

ResponseBytes ::= SEQUENCE { 
responseType OBJECT IDENTIFIER, 
response     OCTET STRING } 

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 }

KeyHash ::= OCTET STRING --SHA-1 hash of responder's public key
            -- (excluding the tag and length fields)

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

ArchiveCutoff ::= GeneralizedTime

AcceptableResponses ::= SEQUENCE OF OBJECT IDENTIFIER

ServiceLocator ::= SEQUENCE {
  issuer    Name,
  locator   AuthorityInfoAccessSyntax }

-- Object Identifiers

id-kp-OCSPSigning          OBJECT IDENTIFIER ::= { id-kp 9 }
id-pkix-ocsp               OBJECT IDENTIFIER ::= { id-ad-ocsp }
id-pkix-ocsp-basic         OBJECT IDENTIFIER ::= { id-pkix-ocsp 1 }
id-pkix-ocsp-nonce         OBJECT IDENTIFIER ::= { id-pkix-ocsp 2 }
id-pkix-ocsp-crl           OBJECT IDENTIFIER ::= { id-pkix-ocsp 3 }
id-pkix-ocsp-response      OBJECT IDENTIFIER ::= { id-pkix-ocsp 4 }
id-pkix-ocsp-nocheck OBJECT IDENTIFIER ::= { id-pkix-ocsp 5 }
id-pkix-ocsp-archive-cutoff OBJECT IDENTIFIER ::= { id-pkix-ocsp 6 }
id-pkix-ocsp-service-locator OBJECT IDENTIFIER ::= { id-pkix-ocsp 7 }

END

3. ASN.1 Module for RFC 2986

PKCS-10

{iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-10(10)
  modules(1) pkcs-10(1)}
DEFINITIONS IMPLICIT TAGS ::= BEGIN
IMPORTS
ALGORITHM, ATTRIBUTE, Name
FROM PKIX1Explicit88
{ iso(1) identified-organization(3) dod(6) internet(1) security(5)
  mechanisms(5) pkix(7) id-mod(0) id-pkix1-explicit(18) };--- Certificate requests
CertificateRequestInfo ::= SEQUENCE {
  version INTEGER { v1(0) } (v1, ... ),
  subject Name,
  subjectPKInfo SubjectPublicKeyInfo({ PKInfoAlgorithms }},
  attributes [0] Attributes{{ CRIAttributes }}
}

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

PKInfoAlgorithms ALGORITHM ::= {
  ... -- add any locally defined algorithms here -- }

Attributes { ATTRIBUTE:IOSet } ::= SET OF Attribute{{ IOSet }}

CRIAttributes ATTRIBUTE ::= {
  ... -- add any locally defined attributes here -- }

Attribute { ATTRIBUTE:IOSet } ::= SEQUENCE {
  type ATTRIBUTE.&id({IOSet}),
  values SET SIZE(1..MAX) OF ATTRIBUTE.&Type({IOSet}{@type})
}
CertificationRequest ::= SEQUENCE {
  certificationRequestInfo CertificationRequestInfo,
  signatureAlgorithm AlgorithmIdentifier{{ SignatureAlgorithms }},
  signature          BIT STRING
}

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

SignatureAlgorithms ALGORITHM ::= {
  ... -- add any locally defined algorithms here -- }

END

4. ASN.1 Module for RFC 3279

PKIX1Algorithms88
  { iso(1) identified-organization(3) dod(6) internet(1) security(5)  
    mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-algorithms(17) }  
DEFINITIONS EXPLICIT TAGS ::= BEGIN  

--  One-way Hash Functions  
--

md2  OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) rsadsi(113549)  
  digestAlgorithm(2) 2 }

md5  OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) rsadsi(113549)  
  digestAlgorithm(2) 5 }

id-sha1  OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) oiw(14) secsig(3)  
  algorithms(2) 26 }

--  DSA Keys and Signatures  
--

--  OID for DSA public key
id-dsa OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) x9-57(10040) x9algorithm(4) 1 }

-- encoding for DSA public key

DSAPublicKey ::= INTEGER -- public key, y

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

-- OID for DSA signature generated with SHA-1 hash

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

-- encoding for DSA signature generated with SHA-1 hash

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

-- RSA Keys and Signatures

-- arc for RSA public key and RSA signature OIDs

pkcs-1 OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) 1 }

-- OID for RSA public keys

rsaEncryption OBJECT IDENTIFIER ::= { pkcs-1 1 }

-- OID for RSA signature generated with MD2 hash

md2WithRSAEncryption OBJECT IDENTIFIER ::= { pkcs-1 2 }

-- OID for RSA signature generated with MD5 hash

md5WithRSAEncryption OBJECT IDENTIFIER ::= { pkcs-1 4 }

-- OID for RSA signature generated with SHA-1 hash

sha1WithRSAEncryption OBJECT IDENTIFIER ::= { pkcs-1 5 }
RSAPublicKey ::= SEQUENCE {
  modulus          INTEGER,  -- n
  publicExponent   INTEGER }  -- e

-- -- Diffie-Hellman Keys
-

dhpublicnumber OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) ansi-x942(10046)
  number-type(2) 1 }

-- encoding for DSA public key

DHPublicKey ::= INTEGER  -- public key, y = g^x mod p

DomainParameters ::= SEQUENCE {
  p        INTEGER,           -- odd prime, p = jq + 1
  g        INTEGER,           -- generator, g
  q        INTEGER,           -- factor of p-1
  j        INTEGER OPTIONAL,  -- subgroup factor, j >= 2
  validationParms  ValidationParms OPTIONAL }

ValidationParms ::= SEQUENCE {
  seed             BIT STRING,
  pgenCounter      INTEGER }

-- -- KEA Keys
-

id-keyExchangeAlgorithm OBJECT IDENTIFIER ::= {
  2 16 840 1 101 2 1 1 22 }

KEA-Parms-Id ::= OCTET STRING

-- -- Elliptic Curve Keys, Signatures, and Curves
--

ansi-X9-62 OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) 10045 }

FIELD-ID ::= TYPE-IDENTIFIER

FieldID ::= -- Finite field
SEQUENCE {
    fieldType   FIELD-ID.
    &id({SupportedFields}),
    parameters  FIELD-ID.
    &Type({SupportedFields}{@fieldType}) OPTIONAL
}

SupportedFields FIELD-ID ::= {fid-prime-field | fid-characteristic-two-field}

-- Arc for ECDSA signature OIDS
id-ecSigType OBJECT IDENTIFIER ::= { ansi-X9-62 signatures(4) }

-- OID for ECDSA signatures with SHA-1
ecdsa-with-SHA1 OBJECT IDENTIFIER ::= { id-ecSigType 1 }

-- OID for an elliptic curve signature
-- format for the value of an ECDSA signature value
ECDSA-Sig-Value ::= SEQUENCE {
    r     INTEGER,
    s     INTEGER }

-- recognized field type OIDs are defined in the following arc
id-fieldType OBJECT IDENTIFIER ::= { ansi-X9-62 fieldType(1) }

-- where fieldType is prime-field, the parameters are of type Prime-p
fid-prime-field FIELD-ID ::= (Prime-p IDENTIFIED BY prime-field)

prime-field OBJECT IDENTIFIER ::= { id-fieldType 1 }

Prime-p ::= INTEGER -- Finite field F(p), where p is an odd prime

-- where fieldType is characteristic-two-field, the parameters are
-- of type Characteristic-two

fid-characteristic-two-field FIELD-ID ::= {Characteristic-two IDENTIFIED BY characteristic-two-field}

characteristic-two-field OBJECT IDENTIFIER ::= { id-fieldType 2 }

CHARACTERISTIC-TWO ::= TYPE-IDENTIFIER

Characteristic-two ::= SEQUENCE {
m INTEGER, -- Field size 2^m
basis CHARACTERISTIC-TWO.
  &id({SupportedCharacteristicTwo}),
parameters CHARACTERISTIC-TWO.
  &Type({SupportedCharacteristicTwo}{@basis})
}

SupportedCharacteristicTwo CHARACTERISTIC-TWO ::= 
  {char2-gnBasis | char2-tpBasis | char2-ppBasis }

-- recognized basis type OIDs are defined in the following arc
id-characteristic-two-basis OBJECT IDENTIFIER ::= { 
  characteristic-two-field basisType(3) }

-- gnbasis is identified by OID gnBasis and indicates
-- parameters are NULL
char2-gnBasis CHARACTERISTIC-TWO ::= {NULL IDENTIFIED BY gnBasis}

gnBasis OBJECT IDENTIFIER ::= { id-characteristic-two-basis 1 }

-- parameters for this basis are NULL

-- trinominal basis is identified by OID tpBasis and indicates
-- parameters of type Pentanomial
char2-tpBasis CHARACTERISTIC-TWO ::= 
  {Trinomial IDENTIFIED BY tpBasis}

tpBasis OBJECT IDENTIFIER ::= { id-characteristic-two-basis 2 }

-- Trinomial basis representation of F2^m
-- Integer k for reduction polynomial xm + xk + 1

Trinomial ::= INTEGER

-- for pentonomial basis is identified by OID ppBasis and indicates
-- parameters of type Pentanomial
char2-ppBasis CHARACTERISTIC-TWO ::= 
  {Pentanomial IDENTIFIED BY ppBasis}

ppBasis OBJECT IDENTIFIER ::= { id-characteristic-two-basis 3 }

-- Pentanomial basis representation of F2^m
-- reduction polynomial integers k1, k2, k3
-- f(x) = x**m + x**k3 + x**k2 + x**k1 + 1
Pentanomial ::= SEQUENCE {
    k1 INTEGER,
    k2 INTEGER,
    k3 INTEGER }

-- The object identifiers gnBasis, tpBasis and ppBasis name
-- three kinds of basis for characteristic-two finite fields

FieldElement ::= OCTET STRING             -- Finite field element

ECPoint ::= OCTET STRING                 -- Elliptic curve point

-- Elliptic Curve parameters may be specified explicitly,
-- specified implicitly through a "named curve", or
-- inherited from the CA

EcpkParameters ::= CHOICE {
    ecParameters ECPParameters,
    namedCurve OBJECT IDENTIFIER,
    implicitlyCA NULL }

ECPParameters ::= SEQUENCE {         -- Elliptic curve parameters
    version ECPVer,
    fieldID FieldID,
    curve Curve,
    base  ECPPoint,                -- Base point G
    order INTEGER,                -- Order n of the base point
    cofactor INTEGER OPTIONAL }     -- The integer h = #E(Fq)/n

ECPVer ::= INTEGER {ecpVer1(1)}

Curve ::= SEQUENCE {
    a FieldElement,            -- Elliptic curve coefficient a
    b FieldElement,            -- Elliptic curve coefficient b
    seed BIT STRING OPTIONAL }  

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

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


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

c-TwoCurve OBJECT IDENTIFIER ::= { ellipticCurve characteristicTwo(0) }

c2pnb163v1  OBJECT IDENTIFIER ::= { c-TwoCurve 1 }
c2pnb163v2  OBJECT IDENTIFIER ::= { c-TwoCurve  2 }
c2pnb163v3  OBJECT IDENTIFIER ::= { c-TwoCurve  3 }
c2pnb176w1  OBJECT IDENTIFIER ::= { c-TwoCurve  4 }
c2tnb191v1  OBJECT IDENTIFIER ::= { c-TwoCurve  5 }
c2tnb191v2  OBJECT IDENTIFIER ::= { c-TwoCurve  6 }
c2tnb191v3  OBJECT IDENTIFIER ::= { c-TwoCurve  7 }
c2onb191v4  OBJECT IDENTIFIER ::= { c-TwoCurve  8 }
c2onb191v5  OBJECT IDENTIFIER ::= { c-TwoCurve  9 }
c2pnb208w1  OBJECT IDENTIFIER ::= { c-TwoCurve 10 }
c2tnb239v1  OBJECT IDENTIFIER ::= { c-TwoCurve 11 }
c2tnb239v2  OBJECT IDENTIFIER ::= { c-TwoCurve 12 }
c2tnb239v3  OBJECT IDENTIFIER ::= { c-TwoCurve 13 }
c2onb239v4  OBJECT IDENTIFIER ::= { c-TwoCurve 14 }
c2onb239v5  OBJECT IDENTIFIER ::= { c-TwoCurve 15 }
c2pnb272w1  OBJECT IDENTIFIER ::= { c-TwoCurve 16 }
c2pnb304w1  OBJECT IDENTIFIER ::= { c-TwoCurve 17 }
c2tnb359v1  OBJECT IDENTIFIER ::= { c-TwoCurve 18 }
c2pnb368w1  OBJECT IDENTIFIER ::= { c-TwoCurve 19 }
c2tnb431r1  OBJECT IDENTIFIER ::= { c-TwoCurve 20 }

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

prime192v1  OBJECT IDENTIFIER ::= { primeCurve  1 }
prime192v2  OBJECT IDENTIFIER ::= { primeCurve  2 }
prime192v3  OBJECT IDENTIFIER ::= { primeCurve  3 }
prime239v1  OBJECT IDENTIFIER ::= { primeCurve  4 }
prime239v2  OBJECT IDENTIFIER ::= { primeCurve  5 }
prime239v3  OBJECT IDENTIFIER ::= { primeCurve  6 }
prime256v1  OBJECT IDENTIFIER ::= { primeCurve  7 }

END

5. ASN.1 Module for RFC 3280 (Explicit)

PKIX1Explicit88
{ iso(1) identified-organization(3) dod(6) internet(1) security(5)
  mechanisms(5) pkix(7) id-mod(0) id-pkix1-explicit(18) }
DEFINITIONS EXPLICIT TAGS ::= BEGIN

id-pkix OBJECT IDENTIFIER ::= { iso(1) identified-organization(3) dod(6) internet(1) security(5)
  mechanisms(5) pkix(7) }

-- PKIX arcs
id-pe OBJECT IDENTIFIER ::=  { id-pkix 1 }
  -- arc for private certificate extensions
id-qt OBJECT IDENTIFIER ::=  { id-pkix 2 }
  -- arc for policy qualifier types
id-kp OBJECT IDENTIFIER ::=  { id-pkix 3 }
  -- arc for extended key purpose OIDS
id-ad OBJECT IDENTIFIER ::=  { id-pkix 48 }
  -- arc for access descriptors

-- policyQualifierIds for Internet policy qualifiers
id-qt-cps OBJECT IDENTIFIER ::=  { id-qt 1 }
  -- OID for CPS qualifier
id-qt-unotice OBJECT IDENTIFIER ::=  { id-qt 2 }
  -- OID for user notice qualifier

-- access descriptor definitions
id-ad-ocsp OBJECT IDENTIFIER ::=  { id-ad 1 }
id-ad-caIssuers OBJECT IDENTIFIER ::=  { id-ad 2 }
id-ad-timeStamping OBJECT IDENTIFIER ::=  { id-ad 3 }
id-ad-caRepository OBJECT IDENTIFIER ::=  { id-ad 5 }

-- attribute data types

ATTRIBUTE ::= TYPE-IDENTIFIER

Attribute ::= SEQUENCE {
  type  ATTRIBUTE.&id({SupportedAttributes}),
  values  SET OF  ATTRIBUTE.&Type({SupportedAttributes}{@type})
}
  -- at least one value is required

SupportedAttributes ATTRIBUTE ::= {
  commonName | x520name | x520LocalityName |
  x520StateOrProvinceName | x520OrganizationName |
  x520OrganizationalUnitName | x520Title | x520dnQualifier |
  x520countryName | x520SerialNumber | x520Pseudonym |
  domainComponent | emailAddress, ... }

AttributeType ::= OBJECT IDENTIFIER

AttributeTypeAndValue ::= SEQUENCE {
  type  ATTRIBUTE.&id({SupportedAttributes}),
  value  ATTRIBUTE.&Type({SupportedAttributes}{@type})
}

-- suggested naming attributes: Definition of the following
-- information object set may be augmented to meet local
requirements. Note that deleting members of the set may prevent interoperability with conforming implementations. Presented in pairs: the AttributeType followed by the type definition for the corresponding AttributeValue.

Arc for standard naming attributes

id-at OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) ds(5) 4 }

-- Naming attributes of type X520name

id-at-name AttributeType ::= { id-at 41 }
id-at-surname AttributeType ::= { id-at 4 }
id-at-givenName AttributeType ::= { id-at 42 }
id-at-initials AttributeType ::= { id-at 43 }
id-at-generationQualifier AttributeType ::= { id-at 44 }

-- Directory string type --

DirectoryString{INTEGER:maxSize} ::= CHOICE {
teletexString TeletexString(SIZE (1..maxSize)),
printableString PrintableString(SIZE(1..maxSize)),
universalString UniversalString(SIZE(1..maxSize)),
utf8String UTF8String(SIZE(1..maxSize)),
bmpString BMPString(SIZE(1..maxSize))
}

x520name ATTRIBUTE ::= { X520name IDENTIFIED BY id-at-name }

X520name ::= DirectoryString {ub-name}

-- Naming attributes of type X520CommonName

id-at-commonName AttributeType ::= { id-at 3 }

commonName ATTRIBUTE ::= {CommonName IDENTIFIED BY id-at-commonName } CommonName ::= DirectoryString {ub-common-name}

-- Naming attributes of type X520LocalityName

id-at-localityName AttributeType ::= { id-at 7 }

x520LocalityName ATTRIBUTE ::= { DirectoryString {ub-locality-name} IDENTIFIED BY id-at-localityName }

-- Naming attributes of type X520StateOrProvinceName

id-at-stateOrProvinceName AttributeType ::= { id-at 8 }

x520StateOrProvinceName ATTRIBUTE ::=
{ DirectoryString {ub-state-name}
   IDENTIFIED BY id-at-stateOrProvinceName }

-- Naming attributes of type X520OrganizationName

id-at-organizationName AttributeType ::= { id-at 10 }

x520OrganizationName ATTRIBUTE ::= { DirectoryString {ub-organization-name}
   IDENTIFIED BY id-at-organizationName }

-- Naming attributes of type X520OrganizationalUnitName

id-at-organizationalUnitName AttributeType ::= { id-at 11 }

x520OrganizationalUnitName ATTRIBUTE ::= { DirectoryString {ub-organizational-unit-name}
   IDENTIFIED BY id-at-organizationalUnitName }

-- Naming attributes of type X520Title

id-at-title AttributeType ::= { id-at 12 }

x520Title ATTRIBUTE ::= { DirectoryString {ub-title}
   IDENTIFIED BY id-at-title }

-- Naming attributes of type X520dnQualifier

id-at-dnQualifier AttributeType ::= { id-at 46 }

x520dnQualifier ATTRIBUTE ::= { PrintableString
   IDENTIFIED BY id-at-dnQualifier }

-- Naming attributes of type X520countryName (digraph from IS 3166)

id-at-countryName AttributeType ::= { id-at 6 }

x520countryName ATTRIBUTE ::= { PrintableString (SIZE (2))
   IDENTIFIED BY id-at-countryName }

-- Naming attributes of type X520SerialNumber

id-at-serialNumber AttributeType ::= { id-at 5 }

x520SerialNumber ATTRIBUTE ::= { PrintableString (SIZE (1..ub-serial-number)) IDENTIFIED BY id-at-serialNumber }

-- Naming attributes of type X520Pseudonym
id-at-pseudonym  
AttributeType ::= { id-at 65 }

x520Pseudonym ATTRIBUTE ::= { DirectoryString {ub-pseudonym}
     IDENTIFIED BY id-at-pseudonym }
-- Naming attributes of type DomainComponent (from RFC 2247)

id-domainComponent  
AttributeType ::= { 0 9 2342 19200300 100 1 25 }

domainComponent ATTRIBUTE ::= {IA5String
     IDENTIFIED BY id-domainComponent }

-- Legacy attributes

pkcs-9 OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) 9 }

id-emailAddress  
AttributeType ::= { pkcs-9 1 }

emailAddress ATTRIBUTE ::= {IA5String
     (SIZE (1..ub-emailaddress-length)) IDENTIFIED BY
     id-emailAddress }

-- naming data types --

Name ::= CHOICE { -- only one possibility for now --
    rdnSequence  RDNSequence }

RDNSequence ::= SEQUENCE OF RelativeDistinguishedName

DistinguishedName ::= RDNSequence

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

-- certificate and CRL specific structures begin here

Certificate ::= SIGNED{TBSCertificate}

TBSCertificate ::= SEQUENCE {
    version [0]  Version DEFAULT v1,
    serialNumber  CertificateSerialNumber,
    signature  AlgorithmIdentifier,
    issuer  Name,
    validity  Validity,
    subject  Name,
    subjectPublicKeyInfo  SubjectPublicKeyInfo,
    ... ,}
Version ::= INTEGER { v1(0), v2(1), v3(2) }

CertificateSerialNumber ::= INTEGER

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

Time ::= CHOICE {
  utcTime      UTCTime,
  generalTime  GeneralizedTime )

UniqueIdentifier ::= BIT STRING

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

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

EXTENSION ::= CLASS {&id OBJECT IDENTIFIER UNIQUE,
  &ExtnType
)WITH SYNTAX {SYNTAX &ExtnType
    IDENTIFIED BY &id
  }

Extension ::= SEQUENCE {
  extnID       EXTENSION.&id({ExtensionSet}),
  critical     BOOLEAN DEFAULT FALSE,
  extnValue    OCTET STRING (CONTAINING
    EXTENSION.&ExtnType({ExtensionSet}[@extnID]))
}

ExtensionSet EXTENSION ::= { ... }

-- CRL structures

CertificateList ::= SIGNED(TBSCertList)

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,
    ...,
}[2:      -- if present, MUST be v2
    crlEntryExtensions Extensions OPTIONAL
}, ...
} OPTIONAL,
...,
[2:      -- if present, MUST be v2
    crlExtensions [0] Extensions OPTIONAL
]}, ... }

-- Version, Time, CertificateSerialNumber, and Extensions were
-- defined earlier for use in the certificate structure

ALGORITHM ::= TYPE-IDENTIFIER

AlgorithmIdentifier ::= SEQUENCE {  
algorithm ALGORITHM.
    &id({SupportedAlgorithms}),
parameters ALGORITHM.
    &Type({SupportedAlgorithms}{@algorithm}) OPTIONAL  
    -- contains a value of the type
    -- registered for use with the
    -- algorithm object identifier value

SupportedAlgorithms ALGORITHM ::= { ... }

-- X.400 address syntax starts here

ORAddress ::= SEQUENCE {  
built-in-standard-attributes BuiltInStandardAttributes,
built-in-domain-defined-attributes
    BuiltInDomainDefinedAttributes OPTIONAL,
    -- see also teletex-domain-defined-attributes
extension-attributes ExtensionAttributes OPTIONAL }

-- Built-in Standard Attributes
BuiltInStandardAttributes ::= SEQUENCE {
    country-name                  CountryName OPTIONAL,
    administration-domain-name   AdministrationDomainName OPTIONAL,
    network-address           [0] IMPLICIT NetworkAddress OPTIONAL,
    -- see also extended-network-address
    terminal-identifier       [1] IMPLICIT TerminalIdentifier OPTIONAL,
    private-domain-name       [2] PrivateDomainName OPTIONAL,
    organization-name         [3] IMPLICIT OrganizationName OPTIONAL,
    -- see also teletex-organization-name
    numeric-user-identifier   [4] IMPLICIT NumericUserIdentifier
                           OPTIONAL,
    personal-name             [5] IMPLICIT PersonalName OPTIONAL,
    -- see also teletex-personal-name
    organizational-unit-names [6] IMPLICIT OrganizationalUnitNames
                           OPTIONAL }
    -- see also teletex-organizational-unit-names

CountryName ::= [APPLICATION 1] CHOICE {
    x121-dcc-code        NumericString
                       (SIZE (ub-country-name-numeric-length)),
    iso-3166-alpha2-code PrintableString
                       (SIZE (ub-country-name-alpha-length)) }

AdministrationDomainName ::= [APPLICATION 2] CHOICE {
    numeric       NumericString (SIZE (0..ub-domain-name-length)),
    printable    PrintableString (SIZE (0..ub-domain-name-length)) }

NetworkAddress ::= X121Address   -- see also extended-network-address

X121Address ::= NumericString (SIZE (1..ub-x121-address-length))

TerminalIdentifier ::= PrintableString (SIZE (1..ub-terminal-id-length))

PrivateDomainName ::= CHOICE {
    numeric       NumericString (SIZE (1..ub-domain-name-length)),
    printable    PrintableString (SIZE (1..ub-domain-name-length)) }

OrganizationName ::= PrintableString
                    (SIZE (1..ub-organization-name-length))
    -- see also teletex-organization-name

NumericUserIdentifier ::= NumericString
                        (SIZE (1..ub-numeric-user-id-length))

PersonalName ::= SET {
    surname     [0] IMPLICIT PrintableString
               (SIZE (1..ub-surname-length)),
    }
given-name  [1]  IMPLICIT  PrintableString
(SIZE (1..ub-given-name-length))  OPTIONAL,
initials    [2]  IMPLICIT  PrintableString
(SIZE (1..ub-initials-length))  OPTIONAL,
(SIZE (1..ub-generation-qualifier-length))
OPTIONAL }
-- see also teletex-personal-name

OrganizationalUnitNames ::= SEQUENCE SIZE (1..ub-organizational-units)
OF OrganizationalUnitName
-- see also teletex-organizational-unit-names

OrganizationalUnitName ::= PrintableString (SIZE
(1..ub-organizational-unit-name-length))

-- Built-in Domain-defined Attributes

BuiltInDomainDefinedAttributes ::= SEQUENCE SIZE
(1..ub-domain-defined-attributes)  OF
BuiltInDomainDefinedAttribute

BuiltInDomainDefinedAttribute ::= SEQUENCE {
type  PrintableString (SIZE
(1..ub-domain-defined-attribute-type-length)),
value  PrintableString (SIZE
(1..ub-domain-defined-attribute-value-length))
}

-- Extension Attributes

ExtensionAttributes ::= SET SIZE (1..ub-extension-attributes)  OF
ExtensionAttribute

EXTENSION-ATTRIBUTE ::= CLASS {
  &id  INTEGER (0..ub-extension-attributes)  UNIQUE,
  &Type
}  WITH SYNTAX { &Type  IDENTIFIED BY &id }

ExtensionAttribute ::=  SEQUENCE {
  extension-attribute-type  [0]  IMPLICIT  EXTENSION-ATTRIBUTE.
  &id({SupportedExtensionAttributes}),
  extension-attribute-value  [1]  EXTENSION-ATTRIBUTE.
  &Type({SupportedExtensionAttributes}
  {@extension-attribute-type}))

SupportedExtensionAttributes  EXTENSION-ATTRIBUTE ::= {  ...  }

-- Extension types and attribute values

Hoffman & Schaad  Expires May 11, 2008  [Page 22]
Internet-Draft          New ASN.1 for PKIX               November 2007

ea-commonName EXTENSION-ATTRIBUTE ::= { PrintableString
   (SIZE (1..ub-common-name-length)) IDENTIFIED BY 1 }

teleutexCommonName EXTENSION-ATTRIBUTE ::= { TeletexString
   (SIZE (1..ub-common-name-length)) IDENTIFIED BY 2 }

teleutexOrganizationName EXTENSION-ATTRIBUTE ::= { TeletexString
   (SIZE (1..ub-organization-name-length)) IDENTIFIED BY 3 }

teleutexPersonalName EXTENSION-ATTRIBUTE ::= {SET {
   surname        [0] IMPLICIT TeletexString
                  (SIZE (1..ub-surname-length)),
   given-name     [1] IMPLICIT TeletexString
                  (SIZE (1..ub-given-name-length)) OPTIONAL,
   initials       [2] IMPLICIT TeletexString
                  (SIZE (1..ub-initials-length)) OPTIONAL,
   generation-qualifier [3] IMPLICIT TeletexString
                  (SIZE (1..ub-generation-qualifier-length))
                  OPTIONAL } IDENTIFIED BY 4 }

teleutexOrganizationalUnitNames EXTENSION-ATTRIBUTE ::= { SEQUENCE SIZE
   (1..ub-organizational-units) OF TeletexOrganizationalUnitName
   IDENTIFIED BY 5 }

TeletexOrganizationalUnitName ::= TeletexString
   (SIZE (1..ub-organizational-unit-name-length))

pDSName EXTENSION-ATTRIBUTE ::= {PrintableString
   (SIZE (1..ub-pds-name-length)) IDENTIFIED BY 7 }

physicalDeliveryCountryName EXTENSION-ATTRIBUTE ::= {CHOICE {
   x121-dcc-code NumericString (SIZE
   (ub-country-name-numeric-length)),
   iso-3166-alpha2-code PrintableString
   (SIZE (ub-country-name-alpha-length)) }
   IDENTIFIED BY 8 }

postalCode EXTENSION-ATTRIBUTE ::= { CHOICE {
   numeric-code NumericString (SIZE (1..ub-postal-code-length)),
   printable-code PrintableString (SIZE (1..ub-postal-code-length))
   IDENTIFIED BY 9 }

physicalDeliveryOfficeName EXTENSION-ATTRIBUTE ::= 
   { PDSParameter IDENTIFIED BY 10 }

physicalDeliveryOfficeNumber EXTENSION-ATTRIBUTE ::= 
   {PDSParameter IDENTIFIED BY 11 }

Hoffman & Schaad          Expires May 11, 2008                 [Page 23]
extensionORAddressComponents EXTENSION-ATTRIBUTE ::= 
{PDSParameter IDENTIFIED BY 12 }

physicalDeliveryPersonalName EXTENSION-ATTRIBUTE ::= 
{PDSParameter IDENTIFIED BY 13}

physicalDeliveryOrganizationName EXTENSION-ATTRIBUTE ::= 
{PDSParameter IDENTIFIED BY 14 }

extensionPhysicalDeliveryAddressComponents EXTENSION-ATTRIBUTE ::= 
{PDSParameter IDENTIFIED BY 15 }

unformattedPostalAddress EXTENSION-ATTRIBUTE ::= 
{ SET { 
    printable-address SEQUENCE SIZE (1..ub-pds-physical-address-lines) 
    OF PrintableString (SIZE (1..ub-pds-parameter-length)) 
    OPTIONAL, 
    teletex-string TeletexString 
    (SIZE (1..ub-unformatted-address-length)) OPTIONAL } 
IDENTIFIED BY 16 }

streetAddress EXTENSION-ATTRIBUTE ::= 
{PDSParameter IDENTIFIED BY 17 }

postOfficeBoxAddress EXTENSION-ATTRIBUTE ::= 
{PDSParameter IDENTIFIED BY 18 }

posteRestanteAddress EXTENSION-ATTRIBUTE ::= 
{PDSParameter IDENTIFIED BY 19 }

uniquePostalName EXTENSION-ATTRIBUTE ::= 
{ PDSParameter IDENTIFIED BY 20 }

localPostalAttributes EXTENSION-ATTRIBUTE ::= 
{PDSParameter IDENTIFIED BY 21 }

PDSParameter ::= SET { 
    printable-string PrintableString 
    (SIZE(1..ub-pds-parameter-length)) OPTIONAL, 
    teletex-string TeletexString 
    (SIZE(1..ub-pds-parameter-length)) OPTIONAL }

extendedNetworkAddress EXTENSION-ATTRIBUTE ::= (CHOICE { 
    e163-4-address SEQUENCE { 
        number [0] IMPLICIT NumericString 
        (SIZE (1..ub-e163-4-number-length)), 
        sub-address [1] IMPLICIT NumericString 
        (SIZE (1..ub-e163-4-sub-address-length)) 
        OPTIONAL }, 

    ... 

)
psap-address [0] IMPLICIT PresentationAddress }
IDENTIFIED BY 22 }

PresentationAddress ::= SEQUENCE {
    pSelector     [0] EXPLICIT OCTET STRING OPTIONAL,
    sSelector     [1] EXPLICIT OCTET STRING OPTIONAL,
    tSelector     [2] EXPLICIT OCTET STRING OPTIONAL,
    nAddresses    [3] EXPLICIT SET SIZE (1..MAX) OF OCTET STRING }

terminalType EXTENSION-ATTRIBUTE ::= {INTEGER {
    telex (3),
    teletex (4),
    g3-facsimile (5),
    g4-facsimile (6),
    ia5-terminal (7),
    videotex (8) } (0..ub-integer-options)
IDENTIFIED BY 23 }

-- Extension Domain-defined Attributes

teletexDomainDefinedAttributes EXTENSION-ATTRIBUTE ::= {SEQUENCE SIZE
(1..ub-domain-defined-attributes) OF TeletexDomainDefinedAttribute
IDENTIFIED BY 6 }

TeletexDomainDefinedAttribute ::= SEQUENCE {
    type TeletexString
    (SIZE (1..ub-domain-defined-attribute-type-length)),
    value TeletexString
    (SIZE (1..ub-domain-defined-attribute-value-length)) }
ub-domain-defined-attributes INTEGER ::= 4
ub-domain-defined-attribute-type-length INTEGER ::= 8
ub-domain-defined-attribute-value-length INTEGER ::= 128
ub-domain-name-length INTEGER ::= 16
ub-extension-attributes INTEGER ::= 256
ub-e163-4-number-length INTEGER ::= 15
ub-e163-4-sub-address-length INTEGER ::= 40
ub-generation-qualifier-length INTEGER ::= 3
ub-given-name-length INTEGER ::= 16
ub-initials-length INTEGER ::= 5
ub-integer-options INTEGER ::= 256
ub-numeric-user-id-length INTEGER ::= 32
ub-organization-name-length INTEGER ::= 64
ub-organizational-unit-name-length INTEGER ::= 32
ub-organizational-units INTEGER ::= 4
ub-pds-name-length INTEGER ::= 16
ub-pds-parameter-length INTEGER ::= 30
ub-pds-physical-address-lines INTEGER ::= 6
ub-postal-code-length INTEGER ::= 16
ub-pseudonym INTEGER ::= 128
ub-surname-length INTEGER ::= 40
ub-terminal-id-length INTEGER ::= 24
ub-unformatted-address-length INTEGER ::= 180
ub-x121-address-length INTEGER ::= 16

-- Note — upper bounds on string types, such as TeletexString, are
-- measured in characters. Excepting PrintableString or IA5String, a
-- significantly greater number of octets will be required to hold
-- such a value. As a minimum, 16 octets, or twice the specified
-- upper bound, whichever is the larger, should be allowed for
-- TeletexString. For UTF8String or UniversalString at least four
-- times the upper bound should be allowed.

-- information object classes used in the definition
-- of certificates and CRLs
-- Parameterized Type SIGNED

SIGNED(ToBeSigned) ::= SEQUENCE {
   toBeSigned   ToBeSigned,
   algorithm    AlgorithmIdentifier,
   signature    BIT STRING
}

END

6. ASN.1 Module for RFC 3280 (Implicit)
PKIX1Implicit88
{ iso(1) identified-organization(3) dod(6) internet(1) security(5)
  mechanisms(5) pkix(7) id-mod(0) id-pkix1-implicit(19) }
DEFINITIONS IMPLICIT TAGS ::= BEGIN
IMPORTS
id-pe, id-kp, id-qt-unotice, id-qt-cps, ORAddress, Name,
  RelativeDistinguishedName, CertificateSerialNumber, Attribute,
  DirectoryString, EXTENSION
FROM PKIX1Explicit88
{ iso(1) identified-organization(3) dod(6) internet(1) security(5)
  mechanisms(5) pkix(7) id-mod(0) id-pkix1-explicit(18) };

-- ISO arc for standard certificate and CRL extensions
id-ce OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) ds(5) 29 }

-- authority key identifier OID and syntax
xt-AuthorityKeyIdentifier EXTENSION ::= { SYNTAX
  AuthorityKeyIdentifier IDENTIFIED BY
  id-ce-authorityKeyIdentifier }

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

AuthorityKeyIdentifier ::= SEQUENCE {
  keyIdentifier             [0] KeyIdentifier            OPTIONAL,
  authorityCertIssuer       [1] GeneralNames             OPTIONAL,
  authorityCertSerialNumber [2] CertificateSerialNumber OPTIONAL }
  -- authorityCertIssuer and authorityCertSerialNumber MUST both
  -- be present or both be absent

KeyIdentifier ::= OCTET STRING

-- subject key identifier OID and syntax
id-ce-subjectKeyIdentifier OBJECT IDENTIFIER ::= { id-ce 14 }

ext-SubjectKeyIdentifier EXTENSION ::= { SYNTAX
  KeyIdentifier IDENTIFIED BY id-ce-subjectKeyIdentifier }

-- key usage extension OID and syntax
ext-KeyUsage EXTENSION ::= { SYNTAX
  KeyUsage IDENTIFIED BY id-ce-keyUsage }
id-ce-keyUsage OBJECT IDENTIFIER ::= { id-ce 15 }

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

-- private key usage period extension OID and syntax

ext-PrivateKeyUsagePeriod EXTENSION ::= { SYNTAX
  PrivateKeyUsagePeriod IDENTIFIED BY id-ce-privateKeyUsagePeriod }

id-ce-privateKeyUsagePeriod OBJECT IDENTIFIER ::= { id-ce 16 }

PrivateKeyUsagePeriod ::= SEQUENCE {
  notBefore       [0]     GeneralizedTime OPTIONAL,
  notAfter        [1]     GeneralizedTime OPTIONAL
  -- either notBefore or notAfter MUST be present

-- certificate policies extension OID and syntax

ext-CertificatePolicies EXTENSION ::= { SYNTAX
  CertificatePolicies IDENTIFIED BY id-ce-certificatePolicies }

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

POLICY-QUALIFIER-INFO ::= TYPE-IDENTIFIER

PolicyQualifierInfo ::= SEQUENCE {
  policyQualifierId  POLICY-QUALIFIER-INFO.
  &id({PolicyQualifierId})},
qualifier POLICY-QUALIFIER-INFO.
   &Type({PolicyQualifierId}@policyQualifierId})

-- Implementations that recognize additional policy qualifiers MUST
-- augment the following definition for PolicyQualifierId

PolicyQualifierId POLICY-QUALIFIER-INFO ::= 
   { pqid-cps | pqid-unotice }

pqid-cps POLICY-QUALIFIER-INFO ::= { CPSuri IDENTIFIED BY id-qt-cps }

pqid-unotice POLICY-QUALIFIER-INFO ::= { UserNotice
   IDENTIFIED BY id-qt-unotice }

-- CPS pointer qualifier

CPSuri ::= IA5String

-- user notice qualifier

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)) }

-- policy mapping extension OID and syntax

ext-PolicyMappings EXTENSION ::= { SYNTAX
   PolicyMappings IDENTIFIED BY id-ce-policyMappings }

id-ce-policyMappings OBJECT IDENTIFIER ::= { id-ce 33 }

PolicyMappings ::= SEQUENCE SIZE (1..MAX) OF SEQUENCE {
   issuerDomainPolicy CertPolicyId,
   subjectDomainPolicy CertPolicyId }

-- subject alternative name extension OID and syntax

ext-SubjectAltName EXTENSION ::= { SYNTAX
   ...
GeneralNames IDENTIFIED BY id-ce-subjectAltName

id-ce-subjectAltName OBJECT IDENTIFIER ::= { id-ce 17 }

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

GeneralName ::= CHOICE {
  otherName                       [0]     INSTANCE OF OTHER-NAME,
  rfc822Name                      [1]     IA5String,
  dNSName                         [2]     IA5String,
  x400Address                     [3]     ORAddress,
  directoryName                   [4]     Name,
  ediPartyName                    [5]     EDIPartyName,
  uniformResourceIdentifier       [6]     IA5String,
  iPAddress                       [7]     OCTET STRING,
  registeredID                    [8]     OBJECT IDENTIFIER }

-- AnotherName replaces OTHER-NAME ::= TYPE-IDENTIFIER, as
-- TYPE-IDENTIFIER is not supported in the ‘88 ASN.1 syntax

OTHER-NAME ::= TYPE-IDENTIFIER

EDIPartyName ::= SEQUENCE {
  nameAssigner    [0]     DirectoryString {ubMax} OPTIONAL,
  partyName       [1]     DirectoryString {ubMax} }

-- issuer alternative name extension OID and syntax

ext-IssuerAltName EXTENSION ::= { SYNTAX
  GeneralNames IDENTIFIED BY id-ce-issuerAltName }

id-ce-issuerAltName OBJECT IDENTIFIER ::= { id-ce 18 }

ext-SubjectDirectoryAttributes EXTENSION ::= { SYNTAX
  SubjectDirectoryAttributes IDENTIFIED BY
  id-ce-subjectDirectoryAttributes }

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

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

-- basic constraints extension OID and syntax

ext-BasicConstraints EXTENSION ::= { SYNTAX
  BasicConstraints IDENTIFIED BY id-ce-basicConstraints }

id-ce-basicConstraints OBJECT IDENTIFIER ::= { id-ce 19 }
BasicConstraints ::= SEQUENCE {
cA                      BOOLEAN DEFAULT FALSE,
pathLenConstraint       INTEGER (0..MAX) OPTIONAL }

-- name constraints extension OID and syntax

ext-NameConstraints EXTENSION ::= { SYNTAX
  NameConstraints IDENTIFIED BY id-ce-nameConstraints }

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)

-- policy constraints extension OID and syntax

ext-PolicyConstraints EXTENSION ::= { SYNTAX
  PolicyConstraints IDENTIFIED BY id-ce-policyConstraints }

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

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

SkipCerts ::= INTEGER (0..MAX)

-- CRL distribution points extension OID and syntax

ext-CRLDistributionPoints EXTENSION ::= { SYNTAX
  CRLDistributionPoints IDENTIFIED BY id-ce-cRLDistributionPoints }

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) }

-- extended key usage extension OID and syntax

ext-ExtKeyUsageSyntax EXTENSION ::= { SYNTAX
  ExtKeyUsageSyntax IDENTIFIED BY id-ce-extKeyUsage }

id-ce-extKeyUsage OBJECT IDENTIFIER ::= {id-ce 37}

ExtKeyUsageSyntax ::= SEQUENCE SIZE (1..MAX) OF KeyPurposeId

KeyPurposeId ::= OBJECT IDENTIFIER

-- permit unspecified key uses

anyExtendedKeyUsage OBJECT IDENTIFIER ::= { id-ce-extKeyUsage 0 }

-- extended key purpose OIDs

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 }

-- inhibit any policy OID and syntax

ext-InhibitAnyPolicy EXTENSION ::= {SYNTAX
  SkipCerts IDENTIFIED BY id-ce-inhibitAnyPolicy }

id-ce-inhibitAnyPolicy OBJECT IDENTIFIER ::= { id-ce 54 }
ext-FreshestCRL EXTENSION ::= {SYNTAX
   CRLDistributionPoints IDENTIFIED BY id-ce-freshestCRL }

id-ce-freshestCRL OBJECT IDENTIFIER ::= { id-ce 46 }

ext-AuthorityInfoAccessSyntax EXTENSION ::= { SYNTAX
   AuthorityInfoAccessSyntax IDENTIFIED BY
   id-pe-authorityInfoAccess }

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

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

AccessDescription ::= SEQUENCE {
   accessMethod OBJECT IDENTIFIER,
   accessLocation GeneralName }

ext-SubjectInfoAccessSyntax EXTENSION ::= { SYNTAX
   SubjectInfoAccessSyntax IDENTIFIED BY id-pe-subjectInfoAccess }

id-pe-subjectInfoAccess OBJECT IDENTIFIER ::= { id-pe 11 }

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

ext-CRLNumber EXTENSION ::= {SYNTAX
   INTEGER (0..MAX) IDENTIFIED BY id-ce-cRLNumber }

id-ce-cRLNumber OBJECT IDENTIFIER ::= { id-ce 20 }

CRLNumber ::= INTEGER (0..MAX)

ext-IssuingDistributionPoint EXTENSION ::= { SYNTAX
   IssuingDistributionPoint IDENTIFIED BY
   id-ce-issuingDistributionPoint }

id-ce-issuingDistributionPoint OBJECT IDENTIFIER ::= { id-ce 28 }
IssuingDistributionPoint ::= SEQUENCE {
    distributionPoint          [0] DistributionPointName OPTIONAL,
    onlyContainsUserCerts      [1] BOOLEAN DEFAULT FALSE,
    onlyContainsCACerts        [2] BOOLEAN DEFAULT FALSE,
    onlySomeReasons            [3] ReasonFlags OPTIONAL,
    indirectCRL                [4] BOOLEAN DEFAULT FALSE,
    onlyContainsAttributeCerts [5] BOOLEAN DEFAULT FALSE }

ext-BaseCRLNumber EXTENSION ::= { SYNTAX
    CRLNumber IDENTIFIED BY id-ce-deltaCRLIndicator }

id-ce-deltaCRLIndicator OBJECT IDENTIFIER ::= { id-ce 27 }

-- CRL reasons extension OID and syntax

ext-CRLReason EXTENSION ::= { SYNTAX
    CRLReason IDENTIFIED BY id-ce-cRLReasons }

id-ce-cRLReasons OBJECT IDENTIFIER ::= { id-ce 21 }

CRLReason ::= ENUMERATED {
    unspecified             (0),
    keyCompromise           (1),
    cACompromise            (2),
    affiliationChanged      (3),
    superseded              (4),
    cessationOfOperation    (5),
    certificateHold         (6),
    removeFromCRL           (8),
    privilegeWithdrawn      (9),
    aACompromise           (10) }

-- certificate issuer CRL entry extension OID and syntax

ext-CertificateIssuer EXTENSION ::= { SYNTAX
    GeneralNames IDENTIFIED BY id-ce-certificateIssuer }

id-ce-certificateIssuer OBJECT IDENTIFIER ::= { id-ce 29 }

-- hold instruction extension OID and syntax

ext-HoldInstructionCode EXTENSION ::= { SYNTAX
    OBJECT IDENTIFIER IDENTIFIED BY id-ce-holdInstructionCode }

id-ce-holdInstructionCode OBJECT IDENTIFIER ::= { id-ce 23 }

-- ANSI x9 holdinstructions
-- ANSI x9 arc holdInstruction arc

holdInstruction OBJECT IDENTIFIER ::=  
  {joint-iso-itu-t(2) member-body(2) us(840) x9cm(10040) 2}

-- ANSI X9 holdinstructions referenced by this standard

id-holdinstruction-none OBJECT IDENTIFIER ::=  
  {holdInstruction 1} -- deprecated

id-holdinstruction-callissuer OBJECT IDENTIFIER ::=  
  {holdInstruction 2}

id-holdinstruction-reject OBJECT IDENTIFIER ::=  
  {holdInstruction 3}

-- invalidity date CRL entry extension OID and syntax

ext-InvalidityDate EXTENSION ::=  
  { SYNTAX  
    GeneralizedTime IDENTIFIED BY id-ce-invalidityDate } 

id-ce-invalidityDate OBJECT IDENTIFIER ::= { id-ce 24 }

ubMax INTEGER ::= 32768

END

7. ASN.1 Module for RFC 3281

PKIXAttributeCertificate
  {iso(1) identified-organization(3) dod(6) internet(1) security(5)  
  mechanisms(5) pkix(7) id-mod(0) id-mod-attribute-cert(12)}
DEFINITIONS IMPLICIT TAGS ::= 
BEGIN

IMPORTS

Attribute, AlgorithmIdentifier, CertificateSerialNumber, Extensions,  
UniqueIdentifier, id-pkix, id-pe, id-kp, id-ad, id-at, SIGNED,  
EXTENSION, ATTRIBUTE
FROM PKIX1Explicit88
  {iso(1) identified-organization(3) dod(6) internet(1) security(5)  
  mechanisms(5) pkix(7) id-mod(0) id-pkix1-explicit(18)}

GeneralName, GeneralNames, id-ce, AuthorityKeyIdentifier,  
AuthorityInfoAccessSyntax, CRLDistributionPoints
FROM PKIX1Implicit88

Hoffman & Schaad          Expires May 11, 2008                 [Page 35]
ExtensionsDefined EXTENSION ::= { auditIdentity | targetInformation |
    ce-authorityKeyIdentifier | ce-authorityInfoAccess |
    ce-cRLDistributionPoints | ce-noRevAvail | pe-ac-proxying |
    pe-aaControls }

auditIdentity EXTENSION ::= { SYNTAX
    OCTET STRING IDENTIFIED BY id-pe-ac-auditIdentity}

targetInformation EXTENSION ::= { SYNTAX
    Targets IDENTIFIED BY id-ce-targetInformation }

ce-authorityKeyIdentifier EXTENSION ::= { SYNTAX
    AuthorityKeyIdentifier IDENTIFIED BY id-ce-authorityKeyIdentifier }

ce-authorityInfoAccess EXTENSION ::= { SYNTAX
    AuthorityInfoAccessSyntax
    IDENTIFIED BY id-ce-authorityInfoAccess}

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

ce-noRevAvail EXTENSION ::= { SYNTAX
    NULL IDENTIFIED BY id-ce-noRevAvail}

pe-ac-proxying EXTENSION ::= { SYNTAX
    ProxyInfo IDENTIFIED BY id-pe-ac-proxying}

pe-aaControls EXTENSION ::= { SYNTAX
    AAControls IDENTIFIED BY id-pe-aaControls}

-- Another way to do the following might be:
-- AttributesDefined ATTRIBUTE ::= { ... , aca-authenticationInfo |
--     aca-accesIdentity | aca-chargingIdentity | aca-group |
--     at-role | at-clearance | aca-encAttrs }

aca-authenticationInfo ATTRIBUTE ::= { SvceAuthInfo
    IDENTIFIED BY id-aca-authenticationInfo}

aca-accesIdentity ATTRIBUTE ::= { SvceAuthInfo
    IDENTIFIED BY id-aca-accesIdentity}

aca-chargingIdentity ATTRIBUTE ::= { IetfAttrSyntax
    IDENTIFIED BY id-aca-chargingIdentity}

aca-group ATTRIBUTE ::= { IetfAttrSyntax
    IDENTIFIED BY id-aca-group}

at-role ATTRIBUTE ::= { RoleSyntax
    IDENTIFIED BY id-at-role}

at-clearance ATTRIBUTE ::= { Clearance
    IDENTIFIED BY id-at-clearance}

aca-encAttrs ATTRIBUTE ::= { ContentInfo
    IDENTIFIED BY id-aca-encAttrs}
id-ce-authorityInfoAccess  OBJECT IDENTIFIER ::= { id-pe 1 }

id-pe-ac-auditIdentity   OBJECT IDENTIFIER ::= { id-pe 4 }

id-pe-aaControls         OBJECT IDENTIFIER ::= { id-pe 6 }

id-pe-ac-proxying        OBJECT IDENTIFIER ::= { id-pe 10 }

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

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

id-ce-targetInformation  OBJECT IDENTIFIER ::= { id-ce 55 }

id-ce-noRevAvail          OBJECT IDENTIFIER ::= { id-ce 56 }

id-aca                  OBJECT IDENTIFIER ::= { id-pkix 10 }

id-aca-authenticationInfo OBJECT IDENTIFIER ::= { id-aca 1 }

id-aca-accessIdentity    OBJECT IDENTIFIER ::= { id-aca 2 }

id-aca-chargingIdentity  OBJECT IDENTIFIER ::= { id-aca 3 }

id-aca-group            OBJECT IDENTIFIER ::= { id-aca 4 }

id-aca-encAttrs          OBJECT IDENTIFIER ::= { id-aca 6 }

id-at-role              OBJECT IDENTIFIER ::= { id-at 72 }

id-at-clearance         OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) ds(5) module(1) selected-attribute-types(5) clearance (55) }

AttributeCertificate ::= SIGNED{AttributeCertificateInfo}

AttributeCertificateInfo ::= SEQUENCE {
  version        AttCertVersion,  -- version is v2,
  holder         Holder,
  issuer         AttCertIssuer,
  signature      AlgorithmIdentifier,
  serialNumber   CertificateSerialNumber,
  attrCertValidityPeriod   AttCertValidityPeriod,
  attributes     SEQUENCE OF Attribute,
  issuerUniqueID UniqueIdentifier OPTIONAL,
  extensions     Extensions     OPTIONAL
}

AttCertVersion ::= INTEGER { v2(1) }

Holder ::= SEQUENCE {
  baseCertificateID  [0] IssuerSerial OPTIONAL,
    -- the issuer and serial number of
    -- the holder’s Public Key Certificate
  entityName        [1] GeneralNames OPTIONAL,
    -- the name of the claimant or role
  objectDigestInfo  [2] ObjectDigestInfo OPTIONAL
    -- used to directly authenticate the
    -- holder, for example, an executable
ObjectDigestInfo ::= SEQUENCE {
    digestedObjectType ENUMERATED {
        publicKey            (0),
        publicKeyCert        (1),
        otherObjectTypes     (2) },
    otherObjectTypeID   OBJECT IDENTIFIER  OPTIONAL,
    digestAlgorithm     AlgorithmIdentifier,
    objectDigest        BIT STRING
}

AttCertIssuer ::= CHOICE {
    v1Form   GeneralNames,  -- MUST NOT be used in this
        -- profile
    v2Form   [0] V2Form     -- v2 only
}

V2Form ::= SEQUENCE {
    issuerName            GeneralNames  OPTIONAL,
    baseCertificateID     [0] IssuerSerial  OPTIONAL,
    objectDigestInfo      [1] ObjectDigestInfo  OPTIONAL
        -- issuerName MUST be present in this profile
        -- baseCertificateID and objectDigestInfo MUST
        -- NOT be present in this profile
}

IssuerSerial ::= SEQUENCE {
    issuer         GeneralNames,
    serial         CertificateSerialNumber,
    issuerUID      UniqueIdentifier OPTIONAL
}

AttCertValidityPeriod ::= SEQUENCE {
    notBeforeTime  GeneralizedTime,
    notAfterTime   GeneralizedTime
}

Targets ::= SEQUENCE OF Target

Target ::= CHOICE {
    targetName     [0] GeneralName,
    targetGroup    [1] GeneralName,
    targetCert     [2] TargetCert
}
TargetCert ::= SEQUENCE {
  targetCertificate IssuerSerial,
  targetName GeneralName OPTIONAL,
  certDigestInfo ObjectDigestInfo OPTIONAL
}

IetfAttrSyntax ::= SEQUENCE {
  policyAuthority[0] GeneralNames OPTIONAL,
  values SEQUENCE OF CHOICE {
    octets OCTET STRING,
    oid OBJECT IDENTIFIER,
    string UTF8String
  }
}

SvceAuthInfo ::= SEQUENCE {
  service GeneralName,
  ident GeneralName,
  authInfo OCTET STRING OPTIONAL
}

RoleSyntax ::= SEQUENCE {
  roleAuthority [0] GeneralNames OPTIONAL,
  roleName [1] GeneralName
}

Clearance ::= SEQUENCE {
  policyId [0] OBJECT IDENTIFIER,
  classList [1] ClassList DEFAULT {unclassified},
}

ClassList ::= BIT STRING {
  unmarked (0),
  unclassified (1),
  restricted (2),
  confidential (3),
  secret (4),
  topSecret (5)
}

SECURITY-CATEGORY ::= TYPE-IDENTIFIER

SecurityCategory ::= SEQUENCE {
  type [0] IMPLICIT TYPE-IDENTIFIER.
    &id({SupportedSecurityCategories}),
  value [1] TYPE-IDENTIFIER.
&Type({SupportedSecurityCategories}{@type})

SupportedSecurityCategories  SECURITY-CATEGORY ::= { ... }

AAControls ::= SEQUENCE {
    pathLenConstraint INTEGER (0..MAX) OPTIONAL,
    permittedAttrs    [0] AttrSpec OPTIONAL,
    excludedAttrs     [1] AttrSpec OPTIONAL,
    permitUnSpecified BOOLEAN DEFAULT TRUE
}

AttrSpec ::= SEQUENCE OF OBJECT IDENTIFIER

ACClearAttrs ::= SEQUENCE {
    acIssuer          GeneralName,
    acSerial          INTEGER,
    attrs             SEQUENCE OF Attribute
}

ProxyInfo ::= SEQUENCE OF Targets

CRLDistPointsSyntax ::= CRLDistributionPoints

ContentInfo ::= INTEGER

END

8. ASN.1 Module for RFC 3852 (Attribute Certificate v1)

AttributeCertificateVersion1
    { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
      smime(16) modules(0) v1AttrCert(15) }
DEFINITIONS EXPLICIT TAGS ::= BEGIN

IMPORTS

AlgorithmIdentifier, Attribute, CertificateSerialNumber, Extensions,
UniqueIdentifier
FROM PKIX1Explicit88
    { iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-pkix1-explicit(18) }

GeneralNames
FROM PKIX1Implicit88
    { iso(1) identified-organization(3) dod(6) internet(1) security(5) }
mechanisms(5) pkix(7) id-mod(0) id-pkix1-implicit(19) }

AttCertValidityPeriod, IssuerSerial
FROM PKIXAttributeCertificate
{ iso(1) identified-organization(3) dod(6) internet(1) security(5)
  mechanisms(5) pkix(7) id-mod(0) id-mod-attribute-cert(12) } ;

-- Definition extracted from X.509-1997 [X.509-97], but
-- different type names are used to avoid collisions.

AttributeCertificateV1 ::= SEQUENCE {
  acInfo AttributeCertificateInfoV1,
  signatureAlgorithm AlgorithmIdentifier,
  signature BIT STRING }

AttributeCertificateInfoV1 ::= SEQUENCE {
  version AttCertVersionV1 DEFAULT v1,
  subject CHOICE {
    baseCertificateID [0] IssuerSerial,
    -- associated with a Public Key Certificate
    subjectName [1] GeneralNames },
    -- associated with a name
    issuer GeneralNames,
    signature AlgorithmIdentifier,
    serialNumber CertificateSerialNumber,
    attCertValidityPeriod AttCertValidityPeriod,
    attributes SEQUENCE OF Attribute,
    issuerUniqueID UniqueIdentifier OPTIONAL,
    extensions Extensions OPTIONAL }

AttCertVersionV1 ::= INTEGER { v1(0) }

END

9. ASN.1 Module for RFC 4210

PKIXCMP
{ iso(1) identified-organization(3) dod(6) internet(1) security(5)
  mechanisms(5) pkix(7) id-mod(0) id-mod-cmp2000(16) }
DEFINITIONS EXPLICIT TAGS :=
BEGIN

IMPORTS

Certificate, CertificateList, Extensions, AlgorithmIdentifier
FROM PKIX1Explicit88
GeneralName, KeyIdentifier
FROM PKIX1Implicit88
  { iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-pkix1-implicit(19) }

CertTemplate, PKIPublicationInfo, EncryptedValue, CertId,
  CertReqMessages
FROM PKIXCRMF-2005
  { iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-crmf2005(36) }
-- see also the behavioral clarifications to CRMF codified in
-- Appendix C of this specification

CertificationRequest
FROM PKCS-10
  { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-10(10)
    modules(1) pkcs-10(1) }
-- (specified in RFC 2986 with 1993 ASN.1 syntax and IMPLICIT
-- tags). Alternatively, implementers may directly include
-- the [PKCS10] syntax in this module

-- the rest of the module contains locally-defined OIDs and
-- constructs

CMPCertificate ::= CHOICE { x509v3PKCert Certificate, ... }
-- This syntax, while bits-on-the-wire compatible with the
-- standard X.509 definition of "Certificate", allows the
-- possibility of future certificate types (such as X.509
-- attribute certificates, WAP WTLS certificates, or other kinds
-- of certificates) within this certificate management protocol,
-- should a need ever arise to support such generality. Those
-- implementations that do not foresee a need to ever support
-- other certificate types MAY, if they wish, comment out the
-- above structure and "un-comment" the following one prior to
-- compiling this ASN.1 module. (Note that interoperability
-- with implementations that don't do this will be unaffected by
-- this change.)

-- CMPCertificate ::= Certificate

PKIMessage ::= SEQUENCE {
  header          PKIHeader,
  body            PKIBody,
  protection      [0] PKIProtection OPTIONAL,
extraCerts  [1] SEQUENCE SIZE (1..MAX) OF CMPCertificate
            OPTIONAL )

PKIMessages ::= SEQUENCE SIZE (1..MAX) OF PKIMessage

PKIHeader ::= SEQUENCE {
    pvno                INTEGER     { cmp1999(1), cmp2000(2) },
    sender              GeneralName,
        -- identifies the sender
    recipient           GeneralName,
        -- identifies the intended recipient
    messageTime     [0] GeneralizedTime         OPTIONAL,
        -- time of production of this message (used when sender
        -- believes that the transport will be "suitable"; i.e.,
        -- that the time will still be meaningful upon receipt)
    protectionAlg   [1] AlgorithmIdentifier     OPTIONAL,
        -- algorithm used for calculation of protection bits
    senderKID       [2] KeyIdentifier           OPTIONAL,
    recipKID        [3] KeyIdentifier           OPTIONAL,
        -- to identify specific keys used for protection
    transactionID   [4] OCTET STRING            OPTIONAL,
        -- identifies the transaction; i.e., this will be the same in
        -- corresponding request, response, certConf, and PKIConf
        -- messages
    senderNonce     [5] OCTET STRING            OPTIONAL,
    recipNonce      [6] OCTET STRING            OPTIONAL,
        -- nonces used to provide replay protection, senderNonce
        -- is inserted by the creator of this message; recipNonce
        -- is a nonce previously inserted in a related message by
        -- the intended recipient of this message
    freeText        [7] PKIFreeText             OPTIONAL,
        -- this may be used to indicate context-specific instructions
        -- (this field is intended for human consumption)
    generalInfo     [8] SEQUENCE SIZE (1..MAX) OF
        InfoTypeAndValue     OPTIONAL
        -- this may be used to convey context-specific information
        -- (this field not primarily intended for human consumption)
}

PKIFreeText ::= SEQUENCE SIZE (1..MAX) OF UTF8String

PKIBody ::= CHOICE {
    -- message-specific body elements
    ir       [0] CertReqMessages, --Initialization Request
PKIProtection ::= BIT STRING

ProtectedPart ::= SEQUENCE {
    header    PKIHeader,
    body      PKIBody }

id-PasswordBasedMac OBJECT IDENTIFIER ::= {1 2 840 113533 7 66 13}

PBMPParameter ::= SEQUENCE {
    salt          OCTET STRING,
-- note: implementations MAY wish to limit acceptable sizes
-- of this string to values appropriate for their environment
-- in order to reduce the risk of denial-of-service attacks
    owf             AlgorithmIdentifier,
-- AlgId for a One-Way Function (SHA-1 recommended)
    iterationCount INTEGER,
-- number of times the OWF is applied
-- note: implementations MAY wish to limit acceptable sizes
-- of this integer to values appropriate for their environment
-- in order to reduce the risk of denial-of-service attacks
    mac             AlgorithmIdentifier
-- the MAC AlgId (e.g., DES-MAC, Triple-DES-MAC [PKCS11]),
id-DHBasedMac OBJECT IDENTIFIER ::= {1 2 840 113533 7 66 30}
DHBMParameter ::= SEQUENCE {
  owf                 AlgorithmIdentifier,
  -- AlgId for a One-Way Function (SHA-1 recommended)
  mac                 AlgorithmIdentifier
  -- the MAC AlgId (e.g., DES-MAC, Triple-DES-MAC [PKCS11],
  -- or HMAC [RFC2104, RFC2202])
}

NestedMessageContent ::= PKIMessages

PKIStatus ::= INTEGER {
  accepted                (0),
  -- you got exactly what you asked for
  grantedWithMods        (1),
  -- you got something like what you asked for; the
  -- requester is responsible for ascertaining the differences
  rejection              (2),
  -- you don’t get it, more information elsewhere in the message
  waiting                (3),
  -- the request body part has not yet been processed; expect to
  -- hear more later (note: proper handling of this status
  -- response MAY use the polling req/rep PKIMessages specified
  -- in Section 5.3.22; alternatively, polling in the underlying
  -- transport layer MAY have some utility in this regard)
  revocationWarning      (4),
  -- this message contains a warning that a revocation is
  -- imminent
  revocationNotification (5),
  -- notification that a revocation has occurred
  keyUpdateWarning       (6)
  -- update already done for the oldCertId specified in
  -- CertReqMsg
}

PKIFailureInfo ::= BIT STRING {
  -- since we can fail in more than one way!
  -- More codes may be added in the future if/when required.
  badAlg              (0),
  -- unrecognized or unsupported Algorithm Identifier
  badMessageCheck     (1),
  -- integrity check failed (e.g., signature did not verify)
  badRequest          (2),
  -- transaction not permitted or supported
  badTime             (3),
}
badCertId (4),
-- no certificate could be found matching the provided criteria
badDataFormat (5),
-- the data submitted has the wrong format
wrongAuthority (6),
-- the authority indicated in the request is different from the
-- one creating the response token
incorrectData (7),
-- the requester's data is incorrect (for notary services)
missingTimeStamp (8),
-- when the timestamp is missing but should be there
-- (by policy)
badPOP (9),
-- the proof-of-possession failed
certRevoked (10),
-- the certificate has already been revoked
certConfirmed (11),
-- the certificate has already been confirmed
wrongIntegrity (12),
-- invalid integrity, password based instead of signature or
-- vice versa
badRecipientNonce (13),
-- invalid recipient nonce, either missing or wrong value
timeNotAvailable (14),
-- the TSA's time source is not available
unacceptedPolicy (15),
-- the requested TSA policy is not supported by the TSA.
unacceptedExtension (16),
-- the requested extension is not supported by the TSA.
addInfoNotAvailable (17),
-- the additional information requested could not be
-- understood or is not available
badSenderNonce (18),
-- invalid sender nonce, either missing or wrong size
badCertTemplate (19),
-- invalid cert. template or missing mandatory information
signerNotTrusted (20),
-- signer of the message unknown or not trusted
transactionIdInUse (21),
-- the transaction identifier is already in use
unsupportedVersion (22),
-- the version of the message is not supported
notAuthorized (23),
-- the sender was not authorized to make the preceding
-- request or perform the preceding action
systemUnavail (24),
-- the request cannot be handled due to system unavailability
systemFailure (25),
-- the request cannot be handled due to system failure
duplicateCertReq (26)
-- certificate cannot be issued because a duplicate
-- certificate already exists
}

PKIStatusInfo ::= SEQUENCE {
  status        PKIStatus,
  statusString  PKIFreeText     OPTIONAL,
  failInfo      PKIFailureInfo  OPTIONAL }

OOBCert ::= CMPCertificate

OOBCertHash ::= SEQUENCE {
  hashAlg     [0] AlgorithmIdentifier     OPTIONAL,
  certId      [1] CertId                  OPTIONAL,
  hashVal         BIT STRING
  -- hashVal is calculated over the DER encoding of the
  -- self-signed certificate with the identifier certID.
}

POPODecKeyChallContent ::= SEQUENCE OF Challenge
-- One Challenge per encryption key certification request (in the
-- same order as these requests appear in CertReqMessages).

Challenge ::= SEQUENCE {
  owf                 AlgorithmIdentifier  OPTIONAL,
  -- MUST be present in the first Challenge; MAY be omitted in
  -- any subsequent Challenge in POPODecKeyChallContent (if
  -- omitted, then the owf used in the immediately preceding
  -- Challenge is to be used).
  witness             OCTET STRING,
  -- the result of applying the one-way function (owf) to a
  -- randomly-generated INTEGER, A. [Note that a different
  -- INTEGER MUST be used for each Challenge.]
  challenge           OCTET STRING
  -- the encryption (under the public key for which the cert.
  -- request is being made) of Rand, where Rand is specified as
  -- Rand ::= SEQUENCE {
  --   int      INTEGER,
  --     -- the randomly-generated INTEGER A (above)
  --   sender   GeneralName
  --     -- the sender’s name (as included in PKIHeader)
  --  }
}
POPODecKeyRespContent ::= SEQUENCE OF INTEGER
-- One INTEGER per encryption key certification request (in the
-- same order as these requests appear in CertReqMessages). The
-- retrieved INTEGER A (above) is returned to the sender of the
-- corresponding Challenge.

CertRepMessage ::= SEQUENCE {
  caPubs [1] SEQUENCE SIZE (1..MAX) OF CMPCertificate
            OPTIONAL,
  response     SEQUENCE OF CertResponse }

CertResponse ::= SEQUENCE {
  certReqId     INTEGER,
  -- to match this response with corresponding request (a value
  -- of -1 is to be used if certReqId is not specified in the
  -- corresponding request)
  status        PKIStatusInfo,
  certifiedKeyPair CertifiedKeyPair OPTIONAL,
  rspInfo       OCTET STRING OPTIONAL
  -- analogous to the id-regInfo-utf8Pairs string defined
  -- for regInfo in CertReqMsg [CRMF]
}

CertifiedKeyPair ::= SEQUENCE {
  certOrEncCert CertOrEncCert,
  privateKey    [0] EncryptedValue OPTIONAL,
  -- see [CRMF] for comment on encoding
  publicationInfo [1] PKIPublicationInfo OPTIONAL }

CertOrEncCert ::= CHOICE {
  certificate    [0] CMPCertificate,
  encryptedCert  [1] EncryptedValue }

KeyRecRepContent ::= SEQUENCE {
  status        PKIStatusInfo,
  newSigCert    [0] CMPCertificate OPTIONAL,
  caCerts       [1] SEQUENCE SIZE (1..MAX) OF
                 CMPCertificate OPTIONAL,
  keyPairHist   [2] SEQUENCE SIZE (1..MAX) OF
                 CertifiedKeyPair OPTIONAL }

RevReqContent ::= SEQUENCE OF RevDetails

RevDetails ::= SEQUENCE {
  certDetails   CertTemplate,
  -- allows requester to specify as much as they can about
  -- the cert. for which revocation is requested
  -- (e.g., for cases in which serialNumber is not available)
crlEntryDetails Extensions OPTIONAL
   -- requested crlEntryExtensions
}

RevRepContent ::= SEQUENCE {
   status       SEQUENCE SIZE (1..MAX) OF PKIStatusInfo,
   -- in same order as was sent in RevReqContent
   revCerts [0] SEQUENCE SIZE (1..MAX) OF CertId
     OPTIONAL,
   -- IDs for which revocation was requested
   -- (same order as status)
   crls     [1] SEQUENCE SIZE (1..MAX) OF CertificateList
     OPTIONAL
   -- the resulting CRLs (there may be more than one)
}

CAKeyUpdAnnContent ::= SEQUENCE {
   oldWithNew   CMPCertificate, -- old pub signed with new priv
   newWithOld   CMPCertificate, -- new pub signed with old priv
   newWithNew   CMPCertificate -- new pub signed with new priv
}

CertAnnContent ::= CMPCertificate

RevAnnContent ::= SEQUENCE {
   status              PKIStatus,
   certId              CertId,
   willBeRevokedAt     GeneralizedTime,
   badSinceDate        GeneralizedTime,
   crlDetails          Extensions OPTIONAL
   -- extra CRL details (e.g., crl number, reason, location, etc.)
}

CRLAnnContent ::= SEQUENCE OF CertificateList

CertConfirmContent ::= SEQUENCE OF CertStatus

CertStatus ::= SEQUENCE {
   certHash    OCTET STRING,
   -- the hash of the certificate, using the same hash algorithm
   -- as is used to create and verify the certificate signature
   certReqId   INTEGER,
   -- to match this confirmation with the corresponding req/rep
   statusInfo  PKIStatusInfo OPTIONAL
}

PKIConfirmContent ::= NULL

INFO-TYPE-AND-VALUE ::= TYPE-IDENTIFIER
InfoTypeAndValue ::= SEQUENCE {
  infoType   INFO-TYPE-AND-VALUE.
    &id({SupportedInfoSet}),
  infoValue   INFO-TYPE-AND-VALUE.
    &Type({SupportedInfoSet}@infoType) }

SupportedInfoSet INFO-TYPE-AND-VALUE ::= { ... }

-- Example InfoTypeAndValue contents include, but are not limited
-- to, the following (un-comment in this ASN.1 module and use as
-- appropriate for a given environment):
--
-- -- id-it-caProtEncCert OBJECT IDENTIFIER ::= {id-it 1}
--   CAProtEncCertValue ::= CMPCertificate
-- -- id-it-signKeyPairTypes OBJECT IDENTIFIER ::= {id-it 2}
--   SignKeyPairTypesValue ::= SEQUENCE OF AlgorithmIdentifier
-- -- id-it-encKeyPairTypes OBJECT IDENTIFIER ::= {id-it 3}
--   EncKeyPairTypesValue ::= SEQUENCE OF AlgorithmIdentifier
-- -- id-it-preferredSymmAlg OBJECT IDENTIFIER ::= {id-it 4}
--   PreferredSymmAlgValue ::= AlgorithmIdentifier
-- -- id-it-caKeyUpdateInfo OBJECT IDENTIFIER ::= {id-it 5}
--   CAKeyUpdateInfoValue ::= CAKeyUpdAnnContent
-- -- id-it-currentCRL OBJECT IDENTIFIER ::= {id-it 6}
--   CurrentCRLValue ::= CertificateList
-- -- id-it-unsupportedOIDs OBJECT IDENTIFIER ::= {id-it 7}
--   UnsupportedOIDsValue ::= SEQUENCE OF OBJECT IDENTIFIER
-- -- id-it-keyPairParamReq OBJECT IDENTIFIER ::= {id-it 10}
--   KeyPairParamReqValue ::= OBJECT IDENTIFIER
-- -- id-it-keyPairParamRep OBJECT IDENTIFIER ::= {id-it 11}
--   KeyPairParamRepValue ::= AlgorithmIdentifier
-- -- id-it-revPassphrase OBJECT IDENTIFIER ::= {id-it 12}
--   RevPassphraseValue ::= EncryptedValue
-- -- id-it-implicitConfirm OBJECT IDENTIFIER ::= {id-it 13}
--   ImplicitConfirmValue ::= NULL
-- -- id-it-confirmWaitTime OBJECT IDENTIFIER ::= {id-it 14}
--   ConfirmWaitTimeValue ::= GeneralizedTime
-- -- id-it-origPKIMessage OBJECT IDENTIFIER ::= {id-it 15}
--   OrigPKIMessageValue ::= PKIMessages
-- -- id-it-suppLangTags OBJECT IDENTIFIER ::= {id-it 16}
--   SuppLangTagsValue ::= SEQUENCE OF UTF8String
--
-- where
--
-- -- id-pkix OBJECT IDENTIFIER ::= {
--   iso(1) identified-organization(3)
--     dod(6) internet(1) security(5) mechanisms(5) pkix(7)}
-- -- and
-- -- id-it OBJECT IDENTIFIER ::= {id-pkix 4}
--
-- This construct MAY also be used to define new PKIX Certificate
-- Management Protocol request and response messages, or general-
-- purpose (e.g., announcement) messages for future needs or for
-- specific environments.

GenMsgContent ::= SEQUENCE OF InfoTypeAndValue

-- May be sent by EE, RA, or CA (depending on message content).
-- The OPTIONAL infoValue parameter of InfoTypeAndValue will
-- typically be omitted for some of the examples given above.
-- The receiver is free to ignore any contained OBJ. IDs that it
-- does not recognize. If sent from EE to CA, the empty set
-- indicates that the CA may send
-- any/all information that it wishes.

GenRepContent ::= SEQUENCE OF InfoTypeAndValue
-- Receiver MAY ignore any contained OIDs that it does not
-- recognize.

ErrorMsgContent ::= SEQUENCE {
  pKIStatusInfo   PKIStatusInfo,
  errorCode       INTEGER     OPTIONAL,
  -- implementation-specific error codes
  errorDetails    PKIFreeText   OPTIONAL
  -- implementation-specific error details
}

PollReqContent ::= SEQUENCE OF SEQUENCE {
  certReqId       INTEGER }

PollRepContent ::= SEQUENCE OF SEQUENCE {
  certReqId       INTEGER,
  checkAfter      INTEGER,  -- time in seconds
  reason          PKIFreeText OPTIONAL }

END

10. ASN.1 Module for RFC 4211

PKIXCRMF-2005
{iso(1) identified-organization(3) dod(6) internet(1) security(5)
  mechanisms(5) pkix(7) id-mod(0) id-mod-crmf2005(36)}
DEFINITIONS IMPLICIT TAGS :=
BEGIN
IMPORTS

Version, AlgorithmIdentifier, Name, Time, SubjectPublicKeyInfo,
Extensions, UniqueIdentifier, Attribute
FROM PKIX1Explicit88

 GeneralName
FROM PKIX1Implicit88

 EnvelopedData
FROM CryptographicMessageSyntax2004

id-pkix OBJECT IDENTIFIER ::= { iso(1) identified-organization(3)
dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) id-pkix1-explicit(18) }

GeneralName
FROM PKIX1Implicit88

EnvelopedData
FROM CryptographicMessageSyntax2004

id-pkix OBJECT IDENTIFIER ::= { iso(1) identified-organization(3)
dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) id-pkix1-implicit(19) }

id-pkix OBJECT IDENTIFIER ::= { iso(1) identified-organization(3)
dod(6) internet(1) security(5) mechanisms(5) 7 }

-- arc for Internet X.509 PKI protocols and their components

id-pkip OBJECT IDENTIFIER ::= { id-pkix 5 }

id-smime OBJECT IDENTIFIER ::= { iso(1) member-body(2)
us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
smime(16) modules(0) cms-2004(24) }; -- found in [CMS]

CertReqMessages ::= SEQUENCE SIZE (1..MAX) OF CertReqMsg

CertReqMsg ::= SEQUENCE {
certReq CertRequest,
popo ProofOfPossession OPTIONAL,
-- content depends upon key type
regInfo SEQUENCE SIZE(1..MAX) OF
AttributeTypeAndValue OPTIONAL }

CertRequest ::= SEQUENCE {
certReqId INTEGER,     -- ID for matching request and reply
certTemplate CertTemplate, -- Selected fields of cert to be issued
controls Controls OPTIONAL } -- Attributes affecting issuance

CertTemplate ::= SEQUENCE {
}
version [0] Version OPTIONAL,
serialNumber [1] INTEGER OPTIONAL,
signingAlg [2] AlgorithmIdentifier OPTIONAL,
issuer [3] Name OPTIONAL,
validity [4] OptionalValidity OPTIONAL,
subject [5] Name OPTIONAL,
publicKey [6] SubjectPublicKeyInfo OPTIONAL,
issuerUID [7] UniqueIdentifier OPTIONAL,
subjectUID [8] UniqueIdentifier OPTIONAL,
extensions [9] Extensions OPTIONAL

OptionalValidity ::= SEQUENCE {
  notBefore [0] Time OPTIONAL,
  notAfter [1] Time OPTIONAL } -- at least one MUST be present

ATTRIBUTE ::= TYPE-IDENTIFIER

Controls ::= SEQUENCE SIZE(1..MAX) OF AttributeTypeAndValue

AttributeTypeAndValue ::= SEQUENCE {
  type ATTRIBUTE.&id({SupportedAttributes}),
  value ATTRIBUTE.&Type({SupportedAttributes})}

SupportedAttributes ATTRIBUTE ::= { ... }

ProofOfPossession ::= CHOICE {
  raVerified [0] NULL,
  -- used if the RA has already verified that the requester is in
  -- possession of the private key
  signature [1] POPOSigningKey,
  keyEncipherment [2] POPOPrivKey,
  keyAgreement [3] POPOPrivKey }

POPOSigningKey ::= SEQUENCE {
  poposkInput [0] POPOSigningKeyInput OPTIONAL,
  algorithmIdentifier AlgorithmIdentifier,
  signature BIT STRING }

-- The signature (using "algorithmIdentifier") is on the
-- DER-encoded value of poposkInput. NOTE: If the CertReqMsg
-- certReq CertTemplate contains the subject and publicKey values,
-- then poposkInput MUST be omitted and the signature MUST be
-- computed over the DER-encoded value of CertReqMsg certReq. If
-- the CertReqMsg certReq CertTemplate does not contain both the
-- public key and subject values (i.e., if it contains only one
-- of these, or neither), then poposkInput MUST be present and
-- MUST be signed.

POPOSigningKeyInput ::= SEQUENCE {
  --
authInfo  CHOICE {
   sender       [0] GeneralName,
   -- used only if an authenticated identity has been
   -- established for the sender (e.g., a DN from a
   -- previously-issued and currently-valid certificate)
   publicKeyMAC PKMACValue },
   -- used if no authenticated GeneralName currently exists for
   -- the sender; publicKeyMAC contains a password-based MAC
   -- on the DER-encoded value of publicKey
   publicKey    SubjectPublicKeyInfo }  -- from CertTemplate

PKMACValue ::= SEQUENCE {
   algId  AlgorithmIdentifier,
   -- algorithm value shall be PasswordBasedMac
   -- {1 2 840 113533 7 66 13}
   -- parameter value is PBMParameter
   value  BIT STRING }

PBMParameter ::= SEQUENCE {
   salt                OCTET STRING,
   owf                 AlgorithmIdentifier,
   -- AlgId for a One-Way Function (SHA-1 recommended)
   iterationCount      INTEGER,
   -- number of times the OWF is applied
   mac                 AlgorithmIdentifier
   -- the MAC AlgId (e.g., DES-MAC, Triple-DES-MAC [PKCS11],
   -- or HMAC [RFC2202])
}

POPOPrivKey ::= CHOICE {
   thisMessage       [0] BIT STRING,  -- Deprecated
   -- possession is proven in this message (which contains
   -- the private key itself (encrypted for the CA))
   subsequentMessage [1] SubsequentMessage,
   -- possession will be proven in a subsequent message
   dhMAC             [2] BIT STRING,  -- Deprecated
   agreeMAC          [3] PKMACValue,
   encryptedKey      [4] EnvelopedData }
   -- for keyAgreement (only), possession is proven in this message
   -- (which contains a MAC (over the DER-encoded value of the
   -- certReq parameter in CertReqMsg, which MUST include both
   -- subject and publicKey) based on a key derived from the end
   -- entity’s private DH key and the CA’s public DH key);

SubsequentMessage ::= INTEGER {
   encrCert (0),
   -- requests that resulting certificate be encrypted for the
   -- end entity (following which, POP will be proven in a
   -- confirmation message)
challengeResp (1) }
   -- requests that CA engage in challenge-response exchange with
   -- end entity in order to prove private key possession

-- Object identifier assignments --

-- Registration Controls in CRMF
   id-regCtrl OBJECT IDENTIFIER ::= { id-pkip 1 }

id-regCtrl-regToken OBJECT IDENTIFIER ::= { id-regCtrl 1 }
   -- with syntax:
   RegToken ::= UTF8String

id-regCtrl-authenticator OBJECT IDENTIFIER ::= { id-regCtrl 2 }
   -- with syntax:
   Authenticator ::= UTF8String

id-regCtrl-pkiPublicationInfo OBJECT IDENTIFIER ::= { id-regCtrl 3 }
   -- with syntax:
   PKIPublicationInfo ::= SEQUENCE {
      action     INTEGER {
         dontPublish (0),
         pleasePublish (1) },
      pubInfos  SEQUENCE SIZE (1..MAX) OF SinglePubInfo OPTIONAL }
   -- pubInfos MUST NOT be present if action is "dontPublish"
   -- (if action is "pleasePublish" and pubInfos is omitted,
   -- "dontCare" is assumed)

SinglePubInfo ::= SEQUENCE {
   pubMethod    INTEGER {
      dontCare    (0),
      x500        (1),
      web         (2),
      ldap        (3) },
   pubLocation  GeneralName OPTIONAL }

id-regCtrl-pkiArchiveOptions OBJECT IDENTIFIER ::= { id-regCtrl 4 }
   -- with syntax:
   PKIArchiveOptions ::= CHOICE {
      encryptedPrivKey     [0] EncryptedKey,
      -- the actual value of the private key
      keyGenParameters     [1] KeyGenParameters,
      -- parameters that allow the private key to be re-generated
      archiveRemGenPrivKey [2] BOOLEAN }
   -- set to TRUE if sender wishes receiver to archive the private
   -- key of a key pair that the receiver generates in response to
   -- this request; set to FALSE if no archival is desired.
EncryptedKey ::= CHOICE {
  encryptedValue        EncryptedValue,   -- Deprecated
  envelopedData     [0] EnvelopedData }
-- The encrypted private key MUST be placed in the envelopedData
-- encryptedContentInfo encryptedContent OCTET STRING.

EncryptedValue ::= SEQUENCE {
  intendedAlg   [0] AlgorithmIdentifier  OPTIONAL,
-- the intended algorithm for which the value will be used
  symmAlg       [1] AlgorithmIdentifier  OPTIONAL,
-- the symmetric algorithm used to encrypt the value
  encSymmKey    [2] BIT STRING           OPTIONAL,
-- the (encrypted) symmetric key used to encrypt the value
  keyAlg        [3] AlgorithmIdentifier  OPTIONAL,
-- algorithm used to encrypt the symmetric key
  valueHint     [4] OCTET STRING         OPTIONAL,
-- a brief description or identifier of the encValue content
-- (may be meaningful only to the sending entity, and used only
-- if EncryptedValue might be re-examined by the sending entity
-- in the future)
  encValue       BIT STRING }
-- the encrypted value itself
-- When EncryptedValue is used to carry a private key (as opposed to
-- a certificate), implementations MUST support the encValue field
-- containing an encrypted PrivateKeyInfo as defined in [PKCS11],
-- section 12.11.  If encValue contains some other format/encoding
-- for the private key, the first octet of valueHint MAY be used
-- to indicate the format/encoding (but note that the possible values
-- of this octet are not specified at this time).  In all cases, the
-- intendedAlg field MUST be used to indicate at least the OID of
-- the intended algorithm of the private key, unless this information
-- is known a priori to both sender and receiver by some other means.

KeyGenParameters ::= OCTET STRING

id-regCtrl-oldCertID OBJECT IDENTIFIER ::= { id-regCtrl 5 }
--with syntax:
OldCertId ::= CertId

CertId ::= SEQUENCE {
  issuer           GeneralName,
  serialNumber     INTEGER }

id-regCtrl-protocolEncrKey OBJECT IDENTIFIER ::= { id-regCtrl 6 }
--with syntax:
ProtocolEncrKey ::= SubjectPublicKeyInfo

-- Registration Info in CRMF
id-regInfo OBJECT IDENTIFIER ::= { id-pkip 2 }

id-regInfo-utf8Pairs OBJECT IDENTIFIER ::= { id-regInfo 1 }
-- with syntax
UTF8Pairs ::= UTF8String

id-regInfo-crtReq OBJECT IDENTIFIER ::= { id-regInfo 2 }
-- with syntax
CrtReq ::= CrtRequest

-- id-ct-encKeyWithID is a new content type used for CMS objects.
-- it contains both a private key and an identifier for key escrow
-- agents to check against recovery requestors.

id-ct-encKeyWithID OBJECT IDENTIFIER ::= {id-ct 21}

EncKeyWithID ::= SEQUENCE {
  privateKey           PrivateKeyInfo,
  identifier CHOICE {
    string             UTF8String,
    generalName        GeneralName
  } OPTIONAL
}

PrivateKeyInfo ::= SEQUENCE {
  version                   INTEGER,
  privateKeyAlgorithm       AlgorithmIdentifier,
  privateKey                OCTET STRING,
  attributes                [0] IMPLICIT Attributes OPTIONAL
}

Attributes ::= SET OF Attribute

END

11. ASN.1 Module for RFC-to-be, SCVP

SCVP
{ iso(1) identified-organization(3) dod(6) internet(1) security(5)
  mechanisms(5) pkix(7) id-mod(0) 21 }

DEFINITIONS IMPLICIT TAGS :=
BEGIN

IMPORTS

AlgorithmIdentifier, Attribute, Certificate, Extensions,
CertificateList, CertificateSerialNumber
FROM PKIX1Explicit88
  { iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) 18 }

GeneralNames, GeneralName, KeyUsage, KeyPurposeId
FROM PKIX1Implicit88
  { iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) 19 }

AttributeCertificate
FROM PKIXAttributeCertificate
  { iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) 12 }

OCSPResponse
FROM OCSP
  { iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) 14 }

ContentInfo
FROM CryptographicMessageSyntax2004
  { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
    smime(16) modules(0) cms-2004(24) } ;

-- SCVP Certificate Validation Request

id-ct OBJECT IDENTIFIER ::= { iso(1) member-body(2)
  us(840) rsadsi(113549) pkcs(1) pkcs9(9) id-smime(16) 1 }

id-ct-scvp-certValRequest OBJECT IDENTIFIER ::= { id-ct 10 }

CVRequest ::= SEQUENCE {
  cvRequestVersion           INTEGER DEFAULT 1,
  query                      Query,
  requestorRef           [0] GeneralNames OPTIONAL,
  requestNonce           [1] OCTET STRING OPTIONAL,
  requesterName          [2] GeneralName OPTIONAL,
  responderName          [3] GeneralName OPTIONAL,
  requestExtensions      [4] Extensions OPTIONAL,
  signatureAlg           [5] AlgorithmIdentifier OPTIONAL,
  hashAlg                [6] OBJECT IDENTIFIER OPTIONAL,
  requestorText          [7] UTF8String (SIZE (1..256)) OPTIONAL }

Query ::= SEQUENCE {
  queriedCerts             CertReferences,
wantBack [1] WantBack OPTIONAL,
validationPolicy ValidationPolicy,
responseFlags ResponseFlags OPTIONAL,
serverContextInfo [2] OCTET STRING OPTIONAL,
intermediateCerts [4] CertBundle OPTIONAL,
revInfos [5] RevocationInfos OPTIONAL,
producedAt [6] GeneralizedTime OPTIONAL,
queryExtensions [7] Extensions OPTIONAL }

CertReferences ::= CHOICE {
pkcRefs [0] SEQUENCE SIZE (1..MAX) OF PKCReference,
acRefs [1] SEQUENCE SIZE (1..MAX) OF ACReference } 

CertReference ::= CHOICE {
pkc PKCReference,
ac ACReference } 

PKCReference ::= CHOICE {
cert [0] Certificate,
pkcRef [1] SCVPCertID } 

ACReference ::= CHOICE {
attrCert [2] AttributeCertificate,
acRef [3] SCVPCertID } 

SCVPCertID ::= SEQUENCE {
certHash OCTET STRING,
issuerSerial SCVPIssuerSerial,
hashAlgorithm AlgorithmIdentifier DEFAULT { algorithm sha-1 } } 

SCVPIssuerSerial ::= SEQUENCE {
issuer GeneralNames,
serialNumber CertificateSerialNumber } 

ValidationPolicy ::= SEQUENCE {
validationPolRef ValidationPolRef,
validationAlg [0] ValidationAlg OPTIONAL,
userPolicySet [1] SEQUENCE SIZE (1..MAX) OF OBJECT IDENTIFIER OPTIONAL,
inhibitPolicyMapping [2] BOOLEAN OPTIONAL,
requireExplicitPolicy [3] BOOLEAN OPTIONAL,
inhibitAnyPolicy [4] BOOLEAN OPTIONAL,
trustAnchors [5] TrustAnchors OPTIONAL,
keyUsages [6] SEQUENCE OF KeyUsage OPTIONAL,
extendedKeyUsages [7] SEQUENCE OF KeyPurposeId OPTIONAL,
specifiedKeyUsages [8] SEQUENCE OF KeyPurposeId OPTIONAL }
CertChecks ::= SEQUENCE SIZE (1..MAX) OF OBJECT IDENTIFIER

WantBack ::= SEQUENCE SIZE (1..MAX) OF OBJECT IDENTIFIER

POLICY ::= TYPE-IDENTIFIER

ValidationPolRef ::= SEQUENCE {
    valPolId             POLICY.&id,
    valPolParams         POLICY.&Type OPTIONAL }

ValidationAlg ::= SEQUENCE {
    valAlgId               POLICY.&id,
    parameters             POLICY.&Type OPTIONAL }

NameValidationAlgParms ::= SEQUENCE {
    nameCompAlgId          OBJECT IDENTIFIER,
    validationNames        GeneralNames }

TrustAnchors ::= SEQUENCE SIZE (1..MAX) OF PKCReference

KeyAgreePublicKey ::= SEQUENCE {
    algorithm           AlgorithmIdentifier,
    publicKey           BIT STRING,
    macAlgorithm        AlgorithmIdentifier,
    kDF                 AlgorithmIdentifier OPTIONAL }

ResponseFlags ::= SEQUENCE {
    fullRequestInResponse      [0] BOOLEAN DEFAULT FALSE,
    responseValidationPolByRef [1] BOOLEAN DEFAULT TRUE,
    protectResponse            [2] BOOLEAN DEFAULT TRUE,
    cachedResponse             [3] BOOLEAN DEFAULT TRUE }

CertBundle ::= SEQUENCE SIZE (1..MAX) OF Certificate

RevocationInfos ::= SEQUENCE SIZE (1..MAX) OF RevocationInfo

RevocationInfo ::= CHOICE {
    crl        [0] CertificateList,
    delta-crl  [1] CertificateList,
    ocsp       [2] OCSPResponse,
    other      [3] OtherRevInfo }

REV-INFO ::= TYPE-IDENTIFIER

OtherRevInfo ::= SEQUENCE {
    riType                     REV-INFO.&id,
    riValue                    REV-INFO.&Type }
--- SCVP Certificate Validation Response

id-ct-scvp-certValResponse OBJECT IDENTIFIER ::= { id-ct 11 }

CVResponse ::= SEQUENCE {
    cvResponseVersion          INTEGER,
    serverConfigurationID      INTEGER,
    producedAt                 GeneralizedTime,
    responseStatus             ResponseStatus,
    respValidationPolicy       [0] RespValidationPolicy OPTIONAL,
    requestRef                 [1] RequestReference OPTIONAL,
    requestorRef               [2] GeneralNames OPTIONAL,
    requestorName              [3] GeneralNames OPTIONAL,
    replyObjects               [4] ReplyObjects OPTIONAL,
    respNonce                  [5] OCTET STRING OPTIONAL,
    serverContextInfo          [6] OCTET STRING OPTIONAL,
    cvResponseExtensions       [7] Extensions OPTIONAL,
    requestorText              [8] UTF8String (SIZE (1..256)) OPTIONAL }

ResponseStatus ::= SEQUENCE {
    statusCode               CVStatusCode DEFAULT okay,
    errorMessage             UTF8String OPTIONAL }

CVStatusCode ::= ENUMERATED {
    okay                        (0),
    skipUnrecognizedItems      (1),
    tooBusy                     (10),
    internalError              (11),
    badStructure               (12),
    unsupportedVersion         (21),
    unsupportedChecks          (22),
    unrecognizedSigKey         (23),
    badSignatureOrMAC          (24),
    unableToDecode             (25),
    notAuthorized              (26),
    unsupportedWantBacks       (28),
    unsupportedSignatureOrMAC  (30),
    protectedResponseUnsupported (31),
    unrecognizedResponderName  (32),
    relayingLoop               (40),
    unrecognizedValPol         (50),
    unrecognizedValAlg         (51),
    fullRequestInResponseUnsupported (52),
    fullPolResponseUnsupported (53),
    inhibitPolicyMappingUnsupported (54),
    ... }
requireExplicitPolicyUnsupported (55),
inhibitAnyPolicyUnsupported (56),
validationTimeUnsupported (57),
unrecognizedCritQueryExt (63),
unrecognizedCritRequestExt (64) }

RespValidationPolicy ::= ValidationPolicy

RequestReference ::= CHOICE {
    requestHash   [0] HashValue, -- hash of CVRequest
    fullRequest   [1] CVRequest
}

HashValue ::= SEQUENCE {
    algorithm         AlgorithmIdentifier DEFAULT { algorithm sha-1 },
    value             OCTET STRING
}

sha-1 OBJECT IDENTIFIER ::= { iso(1) identified-organization(3)
    oiw(14) secsig(3) algorithm(2) 26 }

ReplyObjects ::= SEQUENCE SIZE (1..MAX) OF CertReply

CertReply ::= SEQUENCE {
    cert                       CertReference,
    replyStatus                ReplyStatus DEFAULT success,
    replyValTime               GeneralizedTime,
    replyChecks                ReplyChecks,
    replyWantBacks             ReplyWantBacks,
    validationErrors       [0] SEQUENCE SIZE (1..MAX) OF
        OBJECT IDENTIFIER OPTIONAL,
    nextUpdate             [1] GeneralizedTime OPTIONAL,
    certReplyExtensions    [2] Extensions OPTIONAL
}

ReplyStatus ::= ENUMERATED {
    success                    (0),
    malformedPKC               (1),
    malformedAC                (2),
    unavailableValidationTime  (3),
    referenceCertHashFail      (4),
    certPathConstructFail      (5),
    certPathNotValid           (6),
    certPathNotValidNow        (7),
    wantBackUnsatisfied        (8) }

ReplyChecks ::= SEQUENCE OF ReplyCheck

ReplyCheck ::= SEQUENCE {
    check                      OBJECT IDENTIFIER,
    status                     INTEGER DEFAULT 0 }
ReplyWantBacks ::= SEQUENCE OF ReplyWantBack

ReplyWantBack ::= SEQUENCE {
  wb OBJECT IDENTIFIER,
  value OCTET STRING }

CertBundles ::= SEQUENCE SIZE (1..MAX) OF CertBundle

RevInfoWantBack ::= SEQUENCE {
  revocationInfo RevocationInfos,
  extraCerts CertBundle OPTIONAL }

SCVPResponses ::= SEQUENCE OF ContentInfo

-- SCVP Validation Policies Request

id-ct-scvp-valPolRequest OBJECT IDENTIFIER ::= { id-ct 12 }

ValPolRequest ::= SEQUENCE {
  vpRequestVersion INTEGER DEFAULT 1,
  requestNonce OCTET STRING }

-- SCVP Validation Policies Response

id-ct-scvp-valPolResponse OBJECT IDENTIFIER ::= { id-ct 13 }

ValPolResponse ::= SEQUENCE {
  vpResponseVersion INTEGER,
  maxCVRequestVersion INTEGER,
  maxVPRequestVersion INTEGER,
  serverConfigurationID INTEGER,
  thisUpdate GeneralizedTime,
  nextUpdate GeneralizedTime OPTIONAL,
  supportedChecks CertChecks,
  supportedWantBacks WantBack,
  validationPolicies SEQUENCE OF OBJECT IDENTIFIER,
  validationAlgs SEQUENCE OF OBJECT IDENTIFIER,
  authPolicies SEQUENCE OF AuthPolicy,
  responseTypes ResponseTypes,
  defaultPolicyValues RespValidationPolicy,
  revocationInfoTypes RevocationInfoTypes,
  signatureGeneration SEQUENCE OF AlgorithmIdentifier,
  signatureVerification SEQUENCE OF AlgorithmIdentifier,
  hashAlgorithms SEQUENCE SIZE (1..MAX) OF OBJECT IDENTIFIER,
  serverPublicKeys SEQUENCE OF KeyAgreePublicKey OPTIONAL,
  clockSkew INTEGER DEFAULT 10,
requestNonce OCTET STRING OPTIONAL }

ResponseTypes ::= ENUMERATED {
    cached-only (0),
    non-cached-only (1),
    cached-and-non-cached (2) }

RevocationInfoTypes ::= BIT STRING {
    fullCRLs (0),
    deltaCRLs (1),
    indirectCRLs (2),
    oCSPResponses (3) }

AuthPolicy ::= OBJECT IDENTIFIER

-- SCVP Check Identifiers

id-stc OBJECT IDENTIFIER ::= { iso(1) identified-organization(3)
    dod(6) internet(1) security(5) mechanisms(5) pkix(7) 17 }

id-stc-build-pkc-path OBJECT IDENTIFIER ::= { id-stc 1 }
id-stc-build-valid-pkc-path OBJECT IDENTIFIER ::= { id-stc 2 }
id-stc-build-status-checked-pkc-path
    OBJECT IDENTIFIER ::= { id-stc 3 }

id-stc-build-aa-path OBJECT IDENTIFIER ::= { id-stc 4 }
id-stc-build-valid-aa-path OBJECT IDENTIFIER ::= { id-stc 5 }
id-stc-build-status-checked-aa-path
    OBJECT IDENTIFIER ::= { id-stc 6 }
id-stc-status-check-ac-and-build-status-checked-aa-path
    OBJECT IDENTIFIER ::= { id-stc 7 }

-- SCVP WantBack Identifiers

id-swb OBJECT IDENTIFIER ::= { iso(1) identified-organization(3)
    dod(6) internet(1) security(5) mechanisms(5) pkix(7) 18 }

id-swb-pkc-best-cert-path OBJECT IDENTIFIER ::= { id-swb 1 }
id-swb-pkc-revocation-info OBJECT IDENTIFIER ::= { id-swb 2 }
id-swb-pkc-public-key-info OBJECT IDENTIFIER ::= { id-swb 4 }
id-swb-aa-cert-path OBJECT IDENTIFIER ::= { id-swb 5 }
id-swb-aa-revocation-info OBJECT IDENTIFIER ::= { id-swb 6 }
id-swb-ac-revocation-info OBJECT IDENTIFIER ::= { id-swb 7 }
id-swb-relayed-responses OBJECT IDENTIFIER ::= { id-swb 9 }
id-swb-pkc-cert OBJECT IDENTIFIER ::= { id-swb 10 }
id-swb-ac-cert OBJECT IDENTIFIER ::= { id-swb 11 }
id-swb-pkc-all-cert-paths OBJECT IDENTIFIER ::= { id-swb 12 }
id-swb-pkc-ee-revocation-info OBJECT IDENTIFIER ::= { id-swb 13 }
id-swb-pkc-CAs-revocation-info OBJECT IDENTIFIER ::= { id-swb 14}

-- SCVP Validation Policy and Algorithm Identifiers

id-svp OBJECT IDENTIFIER ::= { iso(1) identified-organization(3)
    dod(6) internet(1) security(5) mechanisms(5) pkix(7) 19 }

id-svp-defaultValPolicy OBJECT IDENTIFIER ::= { id-svp 1 }

-- SCVP Basic Validation Algorithm Identifier

id-svp-basicValAlg OBJECT IDENTIFIER ::= { id-svp 3 }

-- SCVP Basic Validation Algorithm Errors

id-bvae OBJECT IDENTIFIER ::= id-svp-basicValAlg

id-bvae-expired OBJECT IDENTIFIER ::= { id-bvae 1 }

id-bvae-not-yet-valid OBJECT IDENTIFIER ::= { id-bvae 2 }

id-bvae-wrongTrustAnchor OBJECT IDENTIFIER ::= { id-bvae 3 }

id-bvae-noValidCertPath OBJECT IDENTIFIER ::= { id-bvae 4 }

id-bvae-revoked OBJECT IDENTIFIER ::= { id-bvae 5 }

id-bvae-invalidKeyPurpose OBJECT IDENTIFIER ::= { id-bvae 9 }

id-bvae-invalidKeyUsage OBJECT IDENTIFIER ::= { id-bvae 10 }

id-bvae-invalidCertPolicy OBJECT IDENTIFIER ::= { id-bvae 11 }

-- SCVP Name Validation Algorithm Identifier

id-svp-nameValAlg OBJECT IDENTIFIER ::= { id-svp 2 }

-- SCVP Name Validation Algorithm DN comparison algorithm

id-nva-dnCompAlg OBJECT IDENTIFIER ::= { id-svp 4 }

-- SCVP Name Validation Algorithm Errors

id-nvae OBJECT IDENTIFIER ::= id-svp-nameValAlg

id-nvae-name-mismatch OBJECT IDENTIFIER ::= { id-nvae 1 }

id-nvae-no-name OBJECT IDENTIFIER ::= { id-nvae 2 }

id-nvae-unknown-alg OBJECT IDENTIFIER ::= { id-nvae 3 }

id-nvae-bad-name OBJECT IDENTIFIER ::= { id-nvae 4 }

id-nvae-bad-name-type OBJECT IDENTIFIER ::= { id-nvae 5 }

id-nvae-mixed-names OBJECT IDENTIFIER ::= { id-nvae 6 }

-- SCVP Extended Key Usage Key Purpose Identifiers

id-kp OBJECT IDENTIFIER ::= { iso(1) identified-organization(3)
12. Security Considerations

Even though all the RFCs in this document are security-related, the document itself does not have any security considerations. The ASN.1 modules keep the same bits-on-the-wire as the modules that they replace.

13. Normative References

[ASN1-2002]

[NEW-CMS-SMIME]

[RFC2560]

[RFC2986]

[RFC3279]

[RFC3280]

[RFC3281]
Farrell, S. and R. Housley, "An Internet Attribute


Authors’ Addresses

Paul Hoffman
VPN Consortium
127 Segre Place
Santa Cruz, CA  95060
US

Phone: 1-831-426-9827
Email: paul.hoffman@vpnc.org

Jim Schaad
Soaring Hawk Consulting

Email: jimsch@exmsft.com
Full Copyright Statement

Copyright (C) The IETF Trust (2007).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).