PKCS #7:加密消息语法标准（Cryptographic Message Syntax Standard）

An RSA Laboratories Technical Note
Version 1.5
Revised November 1, 1993*

1. 范围

这一标准描述了待加密数据的一般语法，比如数字签名和数字信封。该语法允许递归，如一个信封能够包含在另一个当中，或者一方能够对一已存在的封装数据进行签名。它也允许专有的属性和消息的内容一起被鉴别，比如签名时间，并且提供其他属性如伴随着签名的连属(countersignature)。该语法的一个简化版提供了发布证书和CRL的方法。

这一标准和PEM(Privacy-Enhance Mail)兼容，体现在签名数据和签名并封装的数据内容上，以一种PEM兼容格式构成，并能够在无需任何加密操作的情况下转换成PEM消息。类似地，PEM消息也能转换成签名数据和签名封装数据的内容格式。

这一标准能够支持多种基于认证的密钥管理体系结构，比如它的一个提议已收录在PEM[RFC1422]中。一些体系结构上的决定比如何种证书颁发者才是“顶级”的，何种实体证书颁发者应被授权，何种可辨别名能够被接受以及颁发者应该遵循怎样的证书策略等等这些问题不在本标准讨论范围之内。

由这一标准产生的值可能是BER编码的，这意味着该值会以8位字节串(octet string)的形式表示。众所周知，虽然许多系统能够可靠地传输专有的8位字节串，但很多电子邮件系统并没有这么做。这一标准并不寻找编码8位字节串的机制，像ASCII字符串或者其他保证可靠传输的re-encoding 8位字节串技术。RFC 1421 对该问题提出了可能的解决方法。

*Supersedes June 3, 1991 version, which was also published as NIST/OSI Implementors' Workshop document SEC-SIG-91-22. PKCS documents are available by electronic mail to <pkcs@rsa.com>.

003-903022-150-000-000

2. 参考

FIPS PUB 46–1 National Bureau of Standards. *FIPS PUB 46–1: Data Encryption Standard.* January 1988.

PKCS #1 RSA Laboratories. *PKCS #1: RSA Encryption Standard.* Version 1.5, November 1993.

PKCS #6 RSA Laboratories. *PKCS #6: Extended-Certificate Syntax Standard.* Version 1.5, November 1993.

PKCS #9 RSA Laboratories. *PKCS #9: Selected Attribute Types.* Version 1.1, November 1993.

RFC 1421 J. Linn. *RFC 1421: Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures.* February 1993.

RFC 1422 S. Kent. *RFC 1422: Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management.* February 1993.

RFC 1423 D. Balenson. *RFC 1423: Privacy Enhancement for Internet Electronic Mail: Part III: Algorithms, Modes, and Identifiers.* February 1993.

RFC 1424 B. Kaliski. *RFC 1424: Privacy Enhancement for Internet Electronic Mail: Part IV: Key Certification and Related Services.* February 1993.

RFC 1319 B. Kaliski. *RFC 1319: The MD2 Message-Digest Algorithm.* April 1992.

RFC 1321 R. Rivest. *RFC 1321: The MD5 Message-Digest Algorithm.* April 1992.

X.208 CCITT. *Recommendation X.208: Specification of Abstract Syntax Notation One (ASN.1).* 1988.

X.209 CCITT. *Recommendation X.209: Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1).* 1988.

X.500 CCITT. *Recommendation X.500: The Directory – Overview of Concepts, Models and Services.* 1988.

X.501 CCITT. *Recommendation X.501: The Directory – Models.* 1988.

X.509 CCITT. *Recommendation X.509: The Directory – Authentication Framework.* 1988.

[NIST91] NIST. *Special Publication 500-202: Stable Implementation Agreements for Open Systems Interconnection Protocols.* Version 5, Edition 1, Part 12. December 1991.

[RSA78] R.L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM,* 21(2):120–126, February 1978.

3. 定义

For the purposes of this standard, the following definitions apply.

`AlgorithmIdentifier`: A type that identifies an algorithm (by object identifier) and associated parameters. This type is defined in X.509.

ASN.1: Abstract Syntax Notation One, as defined in X.208.

`Attribute`: A type that contains an attribute type (specified by object identifier) and one or more attribute values. This type is defined in X.501.

BER: Basic Encoding Rules, as defined in X.209.

`Certificate`: A type that binds an entity's distinguished name to a public key with a digital signature. This type is defined in X.509. This type also contains the distinguished name of the certificate issuer (the signer), an issuer-specific serial number, the issuer's signature algorithm identifier, and a validity period.

`CertificateSerialNumber`: A type that uniquely identifies a certificate (and thereby an entity and a public key) among those signed by a particular certificate issuer. This type is defined in X.509.

`CertificateRevocationList`: A type that contains information about certificates whose validity an issuer has prematurely revoked. The information consists of an issuer name, the time of issue, the next scheduled time of issue, and a list of certificate serial numbers and their associated revocation times. The CRL is signed by the issuer. The type intended by this standard is the one defined RFC 1422.

DER: Distinguished Encoding Rules for ASN.1, as defined in X.509, Section 8.7.

DES: Data Encryption Standard, as defined in FIPS PUB 46-1.

`desCBC`: The object identifier for DES in cipher-block chaining (CBC) mode, as defined in [NIST91].

`ExtendedCertificate`: A type that consists of an X.509 public-key certificate and a set of attributes, collectively signed by the issuer of the X.509 public-key certificate. This type is defined in PKCS #6.

MD2: RSA Data Security, Inc.'s MD2 message-digest algorithm, as defined in RFC 1319.

`md2`: The object identifier for MD2, as defined in RFC 1319.

MD5: RSA Data Security, Inc.'s MD5 message-digest algorithm, as defined in RFC 1321.

`md5`: The object identifier for MD5, as defined in RFC 1321.

`Name`: A type that uniquely identifies or "distinguishes" objects in an X.500 directory. This type is defined in X.501. In an X.509 certificate, the type identifies the certificate issuer and the entity whose public key is certified.

PEM: Internet Privacy-Enhanced Mail, as defined in RFCs 1421–1424.

RSA: The RSA public-key cryptosystem, as defined in [RSA78].

`rsaEncryption`: The object identifier for RSA encryption, as defined in PKCS #1.

4. 符号和缩略语

No symbols or abbreviations are defined in this standard.

5. 概述

下面的9节指定了有用的类型，通用的语法，六种内容类型和对象标识符。

语法通常足够支持多种不同的内容类型。这一标准定义了六个：数据，签名数据，封装数据，签名封装数据，摘要数据和加密数据。其他内容类型可在未来再加入。可以使用此标准之外定义的内容类型,但是只限于双方对交换内容达成一致的情况。

这一标准输出一种类型、ContentInfo以及各种对象标志符。

有两种内容类型：基本的和增强的。基本的内容类型仅包含数据，没有进行加密。目前有一种内容类型是属于这一类的，即数据内容类型。增强的内容类型包含一些类型（如加密），并有其他一些密码方面的提高。比如，封装数据内容能包含（将其加密）签名数据内容，签名数据内容又能够包含数据内容。四种非数据内容类型属于增强的类别。增强的内容类型使用封装，引发了术语“外部”内容(包含增强特性的)和“内部”内容(得到增强的)。

这一标准的设计使用不定长BER编码来使增强内容类型能够在一个single-pass中准备好，并用任意BER编码在一个single-pass中处理。如果内容是存储在磁带上或是从其他进程“管道”传递而来，则single-pass操作特别有用。single-pass的一个缺点就是在single-pass的过程中难以输出一个DER编码，因为它的不同组件的长度不能预先知道。由于签名数据、签名封装数据和摘要数据的内容类型都需要DER编码，

当一个非数据内容类型是那些内容类型中的一个内部内容时，就需要一个额外的传递。

6. 有用的类型

这一节定义了在标准中至少两个地方有用的类型

6.1 CertificateRevocationLists

CertificateRevocationLists 类型给定一个证书撤销列表的集合。它表示集合中包含足够的信息来决定集合中的证书是否是"hot listed"的，但是可能有多于必要的证书撤销列表，也可能少于必要的。

```
CertificateRevocationLists ::=
  SET OF CertificateRevocationList
```

6.2 ContentEncryptionAlgorithmIdentifier

ContentEncryptionAlgorithmIdentifier类型确定一个内容加密算法比如DES。一个内容加密算法支持加密和解密操作。加密操作用一个内容加密密钥把一个8位字节串（消息）映射为另一个8位字节串（密文）。解密操作和加密操作相反。由上下文确定使用哪 个操作。

```
ContentEncryptionAlgorithmIdentifier ::=
  AlgorithmIdentifier
```

6.3 DigestAlgorithmIdentifier

DigestAlgorithmIdentifier类型确定一个消息摘要算法。例如MD2和MD5。一个消息摘要算法把一个8位字节串（消息）映射位另一个8位字节串（消息摘要）。

```
DigestAlgorithmIdentifier ::= AlgorithmIdentifier
```

6.4 DigestEncryptionAlgorithmIdentifier

DigestEncryptionAlgorithmIdentifier类型确定一个摘要加密算法（可用来加密消息摘要）。 一个例子就是PKCS #1的rsaEncryption。一个摘要加密算法支持加密和解密操作。加密操作用一个摘要加密密钥把一个8位字节串（消息摘要）映射为另一个8位字节串（加了密的摘要）。解密操作和加密操作相反。由上下文确定使用哪 个操作。

```
DigestEncryptionAlgorithmIdentifier ::=
  AlgorithmIdentifier
```

6.5 ExtendedCertificateOrCertificate

ExtendedCertificateOrCertificate类型指定一个PKCS #6扩展证书或者一个X.509证书。这一类型遵循PKCS #6 第6节推荐的语法：

```
ExtendedCertificateOrCertificate ::= CHOICE {
  certificate Certificate, -- X.509
  extendedCertificate [0] IMPLICIT ExtendedCertificate
}
```

6.6 ExtendedCertificatesAndCertificates

ExtendedCertificatesAndCertificates类型指定一个扩展证书和X.509证书的集合。它表示集合足以包含从可识别的“根”或“顶级CA”到所有签名者的证书链，但是可能有多于必要的证书，也可能少于必要的。

```
ExtendedCertificatesAndCertificates ::=
  SET OF ExtendedCertificateOrCertificate
```

注： 关于“链”的准确含义不在本标准的讨论范围之内。这一标准的一些应用可能会对链的长度加一个上限；其他的可能会在主体到颁发者的证书链之间强制加上一些关联。这一关联的一个例子就在RFC1422的Privacy-Enhanced Mail中被提议。

6.7 IssuerAndSerialNumber

IssuerAndSerialNumber类型用一个证书颁发者可识别名和颁发者特定的证书序列号来确定一份证书（和由此而来的实体及公钥）。

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

6.8 KeyEncryptionAlgorithmIdentifier

KeyEncryptionAlgorithmIdentifier类型确定可用来加密对称密钥的密钥加密算法。一个例子就是PKCS #1的rsaEncryption。一个密钥加密算法支持加密和解密操作。加密操作用一个key-encryption密钥把一个8位字节串（密钥）映射为另一个8位字节串（加了密的密钥）。解密操作和加密操作相反。由上下文确定使用哪个操作。

```
KeyEncryptionAlgorithmIdentifier ::=
  AlgorithmIdentifier
```

6.9 Version

为了这一标准的未来版本的兼容性，Version类型给定一个语法版本号。

```
Version ::= INTEGER
```

7. 通用语法

参照此标准，实体间内容交换的通用语法在内容上结合了一个content type。 该语法应有ASN.1类型的 ContentInfo:

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

ContentType ::= OBJECT IDENTIFIER
```

类型ContentInfo的域有以下意义：

- contentType 简要说明该内容的类型。它是一个对象标识符，即它是一个由定义内容类型的权威机构分配的唯一整数串。这一标准定义了六种内容类型：(见 Section14): data, signedData, envelopedData, signedAndEnvelopedData, digestedData, 和 encryptedData。

- content就是内容. 域是可选的，如果该域不存在，它的intended value一定由其他方式给出。它的类型和contentType的对象标识符一起定义。

注：

1. 下面的方法假设内容的类型能够被contentType唯一标识，故随对象标识符一起定义的类型不应是一个CHOICE type。

2. 当一个 ContentInfo 值是诸如 signed-data 、signed-and-enveloped-data、或者digested-data的内部内容,一个消息摘要算法就应用于该内容的DER编码的字节上。当一个 ContentInfo 值是 enveloped-data 或 signed-and-enveloped-data 的内部内容,一个内容加密算法就应用于该内容的定长BER编码的字节上。

3. 内容域的可选冗长项的存在使得构建“外签名”成为可能，如没有改变或代替所要签名的内容。外签名的情况下，被签名的内容将会被从“inner”封装的ContentInfo值（包含在签名数据内容类型中）中删除。

8. Data内容类型

Data内容类型只是一字节串。它应有ASN.1 类型数据:

```
Data ::= OCTET STRING
```

Data内容类型旨在表示任意的字节串，像ASCII文本文件；其翻译留给应用。这样的串无需任何内部的结构（虽然能够，甚至能够用DER编码）

9. Signed-data 内容类型

signed-data 内容类型由任意类型的内容和加密的（0或多个签名者）消息摘要组成。对于一个签名者来说加了密的摘要就是他对该内容的“数字签名”。任何类型的内容能够同时被任意数量的签名者签名。此外，该语法有一个简化版本 ，其中的内容没有签名者。简化版本提供了一个发布证书和crl的方法。

signed-data内容类型的标准应就是用来表示一个签名者对该内容类型数据的数字签名。另一个标准的应用是发布证书和crl。

签名数据的产生过程有如下几步：

1. 对于每一个签名者，他用自己的消息摘要算法计算出摘要值 (如果两个签名者使用同样的算法，那么摘要值只需计算一次) 。若签名者需要鉴别除内容以外的任何信息(见Section 9.2)，消息摘要和其他信息都由签名者的消息摘要算法进行计算，结果即是“消息摘要”。

2. 对于每一个签名者，消息摘要和相关的信息用自己的私钥加密。

3. 对于每一个签名者，把加密的消息摘要和其他的签名者特定信息放入 `SignerInfo` 值中，在Section 9.2中定义了。每个签名者的证书、crl以及那些并不对应任何签名者的信息也在这一步被收集进来。

4. 把所有签名者的信息摘要算法、他们的`SignerInfo`值和内容一起放进`SignedData`值中，在Section 9.1定义了。

接收者通过用签名者的公钥解开加密的信息摘要来验证他的签名，然后把恢复的消息摘要和独立计算的消息摘要进行比较。签名者的公钥或者包含在签名者信息的证书中，或者由一个颁发者可辨别名和颁发序列号来指引，它能唯一标识该公钥证书。

这一节分为5部分。第一部分描述了顶级的`SignedData`，第二部分讲述了每个签名者的信息类型`SignerInfo`, 第三和第四部分描述了信息摘要和摘要加密的过程，第五部分概述了和Privacy-Enhanced Mail的兼容性。

9.1 SignedData 类型

signed-data 内容类型应该是ASN.1 类型的 SignedData:

```
SignedData ::= SEQUENCE {
  version Version,
  digestAlgorithms DigestAlgorithmIdentifiers,
  contentInfo ContentInfo,
  certificates
      [0] IMPLICIT ExtendedCertificatesAndCertificates
      OPTIONAL,
  crls
    [1] IMPLICIT CertificateRevocationLists OPTIONAL,
  signerInfos SignerInfos }

DigestAlgorithmIdentifiers ::=
  SET OF DigestAlgorithmIdentifier

SignerInfos ::= SET OF SignerInfo
```

类型SignedData的域有以下几点意义:

- version是指语法的版本号，这一标准中版本应该为1 。
- digestAlgorithms是消息摘要算法标识符的集合。 可以包含任意数量的元素，包括0。每一个元素标识一种消息摘要算法（和任何相应的参数），内容就是用某个签名者的算法进行摘要运算的。该集合旨在(以任何顺序)列出所有签名者使用的消息摘要算法以方便one-pass签名验证。这一消息摘要处理过程在Section 9.3中有描述。
- contentInfo是被签名内容。它可以包含任意一种定义了的内容类型。
- certificates是PKCS＃6扩展证书和X.509证书的集合。它表示集合足以包含从可识别的“根”或“顶级CA”到signerInfo域中所有签名者的证书链。可能有多于必要的证书，并且可能有足够的证书来包含链（从两个或多个独立的顶级CA）。 也可能有少于必要的证书，比如验证签名有一个替换的方法来获得必要的证书 (e.g., 从一个先前证书集合中)。
- crls是证书撤销列表的集合。它表示集合包含足够的信息来决定certificates域中的证书是否是“hot listed”的，但这不是必须的。可能有多于必要的crl,也可能有少于必要的crl。

- signerInfos是每个签名者信息的集合。其中可有任意数量的元素，包括0。

注：

1. digestAlgorithms 域是列在contentInfo域前面的，且signerInfos 域是列在它后面的，这使得可以在一个single-pass中处理SignedData 值。(single-pass过程在Section 5中描述了)

2. 版本1的SignedData 和版本0的SignedData (defined in PKCS #7, Version 1.4) 的不同如下：

   - 版本1中的digestAlgorithms和signerInfos域可能包含0元素， 而版本0则不能。
   - 版本1中可有crls 域，而版本0中则不行。

   除了版本号上的不同外，版本0的SignedData值可被版本1接受。实现因此可以处理任一版本的SignedData值，就好像都是版本1的值一样。建议PKCS实现仅产生版本1的SignedData值，但是应能准备处理任何版本的SignedData值。

3. 在没有内容签名者的简化版本中，ContentInfo值被签名是不相关(切题)的。该情况下推荐被“签名”的ContentInfo值的content type是data，且忽略ContentInfo的content域。

9.2 SignerInfo 类型

每个签名者的信息在SignerInfo中呈现：

```
SignerInfo ::= SEQUENCE {
  version Version,
  issuerAndSerialNumber IssuerAndSerialNumber,
  digestAlgorithm DigestAlgorithmIdentifier,
  authenticatedAttributes
    [0] IMPLICIT Attributes OPTIONAL,
  digestEncryptionAlgorithm
    DigestEncryptionAlgorithmIdentifier,
  encryptedDigest EncryptedDigest,
  unauthenticatedAttributes
    [1] IMPLICIT Attributes OPTIONAL }

EncryptedDigest ::= OCTET STRING
```

类型SignerInfo的域有以下意义：

- version是指语法的版本号，这一标准中版本应该为1 。

- issuerAndSerialNumber通过颁发者的可辨别名和颁发序列号来指定签名者的证书(及签名者的可辨别名和公钥)。

- digestAlgorithm指定对内容和待鉴别属性(若存在的话)进行摘要计算的信息摘要算法(及相应的参数)。它应该存在于高级SignerInfo值的digestAlgorithms域中。信息摘要的处理在Section 9.3中描述。

- authenticatedAttributes是经由签名者签名（i.e.鉴别的）的属性的集合。 该域是可选的，但是当被签名的ContentInfo的content type不是data类型时该域必须存在。如果该域存在，它必须包含至少两个属性：

    1. PKCS #9 content-type 属性，当ContentInfo的content type值被签名时。

    2. PKCS #9 message-digest 属性，当值是内容的消息摘要时(见下面)。

  这里可能有用的其他属性类型如签名时间等也在PKCS#9中定义。

- digestEncryptionAlgorithm标识用签名者私钥加密消息摘要和相关信息的摘要加密算法（和相应的参数）。消息加密的处理在Section 9.4中定义。

- encryptedDigest是用签名者私钥加密消息摘要和相关信息后的结果。

- unauthenticatedAttributes是不被签名者签名（i.e.鉴别的）的属性的集合。该域是可选的。这里可能有用的属性类型如countersignatures（连署）等在PKCS#9中定义。

注：

1. 考虑到与PEM兼容性，建议当被签名的ContentInfo的content type是data且没有其他的鉴别属性时忽略authenticatedAttributes域。

2. 版本1的SignerInfo和版本0的SignerInfo不同处就在消息摘要的加密处理中(见Section 9.4)。仅在兼容PEM的处理中是不同的，反映的改变在PEM的签名算法上。在不兼容PEM的摘要加密处理中是没有区别的。

建议在PKCS实现中仅产生版本1的SignedData 值。既然版本0兼容的PEM签名方法已废弃，建议PKCS实现仅接受版本1的SignedData值。

9.3 消息摘要的处理

消息摘要的处理是在待签名的内容或附带签名者鉴别属性的内容上计算信息摘要值的。每一种情况中，初始的输入都是待签名内容的“值”。进一步说，初始的输入是ContentInfo的content域中的DER编码的内容字节。仅仅是该域的DER编码的内容字节进行摘要计算，而不包括 identifier字节或length字节。

消息摘要处理的结果(非正式的就称作“消息摘要”)依赖于authenticatedAttributes域是否存在。当该域不存在时，结果就是内容的消息摘要值。然而当该域存在时，结果是包含authenticatedAttributes中的属性值的完全DER编码的消息摘要。既然Attributes值必须当作和内容的content type、消息摘要一样的属性被包含，那些值就间接的包含在结果中。

当待签名内容的content type是data且authenticatedAttributes域不存在时仅该数据的值(e.g.，文件的内容)进行摘要计算。这样有一个优点，就是在加密处理前无需知道待签名的内容的长度。这个方法和PEM兼容。

虽然identifier字节和length字节没有进行摘要计算，他们仍然通过其他方式来保护。Length字节是由消息摘要算法本身来保护的，因为假设的是两个不同的任意长的消息不可能计算出相同的摘要值。 此外，假设content type唯一决定identifier 字节，identifier字节由两种方法隐式地保护：或者由authenticated属性中content type所包含的，或者由PEM兼容的使用，两者可选其一，在Section9.4中暗含了content type是data。

注：消息摘要是由部分DER编码计算出来的这一事实并不表示DER就是必须用到的描述那部分数据转换的方法。实际上，期望这一标准的实现会使用DER编码以外的方法存储对象，但这样操作并不影响消息摘要的计算。

9.4 摘要加密的处理

摘要加密过程的输入是消息摘要处理的结果和摘要算法标识符(或object identifier)。摘要加密过程的结果是用签名者私钥对DigestInfo类型值的BER编码进行加密:

```
DigestInfo ::= SEQUENCE {
  digestAlgorithm DigestAlgorithmIdentifier,
  digest Digest }

Digest ::= OCTET STRING
```

DigestInfo类型的域有以下意义：

- digestAlgorithm标识用来计算内容和鉴别属性的摘要的消息摘要算法（和相应的参数）。它应该和SignerInfo的digestAlgorithm域有同样的值。
- digest是消息摘要处理的结果。

注：

1. 这里定义的签名过程和PKCS＃1中定义的签名算法的唯一不同在于PKCS＃1中的签名用比特串表示（为了和X509的SIGNED宏一致），这里加密的消息摘要用字节串表示。
2. 加密处理的输入通常有30个或更少字节。如果digestEncryptionAlgorithm是PKCS #1的rsaEncryption,则表示输入能够在一个单块中加密，只要RSA模块的长度至少有328比特（这样是合理的并且和安全建议相一致）。
3. 消息摘要算法标识符包含在DigestInfo值中，to limit the damage resulting from the compromise of one message-digest algorithm. 比如，假设敌人能够通过给定的MD2消息摘要找到消息，那个敌人就能通过找到一个有同样MD2消息摘要值的消息来伪造一个签名，然后把先前的签名和新的消息放一起。仅当签名者用MD2算法时这一攻击才能奏效，因为DigestInfo值包含消息摘要算法。如果一个签名者从不相信MD2算法并总是使用MD5算法，那么MD2的危害将不会影响到该签名者。如果DigestInfo值仅包含消息摘要，则MD2的危害就会影响使用任意签名算法的签名者。
4. 由于DigestInfo值并没有指示digest域是否仅包含内容的消息摘要或是有authenticatedAttributes的完全DER编码的消息摘要，故存在一个潜在的含糊的地方。换句话说，敌人有可能把在鉴别属性上的签名转换成只在内容上的签名，通过改变将要DER编码的内容并去掉authenticatedAttributes值（反向的转换也是可能的，但是需要鉴别属性内容的DER编码值，这不大可能）。这一含糊的地方不是新问题，也不是一个大问题，因为上下文通常会防止误用。实际上，一个敌人把在证书或crl上的签名转换成只在签名数据内容上的签名也是可能的。

9.5 同PEM的兼容性

当待签名的ContentInfo的content type值是data、没有鉴别属性、消息摘要算法是md2或md5、摘要加密算法是PKCS＃1的rsaEncryption时，和PEM的MIC-ONLY

和MIC-CLEAR处理类型兼容。在上述所有情况下，产生的加密消息摘要和PEM产生的一致，因为：

1. PEM中消息摘要算法的输入值和没有鉴别属性的这个标准是一样的。PEM中的MD2、MD5和md2、md5一样。

2. PEM中用签名者私钥加密的值(RFC1423中规定的)和没有鉴别属性的这个标准是一样的。PEM中的RSA私钥加密和PKCS #1的rsaEncryption 是一样的。

signed-data内容类型的其他部分（证书，CRLs，算法标识符等）很容易转成相应的PEM组件或从那些组件中转过来。

10. Enveloped-data 内容类型

enveloped-data内容类型由任意类型的加密内容和加密的一个/多个接收者的内容加密密钥组成。加了密的内容和加了密的内容加密密钥一起构成了接收者的“数字信封”。任意类型的内容能够同时为任意数量的接收者进行封装。

期望的是enveloped-data内容类型的标准应用能够表示一个或多个接收者的data、digested-data或signed-data内容类型的数字信封。

Enveloped-data的组建过程分以下几步：

1. 随机产生一个对应于特定加密算法的内容加密密钥。

2. 内容加密密钥用每个接收者的公钥加密。

3. 对于每一个接收者，把加了密的内容加密密钥和接收者的其他信息放入RecipientInfo值中。

4. 用内容加密密钥加密内容。(内容加密可能会需要填充一些块；见Section 10.3 的讨论)

5. 将所有接收者的RecipientInfo值和加了密的内容放入EnvelopedData值中， Section 10.1中定义了。

接收者用自己的私钥解开加密的内容加密密钥，然后用该密钥解密密文内容。接受者的私钥由一个颁发者可辨别名和颁发序列号来指引，它能唯一标识该公钥证书。

这一节分为4部分。第一部分描述顶级类型的EnvelopedData，第二部分描述每个接受者的信息类型RecipientInfo，第三和第四部分描述内容加密和密钥加密过程。

因为PEM总是包括数字签名，从来不是单独的封装，所以这一内容类型和PEM不兼容(虽然一些过程和PEM的副本兼容)。

10.1 EnvelopedData 类型

enveloped-data内容类型应有ASN.1类型的EnvelopedData:

```
EnvelopedData ::= SEQUENCE {
  version Version,
  recipientInfos RecipientInfos,
  encryptedContentInfo EncryptedContentInfo }

RecipientInfos ::= SET OF RecipientInfo

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

EncryptedContent ::= OCTET STRING
```

EnvelopedData类型域有以下意义：

- version是指语法的版本号，这一标准中版本应该为0。
- recipientInfos是每个接收者信息的集合。这里至少有一个元素。
- encryptedContentInfo是加了密的内容信息。

EncryptedContentInfo类型域有以下意义：

- contentType指示内容的类型。
- contentEncryptionAlgorithm标识内容加密算法（和相应的参数）。内容加密处理在Section 10.3中描述。这一算法对所有接收者都一样。
- encryptedContent是内容加密的结果。该域是可选的，如果该域不存在，它的值应由其他方法提供。

注：recipientInfos域在encryptedContentInfo域前面的这一事实使得在一个single-pass中处理EnvelopedData值成为可能。

10.2 RecipientInfo类型

每个接收者信息用RecipientInfo类型表示:

```
RecipientInfo ::= SEQUENCE {
  version Version,
  issuerAndSerialNumber IssuerAndSerialNumber,
  keyEncryptionAlgorithm
    KeyEncryptionAlgorithmIdentifier,
  encryptedKey EncryptedKey }

EncryptedKey ::= OCTET STRING
```

RecipientInfo类型的域有以下意义：

- version是指语法的版本号，这一标准中版本应该为0。
- issuerAndSerialNumber指定由颁发者可辨别名和颁发序列号确定的接收者证书（和相应的接收者辨别名、公钥）。
- keyEncryptionAlgorithm指定用接收者公钥加密内容加密密钥的密钥加密算法（和相应的参数）。密钥加密过程在Section 10.4中定义。
- encryptedKe是内容加密密钥被接收者公钥加密(见下)后的结果。

10.3 内容加密过程

内容加密过程的输入是待封装的内容“值”。进一步说，输入的是ContentInfo的content域中的定长BER编码的内容字节，封装过程就对这一输入进行处理。仅仅是该域的BER编码的内容字节进行加密，而不包括 identifier字节或length字节；其他字节也跟本不表述。

当被封装的内容是data内容类型，则仅有data的值（如文件内容）被加密。这样有一个优点，就是在加密处理前无需知道待加密的内容的长度。这个方法和PEM兼容。

identifier字节和length字节没有加密。Length字节可能隐含地被加密过程保护，这依赖域加密算法。Identifier字节根本就不保护，虽然它们能够从内容类型中恢复，假设内容类型唯一决定identifier 字节。显式的保护identifier和length字节需要用signed-and-enveloped-data 内容类型，或者成功的应用了digested-data 和enveloped-data内容类型。

注：

1. 需要定长BER编码的原因是指示长度是否为有限或无限的bit位没有在enveloped-data内容类型中的任何地方记录。定长编码更加适合简单类型如8位字节串，所以就选择定长编码。

2. 一些内容加密算法假设输入的长度是k字节的倍数， k>1，并让应用定义方法来处理输入长度不为k字节的倍数的情况。对于这样一个算法，the method shall be to pad the input at the trailing end with $k - (l \bmod k)$ octets all having value $k - (l \bmod k)$, where l is the length of the input. 换句话说，输入在尾端的填充使用如下一种串：

$$\begin{aligned} &01 && \text{— if } l \bmod k = k\text{-}1 \\ &02\ 02 && \text{— if } l \bmod k = k\text{-}2 \\ &\vdots \\ &k\ k \ldots k\ k && \text{— if } l \bmod k = 0 \end{aligned}$$

填充能够明确的移除，因为所有的输入都有填充并且没有填充串是其他的后缀。这一填充方法是定义良好的仅当k<256时；更大的k的方法仍在进一步研究的公开讨论中。

10.4 密钥加密过程

密钥加密过程的输入就是内容加密密钥的“值”，由接收者的密钥加密算法提供。

11. Signed-and-enveloped-data 内容类型

这一节定义了signed-and-enveloped-data内容类型。从简起见，本节的许多内容用Sections9和10的术语进行表述。

signed-and-enveloped-data内容类型由任意类型的加密内容、加了密的一个/多个接收者的内容加密密钥和双重加密的一个/多个签名者的消息摘要。“双重加密”由签名者私钥的加密和内容加密密钥的加密组成。

加了密的内容和加了密的内容加密密钥一起构成了接收者的“数字信封”。恢复的签名者单个加密的消息摘要是恢复的签名者内容的“数字签名”。任意类型的内容能够同时为任意数量的接收者进行封装和被任意数量的签名者进行签名。

期望的是signed-and-enveloped-data内容类型的标准应用能够表示一个签名者的数字签名和一个/多个接收者的数字信封，在data内容类型的内容上。

signed-and-enveloped data的组建过程包括以下几步：

1. 随机产生一个对应于特定加密算法的内容加密密钥。

2. 内容加密密钥用每个接收者的公钥加密。

3. 对于每一个接收者，把加了密的内容加密密钥和接收者的其他信息放入`RecipientInfo`值中，Section 10.2中定义了。

4. 对于每一个签名者，他用自己的消息摘要算法计算出摘要值 (如果两个签名者使用同样的算法，那么摘要值只需计算一次) 。

5. 对于每一个签名者，消息摘要和相关的信息用自己的私钥加密，结果再用内容加密密钥加密。(第二个加密过程可能需要对第一个加密结果进行块的填充，见Section 10.3的讨论)

6. 对于每一个签名者，把双重加密的消息摘要和其他的签名者特定信息放入`SignerInfo` 值中，在Section 定义。

7. 用内容加密密钥加密内容。(见Section 10.3的讨论)

8. 把所有签名者的消息摘要算法、所有签名者的SignerInfo值、所有接收者的RecipientInfo值和加了密的内容一起放入`SignedAndEnvelopedData` 值中，Section 11.1中定义了。

接收者打开信封并验证签名分两部。首先，加了密的内容加密密钥用接收者的私钥解开，并用内容加密密钥解开加密的内容。其次，每个签名者双重加密的消息摘要用恢复的内容加密密钥解开，结果再用签名者公钥解密，恢复的消息摘要再和独立计算的消息摘要进行比较。

接收者的私钥和签名者的公钥在Sections9和10中讨论。

这一节分为三部分。第一部分描述顶级类型的`SignedAndEnvelopedData` ，第二部分描述摘要加密过程。其他类型和过程和Sections 9 、10中一样。第三部分总结了和PEM的兼容性。

注：signed-and-enveloped-data内容类型提供增强加密，和那些signed-data和enveloped-data内容类型有序结合的结果类似。然而，signed-and-enveloped-data内容类型没有鉴别或非鉴别属性，也不提供签名以的外签名者信息的封装，signed-data 和 enveloped-data 内容类型有序结合通常比`SignedAndEnvelopedData` 更情愿被使用，除非有意考虑PEM加密处理的兼容性。

11.1 SignedAndEnvelopedData 类型

signed-and-enveloped-data 内容类型应有 ASN.1 类型的 SignedAndEnvelopedData:

```
SignedAndEnvelopedData ::= SEQUENCE {
  version Version,
  recipientInfos RecipientInfos,
  digestAlgorithms DigestAlgorithmIdentifiers,
  encryptedContentInfo EncryptedContentInfo,
  certificates
      [0] IMPLICIT ExtendedCertificatesAndCertificates
OPTIONAL,
  crls
    [1] IMPLICIT CertificateRevocationLists OPTIONAL,
  signerInfos SignerInfos }
```

SignedAndEnvelopedData类型的域有以下意义：

version是指语法的版本号，这一标准中版本应该为1。

recipientInfos是每个接收者信息的集合，见Section10。至少要有一个元素。

digestAlgorithms是消息摘要算法标识符的集合，见Section9。当没有鉴别属性时消息摘要处理过程和Section 9中一样。

encryptedContentInfo是加了密的内容，见Section 10。它可以是任何定义的内容类型。

certificates是PKCS＃6扩展证书何X509证书的集合，见Section 9.

crls是证书撤销列表的集合，见Section 9。

signerInfos是每个签名者信息集合。至少要有一个元素。SignerInfo值和Section 9中同样的意义，除了encryptedDigest域（见下）。

注：

1. recipientInfos 域和 digestAlgorithms 域出现在 contentInfo域之前以及signerInfos 域出现在它之后的这一事实使得在一个single-pass中处理SignedAndEnvelopedData成为可能。(single-pass在Section 5中描述)

2. 版本1的SignedAndEnvelopedData和版本0的SignedAndEnvelopedData的不同就是在版本1中允许有crls域，而版本0中则不行。除了版本号的不同，版本0的SignedAndEnvelopedData值也能被版本1所接受。因此实现能够处理任何版本的SignedAndEnvelopedData值，就好像都是版本1的值一样。建议PKCS的实现仅产生版本1的SignedAndEnvelopedData值，但是能够处理任意版本的SignedAndEnvelopedData值。

11.2摘要加密处理

摘要加密处理的输入和Section 9中一样，但是本身的处理是不同的。明确的说，该过程分为两部。首先，处理的输入提供给签名者的摘要加密算法，如Section 9中所说。其次，第一步的结果用内容加密密钥加密。这两步中没有用DER编码；第一步输出的“值”直接作为第二步的输入。(见Section 10.3的讨论)

该处理过程和PEM的加密处理过程兼容。

注：第二步的目的是防止敌人恢复内容的信息摘要。否则，敌人就能够通过比较那些消息摘要值和实际的消息摘要值来确定列表上的哪一部分是真实的内容。

11.3和PEM的兼容性

当被签名和封装的ContentInfo值的内容类型是data、消息摘要算法是md2或md5、内容加密算法是CBC模式的DES、摘要加密算法是PKCS＃1的rsaEncryption、密钥加密算法是PKCS＃1的rsaEncryption时，和PEM的加密处理过程是兼容的。在上述所有条件下，双重加密的消息摘要和加密的内容加密密钥同PEM中产生的相符合，原因类似于Section 9.5中所说的，如下：

1. PEM中输入到内容加密算法的值和这个标准是一样的。CBC模式的DES和desCBC是一样的。

2. PEM中输入到密钥加密算法的值和这个标准是一样的（见Section 10.4）。PEM中的RSA公钥钥加密和PKCS＃1的rsaEncryption 是一样的。

3. 这一标准中对消息摘要的双重加密处理过程和PEM中的处理是一样的。

signed-and-enveloped-data内容类型的其他部分（证书，CRLs，算法标识符等）能够很容易的转换成相应的PEM组件或从其转换而来。(CRLs在一个独立的PEM消息中装载。)

12. Digested-data内容类型

digested-data内容类型由任意类型的内容和内容的消息摘要组成。

期望的是digested-data内容类型的标准应用在data内容类型上增加完整性，其结果成为enveloped-data内容类型的内容输入。

digested-data的组建过程分以下几步：

1. 使用消息摘要算法对内容计算摘要值。
2. 把消息摘要算法、消息摘要和内容放入DigestedData值中。

digested-data内容类型应为ASN.1类型的DigestedData:

```
DigestedData ::= SEQUENCE {
  version Version,
  digestAlgorithm DigestAlgorithmIdentifier,
  contentInfo ContentInfo,
  digest Digest }

Digest ::= OCTET STRING
```

DigestedData类型的域有以下意义：

- version是指语法的版本号，这一标准中版本应该为0。
- digestAlgorithm标识对内容进行摘要计算的消息摘要算法（和相应的参数）。(消息摘要的处理过程和Section 9中无鉴别属性的情况一样)
- contentInfo是对其进行摘要计算的内容。可以是任意定义的内容类型。
- digest是消息摘要计算的结果。

注： digestAlgorithm 域在contentInfo域之前且digest域在它之后这一事实使得在一个single-pass中处理DigestedData值成为可能 (Single-pass处理在Section 5中定义了) 。

13. Encrypted-data内容类型

encrypted-data内容类型由任意类型的加密内容组成。不像enveloped-data 内容类型，encrypted-data 内容类型既没有接收者也没有加密的内容加密密钥。假设密钥是由其他方法管理的。

期望的是encrypted-data内容类型的标准应用是来加密本地存储的data内容类型的，可能密钥就是一个password。

encrypted-data内容类型应是ASN.1类型的EncryptedData:

```
EncryptedData ::= SEQUENCE {
  version Version,
  encryptedContentInfo EncryptedContentInfo }
```

EncryptedData类型的域有以下意义：

- version是指语法的版本号，这一标准中版本应该为0。
- encryptedContentInfo是加了密的内容信息，像Section 10中一样。

14. Object identifiers

这一标准定义了7个对象标识符：pkcs-7, data, signedData, envelopedData, signedAndEnvelopedData, digestedData和 encryptedData.

对象标识符pkcs-7指这一标准。

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

对象标识符data, signedData, envelopedData, signedAndEnvelopedData, digestedData和encryptedData分别标识Sections 8–13中定义的数据，签名数据，封装数据，签名封装数据，摘要数据和加密数据内容类型。

```
data OBJECT IDENTIFIER ::= { pkcs-7 1 }
signedData OBJECT IDENTIFIER ::= { pkcs-7 2 }
envelopedData OBJECT IDENTIFIER ::= { pkcs-7 3 }
signedAndEnvelopedData OBJECT IDENTIFIER ::=
  { pkcs-7 4 }
digestedData OBJECT IDENTIFIER ::= { pkcs-7 5 }
encryptedData OBJECT IDENTIFIER ::= { pkcs-7 6 }
```

这些对象标识符用在ContentInfo值的contentType域中(见Section 5)。那些类型的content域应分别有ASN.1类型的Data, SignedData, EnvelopedData, SignedAndEnvelopedData, DigestedData和EncryptedData。这些对象标识符也在PKCS＃9内容类型属性中使用。

Revision history

Versions 1.0–1.3

Versions 1.0–1.3 were distributed to participants in RSA Data Security, Inc.'s Public-Key Cryptography Standards meetings in February and March 1991.

Version 1.4

Version 1.4 is part of the June 3, 1991 initial public release of PKCS. Version 1.4 was published as NIST/OSI Implementors' Workshop document SEC-SIG-91-22.

Version 1.5

Version 1.5 incorporates several editorial changes, including updates to the references and the addition of a revision history. The following substantive changes were made:

- Section 6: `CertificateRevocationLists` type is added.
- Section 9.1: `SignedData` syntax is revised. The new version allows for the dissemination of certificate-revocation lists along with signatures. It also allows for the dissemination of certificates and certificate-revocation lists alone, without any signatures.
- Section 9.2: `SignerInfo` syntax is revised. The new version includes a message-digest encryption process compatible with Privacy-Enhanced Mail as specified in RFC 1423.
- Section 9.3: Meaning of "the DER encoding of the `authenticatedAttributes` field" is clarified as "the DER encoding of the `Attributes` value."
- Section 10.3: Padding method for content-encryption algorithms is described.
- Section 11.1: `SignedAndEnvelopedData` syntax is revised. The new version allows for the dissemination of certificate-revocation lists.
- Section 13: Encrypted-data content type is added. This content type consists of encrypted content of any type.
- Section 14: `encryptedData` object identifier is added.

Author's address

RSA Laboratories
100 Marine Parkway
Redwood City, CA 94065 USA

(415) 595-7703
(415) 595-4126 (fax)
pkcs-editor@rsa.com