

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: June 15, 2011

J. Schaad
Soaring Hawk Consulting
S. Turner
IECA, Inc.
December 12, 2010

Additional New ASN.1 Modules
draft-turner-additional-new-asn-05

Abstract

The Cryptographic Message Syntax (CMS) 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 some auxiliary ASN.1 modules to conform to the 2008 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 Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on June 15, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in [Section 4.e](#) of

Internet-Draft

Additional New ASN.1 Modules

December 2010

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1.	Introduction	3
1.1.	ASN.1 Updates (2002 to 2008)	3
1.2.	Requirements Terminology	4
2.	ASN.1 Module RFC 3274	5
3.	ASN.1 Module RFC 3379	8
4.	ASN.1 Module RFC 4049	11
5.	ASN.1 Module RFC 4073	13
6.	ASN.1 Module RFC 4231	15
7.	ASN.1 Module RFC 4334	18
8.	ASN.1 Module RFC 5083	20
9.	ASN.1 Module RFC 5652	22
10.	ASN.1 Module RFC 5752	33
11.	Module Identifiers in ASN.1	35
12.	Security Considerations	37
13.	IANA Considerations	38
14.	Normative References	39
	Authors' Addresses	41

1. Introduction

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

In this document we have either change the syntax to use the 2008 ASN.1 standard, or done some updates from previous conversions:

[RFC 3274](#), Compressed Data Content Type for Cryptographic Message Syntax (CMS) [[RFC3274](#)].

[RFC 3379](#), Delegated Path Validation and Delegated Path Discovery Protocol Requirements [[RFC3379](#)].

[RFC 4049](#), BinaryTime: An Alternate Format for Representing Date and Time in ASN.1 [[RFC4049](#)].

[RFC 4073](#), Protecting Multiple Contents with the Cryptographic Message Syntax (CMS) [[RFC4073](#)].

[RFC 4231](#), Identifiers and Test Vectors for HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 [[RFC4231](#)].

[RFC 4334](#), Certificate Extensions and Attributes Supporting Authentication in Point-to-Point Protocol (PPP) and Wireless Local Area Networks (WLAN) [[RFC4334](#)].

[RFC 5083](#), Cryptographic Message Syntax (CMS) Authenticated-Enveloped-Data Content Type [[RFC5083](#)].

[RFC 5652](#), Cryptogrphic Message Syntax (CMS) [[RFC5652](#)].

[RFC 5752](#), Multiple Signatures in Cryptographic Message Syntax

(CMS) [[RFC5752](#)].

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 [[RFC5912](#)] and [[RFC5911](#)] can stand on their own and do not need to import definitions from anywhere else.

[1.1.](#) ASN.1 Updates (2002 to 2008)

The modules defined in this document are compatible with the most current ASN.1 specification published in 2008 (see [[ASN1-2008](#)]). The

Schaad & Turner

Expires June 15, 2011

[Page 3]

Internet-Draft

Additional New ASN.1 Modules

December 2010

changes between the 2002 specification and the 2008 specification include the creation of some additional pre-defined types (DATE, DATE-TIME, DURATION, NOT-A-NUMBER, OID-IRI, RELATIVE-OID-IRI, TIME, TIME-OF-DAY). The ability to define different encoding rules (ENCODING-CONTROL, INSTRUCTIONS). None of the newly defined tokens are currently used in any of the ASN.1 specifications published here.

[1.2.](#) Requirements Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [[RFC2119](#)].

2. ASN.1 Module [RFC 3274](#)

We have updated the ASN.1 module associated with this document to be 2008 compliant and to use the set of classes previously defined in [\[RFC5911\]](#).

CompressedDataContent

```
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
  smime(16) modules(0) id-mod-compressedDataContent(54) }
```

DEFINITIONS IMPLICIT TAGS ::=
BEGIN

IMPORTS

CMSVersion, EncapsulatedContentInfo,
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-2009(58) }
```

AlgorithmIdentifier{}, SMIME-CAPS, ParamOptions
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)}
;

--
-- ContentTypes contains the set of content types that are
--   defined in this module.
--
-- The contents of ContentTypes should be added to
--   ContentSet defined in [RFC5652]
--

ContentTypes CONTENT-TYPE ::= {ct-compressedData}

--
-- SMimeCaps contains the set of S/MIME capabilities that
--   are associated with the algorithms defined in this
--   document.
--
-- SMimeCaps are added to SMimeCapsSet defined in [RFC3851].
--

SMimeCaps SMIME-CAPS ::= {cpa-zlibCompress.&smimeCaps}

--

```

```

-- Define the compressed data content type
--

ct-compressedData CONTENT-TYPE ::= {
  TYPE CompressedData IDENTIFIED BY id-ct-compressedData
}

CompressedData ::= SEQUENCE {
  version CMSVersion (v0), -- Always set to 0
  compressionAlgorithm CompressionAlgorithmIdentifier,
  encapContentInfo EncapsulatedContentInfo
}

CompressionAlgorithmIdentifier ::=
  AlgorithmIdentifier{COMPRESS-ALGORITHM, {CompressAlgorithmSet}}

```

```

CompressAlgorithmSet COMPRESS-ALGORITHM ::= {
    cpa-zlibCompress, ...
}

-- Algorithm Identifiers

id-alg-zlibCompress OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) alg(3) 8 }

cpa-zlibCompress COMPRESS-ALGORITHM ::= {
    IDENTIFIER id-alg-zlibCompress
    PARAMS TYPE NULL ARE preferredAbsent
    SMIME-CAPS {IDENTIFIED BY id-alg-zlibCompress}
}

-- Content Type Object Identifiers

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

--
-- Class defined for compression algorithms
--

COMPRESS-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]
}

```

END

We have updated the ASN.1 module associated with [RFC 3379](#) to be ASN.1 2008 compliant and to use the set of classes previously defined in [\[RFC5912\]](#).

```
IPAddrAndASCertExtn { iso(1) identified-organization(3) dod(6)
    internet(1) security(5) mechanisms(5) pkix(7) mod(0)
    id-mod-ip-addr-and-as-ident-2(72) }
DEFINITIONS EXPLICIT TAGS ::=
BEGIN
    EXPORTS ALL;

    IMPORTS

    -- PKIX specific OIDs and arcs --
    id-pe
    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)}

    EXTENSION
    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)}
    ;

    --
    -- Extensions contains the set of extensions defined in this
    -- module
    --
    -- These are intended to be placed in public key certificates
    -- and thus should be added to the CertExtensions extension
    -- set in PKIXImplicit-2009 defined for \[RFC5280\]
    --

    Extensions EXTENSION ::= {
        ext-pe-ipAddrBlocks | ext-pe-autonomousSysIds
    }

    -- IP Address Delegation Extension OID --

    ext-pe-ipAddrBlocks EXTENSION ::= {
        SYNTAX IPAddrBlocks
        IDENTIFIED BY id-pe-ipAddrBlocks
    }
```

```
id-pe-ipAddrBlocks  OBJECT IDENTIFIER ::= { id-pe 7 }

-- IP Address Delegation Extension Syntax --

IPAddrBlocks        ::= SEQUENCE OF IPAddressFamily

IPAddressFamily      ::= SEQUENCE { -- AFI & opt SAFI --
    addressFamily     OCTET STRING (SIZE (2..3)),
    ipAddressChoice   IPAddressChoice }

IPAddressChoice      ::= CHOICE {
    inherit            NULL, -- inherit from issuer --
    addressesOrRanges SEQUENCE OF IPAddressOrRange }

IPAddressOrRange     ::= CHOICE {
    addressPrefix      IPAddress,
    addressRange       IPAddressRange }

IPAddressRange       ::= SEQUENCE {
    min                IPAddress,
    max                IPAddress }

IPAddress            ::= BIT STRING

-- Autonomous System Identifier Delegation Extension OID --

ext-pe-autonomousSysIds EXTENSION ::= {
    SYNTAX ASIdentifiers
    IDENTIFIED BY id-pe-autonomousSysIds
}

id-pe-autonomousSysIds OBJECT IDENTIFIER ::= { id-pe 8 }

-- Autonomous System Identifier Delegation Extension Syntax --

ASIdentifiers        ::= SEQUENCE {
    asnum              [0] ASIdentifierChoice OPTIONAL,
    rdi                [1] ASIdentifierChoice OPTIONAL }
    (WITH COMPONENTS {..., asnum PRESENT} |
    WITH COMPONENTS {..., rdi PRESENT})

ASIdentifierChoice    ::= CHOICE {
    inherit            NULL, -- inherit from issuer --
    asIdsOrRanges      SEQUENCE OF ASIdOrRange }

ASIdOrRange          ::= CHOICE {
```

id range	ASId, ASRange }
-------------	--------------------

Schaad & Turner

Expires June 15, 2011

[Page 9]

Internet-Draft

Additional New ASN.1 Modules

December 2010

ASRange	::= SEQUENCE {
min	ASId,
max	ASId }

ASId	::= INTEGER
------	-------------

END

4. ASN.1 Module [RFC 4049](#)

We have updated the ASN.1 module associated with this document to be 2008 compliant and to use the set of classes previously defined in [\[RFC5911\]](#).

```
BinarySigningTimeModule-2009
  { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
    pkcs-9(9) smime(16) modules(0)
    id-mod-binSigningTime-2009(55) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
  IMPORTS

  -- From PKIX-CommonTypes-2009 [RFC5912]

  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) }
;

--
-- BinaryTime Definition
--
-- BinaryTime contains the number seconds since
-- midnight Jan 1, 1970 UTC.
-- Leap seconds are EXCLUDED from the computation.
--
```

```

BinaryTime ::= INTEGER (0..MAX)

--
-- Signing Binary Time Attribute
--
-- The binary signing time should be added to
-- SignedAttributeSet and tAuthenticatedAttributeSet
-- in CMS [RFC5652] and to AuthEnvDataAttributeSet
-- in [RFC5083].
--

aa-binarySigningTime ATTRIBUTE ::= {
    TYPE BinarySigningTime
    IDENTIFIED BY id-aa-binarySigningTime }

id-aa-binarySigningTime OBJECT IDENTIFIER ::= { iso(1)
    member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
    smime(16) aa(2) 46 }

BinarySigningTime ::= BinaryTime

END

```

5. ASN.1 Module [RFC 4073](#)

We have updated the ASN.1 module associated with this document to be 2008 compliant and to use the set of classes previously defined in [[RFC5911](#)].

```

ContentCollectionModule-2009
    { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
      pkcs-9(9) smime(16) modules(0) id-mod-context-Collect-2009(56) }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
    IMPORTS

-- From CryptographicMessageSyntax-2009 [RFC5911]

CONTENT-TYPE, 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-2009(58) }

AttributeSet{}, 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) }
;

--
-- An object set of all content types defined by this module.
-- This is to be added to ContentSet in the CMS module
--

ContentSet CONTENT-TYPE ::= {
  ct-ContentCollection | ct-ContentWithAttributes, ...
}

--
-- Content Collection Content Type and Object Identifier
--

ct-ContentCollection CONTENT-TYPE ::= {
  TYPE ContentCollection IDENTIFIED BY id-ct-contentCollection }

id-ct-contentCollection OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
  smime(16) ct(1) 19 }

ContentCollection ::= SEQUENCE SIZE (1..MAX) OF ContentInfo

```

```

--
-- Content With Attributes Content Type and Object Identifier
--

ct-ContentWithAttributes CONTENT-TYPE ::= {
  TYPE ContentWithAttributes IDENTIFIED BY id-ct-contentWithAttrs }

id-ct-contentWithAttrs OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
  smime(16) ct(1) 20 }

```

```
ContentWithAttributes ::= SEQUENCE {
    content  ContentInfo,
    attrs    SEQUENCE SIZE (1..MAX) OF AttributeSet
                                     {{ ContentAttributeSet }}
}

ContentAttributeSet ATTRIBUTE ::= { ... }
END
```

[6.](#) ASN.1 Module [RFC 4231](#)

[RFC 4231](#) does not contain an ASN.1 module to be updated. We have therefore created an ASN.1 module to represent the ASN.1 that is

present in the document. Note that the parameters are defined as expecting a parameter for the algorithm identifiers in this module, this is different from most of the algorithms used in PKIX and S/MIME. There is no concept of being able to truncate the MAC value in the ASN.1 unlike the XML definitions. This is reflected by not having a minimum MAC length defined in the ASN.1.

```
HMAC { iso(1) identified-organization(3) dod(6) internet(1)
  security(5) mechanisms(5) pkix(7) mod(0) id-mod-hmac(74) }
DEFINITIONS EXPLICIT TAGS ::=
BEGIN
  EXPORTS ALL;

  IMPORTS

  MAC-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)};

  --
  -- This object set contains all of the MAC algorithms that are
  -- defined in this module.
  -- One would add it to a constraining set of objects such as the
  -- MessageAuthenticationCodeAlgorithmSet in [RFC5652]
  --

  MessageAuthAlgs MAC-ALGORITHM ::= {
    maca-hMAC-SHA224 |
    maca-hMAC-SHA256 |
    maca-hMAC-SHA384 |
    maca-hMAC-SHA512
  }

  --
  -- This object set contains all of the S/MIME capabilities that
  -- have been defined for all the MAC algorithms in this module.
  -- One would add this to an object set that is used to restrict
  -- smime capabilities such as the SMimeCapsSet variable in
  -- the S/MIME message draft
  --

  SMimeCaps SMIME-CAPS ::= {
```

```
    maca-hMAC-SHA224.&smimeCaps      |
    maca-hMAC-SHA256.&smimeCaps      |
    maca-hMAC-SHA384.&smimeCaps      |
    maca-hMAC-SHA512.&smimeCaps
  }

--
-- Define the base OID for the algorithm identifiers
--

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

digestAlgorithm OBJECT IDENTIFIER ::= {rsadsi 2}

--
-- Define the necessary algorithm identifiers
--

id-hmacWithSHA224 OBJECT IDENTIFIER ::= {digestAlgorithm 8}
id-hmacWithSHA256 OBJECT IDENTIFIER ::= {digestAlgorithm 9}
id-hmacWithSHA384 OBJECT IDENTIFIER ::= {digestAlgorithm 10}
id-hmacWithSHA512 OBJECT IDENTIFIER ::= {digestAlgorithm 11}

--
-- Define each of the MAC-ALGORITHM objects to describe the
--   algorithms defined
--

maca-hMAC-SHA224 MAC-ALGORITHM ::= {
    IDENTIFIER id-hmacWithSHA224
    PARAMS TYPE NULL ARE preferredPresent
    IS-KEYED-MAC TRUE
    SMIME-CAPS {IDENTIFIED BY id-hmacWithSHA224}
}

maca-hMAC-SHA256 MAC-ALGORITHM ::= {
    IDENTIFIER id-hmacWithSHA256
    PARAMS TYPE NULL ARE preferredPresent
    IS-KEYED-MAC TRUE
    SMIME-CAPS {IDENTIFIED BY id-hmacWithSHA256}
}

maca-hMAC-SHA384 MAC-ALGORITHM ::= {
    IDENTIFIER id-hmacWithSHA384
```

PARAMS TYPE NULL ARE preferredPresent

Schaad & Turner

Expires June 15, 2011

[Page 16]

Internet-Draft

Additional New ASN.1 Modules

December 2010

```
IS-KEYED-MAC TRUE
SMIME-CAPS {IDENTIFIED BY id-hmacWithSHA384}
}
```

```
maca-hMAC-SHA512 MAC-ALGORITHM ::= {
  IDENTIFIER id-hmacWithSHA512
  PARAMS TYPE NULL ARE preferredPresent
  IS-KEYED-MAC TRUE
  SMIME-CAPS {IDENTIFIED BY id-hmacWithSHA512}
}
```

END

Internet-Draft

Additional New ASN.1 Modules

December 2010

7. ASN.1 Module [RFC 4334](#)

We have updated the ASN.1 module associated with [RFC 4334](#) to be ASN.1 2008 compliant and to use the set of classes previously defined in [\[RFC5912\]](#).

WLANCertExtn

```
{ iso(1) identified-organization(3) dod(6) internet(1)
  security(5) mechanisms(5) pkix(7) id-mod(0)
  id-mod-wlan-extns-2(73) }
```

DEFINITIONS IMPLICIT TAGS ::=

BEGIN

EXPORTS ALL;

IMPORTS

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

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)}
```

id-aca

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)}
```

;

-- Extended Key Usage Values

```
KeyUsageValues OBJECT IDENTIFIER ::= {  
    id-kp-eapOverPPP | id-kp-eapOverLAN  
}
```

```
id-kp-eapOverPPP OBJECT IDENTIFIER ::= { id-kp 13 }
```

```
id-kp-eapOverLAN OBJECT IDENTIFIER ::= { id-kp 14 }
```

-- Wireless LAN SSID Extension

```
ext-pe-wlanSSID EXTENSION ::= {  
    SYNTAX SSIDList  
    IDENTIFIED BY id-pe-wlanSSID  
    CRITICALITY {FALSE}  
}
```

```
id-pe-wlanSSID OBJECT IDENTIFIER ::= { id-pe 13 }
```

```
SSIDList ::= SEQUENCE SIZE (1..MAX) OF SSID
```

```
SSID ::= OCTET STRING (SIZE (1..32))
```

-- Wireless LAN SSID Attribute Certificate Attribute
-- Uses same syntax as the certificate extension: SSIDList

```
at-aca-wlanSSID ATTRIBUTE ::= {  
    TYPE SSIDList  
    IDENTIFIED BY id-aca-wlanSSID  
}
```

```
id-aca-wlanSSID OBJECT IDENTIFIER ::= { id-aca 7 }
```

END

8. ASN.1 Module [RFC 5083](#)

This module is updated from [RFC 5911](#) [[RFC5911](#)] by the following changes:

1. Define separate attribute sets for the unprotected attributes used in EnvelopedData, EncryptedData and AuthenticatedEnvelopedData ([RFC 5083](#)).
2. Define a parameterized type EncryptedContentInfoType so that the basic type can be used with algorithm sets (used for EnvelopedData, EncryptedData and AuthenticatedEnvelopedData ([RFC 5083](#))). The parameterized type is assigned to an unparameterized type of EncryptedContentInfo to minimize the output changes from previous versions.

The use of different attribute sets for EncryptedData and EnvelopedData as well as for AuthenticatedData and AuthEnvelopedData, protocol designers can make use of the '08 ASN.1 constraints to

define different sets of attributes for EncryptedData and EnvelopedData and for AuthenticatedData and AuthEnvelopedData. Previously, attributes could only be constrained based on whether they were in the clear or unauthenticated not on the encapsulating content type.

CMS-AuthEnvelopedData-2009

```
{iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
  smime(16) modules(0) id-mod-cmsAuthEnvData-2009(57) }
```

DEFINITIONS IMPLICIT TAGS ::=

BEGIN

IMPORTS

CMSVersion, EncryptedContentInfoType{},

```
  MessageAuthenticationCode, OriginatorInfo, RecipientInfos,
  CONTENT-TYPE, Attributes{}, ATTRIBUTE, CONTENT-ENCRYPTION,
  AlgorithmIdentifier{},
```

```
  aa-signingTime, aa-messageDigest, aa-contentType
```

FROM CryptographicMessageSyntax-2009

```
{ iso(1) member-body(2) us(840) rsadsi(113549)
  pkcs(1) pkcs-9(9) smime(16) modules(0) id-mod-cms-2009(58) }
```

ContentEncryptionAlgs

FROM CMS-AES-CCM-and-AES-GCM-2009

```
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
  pkcs-9(9) smime(16) modules(0) id-mod-cms-aes-ccm-gcm-02(44) }
```

;

ContentTypes CONTENT-TYPE ::= {ct-authEnvelopedData, ... }

```
ct-authEnvelopedData CONTENT-TYPE ::= {
  TYPE AuthEnvelopedData IDENTIFIED BY id-ct-authEnvelopedData
}
```

id-ct-authEnvelopedData OBJECT IDENTIFIER ::=

```
{iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
  smime(16) ct(1) 23}
```

AuthEnvelopedData ::= SEQUENCE {

```
  version CMSVersion,
  originatorInfo [0] IMPLICIT OriginatorInfo OPTIONAL,
  recipientInfos RecipientInfos,
```

```

    authEncryptedContentInfo EncryptedContentInfo,
    authAttrs [1] IMPLICIT AuthAttributes OPTIONAL,
    mac MessageAuthenticationCode,
    unauthAttrs [2] IMPLICIT UnauthAttributes OPTIONAL
}

EncryptedContentInfo ::=
    EncryptedContentInfoType { AuthContentEncryptionAlgorithmIdentifier }

AuthContentEncryptionAlgorithmIdentifier ::= AlgorithmIdentifier
    {CONTENT-ENCRYPTION, {AuthContentEncryptionAlgorithmSet}}

AuthContentEncryptionAlgorithmSet CONTENT-ENCRYPTION ::= {
    ContentEncryptionAlgs, ...}

AuthAttributes ::= Attributes{{AuthEnvDataAttributeSet}}

UnauthAttributes ::= Attributes{{UnauthEnvDataAttributeSet}}

AuthEnvDataAttributeSet ATTRIBUTE ::= {
    aa-contentType | aa-messageDigest | aa-signingTime, ... }

UnauthEnvDataAttributeSet ATTRIBUTE ::= {...}

END

```

9. ASN.1 Module [RFC 5652](#)

This module is updated from [RFC 5911](#) [[RFC5911](#)] by the following changes:

1. Define separate attribute sets for the unprotected attributes used in EnvelopedData, EncryptedData and AuthenticatedEnvelopedData ([RFC 5083](#)).
2. Define a parameterized type EncryptedContentInfoType so that the basic type can be used with algorithm sets (used for EnvelopedData, EncryptedData and AuthenticatedEnvelopedData ([RFC 5083](#))). The parameterized type is assigned to an unparameterized type of EncryptedContentInfo to minimize the output changes from previous versions.

The use of different attribute sets for EncryptedData and EnvelopedData as well as for AuthenticatedData and AuthEnvelopedData, protocol designers can make use of the '08 ASN.1 constraints to define different sets of attributes for EncryptedData and EnvelopedData and for AuthenticatedData and AuthEnvelopedData. Previously, attributes could only be constrained based on whether they were in the clear or unauthenticated not on the encapsulating content type.

CryptographicMessageSyntax-2009

```
{ iso(1) member-body(2) us(840) rsadsi(113549)
  pkcs(1) pkcs-9(9) smime(16) modules(0) id-mod-cms-2009(58) }
```

DEFINITIONS IMPLICIT TAGS ::=

BEGIN

IMPORTS

```
ParamOptions, DIGEST-ALGORITHM, SIGNATURE-ALGORITHM,
PUBLIC-KEY, KEY-DERIVATION, KEY-WRAP, MAC-ALGORITHM,
KEY-AGREE, KEY-TRANSPORT, CONTENT-ENCRYPTION, 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) }
```

```
SignatureAlgs, MessageDigestAlgs, KeyAgreementAlgs,
MessageAuthAlgs, KeyWrapAlgs, ContentEncryptionAlgs,
KeyTransportAlgs, KeyDerivationAlgs, KeyAgreePublicKeys
```

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) }
```

```
Certificate, CertificateList, CertificateSerialNumber,  
    Name, ATTRIBUTE  
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) }
```

```
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) }
```

```
AttributeCertificateV1  
FROM AttributeCertificateVersion1-2009  
    { iso(1) identified-organization(3) dod(6) internet(1)  
      security(5) mechanisms(5) pkix(7) id-mod(0)  
      id-mod-v1AttrCert-02(49) } ;
```

```
-- Cryptographic Message Syntax
```

```
-- The following are used for version numbers using the ASN.1  
-- idiom "[[n:"  
-- Version 1 = PKCS #7  
-- Version 2 = S/MIME V2  
-- Version 3 = RFC 2630  
-- Version 4 = RFC 3369  
-- Version 5 = RFC 3852
```

```
CONTENT-TYPE ::= CLASS {  
    &id          OBJECT IDENTIFIER UNIQUE,  
    &Type        OPTIONAL  
} WITH SYNTAX {  
    [TYPE &Type] IDENTIFIED BY &id  
}
```

```
ContentType ::= CONTENT-TYPE.&id
```

```
ContentInfo ::= SEQUENCE {  
    contentType      CONTENT-TYPE.  
                    &id({ContentSet}),  
    content          [0] EXPLICIT CONTENT-TYPE.  
                    &Type({ContentSet}{@contentType})}
```

```
ContentSet CONTENT-TYPE ::= {  
    -- Define the set of content types to be recognized.  
    ct-Data | ct-SignedData | ct-EncryptedData | ct-EnvelopedData |  
    ct-AuthenticatedData | ct-DigestedData, ... }
```

Internet-Draft

Additional New ASN.1 Modules

December 2010

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

```
SignerInfos ::= SET OF SignerInfo
```

```
EncapsulatedContentInfo ::= SEQUENCE {  
    eContentType          CONTENT-TYPE.&id({ContentSet}),  
    eContent              [0] EXPLICIT OCTET STRING  
        ( CONTAINING CONTENT-TYPE.  
            &Type({ContentSet}{@eContentType})) OPTIONAL }
```

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

```
SignedAttributes ::= Attributes {{ SignedAttributesSet }}
```

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

```
SignedAttributesSet ATTRIBUTE ::=  
    { aa-signingTime | aa-messageDigest | aa-contentType, ... }
```

```
UnsignedAttributes ATTRIBUTE ::= { aa-countersignature, ... }
```

```
SignatureValue ::= OCTET STRING
```

```
EnvelopedData ::= SEQUENCE {  
    version CMSVersion,
```

```

    originatorInfo [0] IMPLICIT OriginatorInfo OPTIONAL,
    recipientInfos RecipientInfos,
    encryptedContentInfo EncryptedContentInfo,
    ...,
    [[2: unprotectedAttrs [1] IMPLICIT Attributes
        {{ UnprotectedEnvAttributes }} OPTIONAL ]] }

```

```

OriginatorInfo ::= SEQUENCE {
    certs [0] IMPLICIT CertificateSet OPTIONAL,
    crls [1] IMPLICIT RevocationInfoChoices OPTIONAL }

RecipientInfos ::= SET SIZE (1..MAX) OF RecipientInfo

EncryptedContentInfo ::=
    EncryptedContentInfoType { ContentEncryptionAlgorithmIdentifier }

EncryptedContentInfoType { AlgorithmIdentifierType } ::= SEQUENCE {
    contentType          CONTENT-TYPE.&id({ContentSet}),
    contentEncryptionAlgorithm AlgorithmIdentifierType,
    encryptedContent      [0] IMPLICIT OCTET STRING OPTIONAL }

-- If you want to do constraints, you might use:
-- EncryptedContentInfo ::= SEQUENCE {
--     contentType          CONTENT-TYPE.&id({ContentSet}),
--     contentEncryptionAlgorithm ContentEncryptionAlgorithmIdentifier,
--     encryptedContent      [0] IMPLICIT ENCRYPTED {CONTENT-TYPE.
--         &Type({ContentSet}{@contentType}) OPTIONAL }
--     ENCRYPTED {ToBeEncrypted} ::= OCTET STRING ( CONSTRAINED BY
--         { ToBeEncrypted } )

UnprotectedEnvAttributes ATTRIBUTE ::= { ... }
UnprotectedEncAttributes ATTRIBUTE ::= { ... }

RecipientInfo ::= CHOICE {
    ktri          KeyTransRecipientInfo,
    ...,
    [[3: kari [1] KeyAgreeRecipientInfo ]],
    [[4: kekri [2] KEKRecipientInfo]],
    [[5: pwri [3] PasswordRecipientInfo,
        ori [4] OtherRecipientInfo ]] }

```

EncryptedKey ::= OCTET STRING

KeyTransRecipientInfo ::= SEQUENCE {
 version CMSVersion, -- always set to 0 or 2
 rid RecipientIdentifier,
 keyEncryptionAlgorithm AlgorithmIdentifier
 {KEY-TRANSPORT, {KeyTransportAlgorithmSet}},
 encryptedKey EncryptedKey }

KeyTransportAlgorithmSet KEY-TRANSPORT ::= { KeyTransportAlgs, ... }

RecipientIdentifier ::= CHOICE {
 issuerAndSerialNumber IssuerAndSerialNumber,

...,
 [[2: subjectKeyIdentifier [0] SubjectKeyIdentifier]] }
KeyAgreeRecipientInfo ::= SEQUENCE {
 version CMSVersion, -- always set to 3
 originator [0] EXPLICIT OriginatorIdentifierOrKey,
 ukm [1] EXPLICIT UserKeyingMaterial OPTIONAL,
 keyEncryptionAlgorithm AlgorithmIdentifier
 {KEY-AGREE, {KeyAgreementAlgorithmSet}},
 recipientEncryptedKeys RecipientEncryptedKeys }

KeyAgreementAlgorithmSet KEY-AGREE ::= { KeyAgreementAlgs, ... }

OriginatorIdentifierOrKey ::= CHOICE {
 issuerAndSerialNumber IssuerAndSerialNumber,
 subjectKeyIdentifier [0] SubjectKeyIdentifier,
 originatorKey [1] OriginatorPublicKey }

OriginatorPublicKey ::= SEQUENCE {
 algorithm AlgorithmIdentifier {PUBLIC-KEY, {OriginatorKeySet}},
 publicKey BIT STRING }

OriginatorKeySet PUBLIC-KEY ::= { KeyAgreePublicKeys, ... }

RecipientEncryptedKeys ::= SEQUENCE OF RecipientEncryptedKey

RecipientEncryptedKey ::= SEQUENCE {
 rid KeyAgreeRecipientIdentifier,
 encryptedKey EncryptedKey }

```
KeyAgreeRecipientIdentifier ::= CHOICE {  
    issuerAndSerialNumber IssuerAndSerialNumber,  
    rKeyId [0] IMPLICIT RecipientKeyIdentifier }
```

```
RecipientKeyIdentifier ::= SEQUENCE {  
    subjectKeyIdentifier SubjectKeyIdentifier,  
    date GeneralizedTime OPTIONAL,  
    other OtherKeyAttribute OPTIONAL }
```

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

```
KEKRecipientInfo ::= SEQUENCE {  
    version CMSVersion, -- always set to 4  
    kekid KEKIdentifier,  
    keyEncryptionAlgorithm KeyEncryptionAlgorithmIdentifier,  
    encryptedKey EncryptedKey }
```

```
KEKIdentifier ::= SEQUENCE {  
    keyIdentifier OCTET STRING,
```

```
    date GeneralizedTime OPTIONAL,  
    other OtherKeyAttribute OPTIONAL }  
PasswordRecipientInfo ::= SEQUENCE {  
    version CMSVersion, -- always set to 0  
    keyDerivationAlgorithm [0] KeyDerivationAlgorithmIdentifier  
        OPTIONAL,  
    keyEncryptionAlgorithm KeyEncryptionAlgorithmIdentifier,  
    encryptedKey EncryptedKey }
```

```
OTHER-RECIPIENT ::= TYPE-IDENTIFIER
```

```
OtherRecipientInfo ::= SEQUENCE {  
    oriType    OTHER-RECIPIENT.  
        &id({SupportedOtherRecipInfo}),  
    oriValue   OTHER-RECIPIENT.  
        &Type({SupportedOtherRecipInfo}{@oriType})}
```

```
SupportedOtherRecipInfo OTHER-RECIPIENT ::= { ... }
```

```
DigestedData ::= SEQUENCE {  
    version CMSVersion,
```

```
digestAlgorithm DigestAlgorithmIdentifier,  
encapContentInfo EncapsulatedContentInfo,  
digest Digest, ... }
```

Digest ::= OCTET STRING

```
EncryptedData ::= SEQUENCE {  
    version CMSVersion,  
    encryptedContentInfo EncryptedContentInfo,  
    ...,  
    [[2: unprotectedAttrs [1] IMPLICIT Attributes  
        {{UnprotectedEncAttributes}} OPTIONAL ]] }
```

```
AuthenticatedData ::= SEQUENCE {  
    version CMSVersion,  
    originatorInfo [0] IMPLICIT OriginatorInfo OPTIONAL,  
    recipientInfos RecipientInfos,  
    macAlgorithm MessageAuthenticationCodeAlgorithm,  
    digestAlgorithm [1] DigestAlgorithmIdentifier OPTIONAL,  
    encapContentInfo EncapsulatedContentInfo,  
    authAttrs [2] IMPLICIT AuthAttributes OPTIONAL,  
    mac MessageAuthenticationCode,  
    unauthAttrs [3] IMPLICIT UnauthAttributes OPTIONAL }
```

```
AuthAttributes ::= SET SIZE (1..MAX) OF Attribute  
    {{AuthAttributeSet}}
```

```
AuthAttributeSet ATTRIBUTE ::= { aa-contentType | aa-messageDigest  
                                | aa-signingTime, ... }
```

```
MessageAuthenticationCode ::= OCTET STRING
```

```
UnauthAttributes ::= SET SIZE (1..MAX) OF Attribute  
    {{UnauthAttributeSet}}
```

```
UnauthAttributeSet ATTRIBUTE ::= {...}
```

```
--
```

```
-- General algorithm definitions
```

```
--
```

```
DigestAlgorithmIdentifier ::= AlgorithmIdentifier
```

```

    {DIGEST-ALGORITHM, {DigestAlgorithmSet}}

DigestAlgorithmSet DIGEST-ALGORITHM ::= {
    CryptographicMessageSyntaxAlgorithms-2009.MessageDigestAlgs, ... }

SignatureAlgorithmIdentifier ::= AlgorithmIdentifier
    {SIGNATURE-ALGORITHM, {SignatureAlgorithmSet}}

SignatureAlgorithmSet SIGNATURE-ALGORITHM ::=
    { SignatureAlgs, ... }

KeyEncryptionAlgorithmIdentifier ::= AlgorithmIdentifier
    {KEY-WRAP, {KeyEncryptionAlgorithmSet}}

KeyEncryptionAlgorithmSet KEY-WRAP ::= { KeyWrapAlgs, ... }

ContentEncryptionAlgorithmIdentifier ::= AlgorithmIdentifier
    {CONTENT-ENCRYPTION, {ContentEncryptionAlgorithmSet}}

ContentEncryptionAlgorithmSet CONTENT-ENCRYPTION ::=
    { ContentEncryptionAlgs, ... }

MessageAuthenticationCodeAlgorithm ::= AlgorithmIdentifier
    {MAC-ALGORITHM, {MessageAuthenticationCodeAlgorithmSet}}

MessageAuthenticationCodeAlgorithmSet MAC-ALGORITHM ::=
    { MessageAuthAlgs, ... }

KeyDerivationAlgorithmIdentifier ::= AlgorithmIdentifier
    {KEY-DERIVATION, {KeyDerivationAlgs, ...}}

RevocationInfoChoices ::= SET OF RevocationInfoChoice

RevocationInfoChoice ::= CHOICE {

```

```

    crl CertificateList,
    ...,
    [[5: other [1] IMPLICIT OtherRevocationInfoFormat ]] }

OTHER-REVOK-INFO ::= TYPE-IDENTIFIER

OtherRevocationInfoFormat ::= SEQUENCE {

```



```

otherRevInfoFormat    OTHER-REVOK-INFO.
                        &id({SupportedOtherRevokInfo}),
otherRevInfo          OTHER-REVOK-INFO.
                        &Type({SupportedOtherRevokInfo}{@otherRevInfoFormat})}

SupportedOtherRevokInfo OTHER-REVOK-INFO ::= { ... }

CertificateChoices ::= CHOICE {
    certificate Certificate,
    extendedCertificate [0] IMPLICIT ExtendedCertificate,
    -- Obsolete
    ...,
    [[3: v1AttrCert [1] IMPLICIT AttributeCertificateV1]],
    -- Obsolete
    [[4: v2AttrCert [2] IMPLICIT AttributeCertificateV2]],
    [[5: other      [3] IMPLICIT OtherCertificateFormat]] }

AttributeCertificateV2 ::= AttributeCertificate

OTHER-CERT-FMT ::= TYPE-IDENTIFIER

OtherCertificateFormat ::= SEQUENCE {
    otherCertFormat OTHER-CERT-FMT.
                        &id({SupportedCertFormats}),
    otherCert       OTHER-CERT-FMT.
                        &Type({SupportedCertFormats}{@otherCertFormat})}

SupportedCertFormats OTHER-CERT-FMT ::= { ... }

CertificateSet ::= SET OF CertificateChoices

IssuerAndSerialNumber ::= SEQUENCE {
    issuer Name,
    serialNumber CertificateSerialNumber }

CMSVersion ::= INTEGER { v0(0), v1(1), v2(2), v3(3), v4(4), v5(5) }

UserKeyingMaterial ::= OCTET STRING

KEY-ATTRIBUTE ::= TYPE-IDENTIFIER

```

```

OtherKeyAttribute ::= SEQUENCE {
    keyAttrId  KEY-ATTRIBUTE,
                &id({SupportedKeyAttributes}),
    keyAttr    KEY-ATTRIBUTE,
                &Type({SupportedKeyAttributes}{@keyAttrId})}

SupportedKeyAttributes KEY-ATTRIBUTE ::= { ... }

-- Content Type Object Identifiers

id-ct-contentInfo OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs9(9) smime(16) ct(1) 6 }

ct-Data CONTENT-TYPE ::= { IDENTIFIED BY id-data }

id-data OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs7(7) 1 }

ct-SignedData CONTENT-TYPE ::=
    { TYPE SignedData IDENTIFIED BY id-signedData}

id-signedData OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs7(7) 2 }

ct-EnvelopedData CONTENT-TYPE ::=
    { TYPE EnvelopedData IDENTIFIED BY id-envelopedData}

id-envelopedData OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs7(7) 3 }

ct-DigestedData CONTENT-TYPE ::=
    { TYPE DigestedData IDENTIFIED BY id-digestedData}

id-digestedData OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs7(7) 5 }

ct-EncryptedData CONTENT-TYPE ::=
    { TYPE EncryptedData IDENTIFIED BY id-encryptedData}

id-encryptedData OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs7(7) 6 }

ct-AuthenticatedData CONTENT-TYPE ::=
    { TYPE AuthenticatedData IDENTIFIED BY id-ct-authData}

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

```

Internet-Draft

Additional New ASN.1 Modules

December 2010

```
--
-- The CMS Attributes
--

MessageDigest ::= OCTET STRING

SigningTime ::= Time

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

Countersignature ::= SignerInfo

-- Attribute Object Identifiers

aa-contentType ATTRIBUTE ::=
    { TYPE ContentType IDENTIFIED BY id-contentType }
id-contentType OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs9(9) 3 }

aa-messageDigest ATTRIBUTE ::=
    { TYPE MessageDigest IDENTIFIED BY id-messageDigest }
id-messageDigest OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs9(9) 4 }

aa-signingTime ATTRIBUTE ::=
    { TYPE SigningTime IDENTIFIED BY id-signingTime }
id-signingTime OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs9(9) 5 }

aa-countersignature ATTRIBUTE ::=
    { TYPE Countersignature IDENTIFIED BY id-countersignature }
id-countersignature OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs9(9) 6 }

--
-- Obsolete Extended Certificate syntax from PKCS#6
--

ExtendedCertificateOrCertificate ::= CHOICE {
    certificate Certificate,
    extendedCertificate [0] IMPLICIT ExtendedCertificate }
```

```
ExtendedCertificate ::= SEQUENCE {  
    extendedCertificateInfo ExtendedCertificateInfo,  
    signatureAlgorithm SignatureAlgorithmIdentifier,  
    signature Signature }
```

```
ExtendedCertificateInfo ::= SEQUENCE {  
    version CMSVersion,  
    certificate Certificate,  
    attributes UnauthAttributes }
```

```
Signature ::= BIT STRING
```

```
Attribute{ ATTRIBUTE:AttrList } ::= SEQUENCE {  
    attrType ATTRIBUTE.  
        &id({AttrList}),  
    attrValues SET OF ATTRIBUTE.  
        &Type({AttrList}{@attrType}) }
```

```
Attributes { ATTRIBUTE:AttrList } ::=  
    SET SIZE (1..MAX) OF Attribute {{ AttrList }}
```

```
END
```

10. ASN.1 Module [RFC 5752](#)

We have updated the ASN.1 module associated with this document to be 2008 compliant and to use the set of classes previously defined in [\[RFC5911\]](#).

MultipleSignatures-2009

```
{ iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
  smime(16) modules(0) id-mod-multipleSign-2009(59) }
```

DEFINITIONS IMPLICIT TAGS ::=

BEGIN

-- EXPORTS All

-- The types and values defined in this module are exported for use
-- in the other ASN.1 modules. Other applications may use them for
-- their own purposes.

IMPORTS

-- Imports from PKIX-Common-Types-2009 [\[RFC5912\]](#)

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) }
```

-- Imports from CryptographicMessageSyntax-2009 [\[RFC5911\]](#)

DigestAlgorithmIdentifier, SignatureAlgorithmIdentifier

```

FROM CryptographicMessageSyntax-2009
{ iso(1) member-body(2) us(840) rsadsi(113549)
  pkcs(1) pkcs-9(9) smime(16) modules(0) id-mod-cms-2009(58) }

-- Imports from ExtendedSecurityServices-2009 [RFC5911]

ESSCertIDv2
  FROM ExtendedSecurityServices-2009
  { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
    smime(16) modules(0) id-mod-ess-2006-02(42) }
;

--
-- Section 3.0
--
-- at-multipleSignatures should be added ONLY to the
--   SignedAttributesSet defined in [RFC5652]
--

```

```

at-multipleSignatures ATTRIBUTE ::= {
  TYPE MultipleSignatures
  IDENTIFIED BY id-aa-multipleSignatures
}

id-aa-multipleSignatures OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs9(9)
  id-aa(2) 51 }

MultipleSignatures ::= SEQUENCE {
  bodyHashAlg      DigestAlgorithmIdentifier,
  signAlg          SignatureAlgorithmIdentifier,
  signAttrsHash    SignAttrsHash,
  cert             ESSCertIDv2 OPTIONAL
}

SignAttrsHash ::= SEQUENCE {
  algID            DigestAlgorithmIdentifier,
  hash             OCTET STRING
}

END

```

[11](#). Module Identifiers in ASN.1

One potential issue that can occur when updating modules is the fact that a large number of modules may need to be updated if they import from a newly updated module. This section addresses one method that can be used to deal with this problem, but the modules in this document don't currently implement the solution discussed here.

When looking at an import statement, there are three portions: The list of items imported, a textual name for the module and an object identifier for the module. Full implementations of ASN.1 do module matching using first the object identifier and if that is not present the textual name of the module. Note however that some older implementations used the textual name of the module for the purposes of matching. In a full implementation the name assigned to the

module is scoped to the ASN.1 module that it appears in (and thus need to match the module it is importing from).

One can create a module that contains only the module number assignments and import the module assignments from the new module. This means that when a module is replaced, one can replace the previous module, update the module number assignment module and recompile without having to modify any other modules.

A sample module assignment module would be:

```
ModuleNumbers
DEFINITIONS TAGS ::=
BEGIN
    id-mod-CMS ::= { iso(1) member-body(2) us(840) rsadsi(113549)
        pkcs(1) pkcs-9(9) smime(16) modules(0) TBD }

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

This would be used in the following import statement:

IMPORTS

```
id-mod-CMS, id-mod-AlgInfo
FROM ModuleNumber    -- Note it will match on the name since no
                     -- OID is provided
```

```
CMSVersion, EncapsulatedContentInfo, CONTENT-TYPE
FROM CryptographicMessageSyntax-2009
    id-mod-CMS
```



```
AlgorithmIdentifier{}, SMIME-CAPS, ParamOptions  
FROM AlgorithmInformation-2009 id-mod-AlgInfo  
;
```

This document itself does not have any security considerations. The ASN.1 modules keep the same bits-on-the-wire as the modules that they replace.

[13.](#) IANA Considerations

None.

Internet-Draft

Additional New ASN.1 Modules

December 2010

14. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC3274] Gutmann, P., "Compressed Data Content Type for Cryptographic Message Syntax (CMS)", [RFC 3274](#), June 2002.
- [RFC3379] Pinkas, D. and R. Housley, "Delegated Path Validation and Delegated Path Discovery Protocol Requirements", [RFC 3379](#), September 2002.
- [RFC4049] Housley, R., "BinaryTime: An Alternate Format for Representing Date and Time in ASN.1", [RFC 4049](#), April 2005.
- [RFC4073] Housley, R., "Protecting Multiple Contents with the Cryptographic Message Syntax (CMS)", [RFC 4073](#), May 2005.
- [RFC4231] Nystrom, M., "Identifiers and Test Vectors for HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512", [RFC 4231](#), December 2005.
- [RFC4334] Housley, R. and T. Moore, "Certificate Extensions and Attributes Supporting Authentication in Point-to-Point Protocol (PPP) and Wireless Local Area Networks (WLAN)", [RFC 4334](#), February 2006.
- [RFC5083] Housley, R., "Cryptographic Message Syntax (CMS) Authenticated-Enveloped-Data Content Type", [RFC 5083](#), November 2007.
- [RFC5084] Housley, R., "Using AES-CCM and AES-GCM Authenticated Encryption in the Cryptographic Message Syntax (CMS)", [RFC 5084](#), November 2007.
- [RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, [RFC 5652](#), September 2009.

- [RFC5752] Turner, S. and J. Schaad, "Multiple Signatures in Cryptographic Message Syntax (CMS)", [RFC 5752](#), January 2010.
- [RFC5911] Hoffman, P. and J. Schaad, "New ASN.1 Modules for Cryptographic Message Syntax (CMS) and S/MIME", [RFC 5911](#), June 2010.
- [RFC5912] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the

Schaad & Turner

Expires June 15, 2011

[Page 39]

Internet-Draft

Additional New ASN.1 Modules

December 2010

Public Key Infrastructure Using X.509 (PKIX)", [RFC 5912](#), June 2010.

[ASN1-2008]

ITU-T, "ITU-T Recommendations X.680, X.681, X.682, and X.683", 2008.

Internet-Draft

Additional New ASN.1 Modules

December 2010

Authors' Addresses

Jim Schaad
Soaring Hawk Consulting

Email: jimsch@augustcellars.com

Sean Turner
IECA, Inc.
3057 Nutley Street, Suite 106
Fairfax, VA 22031

Email: turners@ieca.com

