ICS 35.040
L 80

中华人民共和国国家标准

GB/T 31503—2015

信息安全技术
电子文档加密与签名消息语法

Information security technology—
Encryption and signature message syntax for electronic document

2015-05-15 发布　　2016-01-01 实施

中华人民共和国国家质量监督检验检疫总局
中国国家标准化管理委员会　发布

目　　次

前　言

本标准按照 GB/T 1.1—2009 给出的规则起草。

本标准由全国信息安全标准化技术委员会(SAC/TC 260)提出并归口。

本标准起草单位:中国科学院数据与通信保护研究教育中心、北京数字认证股份有限公司、上海普华诚信信息技术有限公司、赞嘉电子科技有限公司。

本标准主要起草人:向继、汪婧、王雷、荆继武、高能、林璟锵、管乐、马存庆、查达仁、詹榜华、梁佐泉、张嘉纯。

引　言

本标准主要参考 IETF(互联网工程特别工作组)RFC 5652 文件制定。

本标准规定了用于电子文档密码保护的封装语法。它支持数字签名和加密。该语法允许多重封装,一个封装信封可以嵌套在另一个封装信封之内,同样,一方可以对以前封装过的数据再进行数字签名。它也允许任意属性,如签名时间,同消息内容一起签名,并且提供其他属性如联合签名,同签名关联在一起。

本标准描述的电子文档加密与签名消息语法支持各种基于证书的密钥管理架构。该语法使用抽象语法记法 ASN.1,并采用 BER 编码生成值。这些值通常表示成字节串的形式。虽然很多系统都能够可靠地传输字节串,但仍有很多电子邮件系统不行。本标准不提供字节串编码机制以保证这种环境下的可靠传输。

信息安全技术
电子文档加密与签名消息语法

1 范围

本标准规定了电子文档加密与签名消息语法，此语法可用于对任意消息内容进行数字签名、摘要、鉴别或加密。

本标准适用于电子商务和电子政务中电子文档加密与签名消息的产生、处理以及验证。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本(包括所有的修改单)适用于本文件。

GB/T 16262.1—2006 信息技术 抽象语法记法一(ASN.1) 第1部分：基本记法规范

GB/T 16263.1—2006 信息技术 ASN.1编码规则 第1部分：基本编码规则(BER)、正则编码规则(CER)和非典型编码规则(DER)规范

GB/T 16264.2—2008 信息技术 开放系统互连 目录 第2部分：模型

GB/T 16264.8—2005 信息技术 开放系统互连 目录 第8部分：公钥和属性证书框架

GB/T 19714—2005 信息技术 安全技术 公钥基础设施 证书管理协议

GB/T 20518—2006 信息安全技术 公钥基础设施 数字证书格式

RFC 3281 用于授权的因特网属性证书框架(An Internet Attribute Certificate Profile for Authorization)

RFC 5280 因特网X.509公钥基础设施证书和证书撤销列表轮廓(Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile)

3 术语和定义

GB/T 19714—2005、GB/T 20518—2006界定的以及下列术语和定义适用于本文件。

3.1

算法标识符 algorithm identifier

通过对象标识符标识算法的类型。

3.2

属性 attribute

包括属性类型以及一个或多个属性值，属性类型由对象标识符指定。

4 缩略语

下列缩略语适用于本文件。

ASN.1：抽象语法记法一(Abstract Syntax Notation one)

BER：基本编码规则 (Basic Encoding Rules)

CRL:证书撤销列表 (Certificate Revocation List)

DER:可辨别编码规则 (Distinguished Encoding Rules)

ESMS:电子文档加密与签名消息语法 (Encryption and Signature Message Syntax for Electronic Document)

MAC:消息鉴别码 (Message Authentication Code)

PRNG:伪随机数生成器 (Pseudo-random Number Generator)

UKM:用户密钥生成材料 (User Keying Material)

5 通用语法

下列对象标识符(object identifier)标识了内容信息类型:

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

ESMS 将内容类型标识符与内容关联起来。语法中应含有 ASN.1(见 GB/T 16262.1—2006) ContentInfo 类型:

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

ContentType ::= OBJECT IDENTIFIER
```

ContentInfo 中各个域的意义如下:

a) contentType:表示关联内容的类型,是一个对象标识符,由权威机构分配,用于定义内容类型的唯一整数串。

b) content:表示关联内容。内容的类型由 contentType 唯一确定。本标准定义了 6 种内容类型:数据、签名数据、封装数据、摘要数据、加密数据以及鉴别数据。如果在其他文档中定义了其他内容类型,定义的 ASN.1 类型不应是 CHOICE 类型。

6 数据内容类型

下列对象标识符标识了数据内容类型:

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

数据内容类型用于表示任意字节串,例如 ASCII 文本文件;具体应用具体解释。这类串不需要任何的内部结构,它们可能有自己的 ASN.1 定义或其他结构。

数据内容类型一般装在签名数据、封装数据、摘要数据、加密数据或鉴别数据的内容类型中。

7 签名数据内容类型

7.1 概述

签名数据内容类型包括任何类型的内容,以及零个或多个签名结果。任意数量的签名者可并行地对任意类型的内容进行签名。

签名数据内容类型的典型应用是，签名者对数据内容类型的内容进行数字签名。另一个典型应用是分发证书和 CRL。

构造签名数据的步骤如下：

a) 对于每个签名者，使用签名者指定的消息摘要算法计算出内容的消息摘要值。如果签名者要对除内容之外的其他信息进行签名，那么使用签名者的消息摘要算法对内容摘要值和其他信息一起计算摘要，得到的结果为“消息摘要”。

b) 对于每个签名者，使用签名者的私钥对消息摘要进行数字签名。

c) 对于每个签名者，签名结果和其他签名者指定的信息放在 SignerInfo 值中，见 7.4。每个签名者所对应的证书和 CRL，以及其他不对应于任何签名者的信息，都在这一步收集。

d) 所有签名者的消息摘要算法和 SignerInfo 值，同内容一起放入 SignedData 值中，见 7.2。

接收者独立地计算消息摘要，利用消息摘要以及签名者的公钥验证签名结果。签名者的公钥可以通过两种途径查询：签发者的可辨别名称以及该签发者的签名证书序列号唯一标识了一张证书，证书中包含公钥；或者通过主体密钥标识符，同时适合认证的和未经认证的公钥。签名者的证书可包含在 SignedData 里的证书域中，但并非必须包括。

当存在多个签名时，成功验证某个给定签名者的签名，通常视为该签名者签名成功。但是，也有些应用环境需要其他的规则。若某个应用除了每个签名者对应一个有效签名外，还采用了别的规则，则应阐明这些规则。此外，只匹配签名者标识符并不足以确定这些签名是否由该签名者生成，在应用说明中应描述如何确定哪个签名是由该签名者生成的。签名者选择包含多个签名的主要原因是为了支持多个不同的接收者群体。

本章分为 6 个部分：第一部分描述了顶层类型 SignedData；第二部分描述了 EncapsulatedContentInfo；第三部分描述了每个签名者信息的类型 SignerInfo；第四、第五和第六部分分别描述了消息摘要的计算、签名的生成以及签名的验证过程。

7.2 SignedData 类型

下列对象标识符(object identifier)标识了签名数据的内容类型：

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

签名数据的内容类型应有 ASN.1 SignedData 类型：

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

DigestAlgorithmIdentifiers ::= SET OF DigestAlgorithmIdentifier

SignerInfos ::= SET OF SignerInfo
```

SignedData 类型的结构框图，如图 1 所示：

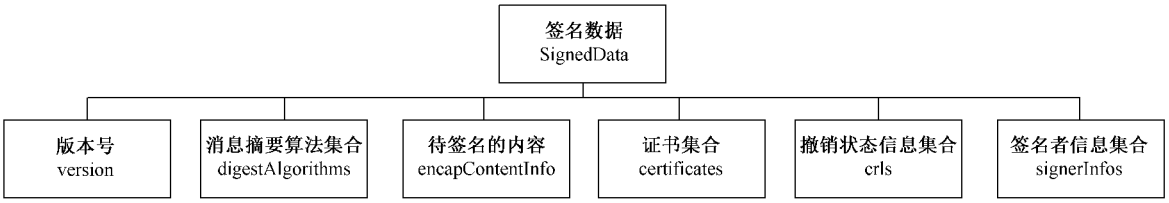

图 1　SignedData 类型结构框图

SignedData 类型的各个域的意义如下：

a) version：是语法的版本号。version 值取决于 certificates，eContentType 和 SignerInfo。version 应如下分配：

IF ((certificates is present) AND
(any certificates with a type of other are present)) OR
((crls is present) AND
(any crls with a type of other are present))
THEN version MUST be 5
ELSE
IF (certificates is present) AND
(any version 2 attribute certificates are present)
THEN version MUST be 4
ELSE
IF ((certificates is present) AND
(any version 1 attribute certificates are present)) OR
(any SignerInfo structures are version 3) OR
(encapContentInfo eContentType is other than id－data)
THEN version MUST be 3
ELSE version MUST be 1

b) digestAlgorithms：是消息摘要算法标识符的集合，集合中可有任意个元素，包括零个。每个元素标识了一个或多个签名者所使用的消息摘要算法及其相关参数。该集合旨在列出所有签名者能使用的消息摘要算法。该集合是无序的，以便单次签名验证。应用中签名若使用了不在该集合内的摘要算法，则签名验证可能会失败。消息摘要的过程描述，见 7.5。

c) encapContentInfo：是待签名的内容，包括内容类型标识符以及内容本身。EncapsulatedContentInfo 类型具体描述，见 7.3 。

d) certificates：是证书集合。预期证书集合包括从信任根或顶级证书机构到 signerInfos 域中所有签名者的证书路径。该集合包含的证书可多于这些必要证书，也可包括从两个或多个独立的顶级证书机构出发的证书路径。如果接收者能通过其他途径获取必要的证书(例如从以前的证书集合中获取)，则可不完全包含这些必要的证书。强烈禁止使用版本 1 属性证书。

e) crls：是撤销状态信息集合。预期该集合包含的信息足以确定 certificates 域中的证书是否有效，但并非必须满足此条件。CRL 是撤销状态信息的主要来源。集合中包含的 CRL 可多于必要的 CRL，也可少于必要的 CRL。

f) signerInfos：是每个签名者信息的集合，可包含任意个元素，包括零个。当该集合含有多个证书时，若能成功验证给定签名者的其中一个签名，应认为该签名者签名成功。但是，一些应用环境也可能需要其他的规则。SignerInfo 类型具体描述，见 7.4。由于每个签名者可采用不同

的数字签名技术，而且以后的规范可能会更新语法，所以，所有实现应处理好它们未实现的SignerInfo 版本。进一步的说，由于所有实现将支持不了每个可能的签名算法，所以，所有实现在遇到未实现的签名算法时应处理得当。

7.3 EncapsulatedContentInfo 类型

EncapsulatedContentInfo 类型中的内容如下表示：

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

ContentType ::= OBJECT IDENTIFIER
```

EncapsulatedContentInfo 类型各个域的意义如下：

a) eContentType：是对象标识符。该对象标识符唯一指定了内容类型。

b) eContent：是内容本身，由字节串表示。eContent 不需要 DER 编码。

可选择省略 EncapsulatedContentInfo 域中的 eContent，从而能够构造“外部签名”。在有外部签名的情况下，包含在签名数据的内容类型中的 EncapsulatedContentInfo 值将缺少签名内容。如果 EncapsulatedContentInfo 中没有 eContent 值，照常计算 signatureValue 并分配 eContentType，同 eContent 值存在一样。

在没有签名者的情况下，“待签名”的 EncapsulatedContentInfo 值是无关紧要的。这种情况下，EncapsulatedContentInfo 值中的内容类型应是 id-data，见第 6 章，而且 EncapsulatedContentInfo 值中的内容域应省略。

7.4 SignerInfo 类型

每个签名者的信息包含在 SignerInfo 类型中：

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

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

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

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

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

AttributeValue ::= ANY

SignatureValue ::= OCTET STRING
```

SignerInfo 类型的结构框如图 2 所示：

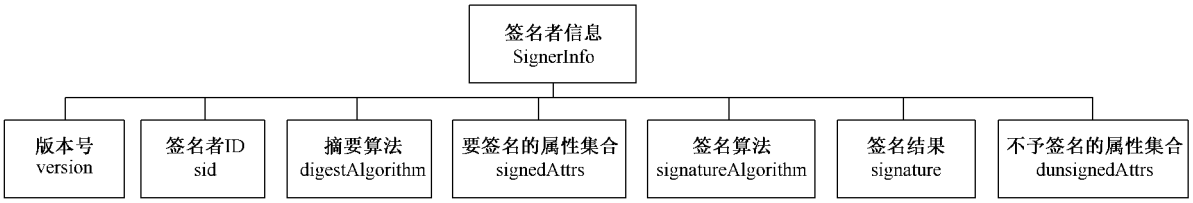

图 2　SignerInfo 类型结构框图

SignerInfo 类型各个域的意义如下：

a) version：是语法的版本号。若 SignerIdentifier 是 issuerAndSerialNumber 选项，那么版本应为 1。若 SignerIdentifier 是 subjectKeyIdentifier 选项，那么版本应为 3。

b) sid：指定了签名者的证书，从而指定了签名者的公钥。接收者需要签名者的公钥以验证签名。SignerIdentifier 提供了两种选择用以指定签名者公钥。issuerAndSerialNumber 选项通过签发者的可辨别名称和证书序列号，标识签名者的证书；subjectKeyIdentifier 选项则是通过密钥标识符，标识签名者的证书。当引用 GB/T 16264.8—2005 证书时，密钥标识符与 GB/T 16264.8—2005 主体密钥标识符的扩展值相匹配。当引用了其他证书格式时，用于规范证书格式以及 ESMS 使用的文档，应包括关于密钥标识符与适当证书域匹配的详细说明。实现应同时支持 SignerIdentifier 的 issuerAndSerialNumber 和 subjectKeyIdentifier 两种类型的接收。当生成 SignerIdentifier 时，实现可支持其中一种类型(issuerAndSerialNumber 或 subjectKeyIdentifier)，并一直使用这种类型，或者实现也可将两种类型任意混合使用。

c) digestAlgorithm：标识了签名者使用的消息摘要算法以及任何相关参数。消息摘要是通过计算待签名的内容或者内容加上签名属性而得到的，消息摘要的计算过程见 7.5。消息摘要算法应包含相关 SignedData 的 digestAlgorithms 域中列出的算法中。若签名使用了不在 SignedData digestAlgorithms 集合里的摘要算法，实现验证这些签名可能会失败。

d) signedAttrs：是要签名的属性集合。该域是可选的，但是，如果待签名的 EncapsulatedContentInfo 值中的内容类型不是 id-data，则该域应存在。SignedAttributes 应是 DER 编码(见 GB/T 16263.1—2006)，即使该结构的其他部分是 BER 编码(见 GB/T 16263.1—2006)。一些有用的属性类型，如签名时间，见第 13 章。如果该域存在，则应至少包括下列两个属性：

   1) 内容类型属性：将待签名的 EncapsulatedContentInfo 值的内容类型作为它的值。13.2 定义了内容类型属性。但是，内容类型属性不应作为联合签名-不予签名属性的一部分使用，联合签名-不予签名属性，见 13.5。

   2) 消息摘要属性：将内容的消息摘要作为它的值。13.3 定义了消息摘要属性。

e) signatureAlgorithm：标识了签名者生成数字签名所使用的签名算法以及任何相关参数。

f) signature：是数字签名产生的结果，使用了消息摘要和签名者的私钥。签名的详细情况取决于采用的签名算法。

g) unsignedAttrs:是不予签名的属性集合。该域是可选的。一些有用的属性类型,如联合签名,定义见第13章。

SignedAttribute 和 UnsignedAttribute 类型中的各个域意义如下:

a) attrType:表示属性的类型,是一个对象标识符。

b) attrValues:构成属性的值的集合。集合中每个值的类型由 attrType 唯一确定。attrType 可对集合中条目的个数加以限制。

7.5 消息摘要计算过程

消息摘要计算过程,是对待签名的内容,或内容加上签名属性,计算消息摘要。无论哪种情况,消息摘要计算过程的初始输入都是待签名的封装内容的"值"。具体来说,初始输入是 encapContentInfo eContent OCTET STRING。只有构成 eContent OCTET STRING 的值的字节才被输入至消息摘要算法中,而不包括标签或长度字节。

消息摘要计算过程的结果取决于 signedAttrs 域是否存在。当该域不存在时,其结果就是上述内容的消息摘要。当该域存在时,其结果是 signedAttrs 域中 SignedAttrs 值的完整 DER 编码的消息摘要。因为当存在 SignedAttrs 值时,其应包括内容类型和消息摘要属性,这两个值间接包含在结果中。内容类型属性不应包含 13.5 中定义的联合签名-不予签名的属性。为了计算消息摘要,要对 signedAttrs 域进行单独编码。DER 编码不使用 signedAttrs 中的 IMPLICIT[0]标签,但要使用 EXPLICIT SET OF 标签。也就是说,EXPLICIT SET OF 标签的 DER 编码,而不是 IMPLICIT[0]标签的 DER 编码,应同 SignedAttributes 值的长度和内容字节一起,包含在消息摘要计算中。当 signedAttrs 域不存在,只有构成 SignedData encapContentInfo eContent OCTET STRING 的值(例如文件内容)的字节,被输入至消息摘要计算中。其优势是,待签名的内容的长度不需要在签名生成过程之前知道。

尽管消息摘要计算中不包括 encapContentInfo eContent OCTET STRING 标签和长度字节,但它们还是能通过其他方式得到保护。长度字节通过消息摘要算法的特性得到保护,因为要找到消息摘要值相等的两个任意长度的不同消息内容是计算上不可行的。

7.6 签名生成过程

签名生成过程的输入包括消息摘要计算过程的结果以及签名者的私钥。签名生成的细节取决于采用的签名算法。对象标识符和参数共同规定了签名者采用的签名算法,其包含在 signatureAlgorithm 域中。签名者生成的签名结果应编码成字节串,并装载在 signature 域中。

7.7 签名验证过程

签名验证过程的输入包括消息摘要计算过程的结果和签名者的公钥。接收者可通过任何途径获得正确的签名者公钥,但是首选的方法是从 SignedData 的 certificates 域中获取证书,然后从证书中获得公钥。签名者公钥的选择和验证可基于证书路径验证,也可基于其他外部背景,但不在本标准的范围内。签名验证的细节取决于采用的签名算法。

接收者不应依赖任何由生成者计算的消息摘要值。若 SignedData signerInfo 包含 signedAttributes,那么内容消息摘要应按照 7.5 中的描述进行计算。接收者计算的消息摘要值应同 SignedData signerInfo 中 signedAttributes 里的消息摘要属性的值相等,签名才可能有效。

若 SignedData signerInfo 包含 signedAttributes,则内容类型属性的值应同 SignedData encapContentInfo eContent 的值相匹配。

8 封装数据内容类型

8.1 概述

封装数据内容类型由任意类型的加密内容以及经过加密的内容加密密钥组成。对接收者而言，加密内容以及经过加密的内容加密密钥组合成一个数字信封。任意类型的内容都可封装给任意数量的接收者，封装可针对每个接收者使用任何支持的密钥管理技术。

封装数据内容类型的典型应用是，将数据内容或签名数据内容类型，封装在一个或多个接收者的数字信封里。

封装数据通过下列步骤构造而成：

a) 随机生成特定内容加密算法所使用的内容加密密钥。

b) 为每个接收者加密内容加密密钥。加密密钥的细节取决于使用的密钥管理算法，有四种支持的通用技术：

   1) 密钥传输：使用接收者的公钥加密内容加密密钥；

   2) 密钥协商：使用接收者的公钥和发送者的私钥生成成对的对称密钥，然后用成对的对称密钥加密内容加密密钥；

   3) 对称密钥加密密钥：使用以前分发的对称密钥加密密钥，加密内容加密密钥；

   4) 口令：使用来源于口令或其他共享秘密值的密钥加密密钥，加密内容加密密钥。

c) 对于每个接收者，经过加密的内容加密密钥和其他接收者相关信息收集在 RecipientInfo 值中，见 8.3。

d) 使用内容加密密钥加密内容。加密内容可能需要对内容进行填充，达到某分组大小的整数倍，见 8.4。

e) 所有接收者的 RecipientInfo 值同加密内容一起，构成 EnvelopedData 值，见 8.2。

接收者通过解密经过加密的内容加密密钥，然后使用恢复出的内容加密密钥解密加密内容，从而打开了数字信封。

本章分为 4 个部分：第一部分描述了顶层类型 EnvelopedData；第二部分描述了每个接收者的信息类型 RecipientInfo；第三和第四部分分别描述了内容加密以及密钥加密的过程。

8.2 EnvelopedData 类型

下列对象标识符标识了封装数据的内容类型：

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

封装数据的内容类型应有 ASN.1 EnvelopedData 类型：

```
EnvelopedData ::= SEQUENCE {
  version ESMSVersion,
  originatorInfo [0] IMPLICIT OriginatorInfo OPTIONAL,
  recipientInfos RecipientInfos,
  encryptedContentInfo EncryptedContentInfo,
  unprotectedAttrs [1] IMPLICIT UnprotectedAttributes OPTIONAL }

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

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

EncryptedContentInfo ::= SEQUENCE {
  contentType ContentType,
  contentEncryptionAlgorithm ContentEncryptionAlgorithmIdentifier,
  encryptedContent [0] IMPLICIT EncryptedContent OPTIONAL }

EncryptedContent ::= OCTET STRING

UnprotectedAttributes ::= SET SIZE (1..MAX) OF Attribute
```

EnvelopedData 类型的结构框图如图 3 所示：

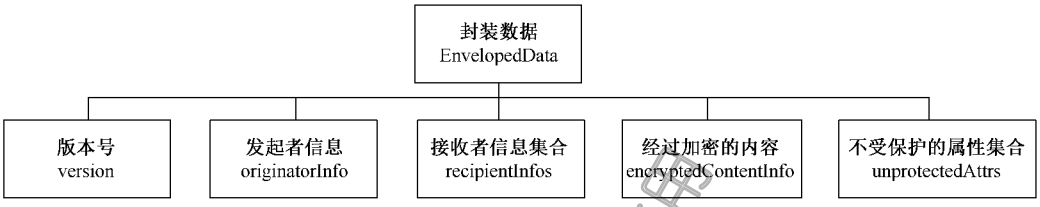

图 3　EnvelopedData 类型结构框图

EnvelopedData 类型的各个域的意义如下：

a) version：是语法的版本号。version 值取决于 originatorInfo，RecipientInfo 和 unprotectedAttrs。version 应如下分配：

```
IF (originatorInfo is present) AND
    ((any certificates with a type of other are present) OR
    (any crls with a type of other are present))
THEN version is 4
ELSE
    IF ((originatorInfo is present) AND
        (any version 2 attribute certificates are present)) OR
        (any RecipientInfo structures include pwri) OR
        (any RecipientInfo structures include ori)
    THEN version is 3
    ELSE
        IF (originatorInfo is absent) AND
            (unprotectedAttrs is absent) AND
            (all RecipientInfo structures are version 0)
        THEN version is 0
        ELSE version is 2
```

b) originatorInfo：选择性地提供有关发起者的信息。只有当密钥管理算法要求时，该域才存在。该域可能包括证书和 CRL：

1) certs：是证书的集合。certs 可能包含与数个不同的密钥管理算法相关的发起者证书。certs 可能还包含与发起者相关的属性证书。certs 包含的证书是为了让所有接收者足以

建立从信任根或顶级证书机构出发的证书路径。但是，该集合包含的证书可多于这些必要证书，也可包括从两个或多个独立的顶级证书机构出发的证书路径。如果接收者能通过其他途径获取必要的证书(例如从以前的证书集合中获取)，则也可不完全包含这些必要的证书。

2) crl：是 CRL 的集合。预期该集合包含的信息足以确定 certs 域中的证书是否有效，但并非必需满足此条件。集合中包含的 CRL 可多于必要的 CRL，也可少于必要的 CRL。

c) recipientInfos：是每个接收者的信息的集合。该集合应至少包含一个元素。

d) encryptedContentInfo：是经过加密的内容信息。

e) unprotectedAttrs：是未经加密的属性的集合。该域是可选的。一些有用属性类型定义于第 13 章。

EncryptedContentInfo 类的各个域的意义如下：

a) contentType：表明内容的类型。

b) contentEncryptionAlgorithm：标识了用于加密内容的内容加密算法以及任何相关参数。内容加密过程，见 8.4。对于所有接收者，使用相同的内容加密算法和内容加密密钥。

c) encryptedContent：是加密内容的结果。该域是可选的。如果该域不存在，其预期值应由其他方式提供。

由于 recipientInfos 域在 encryptedContentInfo 域的前面，所以 EnvelopedData 的值能单次处理。

8.3 RecipientInfo 类型

8.3.1 RecipientInfo 类型概述

每个接收者的信息表示在 RecipientInfo 类型中。RecipientInfo 对于每个支持的密钥管理技术都有不同的格式。对于每个相同加密内容的接收者，可以使用任何密钥管理技术。就所有情况而言，经过加密的内容加密密钥要传送给一个或多个接收者。

因为所有实现将支持不了每个可能的密钥管理算法，所以，所有实现在遇到未实现的算法时应处理得当。

实现应支持密钥传输、密钥协商以及以前分发的对称密钥加密密钥，分别用 ktri、、kari 和 kekri 表示。实现也可支持基于口令的密钥管理，用 pwri 表示。实现也可支持任何其他密钥管理技术，用 ori 表示。因为每个接收者可以采用不同的密钥管理技术，而且以后的规范可能会定义另外的密钥管理技术，所以，所有的实现应处理好 RecipientInfo CHOICE 中未实现的选项，且所有的实现应处理好 RecipientInfo CHOICE 中其他支持选项的未实现版本，所有的实现还应处理好未实现的或未知的 ori 选项。

```
RecipientInfo ::= CHOICE {
  ktri KeyTransRecipientInfo,
  kari [1] KeyAgreeRecipientInfo,
  kekri [2] KEKRecipientInfo,
  pwri [3] PasswordRecipientinfo,
  ori [4] OtherRecipientInfo }
EncryptedKey ::= OCTET STRING
```

8.3.2 KeyTransRecipientInfo 类型

使用密钥传输的每个接收者信息表示在 KeyTransRecipientInfo 类型中。每个 KeyTransRecipientInfo 的实例将内容加密密钥传送给一个接收者。

```
KeyTransRecipientInfo ::= SEQUENCE {
```

```
  version ESMSVersion, -- always set to 0 or 2
  rid RecipientIdentifier,
  keyEncryptionAlgorithm KeyEncryptionAlgorithmIdentifier,
  encryptedKey EncryptedKey }

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

KeyTransRecipientInfo 类型的结构框图如图 4 所示：

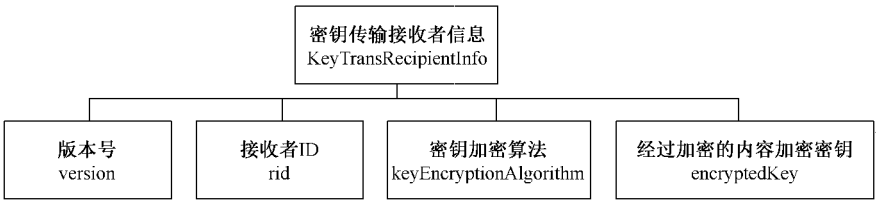

图 4　KeyTransRecipientInfo 类型结构框图

KeyTransRecipientInfo 类型的各个域的意义如下：

a) version：是语法的版本号。若 RecipientIdentifier 是 issuerAndSerialNumber 选项，则版本应是 0。若 RecipientIdentifier 是 subjectKeyIdentifier 选项，则版本应是 2。

b) rid：指定了发送者为了保护内容加密密钥，所使用的接收者的证书或密钥。使用接收者的公钥对内容加密密钥进行加密。RecipientIdentifier 提供了两种选择，用以指定接收者的证书，进而指定接收者的公钥。接收者的证书应包含密钥传输公钥。因此，包含密钥用法扩展的接收者 GB/T 16264.8—2005 版本 3 证书，应使 keyEncipherment 位生效。issuerAndSerialNumber 选项通过签发者的可辨别名称和证书序列号，标识接收者的证书；subjectKeyIdentifier 选项则是通过密钥标识符，标识接收者的证书。当引用 GB/T 16264.8—2005 证书时，密钥标识符与 GB/T 16264.8—2005 主体密钥标识符的扩展值相匹配。当引用了其他证书格式时，用于规范证书格式以及 ESMS 使用的文档，应包括关于密钥标识符与适当证书域匹配的详细说明。对于接收者的处理，实现应同时支持这两种用于指定接收者证书的选项。对于发送者的处理，实现应至少支持其中一种选项。

c) keyEncryptionAlgorithm：标识了用于加密内容加密密钥的密钥加密算法以及任何相关参数。加密密钥过程，见 8.5。

d) encryptedKey：是对内容加密密钥进行加密的结果。

8.3.3　KeyAgreeRecipientInfo 类型

使用密钥协商的接收者信息表示在 KeyAgreeRecipientInfo 类型中。每个 KeyAgreeRecipientInfo 的实例使用相同的密钥协商算法以及算法参数，将内容加密密钥传送给一个或多个接收者。

```
KeyAgreeRecipientInfo ::= SEQUENCE {
  version ESMSVersion, -- always set to 3
  originator [0] EXPLICIT OriginatorIdentifierOrKey,
  ukm [1] EXPLICIT UserKeyingMaterial OPTIONAL,
  keyEncryptionAlgorithm KeyEncryptionAlgorithmIdentifier,
```

```
    recipientEncryptedKeys RecipientEncryptedKeys }

OriginatorIdentifierOrKey ::= CHOICE {
    issuerAndSerialNumber IssuerAndSerialNumber,
    subjectKeyIdentifier [0] SubjectKeyIdentifier,
    originatorKey [1] OriginatorPublicKey }
OriginatorPublicKey ::= SEQUENCE {
    algorithm AlgorithmIdentifier,
    publicKey BIT STRING }

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
```

KeyAgreeRecipientInfo 类型的结构框图如图 5 所示：

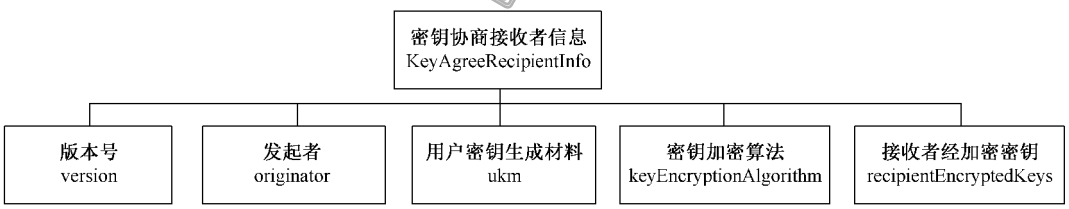

图 5　KeyAgreeRecipientInfo 类型结构框图

KeyAgreeRecipientInfo 类型的各个域的意义如下：

a) version：是语法的版本号。版本应一直是 3。

b) originator：是 CHOICE 类型，有三个选项，指定了发送者的密钥协商公钥。发送者使用相应的私钥以及接收者的公钥，生成成对密钥。用生成的成对密钥加密内容加密密钥。issuerAndSerialNumber 选项通过签发者的可辨别名称和证书序列号，标识接收者的证书，进而标识了发送者的公钥；subjectKeyIdentifier 选项则是通过密钥标识符，标识接收者的证书，进而标识了发送者的公钥。当引用 GB/T 16264.8—2005 证书时，密钥标识符与 GB/T 16264.8—2005 主体密钥标识符的扩展值相匹配。当引用了其他证书格式时，用于规范证书格式以及 ESMS 使用的文档，应包括关于密钥标识符与适当证书域匹配的详细说明。originatorKey 选项包括算法标识符以及发送者的密钥协商公钥。此选项允许发起者匿名，因其公钥未经证明。实现应同时支持这三种用于指定发送者公钥的选项。

c) ukm：是可选的。在有些密钥协商算法中，发送者提供了 UKM，以保证相同两方每一次生成的成对密钥是不同的。实现应能接受包含 ukm 域的 KeyAgreeRecipientInfo SEQUENCE。不支持使用了 UKM 的密钥协商算法的实现，应处理好有 UKM 的情况。

d) keyEncryptionAlgorithm：标识了用密钥加密密钥来加密内容加密密钥的密钥加密算法以及任何相关参数。密钥加密的过程，见 8.5。

e) recipientEncryptedKeys：包括接收者标识符以及给一个或多个接收者的经加密的密钥。KeyAgreeRecipientIdentifier 是 CHOICE 类型，有两个选项，它指定了接收者的证书，进而指定了接收者的公钥，发送者用其公钥生成成对的密钥加密密钥。接收者的证书应包含密钥协商公钥。因此，含有密钥用法扩展的接收者 GB/T 16264.8—2005 版本 3 证书中，keyAgreement 位应生效。内容加密密钥用成对的密钥加密密钥进行加密。issuerAndSerialNumber 选项通过签发者的可辨别名称和证书序列号，标识了接收者的证书；RecipientKeyIdentifier 将在下面描述。encryptedKey 是加密内容加密密钥产生的结果，加密密钥是使用密钥协商算法生成的成对密钥加密密钥。实现应同时支持这两种用于指定接收者证书的选项。

RecipientKeyIdentifier 类型的各个域的意义如下：

a) subjectKeyIdentifier：通过密钥标识符标识了接收者的证书。当引用 GB/T 16264.8—2005 证书时，密钥标识符与 GB/T 16264.8—2005 主体密钥标识符的扩展值相匹配。当引用了其他证书格式时，用于规范证书格式以及 ESMS 使用的文档，应包括关于密钥标识符与适当证书域匹配的详细说明。

b) date：是可选的。当它存在时，指定了发送者所使用的以前发布的接收者 UKM。

c) other：是可选的。当它存在时，该域包括接收者用以找出发送者所使用的公共密钥生成材料的其他信息。

8.3.4 KEKRecipientInfo 类型

使用以前分发的对称密钥的接收者信息表示在 KEKRecipientInfo 类型中。每个 KEKRecipientInfo 的实例将内容加密密钥传送给一个或多个拥有以前分发的密钥加密密钥的接收者。

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

KEKIdentifier ::= SEQUENCE {
  keyIdentifier OCTET STRING,
  date GeneralizedTime OPTIONAL,
  other OtherKeyAttribute OPTIONAL }
```

KEKRecipientInfo 类型的结构框图如图 6 所示：

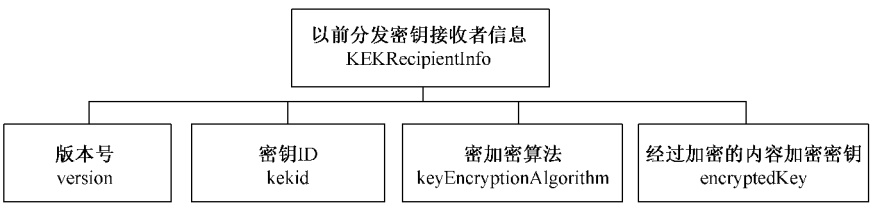

图 6 KEKRecipientInfo 类型结构框图

KEKRecipientInfo 类型的各个域的意义如下：

a) version：是语法的版本号。版本号应一直是 4。

b) kekid：指定了以前分发给发送者以及一个或多个接收者的对称密钥加密密钥。

c) keyEncryptionAlgorithm：标识了用密钥加密密钥来加密内容加密密钥的密钥加密算法以及任何相关参数。密钥加密的过程，见 8.5。

d) encryptedKey：是用密钥加密密钥对内容加密密钥实施加密所形成的结果。

KEKIdentifier 类型的各个域的意义如下：

a) keyIdentifier：标识了以前分发给发送者以及一个或多个接收者的密钥加密密钥。

b) date：是可选的。当它存在时，它从一组以前分发的密钥加密密钥集合中指定了一个密钥。

c) other：是可选的。当它存在时，该域包括接收者用以确定发送者所使用的密钥加密密钥的其他信息。

8.3.5 PasswordRecipientinfo 类型

使用口令或共享秘密值的接收者信息表示在 PasswordRecipientinfo 类型中。每个 PasswordRecipientinfo 的实例将内容加密密钥传送给一个或多个拥有口令或共享秘密值的接收者。

```
PasswordRecipientInfo ::= SEQUENCE {
  version ESMSVersion, -- Always set to 0
  keyDerivationAlgorithm [0] KeyDerivationAlgorithmIdentifier
                                    OPTIONAL,
keyEncryptionAlgorithm KeyEncryptionAlgorithmIdentifier,
encryptedKey EncryptedKey }
```

PasswordRecipientinfo 类型的结构框图如图 7 所示：

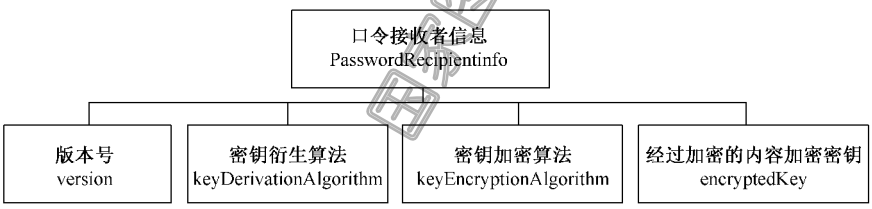

图 7 PasswordRecipientinfo 类型结构框图

PasswordRecipientinfo 类型的各个域的意义如下：

a) version：是语法的版本号。版本应一直是 0。

b) keyDerivationAlgorithm：标识了从口令或共享秘密值衍生出密钥加密密钥的密钥衍生算法以及任何相关参数。若该域不存在，密钥加密密钥则由外部提供，例如硬件密码令牌，如智能卡。

c) keyEncryptionAlgorithm：标识了用密钥加密密钥对内容加密密钥实施加密时所采用的加密算法以及任何相关参数。

d) encryptedKey：是用密钥加密密钥加密内容加密密钥产生的结果。

8.3.6 OtherRecipientInfo 类型

使用其他密钥管理技术的接收者信息表示在 OtherRecipientInfo 类型中。OtherRecipientInfo 类型允许在以后的文档中阐述除密钥传输、密钥协商、以前分发的对称密钥加密密钥以及基于口令的密钥管理之外的密钥管理技术。对象标识符唯一标识了这类密钥管理技术。

```
OtherRecipientInfo ::= SEQUENCE {
  oriType OBJECT IDENTIFIER,
```

oriValue ANY DEFINED BY oriType }

OtherRecipientInfo 类型的各个域的意义如下：

a) oriType：标识了密钥管理技术。

b) oriValue：包括了接收者使用经标识的密钥管理技术所需要的协议数据元。

8.4 内容加密过程

随机生成想要使用的内容加密算法所需的内容加密密钥。将要保护的数据按照下述方法进行填充，然后使用内容加密密钥加密填充数据。加密操作在内容加密密钥的作用下，将一个任意的字节串（数据）映射为另一个字节串（密文）。将加密过的数据装入 EnvelopedData encryptedContentInfo encryptedContent OCTET STRING 中。

有些内容加密算法假定输入长度是 k 字节的整数倍，$k>1$。对于这些算法，输入应如下处理：在输入尾端填充 $k-(l \bmod k)$ 个字节，每个字节的值均为 $k-(l \bmod k)$，l 是输入的长度。也就是说，用下列其中一个字节串对输入进行填充：

$$\begin{gathered} 01 \text{ -- 如果 } l \bmod k = k-1 \\ 02\ 02 \text{ -- 如果 } l \bmod k = k-2 \\ \vdots \\ k\ k\ \cdots\ k\ k \text{ -- 如果 } l \bmod k = 0 \end{gathered}$$

由于所有输入都做了填充处理，输入值都已经是分组大小的整数倍，而且没有一个填充串是另一个填充串的后缀，所以可以明确无疑地将填充部分去除。当且仅当 $k<256$ 时，该填充方法才是定义良好的。

8.5 密钥加密过程

密钥加密过程的输入，即提供给接收者密钥加密算法的值，就是内容加密密钥的值。

9 摘要数据内容类型

摘要数据内容类型由任意类型的内容以及内容的消息摘要组成。

摘要数据内容类型的典型应用是，用以提供内容完整性，而且其结果通常是封装数据内容类型的输入。

摘要数据按照以下步骤进行构造：

a) 用消息摘要算法对内容计算消息摘要。

b) 将消息摘要算法和消息摘要同内容一起收集在 DigestedData 值中。

接收者通过比较收到的消息摘要与自己独立计算的消息摘要是否相等进行验证。

下列对象标识符标识了摘要数据内容类型：

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

摘要数据内容类型应有 ASN.1 DigestedData 类型：

DigestedData ::= SEQUENCE {
version ESMSVersion,
digestAlgorithm DigestAlgorithmIdentifier,
encapContentInfo EncapsulatedContentInfo,
digest Digest }

Digest ::= OCTET STRING

DigestedData 类型的结构框图如图 8 所示：

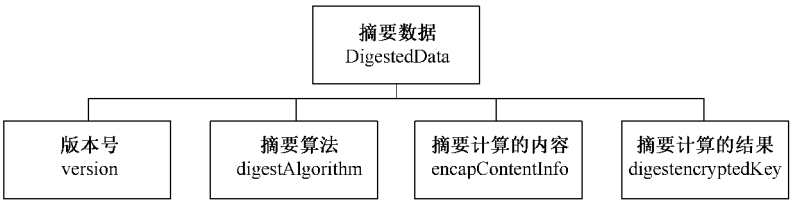

图 8　DigestedData 类型结构框图

DigestedData 类型的各个域的意义如下：

a)　version：是语法的版本号。若封装内容类型是 id-data，则版本号应是 0；此外，版本号应是 2。

b)　digestAlgorithm：标识了计算内容摘要所使用的消息摘要算法以及任何相关参数。消息摘要过程与 8.5 中在没有签名属性情况下的摘要过程相同。

c)　encapContentInfo：是计算摘要的内容，见 8.3。

d)　digest：是消息摘要过程产生的结果。

digestAlgorithm 域、encapContentInfo 域和 digest 域的顺序关系使得能够以单次通过的方式处理 DigestedData 值。

10　加密数据内容类型

加密数据内容类型由任何类型的加密内容组成。不同于封装数据内容类型，加密数据内容类型既没有接收者也没有经过加密的内容加密密钥。密钥应通过其他方式管理。

加密数据内容类型的典型应用是，为本地存储加密数据内容类型的内容，加密密钥可能来源于口令。

下列对象标识符标识了加密数据内容类型：

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

加密数据内容类型应有 ASN.1 EncryptedData 类型：

```
EncryptedData ::= SEQUENCE {
  version ESMSVersion,
  encryptedContentInfo EncryptedContentInfo,
  unprotectedAttrs [1] IMPLICIT UnprotectedAttributes OPTIONAL }
```

EncryptedData 类型的结构框图如图 9 所示：

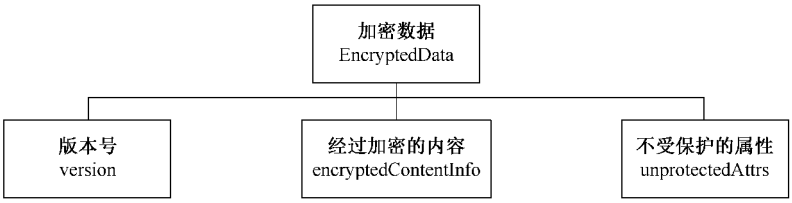

图 9　EncryptedData 类型结构框图

EncryptedData 类型的各个域的意义如下：

a)　version：是语法的版本号。若 unprotectedAttrs 存在，则版本应是 2。若 unprotectedAttrs 不

存在，则版本应是 0。

b) encryptedContentInfo：是经过加密的内容信息，见 8.2。

c) unprotectedAttrs：是未经加密的属性的集合。该域是可选的。一些有用属性类型见第 13 章。

11 鉴别数据内容类型

11.1 概述

鉴别数据内容类型由任意类型的内容、MAC 以及经加密的鉴别密钥组成。接收者需要用 MAC 和经加密的鉴别密钥以验证内容的完整性。可以为任意数量的接收者保护任意类型的内容的完整性。

鉴别数据按照下列步骤进行构造：

a) 随机生成某个特定消息鉴别算法所使用的消息鉴别密钥。

b) 为每个接收者加密消息鉴别密钥。加密的细节取决于使用的密钥管理算法。

c) 对每个接收者，经加密的消息鉴别密钥以及其他与接收者相关的信息包含在 RecipientInfo 值中，见 8.3。

d) 发起者使用消息鉴别密钥，对内容计算 MAC 值。若发起者除内容之外还要鉴别其他任何信息，见 11.3，先对内容计算消息摘要，然后使用消息鉴别密钥对内容的消息摘要以及其他信息做鉴别运算，结果就是“MAC 值”。

11.2 AuthenticatedData 类型

下列对象标识符标识了鉴别数据内容类型：

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

鉴别数据内容类型应有 ASN.1 AuthenticatedData 类型：

```
AuthenticatedData ::= SEQUENCE {
  version ESMSVersion,
  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

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

MessageAuthenticationCode ::= OCTET STRING
```

AuthenticatedData 类型的结构框图如图 10 所示：

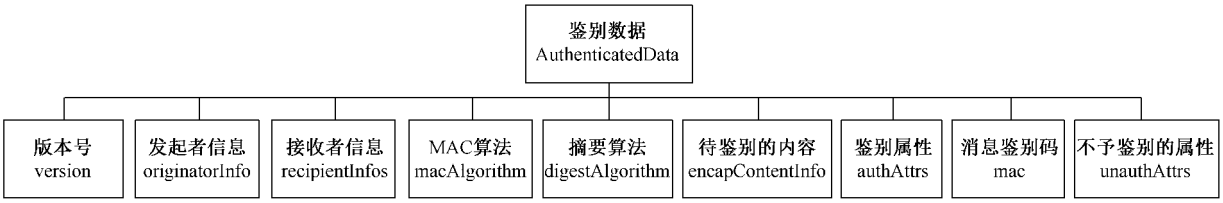

图 10　AuthenticatedData 类型结构框图

AuthenticatedData 类型的各个域的意义如下：

a) version：是语法的版本号。版本应如下分配：

```
IF (originatorInfo is present) AND
    ((any certificates with a type of other are present) OR
    (any crls with a type of other are present))
THEN version is 3
ELSE
  IF ((originatorInfo is present) AND
     (any version 2 attribute certificates are present))
  THEN version is 1
  ELSE version is 0
```

b) originatorInfo：选择性地提供关于发起者的信息。只有当密钥管理算法要求时，该域才存在。该域可能包括证书、属性证书和 CRL，见 8.2。

c) recipientInfos：是每个接收者的信息的集合，见 8.2。该集合应至少包含一个元素。

d) macAlgorithm：是 MAC 算法的标识符。它标识了发起者所使用的 MAC 算法以及任何相关参数。macAlgorithm 域的位置有利于接收者单次处理。

e) digestAlgorithm：标识了当存在鉴别属性时，对封装过的内容计算消息摘要所使用的消息摘要算法以及任何相关参数。消息摘要过程，见 11.3。digestAlgorithm 域的位置有利于接收者进行单次处理。如果存在 digestAlgorithm 域，那么 authAttrs 域也应存在。

f) encapContentInfo：是待鉴别的内容，见 7.3。

g) authAttrs：是鉴别属性的集合。authAttrs 结构是可选的，但是，如果待鉴别的 EncapsulatedContentInfo 值的内容类型不是 id-data，那么它应存在。AuthAttributes 结构应是 DER 编码，即使该结构的其他部分是 BER 编码。一些有用属性类型定义于第 13 章。如果 authAttrs 域存在，那么它应至少包括下列两个属性：

1) 内容类型属性：将待鉴别的 EncapsulatedContentInfo 值的内容类型作为它的值。13.2 定义了内容类型属性。

2) 消息摘要属性：将内容的消息摘要作为它的值。13.3 定义了消息摘要属性。

h) mac：是消息鉴别码。

i) unauthAttrs：是不予鉴别的属性的集合。该域是可选的。至今为止，还没有属性是定义成未鉴定属性使用的，但是有一些其他有用属性类型定义于第 13 章。

11.3　MAC 的生成

MAC 计算过程是对待鉴别的内容，或对发起者的鉴别属性同待鉴别的内容的消息摘要一起，计算 MAC。

如果 authAttrs 不存在，MAC 计算过程的输入就是 encapContentInfo eContent OCTET STRING

值。只有构成 eContent OCTET STRING 值的字节才是 MAC 算法的输入；而删去了标签和长度字节。其好处是在 MAC 生成过程之前，不需要知道待鉴别的内容的长度。

如果 authAttrs 存在，应包括内容类型属性和消息摘要属性，见 13.2 和 13.3，而且 MAC 计算过程的输入是 authAttrs 的 DER 编码。为了 MAC 的计算，要对 authAttrs 域单独执行编码。DER 编码使用 EXPLICIT SET OF 标签，但不使用 authAttrs 域中的 IMPLICIT[2]标签。也就是说，SET OF 标签的 DER 编码，而不是 IMPLICIT[2]标签的 DER 编码，同 authAttrs 值的长度和内容字节一起，包含在 MAC 计算中。

消息摘要计算过程对待鉴别的内容计算消息摘要。消息摘要计算过程的初始输入是待鉴别的封装内容的“值”。具体来说，输入是 encapContentInfo eContent OCTET STRING。只有构成 encapContentInfo eContent OCTET STRING 的值的字节才被输入至消息摘要算法中，而不包括标签或长度字节。其好处是不需要预先知道待鉴别的内容的长度。尽管消息摘要计算中不包括 encapContentInfo eContent OCTET STRING 的标签和长度字节，但它们还是能通过其他方式得到保护。长度字节通过消息摘要算法的特性得到保护，因为要找到消息摘要值相等的两个任意长度的不同消息内容是计算上不可行的。

MAC 计算过程的输入包括上面定义的 MAC 输入数据以及 recipientInfo 结构中传送的鉴别密钥。MAC 计算的详细情况取决于使用的 MAC 算法(如杂凑运算消息鉴别码)。发起者使用的 MAC 算法由对象标识符和任意参数共同指定，它们都装载在 macAlgorithm 域中。最后，发起者生成的 MAC 值被编码成字节串，并装入 mac 域中。

11.4 MAC 的验证

MAC 验证过程的输入包括输入数据(根据 authAttrs 域是否存在确定输入数据，见 11.3)，以及 recipientInfo 结构中传送的鉴别密钥。MAC 验证过程的细节取决于使用的 MAC 算法。

接收者不应依赖任何由发起者计算的 MAC 值或消息摘要值。按照 11.3 所述对内容进行鉴别。如果发起者包括鉴别属性，那么按照 11.3 所述对 authAttrs 的内容进行鉴别。接收者计算的 MAC 值应与 mac 域的值相等，鉴别才算成功。类似的，当 authAttrs 域存在时，接收者计算的内容消息摘要值应与 authAttrs 消息摘要属性中的消息摘要值相等，鉴别才算成功。

如果 AuthenticatedData 包含 authAttrs，那么内容类型属性值应要与 AuthenticatedData encapContentInfo eContentType 值相匹配。

12 有用类型

12.1 算法标识符类型

所有算法标识符都具有相同的类型：AlgorithmIdentifier。AlgorithmIdentifier 的定义来自 GB/T 16264.8—2005。

每个算法类型都有很多其他的选择。

12.1.1 DigestAlgorithmIdentifier

DigestAlgorithmIdentifier 类型标识了消息摘要算法。消息摘要算法将一个字节串(内容)映射为另一个字节串(消息摘要)。

DigestAlgorithmIdentifier ::= AlgorithmIdentifier

12.1.2 SignatureAlgorithmIdentifier

SignatureAlgorithmIdentifier 类型标识了签名算法，而且还可以标识消息摘要算法。签名算法支

持签名的生成和验证操作。签名生成操作使用了消息摘要和签名者的私钥以生成签名结果。签名验证操作使用了消息摘要和签名者的公钥以确定签名结果是否有效。通过上下文决定使用哪种操作。

SignatureAlgorithmIdentifier ::= AlgorithmIdentifier

12.1.3 KeyEncryptionAlgorithmIdentifier

KeyEncryptionAlgorithmIdentifier 类型标识了用于加密内容加密密钥的密钥加密算法。加密操作在密钥加密密钥的作用下,将一个字节串(密钥)映射为另一个字节串(经加密的密钥)。解密操作是加密操作的逆。通过上下文决定使用哪种操作。

加密和解密操作的细节取决于使用的密钥管理算法。有四种支持的密钥管理技术:密钥传输、密钥协商、以前分发的对称密钥加密密钥以及由口令衍生的对称密钥加密密钥。

KeyEncryptionAlgorithmIdentifier ::= AlgorithmIdentifier

12.1.4 ContentEncryptionAlgorithmIdentifier

ContentEncryptionAlgorithmIdentifier 类型标识了内容加密算法。内容加密算法支持加密和解密操作。加密操作在内容加密密钥的作用下,将一个字节串(明文)映射为另一个字节串(密文)。解密操作是加密操作的逆。通过上下文决定使用哪种操作。

ContentEncryptionAlgorithmIdentifier ::= AlgorithmIdentifier

12.1.5 MessageAuthenticationCodeAlgorithm

MessageAuthenticationCodeAlgorithm 类型标识了消息鉴别码(MAC)算法。MAC 算法支持生成和验证操作。MAC 生成和验证操作使用相同的对称密钥。通过上下文决定使用哪种操作。

MessageAuthenticationCodeAlgorithm ::= AlgorithmIdentifier

12.1.6 KeyDerivationAlgorithmIdentifier

KeyDerivationAlgorithmIdentifier 标识了密钥衍生算法。密钥衍生算法将口令或共享秘密值转变为密钥加密密钥。

KeyDerivationAlgorithmIdentifier ::= AlgorithmIdentifier

12.2 其他有用类型

本条定义了在本标准其他地方使用的类型。下列类型没有特定的顺序。

12.2.1 RevocationInfoChoices

RevocationInfoChoices 类型给出一组撤销状态信息选项。预期该集合包含的信息足以确定与该集合相关的证书和属性证书是否已被撤销。但是,集合中包含的撤销状态信息也可多于必要的信息,或者少于必要的信息。GB/T 16264.8—2005 CRL 是撤销状态信息的主要来源,但也可支持任何其他撤销信息格式。该类型提供了 OtherRevocationInfoFormat 选项,用以支持任何其他撤销信息格式,不需要对 ESMS 进行修改。例如,可使用 OtherRevocationInfoFormat 以支持在线证书状态协议。

CertificateList 可能包括 CRL、权威机构撤销列表、增量 CRL 或属性证书撤销列表。所有这些列表共用同一语法。

CertificateList 类型给出了 CRL。CRL 规定于 GB/T 16264.8—2005 中,并且为了 Internet 中的使用,将其描述于 GB/T 20518—2006 中。

CertificateList 的定义来自 GB/T 16264.8—2005。

RevocationInfoChoices ::= SET OF RevocationInfoChoice

```
RevocationInfoChoice ::= CHOICE {
  crl CertificateList,
  other [1] IMPLICIT OtherRevocationInfoFormat }

OtherRevocationInfoFormat ::= SEQUENCE {
  otherRevInfoFormat OBJECT IDENTIFIER,
  otherRevInfo ANY DEFINED BY otherRevInfoFormat }
```

12.2.2 CertificateChoices

CertificateChoices 类型给出了 GB/T 16264.8—2005 证书、GB/T 16264.8—2005 版本 1 属性证书、GB/T 16264.8—2005 版本 2 属性证书，或任何其他证书格式。GB/T 16264.8—2005 证书的 Internet 概要规定于 GB/T 20518—2006。该类型提供了 OtherRevocationInfoFormat 选项，用以支持任何其他证书格式，不需要对 ESMS 进行修改。

Certificate 的定义来自 GB/T 16264.8—2005。

AttributeCertificate 的定义来自 GB/T 16264.8—2005。

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

OtherCertificateFormat ::= SEQUENCE {
  otherCertFormat OBJECT IDENTIFIER,
  otherCert ANY DEFINED BY otherCertFormat }
```


12.2.3 CertificateSet

CertificateSet 类型提供了一组证书。预期该集合足以包括从信任根或顶级证书机构到所有与该集合相关的发送者的证书路径。但是，该集合包含的证书也可多于或少于这些必要证书。

证书路径的精确定义不在本标准的范围之内。GB/T 20518—2006 提供了对 GB/T 16264.8—2005 证书的定义。有些应用可能对证书路径的长度有上界限制；其他一些应用可能对证书路径内证书的主体和签发者强加某些关系。

```
CertificateSet ::= SET OF CertificateChoices
```

12.2.4 IssuerAndSerialNumber

IssuerAndSerialNumber 类型通过证书签发者的可辨别名和签发者签发的证书序列号，标识了证书，从而也标识了实体和公钥。

Name 的定义来自 GB/T 16264.2—2008，而 CertificateSerialNumber 的定义来自 GB/T 16264.8—2005。

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

CertificateSerialNumber ::= INTEGER

12.2.5 ESMSVersion

ESMSVersion 类型给出了语法版本号，目的是与该规范以后的版本能兼容。

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

12.2.6 UserKeyingMaterial

UserKeyingMaterial 类型对 UKM 给出了相应的语法。有些密钥协商算法要求 UKM 以保证每次相同两方生成的成对密钥都是不同的。发送者提供 UKM 与特定密钥协商算法一起使用。

UserKeyingMaterial ::= OCTET STRING

12.2.7 OtherKeyAttribute

OtherKeyAttribute 类型对其他密钥属性所包含的内容给出了相应的语法，这些密钥属性允许接收者选择发送者所使用的密钥。属性对象标识符应与属性本身的语法一起注册。由于该结构可能会有碍互操作性，所以应避免使用。

```
OtherKeyAttribute ::= SEQUENCE {
  keyAttrId OBJECT IDENTIFIER,
  keyAttr ANY DEFINED BY keyAttrId OPTIONAL }
```

13 有用属性

13.1 概述

本章定义了签名数据、封装数据、加密数据和鉴别数据中使用的属性。Attribute 的语法与 GB/T 16264.2—2008 和 GB/T 20518—2006 保持兼容。下列属性没有特定的顺序。

13.2 内容类型

内容类型属性类型规定了签名数据或鉴别数据中 ContentInfo 的内容类型。只要签名数据中存在签名属性或鉴别数据中存在鉴别属性，内容类型属性类型就应存在。内容类型属性值应与签名数据或鉴别数据中的 encapContentInfo eContentType 值相匹配。

内容类型属性应是签名属性或鉴别属性，不应是不予签名的属性或不予鉴别的属性，或不受保护的属性。

下列对象标识符标识了内容类型属性：

```
id-contentType OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs9(9) 3 }
```

内容类型属性值具有 ASN.1 ContentType 类型：

ContentType ::= OBJECT IDENTIFIER

尽管语法定义为一组 AttributeValue 的集合，但是内容类型属性应只有单个属性值，不允许有 0 个或多个 AttributeValue 实例。

SignedAttributes 和 AuthAttributes 语法分别定义为一组 AttributeValue 的集合。signerInfo 中的 SignedAttributes 不应包括多个内容类型属性的实例。相同地，AuthenticatedData 中的 AuthAttributes 也不应包括多个内容类型属性的实例。

13.3 消息摘要

消息摘要属性类型规定了签名数据中待签名的或鉴别数据中待鉴别的 encapContentInfo eContent OCTET STRING，见 7.5 和 11.3。对于签名数据，使用签名者的消息摘要算法计算消息摘要。对于鉴别数据，则是使用发起者的消息摘要算法计算消息摘要。

在签名数据中，当存在任何签名属性时，也应存在消息摘要签名属性类型。在鉴别数据中，当存在任何鉴别属性时，也应存在消息摘要鉴别属性类型。

消息摘要属性应是签名属性或鉴别属性，不应是不予签名的属性、不予鉴别的属性或不受保护的属性。

下列对象标识符标识了消息摘要属性：

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

消息摘要属性值具有 ASN.1 MessageDigest 类型：

MessageDigest ::= OCTET STRING

尽管语法定义为一组 AttributeValue 的集合，但是消息摘要属性应只有单个属性值，不应有 0 个或多个 AttributeValue 实例。

SignedAttributes 和 AuthAttributes 语法分别定义为一组 AttributeValue 的集合。signerInfo 中的 SignedAttributes 只包括单个消息摘要属性的实例。相同地，AuthenticatedData 中的 AuthAttributes 也只包括单个消息摘要属性的实例。

13.4 签名时间

签名时间属性类型规定了签名者执行签名过程据称的时间。签名时间属性类型是为签名数据中的使用而提供的。

签名时间属性应是签名属性或鉴别属性，不应是不予签名的属性、不予鉴别的属性或不受保护的属性。

下列对象标识符标识了签名时间属性：

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

签名时间属性值具有 ASN.1 SigningTime 类型：

SigningTime ::= Time

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

注：Time 的定义与 GB/T 16264.8—2005 中的定义相符。

1950 年 1 月 1 日到 2049 年 12 月 31 日(包括)之间的时期应编码成 UTCTime。任何年份在 1950 年之前或 2049 年之后的日期应编码成 GeneralizedTime。

UTCTime 值应用协调世界时(以前称为格林威治时间(GMT)和祖鲁时钟时间)表示，且应包含秒(也就是说，时间值表示为 YYMMDDHHMMSSZ)，无论秒值是否为 0。半夜十二点应表示成“YYMMDD000000Z”。世纪信息是隐性的，而且应如下确定：

a) 当 YY 大于或等于 50，年份应解释成 19YY。且

b) 当 YY 小于 50，年份应解释成 20YY。

GeneralizedTime 值应用协调世界时表示，且应包含秒(即，时间值表示为 YYYYMMDDH-

HMMSSZ),无论秒值是否为0。GeneralizedTime值不应包含分数形式的秒值。

尽管语法定义为一组AttributeValue的集合,但是签名时间属性应只有单个属性值,不应有0个或多个AttributeValue实例。

SignedAttributes和AuthAttributes语法分别定义为一组AttributeValue的集合。signerInfo中的SignedAttributes不应包括多个签名时间属性的实例。相同地,AuthenticatedData中的AuthAttributes也不应包括多个签名时间属性的实例。

没有针对签名时间正确性的要求,是否接受据称的签名时间由接收者自行决定。但是,有些签名者,如时间戳服务器,被认为是绝对可信的。

13.5 联合签名

联合签名属性类型规定了一个或多个对签名数据的SignerInfo中签名结果字节串的内容字节所进行的签名。也就是说,消息摘要是对构成字节串的值的字节进行计算,而不包括标签或长度字节。因此,联合签名是在另一个签名结果上再进行联合签名(串行签名)。

联合签名属性应是不予签名的属性,不应是签名属性、鉴别属性、不予鉴别的属性或不受保护的属性。

下列对象标识符标识了联合签名属性:

```
id-countersignature OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs9(9) 6 }
```

联合签名属性值具有ASN.1 Countersignature类型:

```
Countersignature ::= SignerInfo
```

对于普通证书而言,Countersignature值与SignerInfo值具有相同的含义,除了:

a) signedAttributes域不应包含内容类型属性,联合签名没有内容类型。

b) 如果signedAttributes域包含任何其他属性,则应包含消息摘要属性。

c) 消息摘要过程的输入是与该属性相关的SignerInfo值中signatureValue域的DER编码的内容字节。

联合签名属性可有多个属性值。其语法定义为一组AttributeValue的集合,且应有一个或多个AttributeValue实例。

UnsignedAttributes语法定义为一组AttributeValue的集合。signerInfo中的UnsignedAttributes可能包含多个联合证书属性的实例。

由于联合证书具有SignerInfo类型,所以它本身也可包含联合证书属性。因此,可以构造一个任意级数的联合签名。

14 ASN.1模块

14.1 ESMS ASN.1模块

```
CryptographicMessageSyntax2004
  { iso(1) member-body(2) us(840) rsadsi(113549)
    pkcs(1) pkcs-9(9) smime(16) modules(0) CMS-2004(24) }

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 RFC 5280, Appendix A.1
        AlgorithmIdentifier, Certificate, CertificateList,
        CertificateSerialNumber, Name
            FROM PKIX1Explicit88
                { iso(1) identified-organization(3) dod(6)
                  internet(1) security(5) mechanisms(5) pkix(7)
                  mod(0) pkix1-explicit(18) }

-- Imports from RFC 3281, Appendix B
        AttributeCertificate
          FROM PKIXAttributeCertificate
              { iso(1) identified-organization(3) dod(6)
                internet(1) security(5) mechanisms(5) pkix(7)
                mod(0) attribute-cert(12) }

-- Imports from Section 14.2 of this document
        AttributeCertificateV1
          FROM AttributeCertificateVersion1
              { iso(1) member-body(2) us(840) rsadsi(113549)
                pkcs(1) pkcs-9(9) smime(16) modules(0)
                v1AttrCert(15) } ;

-- Encryption and Signature Message Syntax

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

ContentType ::= OBJECT IDENTIFIER

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

```
DigestAlgorithmIdentifiers ::= SET OF DigestAlgorithmIdentifier

SignerInfos ::= SET OF SignerInfo

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

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

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

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

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

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

AttributeValue ::= ANY

SignatureValue ::= OCTET STRING

EnvelopedData ::= SEQUENCE {
  version ESMSVersion,
  originatorInfo [0] IMPLICIT OriginatorInfo OPTIONAL,
  recipientInfos RecipientInfos,
  encryptedContentInfo EncryptedContentInfo,
  unprotectedAttrs [1] IMPLICIT UnprotectedAttributes OPTIONAL }

OriginatorInfo ::= SEQUENCE {
  certs [0] IMPLICIT CertificateSet OPTIONAL,
```

```
  crls [1] IMPLICIT RevocationInfoChoices OPTIONAL }

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

EncryptedContentInfo ::= SEQUENCE {
  contentType ContentType,
  contentEncryptionAlgorithm ContentEncryptionAlgorithmIdentifier,
  encryptedContent [0] IMPLICIT EncryptedContent OPTIONAL }

EncryptedContent ::= OCTET STRING

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

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

EncryptedKey ::= OCTET STRING

KeyTransRecipientInfo ::= SEQUENCE {
  version ESMSVersion, -- always set to 0 or 2
  rid RecipientIdentifier,
  keyEncryptionAlgorithm KeyEncryptionAlgorithmIdentifier,
  encryptedKey EncryptedKey }

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

KeyAgreeRecipientInfo ::= SEQUENCE {
  version ESMSVersion, -- always set to 3
  originator [0] EXPLICIT OriginatorIdentifierOrKey,
  ukm [1] EXPLICIT UserKeyingMaterial OPTIONAL,
  keyEncryptionAlgorithm KeyEncryptionAlgorithmIdentifier,
  recipientEncryptedKeys RecipientEncryptedKeys }

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

```
OriginatorPublicKey ::= SEQUENCE {
  algorithm AlgorithmIdentifier,
  publicKey BIT STRING }

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 ESMSVersion, -- always set to 4
  kekid KEKIdentifier,
  keyEncryptionAlgorithm KeyEncryptionAlgorithmIdentifier,
  encryptedKey EncryptedKey }

KEKIdentifier ::= SEQUENCE {
  keyIdentifier OCTET STRING,
  date GeneralizedTime OPTIONAL,
  other OtherKeyAttribute OPTIONAL }

PasswordRecipientInfo ::= SEQUENCE {
  version ESMSVersion, -- always set to 0
  keyDerivationAlgorithm [0] KeyDerivationAlgorithmIdentifier
                             OPTIONAL,
  keyEncryptionAlgorithm KeyEncryptionAlgorithmIdentifier,
  encryptedKey EncryptedKey }

OtherRecipientInfo ::= SEQUENCE {
  oriType OBJECT IDENTIFIER,
  oriValue ANY DEFINED BY oriType }
```

```
DigestedData ::= SEQUENCE {
  version ESMSVersion,
  digestAlgorithm DigestAlgorithmIdentifier,
  encapContentInfo EncapsulatedContentInfo,
  digest Digest }

Digest ::= OCTET STRING

EncryptedData ::= SEQUENCE {
  version ESMSVersion,
  encryptedContentInfo EncryptedContentInfo,
  unprotectedAttrs [1] IMPLICIT UnprotectedAttributes OPTIONAL }

AuthenticatedData ::= SEQUENCE {
  version ESMSVersion,
  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

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

MessageAuthenticationCode ::= OCTET STRING

DigestAlgorithmIdentifier ::= AlgorithmIdentifier

SignatureAlgorithmIdentifier ::= AlgorithmIdentifier

KeyEncryptionAlgorithmIdentifier ::= AlgorithmIdentifier

ContentEncryptionAlgorithmIdentifier ::= AlgorithmIdentifier

MessageAuthenticationCodeAlgorithm ::= AlgorithmIdentifier
KeyDerivationAlgorithmIdentifier ::= AlgorithmIdentifier

RevocationInfoChoices ::= SET OF RevocationInfoChoice
```

```
RevocationInfoChoice ::= CHOICE {
  crl CertificateList,
  other [1] IMPLICIT OtherRevocationInfoFormat }

OtherRevocationInfoFormat ::= SEQUENCE {
  otherRevInfoFormat OBJECT IDENTIFIER,
  otherRevInfo ANY DEFINED BY otherRevInfoFormat }

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

AttributeCertificateV2 ::= AttributeCertificate

OtherCertificateFormat ::= SEQUENCE {
  otherCertFormat OBJECT IDENTIFIER,
  otherCert ANY DEFINED BY otherCertFormat }

CertificateSet ::= SET OF CertificateChoices

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

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

UserKeyingMaterial ::= OCTET STRING

OtherKeyAttribute ::= SEQUENCE {
  keyAttrId OBJECT IDENTIFIER,
  keyAttr ANY DEFINED BY keyAttrId OPTIONAL }

-- 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 }

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

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

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

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

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

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

MessageDigest ::= OCTET STRING

SigningTime ::= Time

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

Countersignature ::= SignerInfo

-- Attribute Object Identifiers

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

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

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

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 ESMSVersion,
    certificate Certificate,
    attributes UnauthAttributes }

  Signature ::= BIT STRING

  END -- of CryptographicMessageSyntax2004
```

14.2 版本 1 属性证书 ASN.1 模块

```
  AttributeCertificateVersion1
      { iso(1) member-body(2) us(840) rsadsi(113549)
        pkcs(1) pkcs-9(9) smime(16) modules(0) v1AttrCert(15) }

  DEFINITIONS EXPLICIT TAGS ::=
  BEGIN

  -- EXPORTS All

  IMPORTS

  -- Imports from RFC 5280, Appendix A.1
       AlgorithmIdentifier, Attribute, CertificateSerialNumber,
       Extensions, UniqueIdentifier
         FROM PKIX1Explicit88
              { iso(1) identified-organization(3) dod(6)
                internet(1) security(5) mechanisms(5) pkix(7)
                mod(0) pkix1-explicit(18) }

  -- Imports from RFC 5280, Appendix A.2
       GeneralNames
       FROM PKIX1Implicit88
              { iso(1) identified-organization(3) dod(6)
                internet(1) security(5) mechanisms(5) pkix(7)
                mod(0) pkix1-implicit(19) }
```

```
-- Imports from RFC 3281, Appendix B
        AttCertValidityPeriod, IssuerSerial
        FROM PKIXAttributeCertificate
                { iso(1) identified-organization(3) dod(6)
                  internet(1) security(5) mechanisms(5) pkix(7)
                  mod(0) attribute-cert(12) } ;

-- Definition extracted from GB/T 16264.8—2005, but
-- different type names are used to avoid collisions.
AttributeCertificateV1 ::= SEQUENCE {
  acInfo AttributeCertificateInfoV1,
  signatureAlgorithm AlgorithmIdentifier,
  signature BIT STRING }

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

AttCertVersionV1 ::= INTEGER { v1(0) }

END -- of AttributeCertificateVersion1
```


15 安全事宜

ESMS 提供了一套办法，用以对数据进行数字签名、摘要、加密以及鉴别。

实现应保护好签名者的私钥。签名者私钥被攻破后，攻击者可以冒充身份。

实现应保护好密钥管理私钥、密钥加密密钥以及内容加密密钥。密钥管理私钥或密钥加密密钥被攻破，可能导致所有用该密钥进行保护的内容遭到泄漏。相同地，内容加密密钥被攻破，可能导致相关加密内容遭到泄漏。

实现应保护好密钥管理私钥和消息鉴别密钥。密钥管理私钥被攻破后，可以伪造鉴别数据。相同地，消息鉴别密钥被攻破后，可能导致鉴别内容遭到不被察觉的修改。

当多于两方共享相同的消息鉴别密钥，数据起源鉴别是无法提供的。知道消息鉴别密钥的任何一方都能够计算有效 MAC，因此，内容可能源自其中任何一方。

实现应随机生成内容加密密钥、消息鉴别密钥、初始向量以及填充内容。而且，公/私钥对的生成要依赖于随机数。使用不够强的 PRNG 生成密钥，会导致安全性降低。攻击者可能更容易复制出制造密钥的 PRNG 环境，然后搜索得到的可能值的集合，而不用穷举整个密钥空间。生成高质量的随机数是比较困难的。

当使用密钥协商算法或以前分发的对称密钥加密密钥时，使用密钥加密密钥加密内容加密密钥。若密钥加密和内容加密算法不同，则实际安全性由两个算法中较弱的决定。例如，如果内容用 168 bit 的内容加密密钥进行加密，内容加密密钥用 40 bit 的密钥加密密钥进行包装，那么最多只提供了 40 bit 的保护。先搜索出 40 bit 的密钥加密密钥值，再恢复出内容加密密钥，然后用该密钥对内容进行解密。因此，实现应保证密钥加密算法同内容加密算法一样强，或比内容加密算法更强。

实现者应意识到，密码算法会随着时间变弱。因为随着新的密码分析技术的出现以及计算性能的提高，破解特定密码算法的工作因数会降低。因此，密码算法的实现应模块化，以便随时插入新的算法。也就是说，实现者应当准备好一组算法，这些算法应支持随时变换。

联合签名-不予签名的属性，包括对内容签名结果进行计算的数字签名。因此，联合签名过程不需要知道原始的签名内容。该结构使得实现高效，但是也可能导致对不正确的签名结果进行联合签名。因此，实行联合签名的实现要么应在联合签名之前验证原始的签名结果(验证需要处理原始内容)，要么应在上下文环境中进行联合签名，保证只有正确的签名结果被联合签名。

中 华 人 民 共 和 国
国 家 标 准
信息安全技术
电子文档加密与签名消息语法
GB/T 31503—2015
*
中 国 标 准 出 版 社 出 版 发 行
北京市朝阳区和平里西街甲 2 号(100029)
北京市西城区三里河北街 16 号(100045)
网址:www.gb168.cn
服务热线:400-168-0010
010-68522006
2015 年 5 月第一版
*
书号:155066·1-51174

GB/T 31503-2015