Internet Engineering Task Force (IETF)
Request for Comments: 5912
Category: Informational
ISSN: 2070-1721

P. Hoffman
VPN Consortium
J. Schaad
Soaring Hawk Consulting

June 2010

New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)

Abstract

The Public Key Infrastructure using X.509 (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.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc5912.

Table of Contents

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, Cryptographic Message Syntax (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 complete.

- RFC 2560, PKIX Online Certificate Status Protocol (OCSP) [RFC2560]
- RFC 2986, PKCS #10 certificate request [RFC2986]
- RFC 3279, PKIX algorithms and identifier [RFC3279]
- RFC 3852, contains PKIX attribute certificates, version 1 [RFC3852]
- RFC 4055, Additional Algorithms and Identifiers for RSA Cryptography [RFC4055]
- RFC 4210, PKIX CMP (Certificate Management Protocol) [RFC4210]
- RFC 4211, PKIX CRMF (Certificate Request Message Format) [RFC4211]
- RFC 5055, PKIX SCVP (Server-based Certificate Validation Protocol) [RFC5055]
- RFC 5272, Certificate Management over CMS (CMC) [RFC5272]
- RFC 5280, PKIX certificate and Certificate Revocation List (CRL) profile [RFC5280] (both the implicit and explicit modules)
- RFC 5755, PKIX attribute certificates, version 2 [RFC5755]

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 [RFC5911] can stand on their own and do not need to import definitions from anywhere else. Also note that the ASN.1 modules in this document have references in their text comments that need to be looked up in original RFCs, and that some of those references may have already been superseded by later RFCs.

The document also includes a module of common definitions called "PKIX-CommonTypes". These definitions are used here and in [RFC5911].

The document also includes a module of common definitions called "AlgorithmInformation". These definitions are used here and in [RFC5911].

1.1. Design Notes

The modules in this document use the object model available in the 2002 ASN.1 documents to a great extent. Objects for each of the different algorithm types are defined. Also, all of the places where the 1988 ASN.1 syntax had ANY holes to allow for variable syntax now use objects.

Much like the way that the PKIX and S/MIME working groups use the prefix of id- for object identifiers, this document has also adopted a set of two-, three-, and four-letter prefixes to allow for quick identification of the type of an object based on its name. This allows, for example, the same back half of the name to be used for the different objects. Thus, "id-sha1" is the object identifier, while "mda-sha1" is the message digest object for "sha1".

One or more object sets for the different types of algorithms are defined. A single consistent name for each different algorithm type is used. For example, an object set named PublicKeys contains the public keys defined in that module. If no public keys are defined, then the object set is not created. When importing these object sets into an ASN.1 module, one needs to be able to distinguish between the different object sets with the same name. This is done by using both the module name (as specified in the IMPORT statement) and the object set name. For example, in the module for RFC 5280:

```
PublicKeys FROM PKIXAlgs-2008 { 1 3 6 1 5 5 7 0 995 }
PublicKeys FROM PKIX1-PSS-OAEP-Algorithms { 1 3 6 1 5 5 7  33 }

PublicKeyAlgorithms PUBLIC-KEY ::= { PKIXAlgs-2008.PublicKeys, ...,
    PKIX1-PSS-OAEP-Algorithms.PublicKeys }
```

2. ASN.1 Module PKIX-CommonTypes

This section contains a module that is imported by many other modules in this document and in [RFC5911]. This module does not come from any existing RFC.

```
PKIX-CommonTypes-2009
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-pkixCommon-02(57)}

DEFINITIONS EXPLICIT TAGS ::=
BEGIN

--  ATTRIBUTE
--
--  Describe the set of data associated with an attribute of some type
--
--  &id is an OID identifying the attribute
--  &Type is the ASN.1 type structure for the attribute; not all
--      attributes have a data structure, so this field is optional
--  &minCount contains the minimum number of times the attribute can
--      occur in an AttributeSet
--  &maxCount contains the maximum number of times the attribute can
--      appear in an AttributeSet
--      Note: this cannot be automatically enforced as the field
--      cannot be defaulted to MAX.
--  &equality-match contains information about how matching should be
--      done
--
--  Currently we are using two different prefixes for attributes.
--
--  at- for certificate attributes
--  aa- for CMS attributes
--

ATTRIBUTE ::= CLASS {
    &id             OBJECT IDENTIFIER UNIQUE,
    &Type           OPTIONAL,
    &equality-match MATCHING-RULE OPTIONAL,
    &minCount       INTEGER DEFAULT 1,
    &maxCount       INTEGER OPTIONAL
} WITH SYNTAX {
    [TYPE &Type]
    [EQUALITY MATCHING RULE &equality-match]
    [COUNTS [MIN &minCount] [MAX &maxCount]]
    IDENTIFIED BY &id
}
```

```
  -- Specification of MATCHING-RULE information object class
  --

  MATCHING-RULE ::= CLASS {
    &ParentMatchingRules   MATCHING-RULE OPTIONAL,
    &AssertionType         OPTIONAL,
    &uniqueMatchIndicator  ATTRIBUTE OPTIONAL,
    &id                    OBJECT IDENTIFIER UNIQUE
  }
  WITH SYNTAX {
    [PARENT &ParentMatchingRules]
    [SYNTAX &AssertionType]
    [UNIQUE-MATCH-INDICATOR &uniqueMatchIndicator]
    ID &id
  }

  --  AttributeSet
  --
  --  Used when a set of attributes is to occur.
  --
  --  type contains the identifier of the attribute
  --  values contains a set of values where the structure of the ASN.1
  --      is defined by the attribute
  --
  --  The parameter contains the set of objects describing
  --      those attributes that can occur in this location.
  --

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

  --  SingleAttribute
  --
  --  Used for a single valued attribute
  --
  --  The parameter contains the set of objects describing the
  --      attributes that can occur in this location
  --

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

  --  EXTENSION
```

```
  --
  --  This class definition is used to describe the association of
  --      object identifier and ASN.1 type structure for extensions
  --
  --  All extensions are prefixed with ext-
  --
  --  &id contains the object identifier for the extension
  --  &ExtnType specifies the ASN.1 type structure for the extension
  --  &Critical contains the set of legal values for the critical field.
  --      This is normally {TRUE|FALSE} but in some instances may be
  --      restricted to just one of these values.
  --

  EXTENSION ::= CLASS {
      &id  OBJECT IDENTIFIER UNIQUE,
      &ExtnType,
      &Critical    BOOLEAN DEFAULT {TRUE | FALSE }
  } WITH SYNTAX {
      SYNTAX &ExtnType IDENTIFIED BY &id
      [CRITICALITY &Critical]
  }

  --  Extensions
  --
  --  Used for a sequence of extensions.
  --
  --  The parameter contains the set of legal extensions that can
  --  occur in this sequence.
  --

  Extensions{EXTENSION:ExtensionSet} ::=
      SEQUENCE SIZE (1..MAX) OF Extension{{ExtensionSet}}

  --  Extension
  --
  --  Used for a single extension
  --
  --  The parameter contains the set of legal extensions that can
  --      occur in this extension.
  --
  --  The restriction on the critical field has been commented out
  --  the authors are not completely sure it is correct.
  --  The restriction could be done using custom code rather than
  --  compiler-generated code, however.
  --

  Extension{EXTENSION:ExtensionSet} ::= SEQUENCE {
      extnID      EXTENSION.&id({ExtensionSet}),
```

```
      critical    BOOLEAN
  --                     (EXTENSION.&Critical({ExtensionSet}{@extnID}))
                       DEFAULT FALSE,
      extnValue   OCTET STRING (CONTAINING
                  EXTENSION.&ExtnType({ExtensionSet}{@extnID}))
                  --  contains the DER encoding of the ASN.1 value
                  --  corresponding to the extension type identified
                  --  by extnID
  }

  --  Security Category
  --
  --  Security categories are used both for specifying clearances and
  --  for labeling objects.  We move this here from RFC 3281 so that
  --  they will use a common single object class to express this
  --  information.
  --

  SECURITY-CATEGORY ::= TYPE-IDENTIFIER

  SecurityCategory{SECURITY-CATEGORY:Supported} ::= SEQUENCE {
      type      [0]  IMPLICIT SECURITY-CATEGORY.
                &id({Supported}),
      value     [1]  EXPLICIT SECURITY-CATEGORY.
                &Type({Supported}{@type})
  }

  END
```

3. ASN.1 Module AlgorithmInformation

This section contains a module that is imported by many other modules in this document. Note that this module is also given in [RFC5911]. This module does not come from any existing RFC.

```
AlgorithmInformation-2009
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0)
    id-mod-algorithmInformation-02(58)}

DEFINITIONS EXPLICIT TAGS ::=
BEGIN
EXPORTS ALL;
IMPORTS

KeyUsage
FROM PKIX1Implicit-2009
    {iso(1) identified-organization(3) dod(6) internet(1)
```

```
    security(5) mechanisms(5) pkix(7) id-mod(0)
    id-mod-pkix1-implicit-02(59)} ;

--  Suggested prefixes for algorithm objects are:
--
--  mda-   Message Digest Algorithms
--  sa-    Signature Algorithms
--  kta-   Key Transport Algorithms (Asymmetric)
--  kaa-   Key Agreement Algorithms  (Asymmetric)
--  kwa-   Key Wrap Algorithms (Symmetric)
--  kda-   Key Derivation Algorithms
--  maca-  Message Authentication Code Algorithms
--  pk-    Public Key
--  cea-   Content (symmetric) Encryption Algorithms
--  cap-   S/MIME Capabilities

ParamOptions ::= ENUMERATED {
   required,         -- Parameters MUST be encoded in structure
   preferredPresent, -- Parameters SHOULD be encoded in structure
   preferredAbsent,  -- Parameters SHOULD NOT be encoded in structure
   absent,           -- Parameters MUST NOT be encoded in structure
   inheritable,      -- Parameters are inherited if not present
   optional,         -- Parameters MAY be encoded in the structure
   ...
}

--  DIGEST-ALGORITHM
--
--  Describes the basic information for ASN.1 and a digest
--      algorithm.
--
--  &id - contains the OID identifying the digest algorithm
--  &Params - if present, contains the type for the algorithm
--               parameters; if absent, implies no parameters
--  &paramPresence - parameter presence requirement
--
--  Additional information such as the length of the hash could have
--      been encoded.  Without a clear understanding of what information
--      is needed by applications, such extraneous information was not
--      considered to be of sufficent importance.
--
--  Example:
--  mda-sha1 DIGEST-ALGORITHM ::= {
--      IDENTIFIER id-sha1
--      PARAMS TYPE NULL ARE preferredAbsent
--  }

DIGEST-ALGORITHM ::= CLASS {
```

```
    &id                 OBJECT IDENTIFIER UNIQUE,
    &Params             OPTIONAL,
    &paramPresence      ParamOptions DEFAULT absent
} WITH SYNTAX {
    IDENTIFIER &id
    [PARAMS [TYPE &Params] ARE &paramPresence ]
}

--  SIGNATURE-ALGORITHM
--
--  Describes the basic properties of a signature algorithm
--
--  &id - contains the OID identifying the signature algorithm
--  &Value - contains a type definition for the value structure of
--              the signature; if absent, implies that no ASN.1
--              encoding is performed on the value
--  &Params - if present, contains the type for the algorithm
--               parameters; if absent, implies no parameters
--  &paramPresence - parameter presence requirement
--  &HashSet - The set of hash algorithms used with this
--                  signature algorithm
--  &PublicKeySet - the set of public key algorithms for this
--                  signature algorithm
--  &smimeCaps - contains the object describing how the S/MIME
--              capabilities are presented.
--
--  Example:
--  sig-RSA-PSS SIGNATURE-ALGORITHM ::= {
--     IDENTIFIER id-RSASSA-PSS
--     PARAMS TYPE RSASSA-PSS-params ARE required
--     HASHES { mda-sha1 | mda-md5, ... }
--     PUBLIC-KEYS { pk-rsa | pk-rsa-pss }
-- }

SIGNATURE-ALGORITHM ::= CLASS {
    &id             OBJECT IDENTIFIER UNIQUE,
    &Value          OPTIONAL,
    &Params         OPTIONAL,
    &paramPresence  ParamOptions DEFAULT absent,
    &HashSet        DIGEST-ALGORITHM OPTIONAL,
    &PublicKeySet   PUBLIC-KEY OPTIONAL,
    &smimeCaps      SMIME-CAPS OPTIONAL
} WITH SYNTAX {
    IDENTIFIER &id
    [VALUE &Value]
    [PARAMS [TYPE &Params] ARE &paramPresence ]
    [HASHES &HashSet]
    [PUBLIC-KEYS &PublicKeySet]
```

```
    [SMIME-CAPS &smimeCaps]
}

--  PUBLIC-KEY
--
--  Describes the basic properties of a public key
--
--  &id - contains the OID identifying the public key
--  &KeyValue - contains the type for the key value
--  &Params - if present, contains the type for the algorithm
--               parameters; if absent, implies no parameters
--  &paramPresence - parameter presence requirement
--  &keyUsage - contains the set of bits that are legal for this
--              key type.  Note that is does not make any statement
--              about how bits may be paired.
--  &PrivateKey - contains a type structure for encoding the private
--              key information.
--
--  Example:
--  pk-rsa-pss PUBLIC-KEY ::= {
--      IDENTIFIER id-RSASSA-PSS
--      KEY RSAPublicKey
--      PARAMS TYPE RSASSA-PSS-params ARE optional
--      CERT-KEY-USAGE { .... }
--  }

PUBLIC-KEY ::= CLASS {
    &id             OBJECT IDENTIFIER UNIQUE,
    &KeyValue       OPTIONAL,
    &Params         OPTIONAL,
    &paramPresence  ParamOptions DEFAULT absent,
    &keyUsage       KeyUsage OPTIONAL,
    &PrivateKey     OPTIONAL
} WITH SYNTAX {
    IDENTIFIER &id
    [KEY &KeyValue]
    [PARAMS [TYPE &Params] ARE &paramPresence]
    [CERT-KEY-USAGE &keyUsage]
    [PRIVATE-KEY &PrivateKey]
}

--  KEY-TRANSPORT
--
--  Describes the basic properties of a key transport algorithm
--
--  &id - contains the OID identifying the key transport algorithm
--  &Params - if present, contains the type for the algorithm
--               parameters; if absent, implies no parameters
```

```
--  &paramPresence - parameter presence requirement
--  &PublicKeySet - specifies which public keys are used with
--                        this algorithm
--  &smimeCaps - contains the object describing how the S/MIME
--              capabilities are presented.
--
--  Example:
--  kta-rsaTransport KEY-TRANSPORT ::= {
--      IDENTIFIER &id
--      PARAMS TYPE NULL ARE required
--      PUBLIC-KEYS  { pk-rsa | pk-rsa-pss }
--  }

KEY-TRANSPORT ::= CLASS {
    &id                 OBJECT IDENTIFIER UNIQUE,
    &Params             OPTIONAL,
    &paramPresence      ParamOptions DEFAULT absent,
    &PublicKeySet       PUBLIC-KEY OPTIONAL,
    &smimeCaps          SMIME-CAPS OPTIONAL
} WITH SYNTAX {
    IDENTIFIER &id
    [PARAMS [TYPE &Params] ARE &paramPresence]
    [PUBLIC-KEYS &PublicKeySet]
    [SMIME-CAPS &smimeCaps]
}

--  KEY-AGREE
--
--  Describes the basic properties of a key agreement algorithm
--
--  &id - contains the OID identifying the key agreement algorithm
--  &Params - if present, contains the type for the algorithm
--               parameters; if absent, implies no parameters
--  &paramPresence - parameter presence requirement
--  &PublicKeySet - specifies which public keys are used with
--                        this algorithm
--  &Ukm - type of user keying material used
--  &ukmPresence - specifies the requirements to define the UKM field
--  &smimeCaps - contains the object describing how the S/MIME
--              capabilities are presented.
--
--  Example:
--  kaa-dh-static-ephemeral KEY-AGREE ::= {
--      IDENTIFIER id-alg-ESDH
--      PARAMS TYPE KeyWrapAlgorithm ARE required
--      PUBLIC-KEYS {
--         {IDENTIFIER dh-public-number KEY DHPublicKey
--            PARAMS TYPE DHDomainParameters ARE inheritable }
```

```
--      }
--      - - UKM should be present but is not separately ASN.1-encoded
--      UKM ARE preferredPresent
--  }

KEY-AGREE ::= CLASS {
    &id             OBJECT IDENTIFIER UNIQUE,
    &Params         OPTIONAL,
    &paramPresence  ParamOptions DEFAULT absent,
    &PublicKeySet   PUBLIC-KEY OPTIONAL,
    &Ukm            OPTIONAL,
    &ukmPresence    ParamOptions DEFAULT absent,
    &smimeCaps      SMIME-CAPS OPTIONAL
} WITH SYNTAX {
    IDENTIFIER &id
    [PARAMS [TYPE &Params] ARE &paramPresence]
    [PUBLIC-KEYS &PublicKeySet]
    [UKM [TYPE &Ukm] ARE &ukmPresence]
    [SMIME-CAPS &smimeCaps]
}

--  KEY-WRAP
--
--  Describes the basic properties of a key wrap algorithm
--
--  &id - contains the OID identifying the key wrap algorithm
--  &Params - if present, contains the type for the algorithm
--               parameters; if absent, implies no parameters
--  &paramPresence - parameter presence requirement
--  &smimeCaps - contains the object describing how the S/MIME
--              capabilities are presented.
--
--  Example:
--  kwa-cms3DESwrap KEY-WRAP ::= {
--      IDENTIFIER id-alg-CMS3DESwrap
--      PARAMS TYPE NULL ARE required
--  }

KEY-WRAP ::= CLASS {
    &id                OBJECT IDENTIFIER UNIQUE,
    &Params            OPTIONAL,
    &paramPresence     ParamOptions DEFAULT absent,
    &smimeCaps         SMIME-CAPS OPTIONAL
} WITH SYNTAX {
    IDENTIFIER &id
    [PARAMS [TYPE &Params] ARE &paramPresence]
    [SMIME-CAPS &smimeCaps]
}
```

```
--  KEY-DERIVATION
--
--  Describes the basic properties of a key derivation algorithm
--
--  &id - contains the OID identifying the key derivation algorithm
--  &Params - if present, contains the type for the algorithm
--               parameters; if absent, implies no parameters
--  &paramPresence - parameter presence requirement
--  &smimeCaps - contains the object describing how the S/MIME
--              capabilities are presented.
--
--  Example:
--  kda-pbkdf2 KEY-DERIVATION ::= {
--      IDENTIFIER id-PBKDF2
--      PARAMS TYPE PBKDF2-params ARE required
--  }

KEY-DERIVATION ::= CLASS {
    &id                OBJECT IDENTIFIER UNIQUE,
    &Params            OPTIONAL,
    &paramPresence     ParamOptions DEFAULT absent,
    &smimeCaps         SMIME-CAPS OPTIONAL
} WITH SYNTAX {
    IDENTIFIER &id
    [PARAMS [TYPE &Params] ARE &paramPresence]
    [SMIME-CAPS &smimeCaps]
}

-- MAC-ALGORITHM
--
--  Describes the basic properties of a message
--      authentication code (MAC) algorithm
--
--  &id - contains the OID identifying the MAC algorithm
--  &Params - if present, contains the type for the algorithm
--               parameters; if absent, implies no parameters
--  &paramPresence - parameter presence requirement
--  &keyed - MAC algorithm is a keyed MAC algorithm
--  &smimeCaps - contains the object describing how the S/MIME
--              capabilities are presented.
--
--  Some parameters that perhaps should have been added would be
--  fields with the minimum and maximum MAC lengths for
--  those MAC algorithms that allow truncations.
--
--  Example:
--  maca-hmac-sha1 MAC-ALGORITHM ::= {
--      IDENTIFIER hMAC-SHA1
```

```
--      PARAMS TYPE NULL ARE preferredAbsent
--      IS KEYED MAC TRUE
--      SMIME-CAPS {IDENTIFIED BY hMAC-SHA1}
--  }

MAC-ALGORITHM ::= CLASS {
    &id                 OBJECT IDENTIFIER UNIQUE,
    &Params             OPTIONAL,
    &paramPresence      ParamOptions DEFAULT absent,
    &keyed              BOOLEAN,
    &smimeCaps          SMIME-CAPS OPTIONAL
} WITH SYNTAX {
    IDENTIFIER &id
    [PARAMS [TYPE &Params] ARE &paramPresence]
    IS-KEYED-MAC &keyed
    [SMIME-CAPS &smimeCaps]
}

--  CONTENT-ENCRYPTION
--
--  Describes the basic properties of a content encryption
--      algorithm
--
--  &id - contains the OID identifying the content
--        encryption algorithm
--  &Params - if present, contains the type for the algorithm
--               parameters; if absent, implies no parameters
--  &paramPresence - parameter presence requirement
--  &smimeCaps - contains the object describing how the S/MIME
--              capabilities are presented.
--
--  Example:
--  cea-3DES-cbc CONTENT-ENCRYPTION ::= {
--      IDENTIFIER des-ede3-cbc
--      PARAMS TYPE IV ARE required
--      SMIME-CAPS { IDENTIFIED BY des-ede3-cbc }
--  }

CONTENT-ENCRYPTION ::= CLASS {
    &id                 OBJECT IDENTIFIER UNIQUE,
    &Params             OPTIONAL,
    &paramPresence      ParamOptions DEFAULT absent,
    &smimeCaps          SMIME-CAPS OPTIONAL
} WITH SYNTAX {
    IDENTIFIER &id
    [PARAMS [TYPE &Params] ARE &paramPresence]
    [SMIME-CAPS &smimeCaps]
}
```

```
-- ALGORITHM
--
-- Describes a generic algorithm identifier
--
--  &id - contains the OID identifying the algorithm
--  &Params - if present, contains the type for the algorithm
--              parameters; if absent, implies no parameters
--  &paramPresence - parameter presence requirement
--  &smimeCaps - contains the object describing how the S/MIME
--             capabilities are presented.
--
--  This would be used for cases where an algorithm of an unknown
--  type is used.  In general however, one should either define
--  a more complete algorithm structure (such as the one above)
--  or use the TYPE-IDENTIFIER class.

ALGORITHM ::= CLASS {
    &id OBJECT   IDENTIFIER UNIQUE,
    &Params      OPTIONAL,
    &paramPresence ParamOptions DEFAULT absent,
    &smimeCaps   SMIME-CAPS OPTIONAL
} WITH SYNTAX {
    IDENTIFIER &id
    [PARAMS [TYPE &Params] ARE &paramPresence]
    [SMIME-CAPS &smimeCaps]
}

-- AlgorithmIdentifier
--
-- Provides the generic structure that is used to encode algorithm
--    identification and the parameters associated with the
--    algorithm.
--
-- The first parameter represents the type of the algorithm being
--    used.
-- The second parameter represents an object set containing the
--    algorithms that may occur in this situation.
--    The initial list of required algorithms should occur to the
--      left of an extension marker; all other algorithms should
--      occur to the right of an extension marker.
--
-- The object class ALGORITHM can be used for generic unspecified
--     items.
-- If new ALGORITHM classes are defined, the fields &id and &Params
--     need to be present as fields in the object in order to use
--     this parameterized type.
--
-- Example:
```

```
--    SignatureAlgorithmIdentifier ::=
--       AlgorithmIdentifier{SIGNATURE-ALGORITHM, {SignatureAlgSet}}

AlgorithmIdentifier{ALGORITHM-TYPE, ALGORITHM-TYPE:AlgorithmSet} ::=
        SEQUENCE {
            algorithm   ALGORITHM-TYPE.&id({AlgorithmSet}),
            parameters  ALGORITHM-TYPE.
                   &Params({AlgorithmSet}{@algorithm}) OPTIONAL
        }

--  S/MIME Capabilities
--
--  We have moved the SMIME-CAPS from the module for RFC 3851 to here
--  because it is used in RFC 4262 (X.509 Certificate Extension for
--  S/MIME Capabilities)
--
--
--  This class is used to represent an S/MIME capability.  S/MIME
--  capabilities are used to represent what algorithm capabilities
--  an individual has.  The classic example was the content encryption
--  algorithm RC2 where the algorithm id and the RC2 key lengths
--  supported needed to be advertised, but the IV used is not fixed.
--  Thus, for RC2 we used
--
--  cap-RC2CBC SMIME-CAPS ::= {
--      TYPE INTEGER ( 40 | 128 ) IDENTIFIED BY rc2-cbc }
--
--  where 40 and 128 represent the RC2 key length in number of bits.
--
--  Another example where information needs to be shown is for
--  RSA-OAEP where only specific hash functions or mask generation
--  functions are supported, but the saltLength is specified by the
--  sender and not the recipient.  In this case, one can either
--  generate a number of capability items,
--  or a new S/MIME capability type could be generated where
--  multiple hash functions could be specified.
--
--
--  SMIME-CAP
--
--  This class is used to associate the type that describes the
--  capabilities with the object identifier.
--

SMIME-CAPS ::= CLASS {
    &id         OBJECT IDENTIFIER UNIQUE,
    &Type       OPTIONAL
}
```

```
WITH SYNTAX { [TYPE &Type] IDENTIFIED BY &id }

--
--  Generic type - this is used for defining values.
--

--  Define a single S/MIME capability encoding

SMIMECapability{SMIME-CAPS:CapabilitySet} ::= SEQUENCE {
    capabilityID        SMIME-CAPS.&id({CapabilitySet}),
    parameters          SMIME-CAPS.&Type({CapabilitySet}
                            {@capabilityID}) OPTIONAL
}

--  Define a sequence of S/MIME capability values

SMIMECapabilities { SMIME-CAPS:CapabilitySet } ::=
        SEQUENCE SIZE (1..MAX) OF SMIMECapability{{CapabilitySet} }

END
```

4. ASN.1 Module for RFC 2560

```
  OCSP-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-ocsp-02(48)}
  DEFINITIONS EXPLICIT TAGS ::=
  BEGIN
  IMPORTS

  Extensions{}, EXTENSION, ATTRIBUTE
  FROM PKIX-CommonTypes-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-pkixCommon-02(57)}

  AlgorithmIdentifier{}, DIGEST-ALGORITHM, SIGNATURE-ALGORITHM
  FROM AlgorithmInformation-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0)
      id-mod-algorithmInformation-02(58)}

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

  Name, CertificateSerialNumber, id-kp, id-ad-ocsp, Certificate
  FROM PKIX1Explicit-2009
```

```
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-explicit-02(51)}

  sa-dsaWithSHA1, sa-rsaWithMD2, sa-rsaWithMD5, sa-rsaWithSHA1
  FROM PKIXAlgs-2009
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0)
    id-mod-pkix1-algorithms2008-02(56)};

  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 {{re-ocsp-nonce |
                                re-ocsp-response, ...}} OPTIONAL }

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

  Version  ::=  INTEGER  {  v1(0) }

  Request ::=     SEQUENCE {
    reqCert                     CertID,
    singleRequestExtensions     [0] EXPLICIT Extensions
                                        { {re-ocsp-service-locator,
                                                ...}} OPTIONAL }

  CertID ::= SEQUENCE {
    hashAlgorithm            AlgorithmIdentifier
                                 {DIGEST-ALGORITHM, {...}},
    issuerNameHash     OCTET STRING, -- Hash of Issuer's DN
    issuerKeyHash      OCTET STRING, -- Hash of Issuer's 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
  }

  RESPONSE ::= TYPE-IDENTIFIER

  ResponseSet RESPONSE ::= {basicResponse, ...}

  ResponseBytes ::=       SEQUENCE {
      responseType        RESPONSE.
                              &id ({ResponseSet}),
      response            OCTET STRING (CONTAINING RESPONSE.
                              &Type({ResponseSet}{@responseType}))}

  basicResponse RESPONSE ::=
      { BasicOCSPResponse IDENTIFIED BY id-pkix-ocsp-basic }

  BasicOCSPResponse       ::= SEQUENCE {
     tbsResponseData      ResponseData,
     signatureAlgorithm   AlgorithmIdentifier{SIGNATURE-ALGORITHM,
                              {sa-dsaWithSHA1 | sa-rsaWithSHA1 |
                                   sa-rsaWithMD5 | sa-rsaWithMD2, ...}},
     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
                                  {{re-ocsp-nonce, ...}} 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{{re-ocsp-crl |
                                                 re-ocsp-archive-cutoff |
                                                  CrlEntryExtensions, ...}
                                                 } 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

  CRLReason ::= INTEGER

  ArchiveCutoff ::= GeneralizedTime

  AcceptableResponses ::= SEQUENCE OF RESPONSE.&id({ResponseSet})

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

  CrlID ::= SEQUENCE {
      crlUrl               [0]     EXPLICIT IA5String OPTIONAL,
      crlNum               [1]     EXPLICIT INTEGER OPTIONAL,
      crlTime              [2]     EXPLICIT GeneralizedTime OPTIONAL }

  --  Request Extensions

  re-ocsp-nonce EXTENSION ::= { SYNTAX OCTET STRING IDENTIFIED
                                   BY id-pkix-ocsp-nonce }
  re-ocsp-response EXTENSION ::= { SYNTAX AcceptableResponses IDENTIFIED
                                      BY id-pkix-ocsp-response }
  re-ocsp-service-locator EXTENSION ::= { SYNTAX ServiceLocator
                                          IDENTIFIED BY
                                          id-pkix-ocsp-service-locator }

  --  Response Extensions

  re-ocsp-crl EXTENSION ::= { SYNTAX CrlID IDENTIFIED BY
                                  id-pkix-ocsp-crl }
  re-ocsp-archive-cutoff EXTENSION ::= { SYNTAX ArchiveCutoff
                                         IDENTIFIED BY
                                         id-pkix-ocsp-archive-cutoff }
```

```
  -- 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
```

5. ASN.1 Module for RFC 2986

```
  PKCS-10
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
        mechanisms(5) pkix(7) id-mod(0) id-mod-pkcs10-2009(69)}
  DEFINITIONS IMPLICIT TAGS ::=
  BEGIN
  IMPORTS

  AlgorithmIdentifier{}, DIGEST-ALGORITHM, SIGNATURE-ALGORITHM,
      PUBLIC-KEY
  FROM AlgorithmInformation-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0)
      id-mod-algorithmInformation-02(58)}

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

  -- Certificate requests
  CertificationRequestInfo ::= SEQUENCE {
      version       INTEGER { v1(0) } (v1, ... ),
      subject       Name,
      subjectPKInfo SubjectPublicKeyInfo{{ PKInfoAlgorithms }},
      attributes    [0] Attributes{{ CRIAttributes }}
  }

  SubjectPublicKeyInfo {PUBLIC-KEY: IOSet} ::= SEQUENCE {
      algorithm        AlgorithmIdentifier {PUBLIC-KEY, {IOSet}},
      subjectPublicKey BIT STRING
  }
```

```
  PKInfoAlgorithms PUBLIC-KEY ::= {
      ... -- 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{SIGNATURE-ALGORITHM,
                                    { SignatureAlgorithms }},
      signature                 BIT STRING
  }

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

  END
```

6. ASN.1 Module for RFC 3279

Note that this module also contains information from RFC 5480 [RFC5480].

```
PKIXAlgs-2009 { iso(1) identified-organization(3) dod(6)
  internet(1) security(5) mechanisms(5) pkix(7) id-mod(0)
  id-mod-pkix1-algorithms2008-02(56) }

DEFINITIONS EXPLICIT TAGS ::=
BEGIN
IMPORTS

PUBLIC-KEY, SIGNATURE-ALGORITHM, DIGEST-ALGORITHM, SMIME-CAPS
FROM AlgorithmInformation-2009
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0)
    id-mod-algorithmInformation-02(58)}

mda-sha224, mda-sha256, mda-sha384, mda-sha512
FROM PKIX1-PSS-OAEP-Algorithms-2009
    {iso(1) identified-organization(3) dod(6) internet(1)
    security(5) mechanisms(5) pkix(7) id-mod(0)
```

```
       id-mod-pkix1-rsa-pkalgs-02(54)} ;

   --
   -- Public Key (pk-) Algorithms
   --

   PublicKeys PUBLIC-KEY ::= {
    pk-rsa  |
    pk-dsa  |
    pk-dh   |
    pk-kea,
    ...,
    pk-ec   |
    pk-ecDH |
    pk-ecMQV
   }

   --
   -- Signature Algorithms (sa-)
   --

   SignatureAlgs SIGNATURE-ALGORITHM ::= {
    sa-rsaWithMD2      |
    sa-rsaWithMD5      |
    sa-rsaWithSHA1     |
    sa-dsaWithSHA1     |
    sa-ecdsaWithSHA1,
    ..., -- Extensible
    sa-dsaWithSHA224   |
    sa-dsaWithSHA256   |
    sa-ecdsaWithSHA224 |
    sa-ecdsaWithSHA256 |
    sa-ecdsaWithSHA384 |
    sa-ecdsaWithSHA512
   }

   --
   -- S/MIME CAPS for algorithms in this document
   --
   -- For all of the algorithms laid out in this document, the
   -- parameters field for the S/MIME capabilities is defined as
   -- ABSENT as there are no specific values that need to be known
   -- by the receiver for negotiation.

   --

   SMimeCaps SMIME-CAPS ::= {
    sa-rsaWithMD2.&smimeCaps      |
```

```
    sa-rsaWithMD5.&smimeCaps      |
    sa-rsaWithSHA1.&smimeCaps     |
    sa-dsaWithSHA1.&smimeCaps     |
    sa-dsaWithSHA224.&smimeCaps   |
    sa-dsaWithSHA256.&smimeCaps   |
    sa-ecdsaWithSHA1.&smimeCaps   |
    sa-ecdsaWithSHA224.&smimeCaps |
    sa-ecdsaWithSHA256.&smimeCaps |
    sa-ecdsaWithSHA384.&smimeCaps |
    sa-ecdsaWithSHA512.&smimeCaps,
    ... }

   -- RSA PK Algorithm, Parameters, and Keys

   pk-rsa PUBLIC-KEY ::= {
    IDENTIFIER rsaEncryption
    KEY RSAPublicKey
    PARAMS TYPE NULL ARE absent
    -- Private key format not in this module --
    CERT-KEY-USAGE {digitalSignature, nonRepudiation,
    keyEncipherment, dataEncipherment, keyCertSign, cRLSign}
   }

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

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

   -- DSA PK Algorithm, Parameters, and Keys

   pk-dsa PUBLIC-KEY ::= {
    IDENTIFIER id-dsa
    KEY DSAPublicKey
    PARAMS TYPE DSA-Params ARE inheritable
    -- Private key format not in this module --
    CERT-KEY-USAGE { digitalSignature, nonRepudiation, keyCertSign,
                         cRLSign }
   }

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

   DSA-Params ::= SEQUENCE {
    p  INTEGER,
```

```
    q  INTEGER,
    g  INTEGER
   }

   DSAPublicKey ::= INTEGER --  public key, y

   -- Diffie-Hellman PK Algorithm, Parameters, and Keys

   pk-dh PUBLIC-KEY ::= {
    IDENTIFIER dhpublicnumber
    KEY DHPublicKey
    PARAMS TYPE DomainParameters ARE inheritable
    -- Private key format not in this module --
    CERT-KEY-USAGE {keyAgreement, encipherOnly, decipherOnly }
   }

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

   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
    validationParams  ValidationParams OPTIONAL
   }

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

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

   -- KEA PK Algorithm and Parameters

   pk-kea PUBLIC-KEY ::= {
    IDENTIFIER id-keyExchangeAlgorithm
    -- key is not encoded --
    PARAMS TYPE KEA-Params-Id ARE required
    -- Private key format not in this module --
    CERT-KEY-USAGE {keyAgreement, encipherOnly, decipherOnly }
   }
   id-keyExchangeAlgorithm OBJECT IDENTIFIER ::= {
       joint-iso-itu-t(2) country(16) us(840) organization(1)
       gov(101) dod(2) infosec(1) algorithms(1) 22 }
```

```
   KEA-Params-Id ::= OCTET STRING

   -- Elliptic Curve (EC) Signatures: Unrestricted Algorithms
   --  (Section 2.1.1 of RFC 5480)
   --
   -- EC Unrestricted Algorithm ID -- -- this is used for ECDSA

   pk-ec PUBLIC-KEY ::= {
    IDENTIFIER id-ecPublicKey
    KEY ECPoint
    PARAMS TYPE ECParameters ARE required
    -- Private key format not in this module --
    CERT-KEY-USAGE { digitalSignature, nonRepudiation, keyAgreement,
                         keyCertSign, cRLSign }
   }

   ECPoint ::= OCTET STRING -- see RFC 5480 for syntax and restrictions

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

   -- Elliptic Curve (EC) Signatures: Restricted Algorithms
   --  (Section 2.1.2 of RFC 5480)
   --
   -- EC Diffie-Hellman Algorithm ID

   pk-ecDH PUBLIC-KEY ::= {
    IDENTIFIER id-ecDH
    KEY ECPoint
    PARAMS TYPE ECParameters ARE required
    -- Private key format not in this module --
    CERT-KEY-USAGE { keyAgreement, encipherOnly, decipherOnly }
   }

   id-ecDH OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) schemes(1)
    ecdh(12) }

   -- EC Menezes-Qu-Vanstone Algorithm ID

   pk-ecMQV PUBLIC-KEY ::= {
    IDENTIFIER id-ecMQV
    KEY ECPoint
    PARAMS TYPE ECParameters ARE required
    -- Private key format not in this module --
    CERT-KEY-USAGE { keyAgreement, encipherOnly, decipherOnly }
   }
```

```
   id-ecMQV OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) schemes(1)
    ecmqv(13) }

   -- Parameters and Keys for both Restricted and Unrestricted EC

   ECParameters ::= CHOICE {
    namedCurve      CURVE.&id({NamedCurve})
    -- implicitCurve   NULL
      -- implicitCurve MUST NOT be used in PKIX
    -- specifiedCurve  SpecifiedCurve
      -- specifiedCurve MUST NOT be used in PKIX
      -- Details for specifiedCurve can be found in [X9.62]
      -- Any future additions to this CHOICE should be coordinated
      -- with ANSI X.9.
   }
   -- If you need to be able to decode ANSI X.9 parameter structures,
   -- uncomment the implicitCurve and specifiedCurve above, and also
   -- uncomment the following:
   --(WITH COMPONENTS {namedCurve PRESENT})

   -- Sec 2.1.1.1 Named Curve

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

   NamedCurve CURVE ::= {
   { ID secp192r1 } | { ID sect163k1 } | { ID sect163r2 } |
   { ID secp224r1 } | { ID sect233k1 } | { ID sect233r1 } |
   { ID secp256r1 } | { ID sect283k1 } | { ID sect283r1 } |
   { ID secp384r1 } | { ID sect409k1 } | { ID sect409r1 } |
   { ID secp521r1 } | { ID sect571k1 } | { ID sect571r1 },
   ... -- Extensible
   }

   -- Note in [X9.62] the curves are referred to as 'ansiX9' as
   -- opposed to 'sec'.  For example, secp192r1 is the same curve as
   -- ansix9p192r1.

   -- Note that in [PKI-ALG] the secp192r1 curve was referred to as
   -- prime192v1 and the secp256r1 curve was referred to as
   -- prime256v1.

   -- Note that [FIPS186-3] refers to secp192r1 as P-192,
   -- secp224r1 as P-224, secp256r1 as P-256, secp384r1 as P-384,
   -- and secp521r1 as P-521.

   secp192r1 OBJECT IDENTIFIER ::= {
```

```
    iso(1) member-body(2) us(840) ansi-X9-62(10045) curves(3)
    prime(1) 1 }

   sect163k1 OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) curve(0) 1 }

   sect163r2 OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) curve(0) 15 }

   secp224r1 OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) curve(0) 33 }

   sect233k1 OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) curve(0) 26 }

   sect233r1 OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) curve(0) 27 }

   secp256r1 OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) ansi-X9-62(10045) curves(3)
    prime(1) 7 }

   sect283k1 OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) curve(0) 16 }

   sect283r1 OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) curve(0) 17 }

   secp384r1 OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) curve(0) 34 }

   sect409k1 OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) curve(0) 36 }

   sect409r1 OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) curve(0) 37 }

   secp521r1 OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) curve(0) 35 }

   sect571k1 OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) curve(0) 38 }

   sect571r1 OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) certicom(132) curve(0) 39 }

   -- RSA with MD-2
```

```
   sa-rsaWithMD2 SIGNATURE-ALGORITHM ::= {
    IDENTIFIER md2WithRSAEncryption
    PARAMS TYPE NULL ARE required
    HASHES { mda-md2 }
    PUBLIC-KEYS { pk-rsa }
    SMIME-CAPS { IDENTIFIED BY md2WithRSAEncryption }
   }

   md2WithRSAEncryption OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
    pkcs-1(1) 2 }

   -- RSA with MD-5

   sa-rsaWithMD5 SIGNATURE-ALGORITHM ::= {
    IDENTIFIER md5WithRSAEncryption
    PARAMS TYPE NULL ARE required
    HASHES { mda-md5 }
    PUBLIC-KEYS { pk-rsa }
    SMIME-CAPS { IDENTIFIED BY md5WithRSAEncryption }
   }

   md5WithRSAEncryption OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
    pkcs-1(1) 4 }

   -- RSA with SHA-1

   sa-rsaWithSHA1 SIGNATURE-ALGORITHM ::= {
    IDENTIFIER sha1WithRSAEncryption
    PARAMS TYPE NULL ARE required
    HASHES { mda-sha1 }
    PUBLIC-KEYS { pk-rsa }
    SMIME-CAPS {IDENTIFIED BY sha1WithRSAEncryption }
   }

   sha1WithRSAEncryption OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
    pkcs-1(1) 5 }

   -- DSA with SHA-1

   sa-dsaWithSHA1 SIGNATURE-ALGORITHM ::= {
    IDENTIFIER dsa-with-sha1
    VALUE DSA-Sig-Value
    PARAMS TYPE NULL ARE absent
    HASHES { mda-sha1 }
    PUBLIC-KEYS { pk-dsa }
```

```
    SMIME-CAPS { IDENTIFIED BY dsa-with-sha1 }
   }

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

   -- DSA with SHA-224

   sa-dsaWithSHA224 SIGNATURE-ALGORITHM ::= {
    IDENTIFIER dsa-with-sha224
    VALUE DSA-Sig-Value
    PARAMS TYPE NULL ARE absent
    HASHES { mda-sha224 }
    PUBLIC-KEYS { pk-dsa }
    SMIME-CAPS { IDENTIFIED BY dsa-with-sha224 }
   }

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

   -- DSA with SHA-256

   sa-dsaWithSHA256 SIGNATURE-ALGORITHM ::= {
    IDENTIFIER dsa-with-sha256
    VALUE DSA-Sig-Value
    PARAMS TYPE NULL ARE absent
    HASHES { mda-sha256 }
    PUBLIC-KEYS { pk-dsa }
    SMIME-CAPS { IDENTIFIED BY dsa-with-sha256 }
   }

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

   -- ECDSA with SHA-1

   sa-ecdsaWithSHA1 SIGNATURE-ALGORITHM ::= {
    IDENTIFIER ecdsa-with-SHA1
    VALUE ECDSA-Sig-Value
    PARAMS TYPE NULL ARE absent
    HASHES { mda-sha1 }
    PUBLIC-KEYS { pk-ec }
    SMIME-CAPS {IDENTIFIED BY ecdsa-with-SHA1 }
   }

   ecdsa-with-SHA1 OBJECT IDENTIFIER ::= {
```

```
    iso(1) member-body(2) us(840) ansi-X9-62(10045)
    signatures(4) 1 }

   -- ECDSA with SHA-224

   sa-ecdsaWithSHA224 SIGNATURE-ALGORITHM ::= {
    IDENTIFIER ecdsa-with-SHA224
    VALUE ECDSA-Sig-Value
    PARAMS TYPE NULL ARE absent
    HASHES { mda-sha224 }
    PUBLIC-KEYS { pk-ec }
    SMIME-CAPS { IDENTIFIED BY ecdsa-with-SHA224 }
   }

   ecdsa-with-SHA224 OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) ansi-X9-62(10045) signatures(4)
    ecdsa-with-SHA2(3) 1 }

   -- ECDSA with SHA-256

   sa-ecdsaWithSHA256 SIGNATURE-ALGORITHM ::= {
    IDENTIFIER ecdsa-with-SHA256
    VALUE ECDSA-Sig-Value
    PARAMS TYPE NULL ARE absent
    HASHES { mda-sha256 }
    PUBLIC-KEYS { pk-ec }
    SMIME-CAPS { IDENTIFIED BY ecdsa-with-SHA256 }
   }

   ecdsa-with-SHA256 OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) ansi-X9-62(10045) signatures(4)
    ecdsa-with-SHA2(3) 2 }

   -- ECDSA with SHA-384

   sa-ecdsaWithSHA384 SIGNATURE-ALGORITHM ::= {
    IDENTIFIER ecdsa-with-SHA384
    VALUE ECDSA-Sig-Value
    PARAMS TYPE NULL ARE absent
    HASHES { mda-sha384 }
    PUBLIC-KEYS { pk-ec }
    SMIME-CAPS { IDENTIFIED BY ecdsa-with-SHA384 }
   }
   ecdsa-with-SHA384 OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) ansi-X9-62(10045) signatures(4)
    ecdsa-with-SHA2(3) 3 }

   -- ECDSA with SHA-512
```

```
   sa-ecdsaWithSHA512 SIGNATURE-ALGORITHM ::= {
    IDENTIFIER ecdsa-with-SHA512
    VALUE ECDSA-Sig-Value
    PARAMS TYPE NULL ARE absent
    HASHES { mda-sha512 }
    PUBLIC-KEYS { pk-ec }
    SMIME-CAPS { IDENTIFIED BY ecdsa-with-SHA512 }
   }

   ecdsa-with-SHA512 OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) ansi-X9-62(10045) signatures(4)
    ecdsa-with-SHA2(3) 4 }

   --
   -- Signature Values
   --

   -- DSA

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

   -- ECDSA

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

   --
   -- Message Digest Algorithms (mda-)
   --

   HashAlgs DIGEST-ALGORITHM ::= {
    mda-md2    |
    mda-md5    |
    mda-sha1,
    ... -- Extensible
   }
   -- MD-2

   mda-md2 DIGEST-ALGORITHM ::= {
    IDENTIFIER id-md2
    PARAMS TYPE NULL ARE preferredAbsent
   }
```

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

   -- MD-5

   mda-md5 DIGEST-ALGORITHM ::= {
    IDENTIFIER id-md5
    PARAMS TYPE NULL ARE preferredAbsent
   }

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

   -- SHA-1

   mda-sha1 DIGEST-ALGORITHM ::= {
    IDENTIFIER id-sha1
    PARAMS TYPE NULL ARE preferredAbsent
   }

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

   END
```

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

```
  AttributeCertificateVersion1-2009
      {iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
      smime(16) modules(0) id-mod-v1AttrCert-02(49)}
  DEFINITIONS EXPLICIT TAGS ::=
  BEGIN
  IMPORTS

  SIGNATURE-ALGORITHM, ALGORITHM, AlgorithmIdentifier{}
  FROM AlgorithmInformation-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0)
      id-mod-algorithmInformation-02(58)}

  AttributeSet{}, Extensions{}, EXTENSION, ATTRIBUTE
  FROM PKIX-CommonTypes-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-pkixCommon-02(57) }
```

```
  CertificateSerialNumber, UniqueIdentifier, SIGNED{}
  FROM PKIX1Explicit-2009
      { iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-explicit-02(51) }

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

  AttCertValidityPeriod, IssuerSerial
  FROM PKIXAttributeCertificate-2009
      { iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-attribute-cert-02(47) } ;

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

  AttributeCertificateV1 ::= SIGNED{AttributeCertificateInfoV1}

  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{SIGNATURE-ALGORITHM, {...}},
     serialNumber CertificateSerialNumber,
     attCertValidityPeriod AttCertValidityPeriod,
     attributes SEQUENCE OF AttributeSet{{AttrList}},
     issuerUniqueID UniqueIdentifier OPTIONAL,
     extensions Extensions{{AttributeCertExtensionsV1}} OPTIONAL }

  AttCertVersionV1 ::= INTEGER { v1(0) }

  AttrList ATTRIBUTE ::= {...}
  AttributeCertExtensionsV1 EXTENSION ::= {...}

  END
```

8. ASN.1 Module for RFC 4055

```
PKIX1-PSS-OAEP-Algorithms-2009
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-rsa-pkalgs-02(54)}
DEFINITIONS EXPLICIT TAGS ::=
BEGIN
IMPORTS

AlgorithmIdentifier{}, ALGORITHM, DIGEST-ALGORITHM, KEY-TRANSPORT,
    SIGNATURE-ALGORITHM, PUBLIC-KEY, SMIME-CAPS
FROM AlgorithmInformation-2009
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0)
    id-mod-algorithmInformation-02(58)}

id-sha1, mda-sha1, pk-rsa, RSAPublicKey
FROM PKIXAlgs-2009
   {iso(1) identified-organization(3) dod(6) internet(1) security(5)
   mechanisms(5) pkix(7) id-mod(0)
   id-mod-pkix1-algorithms2008-02(56)};

-- ============================
--    Object Set exports
-- ============================
--
--  Define top-level symbols with all of the objects defined for
--  export to other modules.  These objects would be included as part
--  of an Object Set to restrict the set of legal values.
--

PublicKeys PUBLIC-KEY ::= { pk-rsaSSA-PSS | pk-rsaES-OAEP, ... }
SignatureAlgs SIGNATURE-ALGORITHM ::= { sa-rsaSSA-PSS, ...}
KeyTransportAlgs KEY-TRANSPORT ::= { kta-rsaES-OAEP, ... }
HashAlgs DIGEST-ALGORITHM ::= { mda-sha224 | mda-sha256 | mda-sha384
                                | mda-sha512, ... }
SMimeCaps SMIME-CAPS ::= {
    sa-rsaSSA-PSS.&smimeCaps |
    kta-rsaES-OAEP.&smimeCaps,
    ...
}

-- ============================
--    Algorithm Objects
-- ============================

--
-- Public key object for PSS signatures
```

```
   --

   pk-rsaSSA-PSS PUBLIC-KEY ::= {
       IDENTIFIER id-RSASSA-PSS
       KEY RSAPublicKey
       PARAMS TYPE RSASSA-PSS-params ARE optional
        -- Private key format not in this module --
       CERT-KEY-USAGE { nonRepudiation, digitalSignature,
                            keyCertSign, cRLSign }
   }

   --
   --  Signature algorithm definition for PSS signatures
   --

   sa-rsaSSA-PSS SIGNATURE-ALGORITHM ::= {
       IDENTIFIER id-RSASSA-PSS
       PARAMS TYPE RSASSA-PSS-params ARE required
       HASHES { mda-sha1 | mda-sha224 | mda-sha256 | mda-sha384
                    | mda-sha512 }
       PUBLIC-KEYS { pk-rsa | pk-rsaSSA-PSS }
       SMIME-CAPS { IDENTIFIED BY id-RSASSA-PSS }
   }

   --
   --  Signature algorithm definitions for PKCS v1.5 signatures
   --

   sa-sha224WithRSAEncryption SIGNATURE-ALGORITHM ::= {
       IDENTIFIER sha224WithRSAEncryption
       PARAMS TYPE NULL ARE required
       HASHES { mda-sha224 }
       PUBLIC-KEYS { pk-rsa }
       SMIME-CAPS { IDENTIFIED BY sha224WithRSAEncryption }
   }
   sha224WithRSAEncryption  OBJECT IDENTIFIER  ::=  { pkcs-1 14 }

   sa-sha256WithRSAEncryption SIGNATURE-ALGORITHM ::= {
       IDENTIFIER sha256WithRSAEncryption
       PARAMS TYPE NULL ARE required
       HASHES { mda-sha256 }
       PUBLIC-KEYS { pk-rsa }
       SMIME-CAPS { IDENTIFIED BY sha256WithRSAEncryption }
   }
   sha256WithRSAEncryption  OBJECT IDENTIFIER  ::=  { pkcs-1 11 }

   sa-sha384WithRSAEncryption SIGNATURE-ALGORITHM ::= {
       IDENTIFIER sha384WithRSAEncryption
```

```
        PARAMS TYPE NULL ARE required
        HASHES { mda-sha384 }
        PUBLIC-KEYS { pk-rsa }
        SMIME-CAPS { IDENTIFIED BY sha384WithRSAEncryption }
    }
    sha384WithRSAEncryption  OBJECT IDENTIFIER  ::=  { pkcs-1 12 }

    sa-sha512WithRSAEncryption SIGNATURE-ALGORITHM ::= {
        IDENTIFIER sha512WithRSAEncryption
        PARAMS TYPE NULL ARE required
        HASHES { mda-sha512 }
        PUBLIC-KEYS { pk-rsa }
        SMIME-CAPS { IDENTIFIED BY sha512WithRSAEncryption }
    }
    sha512WithRSAEncryption  OBJECT IDENTIFIER  ::=  { pkcs-1 13 }

    --
    --  Public key definition for OAEP encryption
    --

    pk-rsaES-OAEP PUBLIC-KEY ::= {
        IDENTIFIER id-RSAES-OAEP
        KEY RSAPublicKey
        PARAMS TYPE RSAES-OAEP-params ARE optional
         -- Private key format not in this module --
        CERT-KEY-USAGE {keyEncipherment, dataEncipherment}
    }

    --
    --  Key transport key lock definition for OAEP encryption
    --

    kta-rsaES-OAEP KEY-TRANSPORT ::= {
        IDENTIFIER id-RSAES-OAEP
        PARAMS TYPE RSAES-OAEP-params ARE required
        PUBLIC-KEYS { pk-rsa | pk-rsaES-OAEP }
        SMIME-CAPS { TYPE RSAES-OAEP-params IDENTIFIED BY id-RSAES-OAEP}
    }
    -- ============================
    --   Basic object identifiers
    -- ============================

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

    -- When rsaEncryption is used in an AlgorithmIdentifier, the
    -- parameters MUST be present and MUST be NULL.
```

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

   -- When id-RSAES-OAEP is used in an AlgorithmIdentifier,
   -- and the parameters field is present, it MUST be
   -- RSAES-OAEP-params.

   id-RSAES-OAEP  OBJECT IDENTIFIER  ::=  { pkcs-1 7 }

   -- When id-mgf1 is used in an AlgorithmIdentifier, the parameters
   -- MUST be present and MUST be a HashAlgorithm.

   id-mgf1  OBJECT IDENTIFIER  ::=  { pkcs-1 8 }

   -- When id-pSpecified is used in an AlgorithmIdentifier, the
   -- parameters MUST be an OCTET STRING.

   id-pSpecified  OBJECT IDENTIFIER  ::=  { pkcs-1 9 }

   -- When id-RSASSA-PSS is used in an AlgorithmIdentifier, and the
   -- parameters field is present, it MUST be RSASSA-PSS-params.

   id-RSASSA-PSS  OBJECT IDENTIFIER  ::=  { pkcs-1 10 }

   -- When the following OIDs are used in an AlgorithmIdentifier, the
   -- parameters SHOULD be absent, but if the parameters are present,
   -- they MUST be NULL.

   --
   -- id-sha1 is imported from RFC 3279.  Additionally, the v1.5
   -- signature algorithms (i.e., rsaWithSHA256) are now solely placed
   -- in that module.
   --

   id-sha224  OBJECT IDENTIFIER  ::=
       { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101)
       csor(3) nistAlgorithms(4) hashalgs(2) 4 }

   mda-sha224 DIGEST-ALGORITHM ::= {
       IDENTIFIER id-sha224
       PARAMS TYPE NULL ARE preferredAbsent
   }

   id-sha256  OBJECT IDENTIFIER  ::=
       { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101)
       csor(3) nistAlgorithms(4) hashalgs(2) 1 }

   mda-sha256 DIGEST-ALGORITHM ::= {
       IDENTIFIER id-sha256
```

```
    PARAMS TYPE NULL ARE preferredAbsent
}
id-sha384  OBJECT IDENTIFIER  ::=
    { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101)
    csor(3) nistAlgorithms(4) hashalgs(2) 2 }

mda-sha384 DIGEST-ALGORITHM ::= {
    IDENTIFIER id-sha384
    PARAMS TYPE NULL ARE preferredAbsent
}
id-sha512  OBJECT IDENTIFIER  ::=
    { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101)
    csor(3) nistAlgorithms(4) hashalgs(2) 3 }

mda-sha512 DIGEST-ALGORITHM ::= {
    IDENTIFIER id-sha512
    PARAMS TYPE NULL ARE preferredAbsent
}

-- =============
--   Constants
-- =============

EncodingParameters ::= OCTET STRING(SIZE(0..MAX))

nullOctetString  EncodingParameters  ::=  ''H

nullParameters NULL  ::=  NULL

-- ========================
--   Algorithm Identifiers
-- ========================

HashAlgorithm  ::=  AlgorithmIdentifier{DIGEST-ALGORITHM,
                        {HashAlgorithms}}

HashAlgorithms DIGEST-ALGORITHM ::=  {
    { IDENTIFIER id-sha1 PARAMS TYPE NULL ARE preferredPresent } |
    { IDENTIFIER id-sha224 PARAMS TYPE NULL ARE preferredPresent } |
    { IDENTIFIER id-sha256 PARAMS TYPE NULL ARE preferredPresent } |
    { IDENTIFIER id-sha384 PARAMS TYPE NULL ARE preferredPresent } |
    { IDENTIFIER id-sha512 PARAMS TYPE NULL ARE preferredPresent }
}

sha1Identifier HashAlgorithm ::= {
    algorithm id-sha1,
    parameters NULL : NULL
}
```

```
   --
   --  We have a default algorithm - create the value here
   --

   MaskGenAlgorithm ::= AlgorithmIdentifier{ALGORITHM,
                           {PKCS1MGFAlgorithms}}

   mgf1SHA1 MaskGenAlgorithm ::= {
       algorithm id-mgf1,
       parameters HashAlgorithm : sha1Identifier
   }

   --
   --  Define the set of mask generation functions
   --
   --  If the identifier is id-mgf1, any of the listed hash
   --    algorithms may be used.
   --

   PKCS1MGFAlgorithms ALGORITHM ::= {
       { IDENTIFIER id-mgf1 PARAMS TYPE HashAlgorithm ARE required },
       ...
   }

   --
   -- Define the set of known source algorithms for PSS
   --

   PSourceAlgorithm ::= AlgorithmIdentifier{ALGORITHM,
                                               {PSS-SourceAlgorithms}}

   PSS-SourceAlgorithms ALGORITHM ::= {
       { IDENTIFIER id-pSpecified PARAMS TYPE EncodingParameters
             ARE required },
       ...
   }
   pSpecifiedEmpty PSourceAlgorithm ::=  {
       algorithm id-pSpecified,
       parameters EncodingParameters : nullOctetString
   }

   -- ====================
   --   Main structures
   -- ====================

   -- AlgorithmIdentifier parameters for id-RSASSA-PSS.
   -- Note that the tags in this Sequence are explicit.
   -- Note: The hash algorithm in hashAlgorithm and in
```

```
   -- maskGenAlgorithm should be the same.

   RSASSA-PSS-params  ::=  SEQUENCE  {
       hashAlgorithm     [0] HashAlgorithm DEFAULT sha1Identifier,
       maskGenAlgorithm  [1] MaskGenAlgorithm DEFAULT mgf1SHA1,
       saltLength        [2] INTEGER DEFAULT 20,
       trailerField      [3] INTEGER DEFAULT 1
   }

   -- AlgorithmIdentifier parameters for id-RSAES-OAEP.
   -- Note that the tags in this Sequence are explicit.
   -- Note: The hash algorithm in hashFunc and in
   -- maskGenFunc should be the same.

   RSAES-OAEP-params  ::=  SEQUENCE  {
       hashFunc          [0] HashAlgorithm DEFAULT sha1Identifier,
       maskGenFunc       [1] MaskGenAlgorithm DEFAULT mgf1SHA1,
       pSourceFunc       [2] PSourceAlgorithm DEFAULT
                                pSpecifiedEmpty
   }

   END
```

9. ASN.1 Module for RFC 4210

```
 PKIXCMP-2009
     { iso(1) identified-organization(3) dod(6) internet(1) security(5)
     mechanisms(5) pkix(7) id-mod(0) id-mod-cmp2000-02(50) }
 DEFINITIONS EXPLICIT TAGS ::=
 BEGIN
 IMPORTS

 AttributeSet{}, Extensions{}, EXTENSION, ATTRIBUTE
 FROM PKIX-CommonTypes-2009
     {iso(1) identified-organization(3) dod(6) internet(1) security(5)
     mechanisms(5) pkix(7) id-mod(0) id-mod-pkixCommon-02(57)}

 AlgorithmIdentifier{}, SIGNATURE-ALGORITHM, ALGORITHM,
     DIGEST-ALGORITHM, MAC-ALGORITHM
 FROM AlgorithmInformation-2009
     {iso(1) identified-organization(3) dod(6) internet(1) security(5)
     mechanisms(5) pkix(7) id-mod(0)
     id-mod-algorithmInformation-02(58)}

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

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

 CertTemplate, PKIPublicationInfo, EncryptedValue, CertId,
     CertReqMessages
 FROM PKIXCRMF-2009
     { iso(1) identified-organization(3) dod(6) internet(1) security(5)
     mechanisms(5) pkix(7) id-mod(0) id-mod-crmf2005-02(55) }
 -- see also the behavioral clarifications to CRMF codified in
 -- Appendix C of this specification

 CertificationRequest
 FROM PKCS-10
       {iso(1) identified-organization(3) dod(6) internet(1) security(5)
       mechanisms(5) pkix(7) id-mod(0) id-mod-pkcs10-2009(69)}
 -- (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 "uncomment" 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{ALGORITHM, {...}}
                            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
    -- text encoded as UTF-8 String [RFC3629] (note: each
    -- UTF8String MAY include an [RFC3066] language tag
    -- to indicate the language of the contained text;
    -- see [RFC2482] for details)

PKIBody ::= CHOICE {       -- message-specific body elements
    ir       [0]  CertReqMessages,        --Initialization Request
    ip       [1]  CertRepMessage,         --Initialization Response
    cr       [2]  CertReqMessages,        --Certification Request
    cp       [3]  CertRepMessage,         --Certification Response
```

```
    p10cr    [4]  CertificationRequest,   --imported from [PKCS10]
    popdecc  [5]  POPODecKeyChallContent, --pop Challenge
    popdecr  [6]  POPODecKeyRespContent,  --pop Response
    kur      [7]  CertReqMessages,        --Key Update Request
    kup      [8]  CertRepMessage,         --Key Update Response
    krr      [9]  CertReqMessages,        --Key Recovery Request
    krp      [10] KeyRecRepContent,       --Key Recovery Response
    rr       [11] RevReqContent,          --Revocation Request
    rp       [12] RevRepContent,          --Revocation Response
    ccr      [13] CertReqMessages,        --Cross-Cert. Request
    ccp      [14] CertRepMessage,         --Cross-Cert. Response
    ckuann   [15] CAKeyUpdAnnContent,     --CA Key Update Ann.
    cann     [16] CertAnnContent,         --Certificate Ann.
    rann     [17] RevAnnContent,          --Revocation Ann.
    crlann   [18] CRLAnnContent,          --CRL Announcement
    pkiconf  [19] PKIConfirmContent,      --Confirmation
    nested   [20] NestedMessageContent,   --Nested Message
    genm     [21] GenMsgContent,          --General Message
    genp     [22] GenRepContent,          --General Response
    error    [23] ErrorMsgContent,        --Error Message
    certConf [24] CertConfirmContent,     --Certificate confirm
    pollReq  [25] PollReqContent,         --Polling request
    pollRep  [26] PollRepContent          --Polling response
}

PKIProtection ::= BIT STRING

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

id-PasswordBasedMac OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    usa(840) nt(113533) nsn(7) algorithms(66) 13 }
PBMParameter ::= 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{DIGEST-ALGORITHM, {...}},
    -- 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{MAC-ALGORITHM, {...}}
    -- the MAC AlgId (e.g., DES-MAC, Triple-DES-MAC [PKCS11],
    -- or HMAC [RFC2104, RFC2202])
```

```
}

id-DHBasedMac OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    usa(840) nt(113533) nsn(7) algorithms(66) 30 }
DHBMParameter ::= SEQUENCE {
    owf                 AlgorithmIdentifier{DIGEST-ALGORITHM, {...}},
    -- AlgId for a One-Way Function (SHA-1 recommended)
    mac                 AlgorithmIdentifier{MAC-ALGORITHM, {...}}
    -- the MAC AlgId (e.g., DES-MAC, Triple-DES-MAC [PKCS11],
    -- or HMAC [RFC2104, RFC2202])
}

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),
    -- messageTime was not sufficiently close to the system time,
    -- as defined by local policy
```

```
    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{DIGEST-ALGORITHM, {...}}
                        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{DIGEST-ALGORITHM, {...}}
                            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 the 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 [RFC4211]
}

CertifiedKeyPair ::= SEQUENCE {
    certOrEncCert       CertOrEncCert,
    privateKey      [0] EncryptedValue      OPTIONAL,
    -- see [RFC4211] 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
 PKIConfirmContent ::= NULL

 NestedMessageContent ::= PKIMessages

 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 (uncomment 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    ::= AlgorithmIdentifer
 --   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 OBJECT 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
 }

 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 }

 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-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-crmf2005-02(55)}
  DEFINITIONS IMPLICIT TAGS ::=
  BEGIN
  IMPORTS

  AttributeSet{}, Extensions{}, EXTENSION, ATTRIBUTE,
      SingleAttribute{}
  FROM PKIX-CommonTypes-2009
      {iso(1) identified-organization(3) dod(6) internet(1)
      security(5) mechanisms(5) pkix(7) id-mod(0)
      id-mod-pkixCommon-02(57) }

  AlgorithmIdentifier{}, SIGNATURE-ALGORITHM, ALGORITHM,
      DIGEST-ALGORITHM, MAC-ALGORITHM, PUBLIC-KEY
  FROM AlgorithmInformation-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0)
      id-mod-algorithmInformation-02(58)}

  Version, Name, Time, SubjectPublicKeyInfo, UniqueIdentifier, id-pkix,
      SignatureAlgorithms
  FROM PKIX1Explicit-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-explicit-02(51)}

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

  EnvelopedData, CONTENT-TYPE
  FROM CryptographicMessageSyntax-2009
      { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
      smime(16) modules(0) id-mod-cms-2004-02(41)}
  maca-hMAC-SHA1
  FROM CryptographicMessageSyntaxAlgorithms-2009
      { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
      smime(16) modules(0) id-mod-cmsalg-2001-02(37) }

  mda-sha1
  FROM PKIXAlgs-2009
      { iso(1) identified-organization(3) dod(6)
      internet(1) security(5) mechanisms(5) pkix(7) id-mod(0)
      id-mod-pkix1-algorithms2008-02(56) } ;
```

```
  -- 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) pkcs9(9) 16 }

  id-ct   OBJECT IDENTIFIER ::= { id-smime  1 }  -- content types

  -- Core definitions for this module

  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
          SingleAttribute{{RegInfoSet}} 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{SIGNATURE-ALGORITHM,
                           {SignatureAlgorithms}}   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{{CertExtensions}}  OPTIONAL }

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

  Controls  ::= SEQUENCE SIZE(1..MAX) OF SingleAttribute
                    {{RegControlSet}}
```

```
  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-ALGORITHM,
                                {SignatureAlgorithms}},
      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{MAC-ALGORITHM,
                 {Password-MACAlgorithms}},
      value  BIT STRING }

  --
  --  Define the currently only acceptable MAC algorithm to be used
  --  for the PKMACValue structure
  --

  id-PasswordBasedMac OBJECT IDENTIFIER ::= { iso(1) member-body(2)
      usa(840) nt(113533) nsn(7) algorithms(66) 13 }
```

```
  Password-MACAlgorithms MAC-ALGORITHM ::= {
      {IDENTIFIER id-PasswordBasedMac
       PARAMS TYPE PBMParameter ARE required
       IS-KEYED-MAC TRUE
      }, ...
  }

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

  DigestAlgorithms DIGEST-ALGORITHM ::= {
     mda-sha1, ...
  }

  MACAlgorithms MAC-ALGORITHM ::= {
      -- The modules containing the ASN.1 for the DES and 3DES MAC
      -- algorithms have not been updated at the time that this is
      -- being published.  Users of this module should define the
      -- appropriate MAC-ALGORITHM objects and uncomment the
      -- following lines if they support these MAC algorithms.
      -- maca-des-mac | maca-3des-mac --
      maca-hMAC-SHA1,
      ...
  }

  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

  --
  -- id-ct-encKeyWithID content type used as the content type for the
  -- EnvelopedData in POPOPrivKey.
  -- It contains both a private key and an identifier for key escrow
  -- agents to check against recovery requestors.
  --

  ct-encKeyWithID CONTENT-TYPE ::=
      { EncKeyWithID IDENTIFIED BY id-ct-encKeyWithID }

  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{PUBLIC-KEY, {...}},
     privateKey                OCTET STRING,
                --  Structure of public key is in PUBLIC-KEY.&PrivateKey
     attributes                [0] IMPLICIT Attributes OPTIONAL
  }

  Attributes ::= SET OF AttributeSet{{PrivateKeyAttributes}}
  PrivateKeyAttributes ATTRIBUTE ::= {...}

  --
  -- 6.  Registration Controls in CRMF
  --

  id-regCtrl OBJECT IDENTIFIER ::= { id-pkip 1 }

  RegControlSet ATTRIBUTE ::= {
      regCtrl-regToken | regCtrl-authenticator |
```

```
      regCtrl-pkiPublicationInfo | regCtrl-pkiArchiveOptions |
      regCtrl-oldCertID | regCtrl-protocolEncrKey, ... }

  --
  --  6.1.  Registration Token Control
  --

  regCtrl-regToken ATTRIBUTE ::=
      { TYPE RegToken IDENTIFIED BY id-regCtrl-regToken }

  id-regCtrl-regToken OBJECT IDENTIFIER ::= { id-regCtrl 1 }

  RegToken ::= UTF8String

  --
  --  6.2.  Authenticator Control
  --

  regCtrl-authenticator ATTRIBUTE ::=
      { TYPE Authenticator IDENTIFIED BY id-regCtrl-authenticator }

  id-regCtrl-authenticator OBJECT IDENTIFIER ::= { id-regCtrl 2 }

  Authenticator ::= UTF8String

  --
  --  6.3.  Publication Information Control
  --

  regCtrl-pkiPublicationInfo ATTRIBUTE ::=
      { TYPE PKIPublicationInfo IDENTIFIED BY
          id-regCtrl-pkiPublicationInfo }

  id-regCtrl-pkiPublicationInfo OBJECT IDENTIFIER ::= { id-regCtrl 3 }

  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 }

--
--  6.4.  Archive Options Control
--

regCtrl-pkiArchiveOptions ATTRIBUTE ::=
    { TYPE PKIArchiveOptions IDENTIFIED BY
        id-regCtrl-pkiArchiveOptions }

id-regCtrl-pkiArchiveOptions OBJECT IDENTIFIER ::= { id-regCtrl 4 }

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 archive is desired.

EncryptedKey ::= CHOICE {
    encryptedValue        EncryptedValue,   -- Deprecated
    envelopedData     [0] EnvelopedData }
    -- The encrypted private key MUST be placed in the envelopedData
    -- encryptedContentInfo encryptedContent OCTET STRING.

--
--  We skipped doing the full constraints here since this structure
--      has been deprecated in favor of EnvelopedData
--

EncryptedValue ::= SEQUENCE {
    intendedAlg   [0] AlgorithmIdentifier{ALGORITHM, {...}}  OPTIONAL,
    -- the intended algorithm for which the value will be used
    symmAlg       [1] AlgorithmIdentifier{ALGORITHM, {...}}  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{ALGORITHM, {...}}  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

  --
  --  6.5.  OldCert ID Control
  --

  regCtrl-oldCertID ATTRIBUTE ::=
      { TYPE OldCertId IDENTIFIED BY id-regCtrl-oldCertID }

  id-regCtrl-oldCertID  OBJECT IDENTIFIER ::= { id-regCtrl 5 }

  OldCertId ::= CertId

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

  --
  -- 6.6.  Protocol Encryption Key Control
  --

  regCtrl-protocolEncrKey ATTRIBUTE ::=
      { TYPE ProtocolEncrKey IDENTIFIED BY id-regCtrl-protocolEncrKey }
  id-regCtrl-protocolEncrKey    OBJECT IDENTIFIER ::= { id-regCtrl 6 }

  ProtocolEncrKey ::= SubjectPublicKeyInfo

  --
  -- 7.  Registration Info in CRMF
  --

  id-regInfo OBJECT IDENTIFIER ::= { id-pkip 2 }

  RegInfoSet ATTRIBUTE ::=
```

```
      { regInfo-utf8Pairs | regInfo-certReq }

  --
  -- 7.1.  utf8Pairs RegInfo Control
  --

  regInfo-utf8Pairs ATTRIBUTE ::=
      { TYPE UTF8Pairs IDENTIFIED BY id-regInfo-utf8Pairs }

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

  --
  --  7.2.  certReq RegInfo Control
  --

  regInfo-certReq ATTRIBUTE ::=
      { TYPE CertReq IDENTIFIED BY id-regInfo-certReq }

  id-regInfo-certReq       OBJECT IDENTIFIER ::= { id-regInfo 2 }
  --with syntax
  CertReq ::= CertRequest

  END
```

11. ASN.1 Module for RFC 5055

```
  SCVP-2009
      { iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-scvp-02(52) }
  DEFINITIONS IMPLICIT TAGS ::=
  BEGIN
  IMPORTS

  Extensions{}, EXTENSION, ATTRIBUTE
  FROM PKIX-CommonTypes-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-pkixCommon-02(57) }

  AlgorithmIdentifier{}, SIGNATURE-ALGORITHM, PUBLIC-KEY, KEY-AGREE,
      DIGEST-ALGORITHM, KEY-DERIVATION, MAC-ALGORITHM
  FROM AlgorithmInformation-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0)
      id-mod-algorithmInformation-02(58)}

  Certificate, CertificateList, CertificateSerialNumber,
```

```
    SignatureAlgorithms, SubjectPublicKeyInfo
  FROM PKIX1Explicit-2009
    { iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-explicit-02(51) }

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

  AttributeCertificate
  FROM PKIXAttributeCertificate-2009
    { iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-attribute-cert-02(47) }

  OCSPResponse
  FROM OCSP-2009
    { iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-ocsp-02(48) }

  ContentInfo, CONTENT-TYPE
  FROM CryptographicMessageSyntax-2009
    { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
    smime(16) modules(0) id-mod-cms-2004-02(41) }

  mda-sha1
  FROM PKIXAlgs-2009
    { iso(1) identified-organization(3) dod(6)
    internet(1) security(5) mechanisms(5) pkix(7) id-mod(0)
    id-mod-pkix1-algorithms2008-02(56) } ;

  ContentTypes CONTENT-TYPE ::= {ct-scvp-certValRequest |
    ct-scvp-certValResponse | ct-scvp-valPolRequest |
    ct-scvp-valPolResponse, ... }

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

  ct-scvp-certValRequest CONTENT-TYPE ::=
    { CVRequest IDENTIFIED BY id-ct-scvp-certValRequest }

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

  -- SCVP Certificate Validation Request

  CVRequest ::= SEQUENCE {
    cvRequestVersion          INTEGER DEFAULT 1,
```

```
      query                       Query,
      requestorRef           [0] GeneralNames OPTIONAL,
      requestNonce           [1] OCTET STRING OPTIONAL,
      requestorName          [2] GeneralName OPTIONAL,
      responderName          [3] GeneralName OPTIONAL,
      requestExtensions      [4] Extensions{{RequestExtensions}}
                                     OPTIONAL,
      signatureAlg           [5] AlgorithmIdentifier
                                     {SIGNATURE-ALGORITHM,
                                         {SignatureAlgorithms}}
                                     OPTIONAL,
      hashAlg                [6] OBJECT IDENTIFIER OPTIONAL,
      requestorText          [7] UTF8String (SIZE (1..256)) OPTIONAL
  }

  -- Set of signature algorithms is coming from RFC 5280
  -- SignatureAlgorithms SIGNATURE-ALGORITHM ::= {...}

  -- Add supported request extensions here; all new items should
  --      be added after the extension marker

  RequestExtensions EXTENSION ::= {...}

  Query ::= SEQUENCE {
      queriedCerts             CertReferences,
      checks                   CertChecks,
      wantBack             [1] WantBack OPTIONAL,
      validationPolicy         ValidationPolicy,
      responseFlags            ResponseFlags OPTIONAL,
      serverContextInfo    [2] OCTET STRING OPTIONAL,
      validationTime       [3] GeneralizedTime OPTIONAL,
      intermediateCerts    [4] CertBundle OPTIONAL,
      revInfos             [5] RevocationInfos OPTIONAL,
      producedAt           [6] GeneralizedTime OPTIONAL,
      queryExtensions      [7] Extensions{{QueryExtensions}} OPTIONAL
  }

  -- Add supported query extensions here; all new items should be added
  --      after the extension marker

  QueryExtensions EXTENSION ::= {...}

  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
  }

  HashAlgorithm ::= AlgorithmIdentifier{DIGEST-ALGORITHM,
                        {mda-sha1, ...}}

  SCVPCertID ::= SEQUENCE {
     certHash        OCTET STRING,
     issuerSerial    SCVPIssuerSerial,
     hashAlgorithm   HashAlgorithm
                         DEFAULT { algorithm mda-sha1.&id }
  }

  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 (CertCheckSet | ACertCheckSet, ... )

  WantBack ::= SEQUENCE SIZE (1..MAX) OF
      WANT-BACK.&id ({AllWantBacks})
```

```
  POLICY ::= ATTRIBUTE

  ValidationPolRefSet POLICY ::= {
      svp-defaultValPolicy, ...
  }

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

  ValidationAlgSet POLICY ::= {
          svp-basicValAlg, ...
  }

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

  NameValidationAlgSet POLICY ::= {
      svp-nameValAlg, ...
  }

  NameValidationAlgParams ::= SEQUENCE {
      nameCompAlgId          OBJECT IDENTIFIER (NameCompAlgSet, ... ),
      validationNames        GeneralNames
  }

  TrustAnchors ::= SEQUENCE SIZE (1..MAX) OF PKCReference
  KeyAgreePublicKey ::= SEQUENCE {
      algorithm           AlgorithmIdentifier{KEY-AGREE,
                              {SupportedKeyAgreePublicKeys}},
      publicKey           BIT STRING,
      macAlgorithm        AlgorithmIdentifier{MAC-ALGORITHM,
                              {SupportedMACAlgorithms}},
      kDF                 AlgorithmIdentifier{KEY-DERIVATION,
                              {SupportedKeyDerivationFunctions}}
                              OPTIONAL
  }

  SupportedKeyAgreePublicKeys KEY-AGREE ::= {...}
  SupportedMACAlgorithms MAC-ALGORITHM ::= {...}
  SupportedKeyDerivationFunctions KEY-DERIVATION ::= {...}

  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

ct-scvp-certValResponse CONTENT-TYPE ::=
    { CVResponse IDENTIFIED BY id-ct-scvp-certValResponse }

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{{CVResponseExtensions}}
                                   OPTIONAL,
    requestorText          [8] UTF8String (SIZE (1..256)) OPTIONAL
}

--  This document defines no extensions
CVResponseExtensions EXTENSION ::= {...}
```

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

  CVStatusCode ::= ENUMERATED {
     okay                                (0),
     skipUnrecognizedItems               (1),
     tooBusy                            (10),
     invalidRequest                     (11),
     internalError                      (12),
     badStructure                       (20),
     unsupportedVersion                 (21),
     abortUnrecognizedItems             (22),
     unrecognizedSigKey                 (23),
     badSignatureOrMAC                  (24),
     unableToDecode                     (25),
     notAuthorized                      (26),
     unsupportedChecks                  (27),
     unsupportedWantBacks               (28),
     unsupportedSignatureOrMAC          (29),
     invalidSignatureOrMAC              (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         HashAlgorithm
                            DEFAULT { algorithm mda-sha1.&id },
      value             OCTET STRING }
```

```
  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 ( BasicValidationErrorSet |
                              NameValidationErrorSet,
                              ... ) 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 (CertCheckSet | ACertCheckSet, ... ),
      status   INTEGER DEFAULT 0
  }

  ReplyWantBacks ::= SEQUENCE OF ReplyWantBack

  ReplyWantBack::= SEQUENCE {
      wb     WANT-BACK.&id({AllWantBacks}),
      value  OCTET STRING
                 (CONTAINING WANT-BACK.&Type({AllWantBacks}{@wb}))
  }

  WANT-BACK ::= TYPE-IDENTIFIER

  AllWantBacks WANT-BACK ::= {
      WantBackSet | ACertWantBackSet | AnyWantBackSet, ...
  }
```

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

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

  SCVPResponses ::= SEQUENCE OF ContentInfo

  -- SCVP Validation Policies Request

  ct-scvp-valPolRequest CONTENT-TYPE ::=
      { ValPolRequest IDENTIFIED BY id-ct-scvp-valPolRequest }

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

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

  -- SCVP Validation Policies Response

  ct-scvp-valPolResponse CONTENT-TYPE ::=
      { ValPolResponse IDENTIFIED BY id-ct-scvp-valPolResponse }

  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
                                  {SIGNATURE-ALGORITHM,
                                      {SignatureAlgorithms}},
      signatureVerification   SEQUENCE OF AlgorithmIdentifier
                                  {SIGNATURE-ALGORITHM,
                                      {SignatureAlgorithms}},
```

```
      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 }

  CertCheckSet OBJECT IDENTIFIER ::= {
      id-stc-build-pkc-path | id-stc-build-valid-pkc-path |
      id-stc-build-status-checked-pkc-path, ... }

  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 }

  ACertCheckSet OBJECT IDENTIFIER ::= {
      id-stc-build-aa-path | id-stc-build-valid-aa-path |
      id-stc-build-status-checked-aa-path |
      id-stc-status-check-ac-and-build-status-checked-aa-path
  }

  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 }

  WantBackSet WANT-BACK ::= {
      swb-pkc-cert | swb-pkc-best-cert-path |
      swb-pkc-revocation-info | swb-pkc-public-key-info |
      swb-pkc-all-cert-paths | swb-pkc-ee-revocation-info |
      swb-pkc-CAs-revocation-info
  }

  ACertWantBackSet WANT-BACK ::= {
      swb-ac-cert | swb-aa-cert-path |
      swb-aa-revocation-info | swb-ac-revocation-info
  }

  AnyWantBackSet WANT-BACK ::= { swb-relayed-responses }

  swb-pkc-best-cert-path WANT-BACK ::=
      { CertBundle IDENTIFIED BY id-swb-pkc-best-cert-path }
  id-swb-pkc-best-cert-path       OBJECT IDENTIFIER ::= { id-swb 1 }
  swb-pkc-revocation-info WANT-BACK ::=
      { RevInfoWantBack IDENTIFIED BY id-swb-pkc-revocation-info }
  id-swb-pkc-revocation-info      OBJECT IDENTIFIER ::= { id-swb 2 }

  swb-pkc-public-key-info WANT-BACK ::=
      { SubjectPublicKeyInfo IDENTIFIED BY id-swb-pkc-public-key-info }
  id-swb-pkc-public-key-info      OBJECT IDENTIFIER ::= { id-swb 4 }

  swb-aa-cert-path WANT-BACK ::=
      {CertBundle IDENTIFIED BY id-swb-aa-cert-path }
  id-swb-aa-cert-path             OBJECT IDENTIFIER ::= { id-swb 5 }

  swb-aa-revocation-info WANT-BACK ::=
      { RevInfoWantBack IDENTIFIED BY id-swb-aa-revocation-info }
  id-swb-aa-revocation-info       OBJECT IDENTIFIER ::= { id-swb 6 }

  swb-ac-revocation-info WANT-BACK ::=
      { RevInfoWantBack IDENTIFIED BY id-swb-ac-revocation-info }
  id-swb-ac-revocation-info       OBJECT IDENTIFIER ::= { id-swb 7 }

  swb-relayed-responses WANT-BACK ::=
      {SCVPResponses IDENTIFIED BY id-swb-relayed-responses }
```

```
  id-swb-relayed-responses        OBJECT IDENTIFIER ::= { id-swb 9 }

  swb-pkc-all-cert-paths WANT-BACK ::=
      {CertBundles IDENTIFIED BY id-swb-pkc-all-cert-paths }
  id-swb-pkc-all-cert-paths       OBJECT IDENTIFIER ::= { id-swb 12}

  swb-pkc-ee-revocation-info WANT-BACK ::=
      { RevInfoWantBack IDENTIFIED BY id-swb-pkc-ee-revocation-info }
  id-swb-pkc-ee-revocation-info   OBJECT IDENTIFIER ::= { id-swb 13}

  swb-pkc-CAs-revocation-info WANT-BACK ::=
      { RevInfoWantBack IDENTIFIED BY id-swb-pkc-CAs-revocation-info }
  id-swb-pkc-CAs-revocation-info  OBJECT IDENTIFIER ::= { id-swb 14}

  swb-pkc-cert WANT-BACK ::=
      { Certificate IDENTIFIED BY id-swb-pkc-cert }
  id-swb-pkc-cert OBJECT IDENTIFIER ::= { id-swb 10}

  swb-ac-cert WANT-BACK ::=
      { AttributeCertificate IDENTIFIED BY id-swb-ac-cert }
  id-swb-ac-cert OBJECT IDENTIFIER ::= { id-swb 11}

  -- 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 }

  svp-defaultValPolicy POLICY ::=
      { IDENTIFIED BY id-svp-defaultValPolicy }

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

  -- SCVP Basic Validation Algorithm Identifier

  svp-basicValAlg POLICY ::= {IDENTIFIED BY id-svp-basicValAlg }

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

  -- SCVP Basic Validation Algorithm Errors

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

  BasicValidationErrorSet OBJECT IDENTIFIER ::= {
      id-bvae-expired | id-bvae-not-yet-valid |
      id-bvae-wrongTrustAnchor | id-bvae-noValidCertPath |
      id-bvae-revoked | id-bvae-invalidKeyPurpose |
      id-bvae-invalidKeyUsage | id-bvae-invalidCertPolicy
```

```
  }

  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

  svp-nameValAlg POLICY ::=
      {TYPE NameValidationAlgParams IDENTIFIED BY id-svp-nameValAlg }

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

  -- SCVP Name Validation Algorithm DN comparison algorithm

  NameCompAlgSet OBJECT IDENTIFIER ::= {
      id-nva-dnCompAlg
  }

  id-nva-dnCompAlg   OBJECT IDENTIFIER ::= { id-svp 4 }
  -- SCVP Name Validation Algorithm Errors

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

  NameValidationErrorSet OBJECT IDENTIFIER ::= {
      id-nvae-name-mismatch | id-nvae-no-name | id-nvae-unknown-alg |
      id-nvae-bad-name | id-nvae-bad-name-type | id-nvae-mixed-names
  }

  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) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) 3 }

  SvcpExtKeyUsageSet OBJECT IDENTIFIER ::= {
```

```
      id-kp-scvpServer | id-kp-scvpClient
  }

  id-kp-scvpServer   OBJECT IDENTIFIER ::= { id-kp 15 }

  id-kp-scvpClient   OBJECT IDENTIFIER ::= { id-kp 16 }

  END
```

12. ASN.1 Module for RFC 5272

```
  EnrollmentMessageSyntax-2009
      {iso(1) identified-organization(3) dod(6) internet(1)
      security(5) mechanisms(5) pkix(7) id-mod(0) id-mod-cmc2002-02(53)}
  DEFINITIONS IMPLICIT TAGS ::=
  BEGIN
  EXPORTS ALL;
  IMPORTS

  AttributeSet{}, Extension{}, EXTENSION, ATTRIBUTE
  FROM PKIX-CommonTypes-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-pkixCommon-02(57)}
  AlgorithmIdentifier{}, DIGEST-ALGORITHM, KEY-WRAP, KEY-DERIVATION,
      MAC-ALGORITHM, SIGNATURE-ALGORITHM, PUBLIC-KEY
  FROM AlgorithmInformation-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0)
      id-mod-algorithmInformation-02(58)}

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

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

  ContentInfo, IssuerAndSerialNumber, CONTENT-TYPE
  FROM CryptographicMessageSyntax-2009
      { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
      smime(16) modules(0) id-mod-cms-2004-02(41)}

  CertReqMsg, PKIPublicationInfo, CertTemplate
  FROM PKIXCRMF-2009
```

```
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-crmf2005-02(55)}

  mda-sha1
  FROM PKIXAlgs-2009
       { iso(1) identified-organization(3) dod(6)
       internet(1) security(5) mechanisms(5) pkix(7) id-mod(0)
       id-mod-pkix1-algorithms2008-02(56)}

  kda-PBKDF2, maca-hMAC-SHA1
  FROM CryptographicMessageSyntaxAlgorithms-2009
      { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
      smime(16) modules(0) id-mod-cmsalg-2001-02(37) }

  mda-sha256
  FROM PKIX1-PSS-OAEP-Algorithms-2009
       { iso(1) identified-organization(3) dod(6)
         internet(1) security(5) mechanisms(5) pkix(7) id-mod(0)
         id-mod-pkix1-rsa-pkalgs-02(54) } ;

  --  CMS Content types defined in this document
  CMC-ContentTypes CONTENT-TYPE ::= { ct-PKIData | ct-PKIResponse, ... }

  --  Signature Algorithms defined in this document

  SignatureAlgs SIGNATURE-ALGORITHM ::= { sa-noSignature }

  --  CMS Unsigned Attributes

  CMC-UnsignedAtts ATTRIBUTE ::= { aa-cmc-unsignedData }

  --
  --

  id-cmc OBJECT IDENTIFIER ::= {id-pkix 7}   -- CMC controls
  id-cct OBJECT IDENTIFIER ::= {id-pkix 12}  -- CMC content types

  -- This is the content type for a request message in the protocol

  ct-PKIData CONTENT-TYPE ::=
      { PKIData IDENTIFIED BY id-cct-PKIData }
  id-cct-PKIData OBJECT IDENTIFIER ::= { id-cct 2 }

  PKIData ::= SEQUENCE {
      controlSequence    SEQUENCE SIZE(0..MAX) OF TaggedAttribute,
      reqSequence        SEQUENCE SIZE(0..MAX) OF TaggedRequest,
      cmsSequence        SEQUENCE SIZE(0..MAX) OF TaggedContentInfo,
      otherMsgSequence   SEQUENCE SIZE(0..MAX) OF OtherMsg
```

```
  }

  BodyPartID ::= INTEGER(0..4294967295)

  TaggedAttribute ::= SEQUENCE {
      bodyPartID         BodyPartID,
      attrType           CMC-CONTROL.&id({Cmc-Control-Set}),
      attrValues         SET OF CMC-CONTROL.
                             &Type({Cmc-Control-Set}{@attrType})
  }

  Cmc-Control-Set CMC-CONTROL ::= {
      cmc-identityProof | cmc-dataReturn | cmc-regInfo |
      cmc-responseInfo | cmc-queryPending | cmc-popLinkRandom |
      cmc-popLinkWitness | cmc-identification | cmc-transactionId |
      cmc-senderNonce | cmc-recipientNonce | cmc-statusInfo |
      cmc-addExtensions | cmc-encryptedPOP | cmc-decryptedPOP |
      cmc-lraPOPWitness | cmc-getCert | cmc-getCRL |
      cmc-revokeRequest | cmc-confirmCertAcceptance |
      cmc-statusInfoV2 | cmc-trustedAnchors | cmc-authData |
      cmc-batchRequests | cmc-batchResponses | cmc-publishCert |
      cmc-modCertTemplate | cmc-controlProcessed |
      cmc-identityProofV2 | cmc-popLinkWitnessV2, ... }

  OTHER-REQUEST ::= TYPE-IDENTIFIER

  --  We do not define any other requests in this document;
  --     examples might be attribute certification requests

  OtherRequests OTHER-REQUEST ::= {...}

  TaggedRequest ::= CHOICE {
      tcr               [0] TaggedCertificationRequest,
      crm               [1] CertReqMsg,
      orm               [2] SEQUENCE {
          bodyPartID            BodyPartID,
          requestMessageType    OTHER-REQUEST.&id({OtherRequests}),
          requestMessageValue   OTHER-REQUEST.&Type({OtherRequests}
                                    {@.requestMessageType})
      }
  }

  TaggedCertificationRequest ::= SEQUENCE {
      bodyPartID            BodyPartID,
      certificationRequest  CertificationRequest
  }

  AttributeList ATTRIBUTE ::= {at-extension-req, ...}
```

```
  CertificationRequest ::= SEQUENCE {
     certificationRequestInfo  SEQUENCE {
         version                   INTEGER,
         subject                   Name,
         subjectPublicKeyInfo      SEQUENCE {
             algorithm                 AlgorithmIdentifier{PUBLIC-KEY,
                                           {PublicKeyAlgorithms}},
             subjectPublicKey          BIT STRING
         },
         attributes                [0] IMPLICIT SET OF
                                       AttributeSet{{AttributeList}}
     },
     signatureAlgorithm        AlgorithmIdentifier
                                   {SIGNATURE-ALGORITHM,
                                       {SignatureAlgorithms}},
     signature                 BIT STRING
  }

  TaggedContentInfo ::= SEQUENCE {
      bodyPartID              BodyPartID,
      contentInfo             ContentInfo
  }

  OTHER-MSG ::= TYPE-IDENTIFIER

  --  No other messages currently defined

  OtherMsgSet OTHER-MSG ::= {...}

  OtherMsg ::= SEQUENCE {
      bodyPartID        BodyPartID,
      otherMsgType      OTHER-MSG.&id({OtherMsgSet}),
      otherMsgValue     OTHER-MSG.&Type({OtherMsgSet}{@otherMsgType}) }

  --  This defines the response message in the protocol

  ct-PKIResponse CONTENT-TYPE ::=
      { PKIResponse IDENTIFIED BY id-cct-PKIResponse }
  id-cct-PKIResponse OBJECT IDENTIFIER ::= { id-cct 3 }

  ResponseBody ::= PKIResponse

  PKIResponse ::= SEQUENCE {
      controlSequence   SEQUENCE SIZE(0..MAX) OF TaggedAttribute,
      cmsSequence       SEQUENCE SIZE(0..MAX) OF TaggedContentInfo,
      otherMsgSequence  SEQUENCE SIZE(0..MAX) OF OtherMsg
  }
```

```
  CMC-CONTROL ::= TYPE-IDENTIFIER

  -- The following controls have the type OCTET STRING

  cmc-identityProof CMC-CONTROL ::=
      { OCTET STRING IDENTIFIED BY id-cmc-identityProof }
  id-cmc-identityProof OBJECT IDENTIFIER ::= {id-cmc 3}

  cmc-dataReturn CMC-CONTROL ::=
      { OCTET STRING IDENTIFIED BY id-cmc-dataReturn }
  id-cmc-dataReturn OBJECT IDENTIFIER ::= {id-cmc 4}

  cmc-regInfo CMC-CONTROL ::=
      { OCTET STRING IDENTIFIED BY id-cmc-regInfo }
  id-cmc-regInfo OBJECT IDENTIFIER ::= {id-cmc 18}

  cmc-responseInfo CMC-CONTROL ::=
      { OCTET STRING IDENTIFIED BY id-cmc-responseInfo }
  id-cmc-responseInfo OBJECT IDENTIFIER ::= {id-cmc 19}

  cmc-queryPending CMC-CONTROL ::=
      { OCTET STRING IDENTIFIED BY id-cmc-queryPending }
  id-cmc-queryPending OBJECT IDENTIFIER ::= {id-cmc 21}

  cmc-popLinkRandom CMC-CONTROL ::=
      { OCTET STRING IDENTIFIED BY id-cmc-popLinkRandom }
  id-cmc-popLinkRandom OBJECT IDENTIFIER ::= {id-cmc 22}

  cmc-popLinkWitness CMC-CONTROL ::=
      { OCTET STRING IDENTIFIED BY id-cmc-popLinkWitness }
  id-cmc-popLinkWitness OBJECT IDENTIFIER ::= {id-cmc 23}

  -- The following controls have the type UTF8String

  cmc-identification CMC-CONTROL ::=
      { UTF8String IDENTIFIED BY id-cmc-identification }
  id-cmc-identification OBJECT IDENTIFIER ::= {id-cmc 2}

  -- The following controls have the type INTEGER

  cmc-transactionId CMC-CONTROL ::=
      { INTEGER IDENTIFIED BY id-cmc-transactionId }
  id-cmc-transactionId OBJECT IDENTIFIER ::= {id-cmc 5}

  -- The following controls have the type OCTET STRING

  cmc-senderNonce CMC-CONTROL ::=
      { OCTET STRING IDENTIFIED BY id-cmc-senderNonce }
```

```
  id-cmc-senderNonce OBJECT IDENTIFIER ::= {id-cmc 6}

  cmc-recipientNonce CMC-CONTROL ::=
      { OCTET STRING IDENTIFIED BY id-cmc-recipientNonce }
  id-cmc-recipientNonce OBJECT IDENTIFIER ::= {id-cmc 7}

  -- Used to return status in a response

  cmc-statusInfo CMC-CONTROL ::=
      { CMCStatusInfo IDENTIFIED BY id-cmc-statusInfo }
  id-cmc-statusInfo OBJECT IDENTIFIER ::= {id-cmc 1}

  CMCStatusInfo ::= SEQUENCE {
      cMCStatus       CMCStatus,
      bodyList        SEQUENCE SIZE (1..MAX) OF BodyPartID,
      statusString    UTF8String OPTIONAL,
      otherInfo       CHOICE {
         failInfo         CMCFailInfo,
         pendInfo         PendInfo
      } OPTIONAL
  }

  PendInfo ::= SEQUENCE {
      pendToken        OCTET STRING,
      pendTime         GeneralizedTime
  }

  CMCStatus ::= INTEGER {
      success         (0),
      failed          (2),
      pending         (3),
      noSupport       (4),
      confirmRequired (5),
      popRequired     (6),
      partial         (7)
  }

  -- Note:
  -- The spelling of unsupportedExt is corrected in this version.
  -- In RFC 2797, it was unsuportedExt.

  CMCFailInfo ::= INTEGER {
      badAlg          (0),
      badMessageCheck (1),
      badRequest      (2),
      badTime         (3),
      badCertId       (4),
      unsuportedExt   (5),
```

```
      mustArchiveKeys (6),
      badIdentity     (7),
      popRequired     (8),
      popFailed       (9),
      noKeyReuse      (10),
      internalCAError (11),
      tryLater        (12),
      authDataFail    (13)
  }

  -- Used for RAs to add extensions to certification requests

  cmc-addExtensions CMC-CONTROL ::=
      { AddExtensions IDENTIFIED BY id-cmc-addExtensions }
  id-cmc-addExtensions OBJECT IDENTIFIER ::= {id-cmc 8}

  AddExtensions ::= SEQUENCE {
      pkiDataReference    BodyPartID,
      certReferences      SEQUENCE OF BodyPartID,
      extensions          SEQUENCE OF Extension{{CertExtensions}}
  }

  cmc-encryptedPOP CMC-CONTROL ::=
      { EncryptedPOP IDENTIFIED BY id-cmc-encryptedPOP }
  cmc-decryptedPOP CMC-CONTROL ::=
      { DecryptedPOP IDENTIFIED BY id-cmc-decryptedPOP }
  id-cmc-encryptedPOP OBJECT IDENTIFIER ::= {id-cmc 9}
  id-cmc-decryptedPOP OBJECT IDENTIFIER ::= {id-cmc 10}

  EncryptedPOP ::= SEQUENCE {
      request       TaggedRequest,
      cms             ContentInfo,
      thePOPAlgID     AlgorithmIdentifier{MAC-ALGORITHM, {POPAlgs}},
      witnessAlgID    AlgorithmIdentifier{DIGEST-ALGORITHM,
                          {WitnessAlgs}},
      witness         OCTET STRING
  }

  POPAlgs MAC-ALGORITHM ::= {maca-hMAC-SHA1, ...}
  WitnessAlgs DIGEST-ALGORITHM ::= {mda-sha1, ...}

  DecryptedPOP ::= SEQUENCE {
      bodyPartID      BodyPartID,
      thePOPAlgID     AlgorithmIdentifier{MAC-ALGORITHM, {POPAlgs}},
      thePOP          OCTET STRING
  }

  cmc-lraPOPWitness CMC-CONTROL ::=
```

```
      { LraPopWitness IDENTIFIED BY id-cmc-lraPOPWitness }

  id-cmc-lraPOPWitness OBJECT IDENTIFIER ::= {id-cmc 11}

  LraPopWitness ::= SEQUENCE {
      pkiDataBodyid   BodyPartID,
      bodyIds         SEQUENCE OF BodyPartID
  }

  --

  cmc-getCert CMC-CONTROL ::=
      { GetCert IDENTIFIED BY id-cmc-getCert }
  id-cmc-getCert OBJECT IDENTIFIER ::= {id-cmc 15}

  GetCert ::= SEQUENCE {
      issuerName      GeneralName,
      serialNumber    INTEGER }

  cmc-getCRL CMC-CONTROL ::=
      { GetCRL IDENTIFIED BY id-cmc-getCRL }
  id-cmc-getCRL OBJECT IDENTIFIER ::= {id-cmc 16}
  GetCRL ::= SEQUENCE {
      issuerName    Name,
      cRLName       GeneralName OPTIONAL,
      time          GeneralizedTime OPTIONAL,
      reasons       ReasonFlags OPTIONAL }

  cmc-revokeRequest CMC-CONTROL ::=
      { RevokeRequest IDENTIFIED BY id-cmc-revokeRequest}
  id-cmc-revokeRequest OBJECT IDENTIFIER ::= {id-cmc 17}

  RevokeRequest ::= SEQUENCE {
      issuerName            Name,
      serialNumber          INTEGER,
      reason                CRLReason,
      invalidityDate         GeneralizedTime OPTIONAL,
      passphrase            OCTET STRING OPTIONAL,
      comment               UTF8String OPTIONAL }

  cmc-confirmCertAcceptance CMC-CONTROL ::=
      { CMCCertId IDENTIFIED BY id-cmc-confirmCertAcceptance }
  id-cmc-confirmCertAcceptance OBJECT IDENTIFIER ::= {id-cmc 24}

  CMCCertId ::= IssuerAndSerialNumber

  -- The following is used to request v3 extensions be added
  --     to a certificate
```

```
  at-extension-req ATTRIBUTE ::=
      { TYPE ExtensionReq IDENTIFIED BY id-ExtensionReq }
  id-ExtensionReq OBJECT IDENTIFIER ::= {iso(1) member-body(2) us(840)
      rsadsi(113549) pkcs(1) pkcs-9(9) 14}

  ExtensionReq ::= SEQUENCE SIZE (1..MAX) OF
      Extension{{CertExtensions}}

  -- The following allows Diffie-Hellman Certification Request
  --     Messages to be well-formed

  sa-noSignature SIGNATURE-ALGORITHM ::= {
      IDENTIFIER id-alg-noSignature
      VALUE NoSignatureValue
      PARAMS TYPE NULL ARE required
      HASHES { mda-sha1 }
  }
  id-alg-noSignature OBJECT IDENTIFIER ::= {id-pkix id-alg(6) 2}

  NoSignatureValue ::= OCTET STRING
  --  Unauthenticated attribute to carry removable data.

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

  aa-cmc-unsignedData ATTRIBUTE ::=
      { TYPE CMCUnsignedData IDENTIFIED BY id-aa-cmc-unsignedData }
  id-aa-cmc-unsignedData OBJECT IDENTIFIER ::= {id-aa 34}

  CMCUnsignedData ::= SEQUENCE {
      bodyPartPath        BodyPartPath,
      identifier          TYPE-IDENTIFIER.&id,
      content             TYPE-IDENTIFIER.&Type
  }

  --  Replaces CMC Status Info
  --

  cmc-statusInfoV2 CMC-CONTROL ::=
      { CMCStatusInfoV2 IDENTIFIED BY id-cmc-statusInfoV2 }
  id-cmc-statusInfoV2 OBJECT IDENTIFIER ::= {id-cmc 25}

  EXTENDED-FAILURE-INFO ::= TYPE-IDENTIFIER

  ExtendedFailures EXTENDED-FAILURE-INFO ::= {...}

  CMCStatusInfoV2 ::= SEQUENCE {
     cMCStatus             CMCStatus,
```

```
    bodyList              SEQUENCE SIZE (1..MAX) OF
                                  BodyPartReference,
    statusString          UTF8String OPTIONAL,
    otherInfo             CHOICE {
        failInfo               CMCFailInfo,
        pendInfo               PendInfo,
        extendedFailInfo       [1] SEQUENCE {
           failInfoOID             TYPE-IDENTIFIER.&id
                                       ({ExtendedFailures}),
           failInfoValue           TYPE-IDENTIFIER.&Type
                                       ({ExtendedFailures}
                                           {@.failInfoOID})
        }
     } OPTIONAL
  }

  BodyPartReference ::= CHOICE {
     bodyPartID           BodyPartID,
     bodyPartPath         BodyPartPath
  }

  BodyPartPath ::= SEQUENCE SIZE (1..MAX) OF BodyPartID

  --  Allow for distribution of trust anchors
  --

  cmc-trustedAnchors CMC-CONTROL ::=
      { PublishTrustAnchors IDENTIFIED BY id-cmc-trustedAnchors }
  id-cmc-trustedAnchors OBJECT IDENTIFIER ::= {id-cmc 26}

  PublishTrustAnchors ::= SEQUENCE {
      seqNumber      INTEGER,
      hashAlgorithm  AlgorithmIdentifier{DIGEST-ALGORITHM,
                         {HashAlgorithms}},
      anchorHashes   SEQUENCE OF OCTET STRING
  }

  HashAlgorithms DIGEST-ALGORITHM ::= {
     mda-sha1 | mda-sha256, ...
  }

  cmc-authData CMC-CONTROL ::=
      { AuthPublish IDENTIFIED BY id-cmc-authData }
  id-cmc-authData OBJECT IDENTIFIER ::= {id-cmc 27}

  AuthPublish ::= BodyPartID

  --   These two items use BodyPartList
```

```
  cmc-batchRequests CMC-CONTROL ::=
      { BodyPartList IDENTIFIED BY id-cmc-batchRequests }
  id-cmc-batchRequests OBJECT IDENTIFIER ::= {id-cmc 28}

  cmc-batchResponses CMC-CONTROL ::=
      { BodyPartList IDENTIFIED BY id-cmc-batchResponses }
  id-cmc-batchResponses OBJECT IDENTIFIER ::= {id-cmc 29}

  BodyPartList ::= SEQUENCE SIZE (1..MAX) OF BodyPartID

  cmc-publishCert CMC-CONTROL ::=
      { CMCPublicationInfo IDENTIFIED BY id-cmc-publishCert }
  id-cmc-publishCert OBJECT IDENTIFIER ::= {id-cmc 30}

  CMCPublicationInfo ::= SEQUENCE {
      hashAlg        AlgorithmIdentifier{DIGEST-ALGORITHM,
                           {HashAlgorithms}},
      certHashes     SEQUENCE OF OCTET STRING,
      pubInfo        PKIPublicationInfo
  }

  cmc-modCertTemplate CMC-CONTROL ::=
      { ModCertTemplate IDENTIFIED BY id-cmc-modCertTemplate }
  id-cmc-modCertTemplate OBJECT IDENTIFIER ::= {id-cmc 31}

  ModCertTemplate ::= SEQUENCE {
      pkiDataReference             BodyPartPath,
      certReferences               BodyPartList,
      replace                      BOOLEAN DEFAULT TRUE,
      certTemplate                 CertTemplate
  }

  -- Inform follow-on servers that one or more controls have
  --     already been processed

  cmc-controlProcessed CMC-CONTROL ::=
      { ControlsProcessed IDENTIFIED BY id-cmc-controlProcessed }
  id-cmc-controlProcessed OBJECT IDENTIFIER ::= {id-cmc 32}

  ControlsProcessed ::= SEQUENCE {
      bodyList              SEQUENCE SIZE(1..MAX) OF BodyPartReference
  }

  --  Identity Proof control w/ algorithm agility

  cmc-identityProofV2 CMC-CONTROL ::=
      { IdentityProofV2 IDENTIFIED BY id-cmc-identityProofV2 }
  id-cmc-identityProofV2 OBJECT IDENTIFIER ::= { id-cmc 33 }
```

```
  IdentityProofV2 ::= SEQUENCE {
      proofAlgID       AlgorithmIdentifier{DIGEST-ALGORITHM,
                           {WitnessAlgs}},
      macAlgId         AlgorithmIdentifier{MAC-ALGORITHM, {POPAlgs}},
      witness          OCTET STRING
  }

  cmc-popLinkWitnessV2 CMC-CONTROL ::=
      { PopLinkWitnessV2 IDENTIFIED BY id-cmc-popLinkWitnessV2 }
  id-cmc-popLinkWitnessV2 OBJECT IDENTIFIER ::= { id-cmc 34 }

  PopLinkWitnessV2 ::= SEQUENCE {
      keyGenAlgorithm   AlgorithmIdentifier{KEY-DERIVATION,
                            {KeyDevAlgs}},
      macAlgorithm      AlgorithmIdentifier{MAC-ALGORITHM, {POPAlgs}},
      witness           OCTET STRING
  }

  KeyDevAlgs KEY-DERIVATION ::= {kda-PBKDF2, ...}

  END
```

13. ASN.1 Module for RFC 5755

```
   PKIXAttributeCertificate-2009
       {iso(1) identified-organization(3) dod(6) internet(1) security(5)
       mechanisms(5) pkix(7) id-mod(0) id-mod-attribute-cert-02(47)}
   DEFINITIONS IMPLICIT TAGS ::=
   BEGIN
   IMPORTS

   AttributeSet{}, Extensions{}, SecurityCategory{},
           EXTENSION, ATTRIBUTE, SECURITY-CATEGORY
   FROM PKIX-CommonTypes-2009
       {iso(1) identified-organization(3) dod(6) internet(1) security(5)
       mechanisms(5) pkix(7) id-mod(0) id-mod-pkixCommon-02(57) }

   AlgorithmIdentifier{}, SIGNATURE-ALGORITHM, DIGEST-ALGORITHM
   FROM AlgorithmInformation-2009
       {iso(1) identified-organization(3) dod(6) internet(1) security(5)
       mechanisms(5) pkix(7) id-mod(0)
       id-mod-algorithmInformation-02(58)}

      -- IMPORTed module OIDs MAY change if [PKIXPROF] changes
      -- PKIX Certificate Extensions

   CertificateSerialNumber, UniqueIdentifier, id-pkix, id-pe, id-kp,
       id-ad, id-at, SIGNED{}, SignatureAlgorithms
```

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

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

   ContentInfo
     FROM CryptographicMessageSyntax-2009
       { iso(1) member-body(2) us(840) rsadsi(113549)
       pkcs(1) pkcs-9(9) smime(16) modules(0) id-mod-cms-2004-02(41) };
   --  Define the set of extensions that can appear.
   --  Some of these are imported from PKIX Cert

   AttributeCertExtensions EXTENSION ::= {
       ext-auditIdentity | ext-targetInformation |
       ext-AuthorityKeyIdentifier | ext-AuthorityInfoAccess |
       ext-CRLDistributionPoints | ext-noRevAvail | ext-ac-proxying |
       ext-aaControls, ... }

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

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

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

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

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

   -- Define the set of attributes used here

   AttributesDefined ATTRIBUTE ::= {  at-authenticationInfo |
        at-accesIdentity | at-chargingIdentity | at-group |
        at-role | at-clearance | at-encAttrs, ...}

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

   at-accesIdentity ATTRIBUTE ::= { TYPE SvceAuthInfo
```

```
       IDENTIFIED BY id-aca-accessIdentity}

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

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

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

   at-clearance ATTRIBUTE ::= { TYPE Clearance
       IDENTIFIED BY id-at-clearance}
   at-clearance-RFC3281 ATTRIBUTE ::= {TYPE Clearance-rfc3281
       IDENTIFIED BY id-at-clearance-rfc3281 }

   at-encAttrs ATTRIBUTE ::= { TYPE ContentInfo
       IDENTIFIED BY id-aca-encAttrs}

   --
   --  OIDs used by Attribute Certificate Extensions
   --

   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-targetInformation      OBJECT IDENTIFIER ::= { id-ce 55 }
   id-ce-noRevAvail             OBJECT IDENTIFIER ::= { id-ce 56 }

   --
   --  OIDs used by Attribute Certificate Attributes
   --

   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 5 } is reserved
   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) attributeType(4) clearance (55) }

   -- Uncomment the following declaration and comment the above line if
   -- using the id-at-clearance attribute as defined in [RFC3281]
```

```
   -- id-at-clearance ::= id-at-clearance-3281

   id-at-clearance-rfc3281              OBJECT IDENTIFIER ::= {
       joint-iso-ccitt(2) ds(5) module(1) selected-attribute-types(5)
       clearance (55) }

   --
   --  The syntax of an Attribute Certificate
   --

   AttributeCertificate ::= SIGNED{AttributeCertificateInfo}

   AttributeCertificateInfo ::= SEQUENCE {
       version        AttCertVersion,  -- version is v2
       holder         Holder,
       issuer         AttCertIssuer,
       signature      AlgorithmIdentifier{SIGNATURE-ALGORITHM,
                          {SignatureAlgorithms}},
       serialNumber   CertificateSerialNumber,
       attrCertValidityPeriod   AttCertValidityPeriod,
       attributes     SEQUENCE OF
                          AttributeSet{{AttributesDefined}},
       issuerUniqueID UniqueIdentifier OPTIONAL,
       extensions     Extensions{{AttributeCertExtensions}} 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) },
               -- otherObjectTypes MUST NOT
               -- be used in this profile
       otherObjectTypeID   OBJECT IDENTIFIER  OPTIONAL,
       digestAlgorithm     AlgorithmIdentifier{DIGEST-ALGORITHM, {...}},
```

```
      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
   }

   --
   -- Syntax used by Attribute Certificate Extensions
   --

   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
   }

   AAControls ::= SEQUENCE {
```

```
        pathLenConstraint INTEGER (0..MAX) OPTIONAL,
        permittedAttrs    [0] AttrSpec OPTIONAL,
        excludedAttrs     [1] AttrSpec OPTIONAL,
        permitUnSpecified BOOLEAN DEFAULT TRUE
    }

    AttrSpec::= SEQUENCE OF OBJECT IDENTIFIER

    ProxyInfo ::= SEQUENCE OF Targets

    --
    --  Syntax used by Attribute Certificate Attributes
    --
    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            OBJECT IDENTIFIER,
        classList           ClassList DEFAULT {unclassified},
        securityCategories  SET OF SecurityCategory
                                 {{SupportedSecurityCategories}} OPTIONAL
    }

    -- Uncomment the following lines to support deprecated clearance
    -- syntax and comment out previous Clearance.

    -- Clearance ::= Clearance-rfc3281

    Clearance-rfc3281  ::=  SEQUENCE {
        policyId       [0] OBJECT IDENTIFIER,
        classList      [1] ClassList DEFAULT {unclassified},
```

```
        securityCategories [2] SET OF SecurityCategory-rfc3281
                               {{SupportedSecurityCategories}} OPTIONAL
    }

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

    SecurityCategory-rfc3281{SECURITY-CATEGORY:Supported} ::= SEQUENCE {
        type      [0]  IMPLICIT SECURITY-CATEGORY.
                  &id({Supported}),
        value     [1]  EXPLICIT SECURITY-CATEGORY.
                  &Type({Supported}{@type})
    }

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

    END
```

14. ASN.1 Module for RFC 5280, Explicit and Implicit

Note that many of the changes in this module are similar or the same as the changes made in more recent versions of X.509 itself.

```
  PKIX1Explicit-2009
      {iso(1) identified-organization(3) dod(6) internet(1)
      security(5) mechanisms(5) pkix(7) id-mod(0)
      id-mod-pkix1-explicit-02(51)}
  DEFINITIONS EXPLICIT TAGS ::=
  BEGIN

  IMPORTS

  Extensions{}, EXTENSION, ATTRIBUTE, SingleAttribute{}
  FROM PKIX-CommonTypes-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-pkixCommon-02(57)}
```

```
  AlgorithmIdentifier{}, PUBLIC-KEY, SIGNATURE-ALGORITHM
  FROM AlgorithmInformation-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0)
      id-mod-algorithmInformation-02(58)}

  CertExtensions, CrlExtensions, CrlEntryExtensions
  FROM PKIX1Implicit-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-implicit-02(59)}
  SignatureAlgs, PublicKeys
  FROM PKIXAlgs-2009
      {iso(1) identified-organization(3) dod(6)
      internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) 56}

  SignatureAlgs, PublicKeys
  FROM PKIX1-PSS-OAEP-Algorithms-2009
      {iso(1) identified-organization(3) dod(6)
      internet(1) security(5) mechanisms(5) pkix(7) id-mod(0)
      id-mod-pkix1-rsa-pkalgs-02(54)}

  ORAddress
  FROM PKIX-X400Address-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-x400address-02(60)};

  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
  AttributeType           ::=  ATTRIBUTE.&id

  --  Replaced by SingleAttribute{}
  --
  -- 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.
  -- All attributes are 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 }
  at-name ATTRIBUTE ::= { TYPE X520name IDENTIFIED BY id-at-name }

  id-at-surname           AttributeType ::= { id-at 4 }
  at-surname ATTRIBUTE ::= { TYPE X520name IDENTIFIED BY id-at-surname }

  id-at-givenName         AttributeType ::= { id-at 42 }
  at-givenName ATTRIBUTE ::=
      { TYPE X520name IDENTIFIED BY id-at-givenName }

  id-at-initials          AttributeType ::= { id-at 43 }
  at-initials ATTRIBUTE ::=
      { TYPE X520name IDENTIFIED BY id-at-initials }

  id-at-generationQualifier AttributeType ::= { id-at 44 }
  at-generationQualifier ATTRIBUTE ::=
      { TYPE X520name IDENTIFIED BY id-at-generationQualifier }
```

```
  -- Directory string type --

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

  X520name ::= DirectoryString {ub-name}

  -- Naming attributes of type X520CommonName

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

  at-x520CommonName ATTRIBUTE ::=
      {TYPE X520CommonName IDENTIFIED BY id-at-commonName }

  X520CommonName ::= DirectoryString {ub-common-name}

  -- Naming attributes of type X520LocalityName

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

  at-x520LocalityName ATTRIBUTE ::=
      { TYPE X520LocalityName IDENTIFIED BY id-at-localityName }
  X520LocalityName ::= DirectoryString {ub-locality-name}

  -- Naming attributes of type X520StateOrProvinceName

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

  at-x520StateOrProvinceName ATTRIBUTE ::=
      { TYPE DirectoryString {ub-state-name}
          IDENTIFIED BY id-at-stateOrProvinceName }
  X520StateOrProvinceName ::= DirectoryString {ub-state-name}

  -- Naming attributes of type X520OrganizationName

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

  at-x520OrganizationName ATTRIBUTE ::=
      { TYPE DirectoryString {ub-organization-name}
          IDENTIFIED BY id-at-organizationName }
  X520OrganizationName ::= DirectoryString {ub-organization-name}

  -- Naming attributes of type X520OrganizationalUnitName
```

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

  at-x520OrganizationalUnitName ATTRIBUTE ::=
      { TYPE DirectoryString  {ub-organizational-unit-name}
          IDENTIFIED BY id-at-organizationalUnitName }
  X520OrganizationalUnitName ::= DirectoryString
                                    {ub-organizational-unit-name}

  -- Naming attributes of type X520Title

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

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

  -- Naming attributes of type X520dnQualifier

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

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

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

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

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

  -- Naming attributes of type X520SerialNumber

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

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

  -- Naming attributes of type X520Pseudonym

  id-at-pseudonym         AttributeType ::= { id-at 65 }

  at-x520Pseudonym ATTRIBUTE ::= { TYPE DirectoryString {ub-pseudonym}
      IDENTIFIED BY id-at-pseudonym }

  -- Naming attributes of type DomainComponent (from RFC 2247)

  id-domainComponent      AttributeType ::=
       { itu-t(0) data(9) pss(2342) ucl(19200300) pilot(100)
       pilotAttributeType(1) 25 }
```

```
  at-domainComponent ATTRIBUTE ::= {TYPE 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 }

  at-emailAddress ATTRIBUTE ::= {TYPE 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 SingleAttribute { {SupportedAttributes} }

  --  These are the known name elements for a DN

  SupportedAttributes ATTRIBUTE ::= {
      at-name | at-surname | at-givenName | at-initials |
      at-generationQualifier | at-x520CommonName |
      at-x520LocalityName | at-x520StateOrProvinceName |
      at-x520OrganizationName | at-x520OrganizationalUnitName |
      at-x520Title | at-x520dnQualifier | at-x520countryName |
      at-x520SerialNumber | at-x520Pseudonym | at-domainComponent |
      at-emailAddress, ... }

  --
  -- Certificate- and CRL-specific structures begin here
  --

  Certificate  ::=  SIGNED{TBSCertificate}

  TBSCertificate  ::=  SEQUENCE  {
      version         [0]  Version DEFAULT v1,
      serialNumber         CertificateSerialNumber,
      signature            AlgorithmIdentifier{SIGNATURE-ALGORITHM,
                               {SignatureAlgorithms}},
      issuer               Name,
```

```
      validity             Validity,
      subject              Name,
      subjectPublicKeyInfo SubjectPublicKeyInfo,
      ... ,
      [[2:                -- If present, version MUST be v2
      issuerUniqueID  [1]  IMPLICIT UniqueIdentifier OPTIONAL,
      subjectUniqueID [2]  IMPLICIT UniqueIdentifier OPTIONAL
      ]],
      [[3:                -- If present, version MUST be v3 --
      extensions      [3]  Extensions{{CertExtensions}} OPTIONAL
      ]], ... }

  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{PUBLIC-KEY,
                               {PublicKeyAlgorithms}},
      subjectPublicKey     BIT STRING  }

  -- CRL structures

  CertificateList  ::=  SIGNED{TBSCertList}

  TBSCertList  ::=  SEQUENCE  {
      version              Version OPTIONAL,
                                 -- if present, MUST be v2
      signature            AlgorithmIdentifier{SIGNATURE-ALGORITHM,
                               {SignatureAlgorithms}},
      issuer               Name,
      thisUpdate           Time,
      nextUpdate           Time OPTIONAL,
      revokedCertificates  SEQUENCE SIZE (1..MAX) OF SEQUENCE {
          userCertificate  CertificateSerialNumber,
          revocationDate   Time,
          ... ,
          [[2:                 -- if present, version MUST be v2
```

```
          crlEntryExtensions  Extensions{{CrlEntryExtensions}}
                                      OPTIONAL
          ]], ...
      } OPTIONAL,
      ... ,
      [[2:                          -- if present, version MUST be v2
      crlExtensions       [0] Extensions{{CrlExtensions}}
                                      OPTIONAL
      ]], ... }

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

  --
  --  The two object sets below should be expanded to include
  --  those algorithms which are supported by the system.
  --
  --  For example:
  --  SignatureAlgorithms SIGNATURE-ALGORITHM ::= {
  --    PKIXAlgs-2008.SignatureAlgs, ...,
  --        - - RFC 3279 provides the base set
  --    PKIX1-PSS-OAEP-ALGORITHMS.SignatureAlgs |
  --        - - RFC 4055 provides extension algs
  --    OtherModule.SignatureAlgs
  --        - - RFC XXXX provides additional extension algs
  --  }

  SignatureAlgorithms SIGNATURE-ALGORITHM ::= {
      PKIXAlgs-2009.SignatureAlgs, ...,
      PKIX1-PSS-OAEP-Algorithms-2009.SignatureAlgs }

  PublicKeyAlgorithms PUBLIC-KEY ::= {
      PKIXAlgs-2009.PublicKeys, ...,
      PKIX1-PSS-OAEP-Algorithms-2009.PublicKeys}

  -- Upper Bounds

  ub-state-name INTEGER ::= 128
  ub-organization-name INTEGER ::= 64
  ub-organizational-unit-name INTEGER ::= 64
  ub-title INTEGER ::= 64
  ub-serial-number INTEGER ::= 64
  ub-pseudonym INTEGER ::= 128
  ub-emailaddress-length INTEGER ::= 255
  ub-locality-name INTEGER ::= 128
  ub-common-name INTEGER ::= 64
  ub-name INTEGER ::= 32768
```

```
  -- 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
  --
  -- Three different versions of doing SIGNED:
  --  1.  Simple and close to the previous version
  --
  --  SIGNED{ToBeSigned} ::= SEQUENCE {
  --    toBeSigned  ToBeSigned,
  --    algorithm   AlgorithmIdentifier{SIGNATURE-ALGORITHM,
  --                    {SignatureAlgorithms}},
  --    signature   BIT STRING
  --  }

  --  2.  From Authenticated Framework
  --
  --  SIGNED{ToBeSigned} ::= SEQUENCE {
  --    toBeSigned        ToBeSigned,
  --    COMPONENTS OF SIGNATURE{ToBeSigned}
  --  }
  --  SIGNATURE{ToBeSigned} ::= SEQUENCE {
  --    algorithmIdentifier   AlgorithmIdentifier,
  --    encrypted             ENCRYPTED-HASH{ToBeSigned}
  --  }
  --  ENCRYPTED-HASH{ToBeSigned} ::=
  --    BIT STRING
  --      (CONSTRAINED BY {
  --        shall be the result of applying a hashing procedure to
  --        the DER-encoded (see 4.1) octets of a value of
  --        ToBeSigned and then applying an encipherment procedure
  --        to those octets
  --      })
  --
  --
  --  3.  A more complex version, but one that automatically ties
  --      together both the signature algorithm and the
  --      signature value for automatic decoding.
  --
  SIGNED{ToBeSigned} ::= SEQUENCE {
```

```
    toBeSigned           ToBeSigned,
    algorithmIdentifier  SEQUENCE {
        algorithm        SIGNATURE-ALGORITHM.
                           &id({SignatureAlgorithms}),
        parameters       SIGNATURE-ALGORITHM.
                           &Params({SignatureAlgorithms}
                             {@algorithmIdentifier.algorithm}) OPTIONAL
    },
    signature BIT STRING (CONTAINING SIGNATURE-ALGORITHM.&Value(
                             {SignatureAlgorithms}
                             {@algorithmIdentifier.algorithm}))
}

END
```

```
PKIX1Implicit-2009
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-implicit-02(59)}
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
IMPORTS

AttributeSet{}, EXTENSION, ATTRIBUTE
FROM PKIX-CommonTypes-2009
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-pkixCommon-02(57) }

id-pe, id-kp, id-qt-unotice, id-qt-cps, ORAddress, Name,
    RelativeDistinguishedName, CertificateSerialNumber,
    DirectoryString{}, SupportedAttributes
FROM PKIX1Explicit-2009
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-explicit-02(51) };

CertExtensions EXTENSION ::= {
        ext-AuthorityKeyIdentifier | ext-SubjectKeyIdentifier |
        ext-KeyUsage | ext-PrivateKeyUsagePeriod |
        ext-CertificatePolicies | ext-PolicyMappings |
        ext-SubjectAltName | ext-IssuerAltName |
        ext-SubjectDirectoryAttributes |
        ext-BasicConstraints | ext-NameConstraints |
        ext-PolicyConstraints | ext-ExtKeyUsage |
        ext-CRLDistributionPoints | ext-InhibitAnyPolicy |
        ext-FreshestCRL | ext-AuthorityInfoAccess |
        ext-SubjectInfoAccessSyntax, ... }

CrlExtensions EXTENSION ::= {
```

```
            ext-AuthorityKeyIdentifier | ext-IssuerAltName |
            ext-CRLNumber | ext-DeltaCRLIndicator |
            ext-IssuingDistributionPoint |  ext-FreshestCRL, ... }

   CrlEntryExtensions EXTENSION ::= {
            ext-CRLReason | ext-CertificateIssuer |
            ext-HoldInstructionCode | ext-InvalidityDate, ... }
   -- Shared arc for standard certificate and CRL extensions

   id-ce OBJECT IDENTIFIER  ::=  { joint-iso-ccitt(2) ds(5) 29 }

   -- authority key identifier OID and syntax

   ext-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 }
   (WITH COMPONENTS {
      ...,
      authorityCertIssuer        PRESENT,
      authorityCertSerialNumber  PRESENT
    } |
    WITH COMPONENTS {
      ...,
      authorityCertIssuer        ABSENT,
      authorityCertSerialNumber  ABSENT
    })

   KeyIdentifier ::= OCTET STRING

   -- subject key identifier OID and syntax

   ext-SubjectKeyIdentifier EXTENSION ::= { SYNTAX
       KeyIdentifier IDENTIFIED BY id-ce-subjectKeyIdentifier }
   id-ce-subjectKeyIdentifier OBJECT IDENTIFIER ::=  { id-ce 14 }

   -- 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), --  recent editions of X.509 have
                                     --  renamed this bit to
                                     --  contentCommitment
        keyEncipherment         (2),
        dataEncipherment        (3),
        keyAgreement            (4),
        keyCertSign             (5),
        cRLSign                 (6),
        encipherOnly            (7),
        decipherOnly            (8)
    }

   -- 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 }
   (WITH COMPONENTS {..., notBefore  PRESENT } |
    WITH COMPONENTS {..., notAfter  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 }

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

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

   CertPolicyId ::= OBJECT IDENTIFIER

   CERT-POLICY-QUALIFIER ::= TYPE-IDENTIFIER

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

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

   PolicyQualifierId CERT-POLICY-QUALIFIER ::=
       { pqid-cps | pqid-unotice, ... }

   pqid-cps CERT-POLICY-QUALIFIER ::= { CPSuri IDENTIFIED BY id-qt-cps }
   pqid-unotice CERT-POLICY-QUALIFIER ::= { UserNotice
       IDENTIFIED BY id-qt-unotice }

   -- CPS pointer qualifier

   CPSuri ::= IA5String

   -- user notice qualifier

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

   --
   --  This is not made explicit in the text
   --
   -- {WITH COMPONENTS {..., noticeRef PRESENT} |
   --  WITH COMPONENTS {..., DisplayText PRESENT }}

   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
       AttributeSet{{SupportedAttributes}}

   -- 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
   }
   --
   --  This is a constraint in the issued certificates by CAs, but is
   --  not a requirement on EEs.
   --
   -- (WITH COMPONENTS { ..., permittedSubtrees PRESENT} |
   --  WITH COMPONENTS { ..., excludedSubtrees PRESENT }}

   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 }
   --
   --  This is a constraint in the issued certificates by CAs,
   --  but is not a requirement for EEs
   --
   -- (WITH COMPONENTS { ..., requireExplicitPolicy PRESENT} |
   --  WITH COMPONENTS { ..., inhibitPolicyMapping PRESENT})

   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
   }
   --
   --  This is not a requirement in the text, but it seems as if it
   --      should be
   --
   --(WITH COMPONENTS {..., distributionPoint PRESENT} |
   -- WITH COMPONENTS {..., cRLIssuer PRESENT})

   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-ExtKeyUsage 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 }

   -- freshest (delta)CRL extension OID and syntax

   ext-FreshestCRL EXTENSION ::= {SYNTAX
       CRLDistributionPoints IDENTIFIED BY id-ce-freshestCRL }
   id-ce-freshestCRL OBJECT IDENTIFIER ::=  { id-ce 46 }

   -- authority info access

   ext-AuthorityInfoAccess 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  }

   -- subject info access

   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

   -- CRL number extension OID and syntax
```

```
   ext-CRLNumber EXTENSION ::= {SYNTAX
       INTEGER (0..MAX) IDENTIFIED BY id-ce-cRLNumber }
   id-ce-cRLNumber OBJECT IDENTIFIER ::= { id-ce 20 }

   CRLNumber ::= INTEGER (0..MAX)
   -- issuing distribution point extension OID and syntax

   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
   }
           -- at most one of onlyContainsUserCerts, onlyContainsCACerts,
           -- or onlyContainsAttributeCerts may be set to TRUE.

   ext-DeltaCRLIndicator 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

   holdInstruction OBJECT IDENTIFIER ::=
             {joint-iso-itu-t(2) member-body(2) us(840) x9cm(10040) 2}
   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 }
   -- Upper bounds
   ubMax INTEGER ::= 32768

   END


  --
  --  This module is used to isolate all the X.400 naming information.
  --  There is no reason to expect this to occur in a PKIX certificate.
  --

  PKIX-X400Address-2009
      {iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-x400address-02(60) }
  DEFINITIONS EXPLICIT TAGS ::=
  BEGIN

  -- 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,
     generation-qualifier [3] IMPLICIT PrintableString
                      (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 ::= {
      ea-commonName | ea-teletexCommonName | ea-teletexOrganizationName
      | ea-teletexPersonalName | ea-teletexOrganizationalUnitNames |
      ea-pDSName | ea-physicalDeliveryCountryName | ea-postalCode |
      ea-physicalDeliveryOfficeName | ea-physicalDeliveryOfficeNumber |
      ea-extensionORAddressComponents | ea-physicalDeliveryPersonalName
      | ea-physicalDeliveryOrganizationName |
      ea-extensionPhysicalDeliveryAddressComponents |
      ea-unformattedPostalAddress | ea-streetAddress |
      ea-postOfficeBoxAddress | ea-posteRestanteAddress |
      ea-uniquePostalName | ea-localPostalAttributes |
      ea-extendedNetworkAddress | ea-terminalType |
      ea-teletexDomainDefinedAttributes, ... }

  -- Extension types and attribute values

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

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

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

  ea-teletexPersonalName 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 }

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

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

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

```
  ea-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 }

  ea-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 }

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

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

  ea-extensionORAddressComponents EXTENSION-ATTRIBUTE ::=
      {PDSParameter IDENTIFIED BY 12 }

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

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

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

  ea-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 }

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

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

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

  ea-uniquePostalName EXTENSION-ATTRIBUTE ::=
```

```
      { PDSParameter IDENTIFIED BY 20 }

  ea-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 }

  ea-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 }

  ea-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

  ea-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)) }
```

```
  --  specifications of Upper Bounds MUST be regarded as mandatory
  --  from Annex B of ITU-T X.411 Reference Definition of MTS Parameter
  --  Upper Bounds
  -- Upper Bounds
  ub-match INTEGER ::= 128
  ub-common-name-length INTEGER ::= 64
  ub-country-name-alpha-length INTEGER ::= 2
  ub-country-name-numeric-length INTEGER ::= 3
  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-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.

  END
```

15. 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.

16. Normative References

[ASN1-2002] ITU-T, "ITU-T Recommendation X.680, X.681, X.682, and X.683", ITU-T X.680, X.681, X.682, and X.683, 2002.

[RFC2560] Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 2560, June 1999.

[RFC2986] Nystrom, M. and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7", RFC 2986, November 2000.

[RFC3279] Bassham, L., Polk, W., and R. Housley, "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3279, April 2002.

[RFC3852] Housley, R., "Cryptographic Message Syntax (CMS)", RFC 3852, July 2004.

[RFC4055] Schaad, J., Kaliski, B., and R. Housley, "Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 4055, June 2005.

[RFC4210] Adams, C., Farrell, S., Kause, T., and T. Mononen, "Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)", RFC 4210, September 2005.

[RFC4211] Schaad, J., "Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)", RFC 4211, September 2005.

[RFC5055] Freeman, T., Housley, R., Malpani, A., Cooper, D., and W. Polk, "Server-Based Certificate Validation Protocol (SCVP)", RFC 5055, December 2007.

[RFC5272] Schaad, J. and M. Myers, "Certificate Management over CMS (CMC)", RFC 5272, June 2008.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[RFC5480] Turner, S., Brown, D., Yiu, K., Housley, R., and T. Polk, "Elliptic Curve Cryptography Subject Public Key Information", RFC 5480, March 2009.

[RFC5755] Farrell, S., Housley, R., and S. Turner, "An Internet Attribute Certificate Profile for Authorization", RFC 5755, January 2010.

[RFC5911] Hoffman, P. and J. Schaad, "New ASN.1 Modules for Cryptographic Message Syntax (CMS) and S/MIME", RFC 5911, June 2010.

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