Network Working Group
Request for Comments: 2312
Category: Informational

S. Dusse
RSA Data Security
P. Hoffman
Internet Mail Consortium
B. Ramsdell
Worldtalk
J. Weinstein
Netscape
March 1998

S/MIME Version 2 Certificate Handling

Status of this Memo

1. Overview

S/MIME (Secure/Multipurpose Internet Mail Extensions), described in [SMIME-MSG], provides a method to send and receive secure MIME messages. In order to validate the keys of a message sent to it, an S/MIME agent needs to certify that the key is valid. This memo describes the mechanisms S/MIME uses to create and validate keys using certificates.

This specification is compatible with PKCS #7 in that it uses the data types defined by PKCS #7. It also inherits all the varieties of architectures for certificate-based key management supported by PKCS #7. Note that the method S/MIME messages make certificate requests is defined in [SMIME-MSG].

In order to handle S/MIME certificates, an agent has to follow specifications in this memo, as well as some of the specifications listed in the following documents:

- "PKCS #1: RSA Encryption", [PKCS-1].
- "PKCS #7: Cryptographic Message Syntax", [PKCS-7]
- "PKCS #10: Certification Request Syntax", [PKCS-10].

Please note: The information in this document is historical material being published for the public record. It is not an IETF standard. The use of the word "standard" in this document indicates a standard for adopters of S/MIME version 2, not an IETF standard.

1.1 Definitions

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

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

BER: Basic Encoding Rules for ASN.1, as defined in CCITT 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 CCITT X.509 [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.

Certificate Revocation List (CRL): 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 specification is the one defined in [KEYM].

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

1.2 Compatibility with Prior Practice of S/MIME

Appendix C contains important information about how S/MIME agents following this specification should act in order to have the greatest interoperability with earlier implementations of S/MIME.

1.3 Terminology

Throughout this memo, the terms MUST, MUST NOT, SHOULD, and SHOULD NOT are used in capital letters. This conforms to the definitions in [MUSTSHOULD]. [MUSTSHOULD] defines the use of these key words to help make the intent of standards track documents as clear as possible. The same key words are used in this document to help implementors achieve interoperability.

2. PKCS #7 Options

The PKCS #7 message format allows for a wide variety of options in content and algorithm support. This section puts forth a number of support requirements and recommendations in order to achieve a base level of interoperability among all S/MIME implementations. Most of the PKCS #7 format for S/MIME messages is defined in [SMIME-MSG].

2.1 CertificateRevocationLists

Receiving agents MUST support for the Certificate Revocation List (CRL) format defined in [KEYM]. If sending agents include CRLs in outgoing messages, the CRL format defined in [KEYM] MUST be used.

All agents MUST validate CRLs and check certificates against CRLs, if available, in accordance with [KEYM]. All agents SHOULD check the nextUpdate field in the CRL against the current time. If the current time is later than the nextUpdate time, the action that the agent takes is a local decision. For instance, it could warn a human user, it could retrieve a new CRL if able, and so on.

Receiving agents MUST recognize CRLs in received S/MIME messages.

Clients MUST use revocation information included as a CRL in an S/MIME message when verifying the signature and certificate path validity in that message. Clients SHOULD store CRLs received in messages for use in processing later messages.

Clients MUST handle multiple valid Certificate Authority (CA) certificates containing the same subject name and the same public keys but with overlapping validity intervals.

2.2 ExtendedCertificateOrCertificate

Receiving agents MUST support X.509 v1 and X.509 v3 certificates. See [KEYM] for details about the profile for certificate formats. End entity certificates MUST include an Internet mail address, as described in section 3.1.

2.2.1 Historical Note About PKCS #7 Certificates

The PKCS #7 message format supports a choice of certificate two formats for public key content types: X.509 and PKCS #6 Extended Certificates. The PKCS #6 format is not in widespread use. In addition, proposed revisions of X.509 certificates address much of the same functionality and flexibility as was intended in the PKCS #6. Thus, sending and receiving agents MUST NOT use PKCS #6 extended certificates.

2.3 ExtendedCertificateAndCertificates

Receiving agents MUST be able to handle an arbitrary number of certificates of arbitrary relationship to the message sender and to each other in arbitrary order. In many cases, the certificates included in a signed message may represent a chain of certification from the sender to a particular root. There may be, however, situations where the certificates in a signed message may be unrelated and included for convenience.

Sending agents SHOULD include any certificates for the user's public key(s) and associated issuer certificates. This increases the likelihood that the intended recipient can establish trust in the originator's public key(s). This is especially important when sending a message to recipients that may not have access to the sender's public key through any other means or when sending a signed message to a new recipient. The inclusion of certificates in outgoing messages can be omitted if S/MIME objects are sent within a group of correspondents that has established access to each other's certificates by some other means such as a shared directory or manual certificate distribution. Receiving S/MIME agents SHOULD be able to handle messages without certificates using a database or directory lookup scheme.

A sending agent SHOULD include at least one chain of certificates up to, but not including, a Certificate Authority (CA) that it believes that the recipient may trust as authoritative. A receiving agent SHOULD be able to handle an arbitrarily large number of certificates and chains.

Clients MAY send CA certificates, that is, certificates that are self-signed and can be considered the "root" of other chains. Note that receiving agents SHOULD NOT simply trust any self-signed certificates as valid CAs, but SHOULD use some other mechanism to determine if this is a CA that should be trusted.

Receiving agents MUST support chaining based on the distinguished name fields. Other methods of building certificate chains may be supported but are not currently recommended.

3. Distinguished Names in Certificates

3.1 Using Distinguished Names for Internet Mail

The format of an X.509 certificate includes fields for the subject name and issuer name. The subject name identifies the owner of a particular public key/private key pair while the issuer name is meant to identify the entity that "certified" the subject (that is, who signed the subject's certificate). The subject name and issuer name are defined by X.509 as Distinguished Names.

Distinguished Names are defined by a CCITT standard X.501 [X.501]. A Distinguished Name is broken into one or more Relative Distinguished Names. Each Relative Distinguished Name is comprised of one or more Attribute-Value Assertions. Each Attribute-Value Assertion consists of a Attribute Identifier and its corresponding value information, such as CountryName=US. Distinguished Names were intended to identify entities in the X.500 directory tree [X.500]. Each Relative Distinguished Name can be thought of as a node in the tree which is described by some collection of Attribute-Value Assertions. The entire Distinguished Name is some collection of nodes in the tree that traverse a path from the root of the tree to some end node which represents a particular entity.

The goal of the directory was to provide an infrastructure to uniquely name every communications entity everywhere. However, adoption of a global X.500 directory infrastructure has been slower than expected. Consequently, there is no requirement for X.500 directory service provision in the S/MIME environment, although such provision would almost undoubtedly be of great value in facilitating key management for S/MIME.

The use of Distinguished Names in accordance with the X.500 directory is not very widespread. By contrast, Internet mail addresses, as described in RFC 822 [RFC-822], are used almost exclusively in the Internet environment to identify originators and recipients of messages. However, Internet mail addresses bear no resemblance to X.500 Distinguished Names (except, perhaps, that they are both hierarchical in nature). Some method is needed to map Internet mail addresses to entities that hold public keys. Some people have

suggested that the X.509 certificate format should be abandoned in favor of other binding mechanisms. Instead, S/MIME keeps the X.509 certificate and Distinguished Name mechanisms while tailoring the content of the naming information to suit the Internet mail environment.

End-entity certificates MUST contain an Internet mail address as described in [RFC-822]. The address must be an "addr-spec" as defined in Section 6.1 of that specification.

Receiving agents MUST recognize email addresses in the subjectAltName field. Receiving agents MUST recognize email addresses in the Distinguished Name field.

Sending agents SHOULD make the address in the From header in a mail message match an Internet mail address in the signer's certificate. Receiving agents MUST check that the address in the From header of a mail message matches an Internet mail address in the signer's certificate. A receiving agent MUST provide some explicit alternate processing of the message if this comparison fails, which may be to reject the message.

3.2 Required Name Attributes

Receiving agents MUST support parsing of zero, one, or more instances of each of the following set of name attributes within the Distinguished Names in certificates.

Sending agents MUST include the Internet mail address during Distinguished Name creation. Guidelines for the inclusion, omission, and ordering of the remaining name attributes during the creation of a distinguished name will most likely be dictated by the policies associated with the certification service which will certify the corresponding name and public key.

CountryName
StateOrProvinceName
Locality
CommonName
Title
Organization
OrganizationalUnit
StreetAddress
PostalCode
PhoneNumber
EmailAddress

All attributes other than EmailAddress are described in X.520 [X.520]. EmailAddress is an IA5String that can have multiple attribute values.

4. Certificate Processing

A receiving agent needs to provide some certificate retrieval mechanism in order to gain access to certificates for recipients of digital envelopes. There are many ways to implement certificate retrieval mechanisms. X.500 directory service is an excellent example of a certificate retrieval-only mechanism that is compatible with classic X.500 Distinguished Names. The PKIX Working Group is investigating other mechanisms. Another method under consideration by the IETF is to provide certificate retrieval services as part of the existing Domain Name System (DNS). Until such mechanisms are widely used, their utility may be limited by the small number of correspondent's certificates that can be retrieved. At a minimum, for initial S/MIME deployment, a user agent could automatically generate a message to an intended recipient requesting that recipient's certificate in a signed return message.

Receiving and sending agents SHOULD also provide a mechanism to allow a user to "store and protect" certificates for correspondents in such a way so as to guarantee their later retrieval. In many environments, it may be desirable to link the certificate retrieval/storage mechanisms together in some sort of certificate database. In its simplest form, a certificate database would be local to a particular user and would function in a similar way as a "address book" that stores a user's frequent correspondents. In this way, the certificate retrieval mechanism would be limited to the certificates that a user has stored (presumably from incoming messages). A comprehensive certificate retrieval/storage solution may combine two or more mechanisms to allow the greatest flexibility and utility to the user. For instance, a secure Internet mail agent may resort to checking a centralized certificate retrieval mechanism for a certificate if it can not be found in a user's local certificate storage/retrieval database.

Receiving and sending agents SHOULD provide a mechanism for the import and export of certificates, using a PKCS #7 certs-only message. This allows for import and export of full certificate chains as opposed to just a single certificate. This is described in [SMIME-MSG].

4.1 Certificate Revocation Lists

A receiving agent SHOULD have access to some certificate-revocation list (CRL) retrieval mechanism in order to gain access to certificate-revocation information when validating certificate chains. A receiving or sending agent SHOULD also provide a mechanism to allow a user to store incoming certificate-revocation information for correspondents in such a way so as to guarantee its later retrieval. However, it is always better to get the latest information from the CA than to get information stored away from incoming messages.

Receiving and sending agents SHOULD retrieve and utilize CRL information every time a certificate is verified as part of a certificate chain validation even if the certificate was already verified in the past. However, in many instances (such as off-line verification) access to the latest CRL information may be difficult or impossible. The use of CRL information, therefore, may be dictated by the value of the information that is protected. The value of the CRL information in a particular context is beyond the scope of this memo but may be governed by the policies associated with particular certificate hierarchies.

4.2 Certificate Chain Validation

In creating a user agent for secure messaging, certificate, CRL, and certificate chain validation SHOULD be highly automated while still acting in the best interests of the user. Certificate, CRL, and chain validation MUST be performed when validating a correspondent's public key. This is necessary when a) verifying a signature from a correspondent and, b) creating a digital envelope with the correspondent as the intended recipient.

Certificates and CRLs are made available to the chain validation procedure in two ways: a) incoming messages, and b) certificate and CRL retrieval mechanisms. Certificates and CRLs in incoming messages are not required to be in any particular order nor are they required to be in any way related to the sender or recipient of the message (although in most cases they will be related to the sender). Incoming certificates and CRLs SHOULD be cached for use in chain validation and optionally stored for later use. This temporary certificate and CRL cache SHOULD be used to augment any other certificate and CRL retrieval mechanisms for chain validation on incoming signed messages.

4.3 Certificate and CRL Signing Algorithms

Certificates and Certificate-Revocation Lists (CRLs) are signed by the certificate issuer. A receiving agent MUST be capable of verifying the signatures on certificates andCRLs made with md5WithRSAEncryption and sha-1WithRSAEncryption signature algorithms with key sizes from 512 bits to 2048 bits described in [SMIME-MSG]. A receiving agent SHOULD be capable of verifying the signatures on certificates and CRLs made with the md2WithRSAEncryption signature algorithm with key sizes from 512 bits to 2048 bits.

4.4 X.509 Version 3 Certificate Extensions

The X.509 v3 standard describes an extensible framework in which the basic certificate information can be extended and how such extensions can be used to control the process of issuing and validating certificates. The PKIX Working Group has ongoing efforts to identify and create extensions which have value in particular certification environments. As such, there is still a fair amount of profiling work to be done before there is widespread agreement on which v3 extensions will be used. Further, there are active efforts underway to issue X.509 v3 certificates for business purposes. This memo identifies the minumum required set of certificate extensions which have the greatest value in the S/MIME environment. The basicConstraints, and keyUsage extensions are defined in [X.509].

Sending and receiving agents MUST correctly handle the v3 Basic Constraints Certificate Extension, the Key Usage Certificate Extension, authorityKeyID, subjectKeyID, and the subjectAltNames when they appear in end-user certificates. Some mechanism SHOULD exist to handle the defined v3 certificate extensions when they appear in intermediate or CA certificates.

Certificates issued for the S/MIME environment SHOULD NOT contain any critical extensions other than those listed here. These extensions SHOULD be marked as non-critical unless the proper handling of the extension is deemed critical to the correct interpretation of the associated certificate. Other extensions may be included, but those extensions SHOULD NOT be marked as critical.

4.4.1 Basic Constraints Certificate Extension

The basic constraints extension serves to delimit the role and position of an issuing authority or end-user certificate plays in a chain of certificates.

For example, certificates issued to CAs and subordinate CAs contain a basic constraint extension that identifies them as issuing authority certificates. End-user subscriber certificates contain an extension that constrains the certificate from being an issuing authority certificate.

Certificates SHOULD contain a basicContstraints extension.

4.4.2 Key Usage Certificate Extension

The key usage extension serves to limit the technical purposes for which a public key listed in a valid certificate may be used. Issuing authority certificates may contain a key usage extension that restricts the key to signing certificates, certificate revocation lists and other data.

For example, a certification authority may create subordinate issuer certificates which contain a keyUsage extension which specifies that the corresponding public key can be used to sign end user certs and sign CRLs.

5. Generating Keys and Certification Requests

5.1 Binding Names and Keys

An S/MIME agent or some related administrative utility or function MUST be capable of generating a certification request given a user's public key and associated name information. In most cases, the user's public key/private key pair will be generated simultaneously. However, there are cases where the keying information may be generated by an external process (such as when a key pair is generated on a cryptographic token or by a "key recovery" service).

There SHOULD NOT be multiple valid (that is, non-expired and non-revoked) certificates for the same key pair bound to different Distinguished Names. Otherwise, a security flaw exists where an attacker can substitute one valid certificate for another in such a way that can not be detected by a message recipient. If a users wishes to change their name (or create an alternate name), the user agent SHOULD generate a new key pair. If the user wishes to reuse an existing key pair with a new or alternate name, the user SHOULD first have any valid certificates for the existing public key revoked.

In general, it is possible for a user to request certification for the same name and different public key from the same or different certification authorities. This is acceptable both for end-entity and issuer certificates and can be useful in supporting a change of issuer keys in a smooth fashion.

CAs that re-use their own name with distinct keys MUST include the AuthorityKeyIdentifier extension in certificates that they issue, and MUST have the SubjectKeyIdentifier extension in their own certificate. CAs SHOULD use these extensions uniformly.

Clients SHOULD handle multiple valid CA certificates that certify different public keys but contain the same subject name (in this case, that CA's name).

When selecting an appropriate issuer's certificate to use to verify a given certificate, clients SHOULD process the AuthorityKeyIdentifier and SubjectKeyIdentifier extensions.

5.2 Using PKCS #10 for Certification Requests

PKCS #10 is a flexible and extensible message format for representing the results of cryptographic operations on some data. The choice of naming information is largely dictated by the policies and procedures associated with the intended certification service.

In addition to key and naming information, the PKCS #10 format supports the inclusion of optional attributes, signed by the entity requesting certification. This allows for information to be conveyed in a certification request which may be useful to the request process, but not necessarily part of the Distinguished Name being certified.

Receiving agents MUST support the identification of an RSA key with the rsa defined in X.509 and the rsaEncryption OID. Certification authorities MUST support sha-1WithRSAEncryption and md5WithRSAEncryption and SHOULD support MD2WithRSAEncryption for verification of signatures on certificate requests as described in [SMIME-MSG].

For the creation and submission of certification-requests, RSA keys SHOULD be identified with the rsaEncryption OID and signed with the sha-1WithRSAEncryption signature algorithm. Certification-requests MUST NOT be signed with the md2WithRSAEncryption signature algorithm.

Certification requests MUST include a valid Internet mail address, either as part of the certificate (as described in 3.2) or as part of the PKCS #10 attribute list. Certification authorities MUST check that the address in the "From:" header matches either of these addresses. CAs SHOULD allow the CA operator to configure processing of messages whose addresses do not match.

Certification authorities SHOULD support parsing of zero or one instance of each of the following set of certification-request attributes on incoming messages. Attributes that a particular implementation does not support may generate a warning message to the requestor, or may be silently ignored. Inclusion of the following attributes during the creation and submission of a certification-request will most likely be dictated by the policies associated with the certification service which will certify the corresponding name and public key.

postalAddress
challengePassword
unstructuredAddress

postalAddress is described in [X.520].

5.2.1 Challenge Password

The challenge-password attribute type specifies a password by which an entity may request certificate revocation. The interpretation of the password is intended to be specified by the issuer of the certificate; no particular interpretation is required. The challenge-password attribute type is intended for PKCS #10 certification requests.

Challenge-password attribute values have ASN.1 type ChallengePassword:

```
ChallengePassword ::= CHOICE {
  PrintableString, T61String }
```

A challenge-password attribute must have a single attribute value.

It is expected that if UCS becomes an ASN.1 type
(e.g., UNIVERSAL STRING),
ChallengePassword will become a CHOICE type:

```
ChallengePassword ::= CHOICE {
    PrintableString, T61String, UNIVERSAL STRING }
```

5.2.2 Unstructured Address

The unstructured-address attribute type specifies the address or addresses of the subject of a certificate as an unstructured ASCII or T.61 string. The interpretation of the addresses is intended to be specified by the issuer of the certificate; no particular interpretation is required. A likely interpretation is as an alternative to the X.520 postalAddress attribute type. The unstructured-address attribute type is intended for PKCS #10

certification requests.

Unstructured-address attribute values have
ASN.1 type UnstructuredAddress:

```
UnstructuredAddress ::= CHOICE {
  PrintableString, T61String }
```

An unstructured-address attribute can have multiple attribute values.

Note: T.61's newline character (hexadecimal code 0d) is recommended as a line separator in multi-line addresses.

It is expected that if UCS becomes an ASN.1 type (e.g., UNIVERSAL STRING), UnstructuredAddress will become a CHOICE type:

```
UnstructuredAddress ::= CHOICE {
    PrintableString, T61String, UNIVERSAL STRING }
```

5.3 Fulfilling a Certification Request

Certification authorities SHOULD use the sha-1WithRSAEncryption signature algorithms when signing certificates.

5.4 Using PKCS #7 for Fulfilled Certificate Response

[PKCS-7] supports a degenerate case of the SignedData content type where there are no signers on the content (and hence, the content value is "irrelevant"). This degenerate case is used to convey certificate and CRL information. Certification authorities MUST use this format for returning certificate information resulting from the successful fulfillment of a certification request. At a minimum, the fulfilled certificate response MUST include the actual subject certificate (corresponding to the information in the certification request). The response SHOULD include other certificates which link the issuer to higher level certification authorities and corresponding certificate-revocation lists. Unrelated certificates and revocation information is also acceptable.

Receiving agents MUST parse this degenerate PKCS #7 message type and handle the certificates and CRLs according to the requirements and recommendations in Section 4.

6. Security Considerations

All of the security issues faced by any cryptographic application must be faced by a S/MIME agent. Among these issues are protecting the user's private key, preventing various attacks, and helping the user avoid mistakes such as inadvertently encrypting a message for the wrong recipient. The entire list of security considerations is beyond the scope of this document, but some significant concerns are listed here.

When processing certificates, there are many situations where the processing might fail. Because the processing may be done by a user agent, a security gateway, or other program, there is no single way to handle such failures. Just because the methods to handle the failures has not been listed, however, the reader should not assume that they are not important. The opposite is true: if a certificate is not provably valid and associated with the message, the processing software should take immediate and noticable steps to inform the end user about it.

Some of the many places where signature and certificate checking might fail include:

- no Internet mail addresses in a certificate match the sender of a message
- no certificate chain leads to a trusted CA
- no ability to check the CRL for a certificate
- an invalid CRL was received
- the CRL being checked is expired
- the certificate is expired
- the certificate has been revoked

There are certainly other instances where a certificate may be invalid, and it is the responsibility of the processing software to check them all thoroughly, and to decide what to do if the check fails.

A. Object Identifiers and Syntax

Sections A.1 through A.4 are adopted from [SMIME-MSG].

A.5 Name Attributes

```
emailAddress OBJECT IDENTIFIER ::=

     {iso(1) member-body(2) US(840) rsadsi(113549) pkcs(1) pkcs-9(9) 1}

CountryName OBJECT IDENTIFIER ::=
     {joint-iso-ccitt(2) ds(5) attributeType(4) 6}

StateOrProvinceName OBJECT IDENTIFIER ::=
     {joint-iso-ccitt(2) ds(5) attributeType(4) 8}

locality OBJECT IDENTIFIER ::=
     {joint-iso-ccitt(2) ds(5) attributeType(4) 7}

CommonName OBJECT IDENTIFIER ::=
     {joint-iso-ccitt(2) ds(5) attributeType(4) 3}

Title OBJECT IDENTIFIER ::=
     {joint-iso-ccitt(2) ds(5) attributeType(4) 12}

Organization OBJECT IDENTIFIER ::=
     {joint-iso-ccitt(2) ds(5) attributeType(4) 10}

OrganizationalUnit OBJECT IDENTIFIER ::=
     {joint-iso-ccitt(2) ds(5) attributeType(4) 11}

StreetAddress OBJECT IDENTIFIER ::=
     {joint-iso-ccitt(2) ds(5) attributeType(4) 9}

Postal Code OBJECT IDENTIFIER ::=
     {joint-iso-ccitt(2) ds(5) attributeType(4) 17}

Phone Number OBJECT IDENTIFIER ::=
     {joint-iso-ccitt(2) ds(5) attributeType(4) 20}
```

A.6 Certification Request Attributes

```
postalAddress OBJECT IDENTIFIER ::=
     {joint-iso-ccitt(2) ds(5) attributeType(4) 16}

challengePassword OBJECT IDENTIFIER ::=
     {iso(1) member-body(2) US(840) rsadsi(113549) pkcs(1) pkcs-9(9) 7}
```

```
unstructuredAddress OBJECT IDENTIFIER ::=
     {iso(1) member-body(2) US(840) rsadsi(113549) pkcs(1) pkcs-9(9) 8}
```

A.7 X.509 V3 Certificate Extensions

```
basicConstraints OBJECT IDENTIFIER ::=

     {joint-iso-ccitt(2) ds(5) 29 19 }
```

The ASN.1 definition of basicConstraints certificate extension is:

```
basicConstraints basicConstraints EXTENSION ::= {
     SYNTAX  BasicConstraintsSyntax
     IDENTIFIED BY { id-ce 19 } }

BasicConstraintsSyntax ::= SEQUENCE {
     cA                 BOOLEAN DEFAULT FALSE,
     pathLenConstraint  INTEGER (0..MAX) OPTIONAL }

keyUsage OBJECT IDENTIFIER ::=
     {joint-iso-ccitt(2) ds(5) 29 15 }
```

The ASN.1 definition of keyUsage certificate extension is:

```
keyUsage EXTENSION ::= {
     SYNTAX  KeyUsage
     IDENTIFIED BY { id-ce 15 }}

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

B. References

[KEYM] PKIX Part 1. At the time of this writing, PKIX is a Work in Progress, but it is expected that there will be standards-track RFCs at some point in the future.

[MUSTSHOULD] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 1l4, RFC 2119, March 1997.

[PKCS-1] Kaliski, B., "PKCS #1: RSA Encryption Version 1.5", RFC 2313, March 1998.

[PKCS-7] Kaliski, B., "PKCS #7: Cryptographic Message Syntax Version 1.5", RFC 2315, March 1998.

[PKCS-10] Kaliski, B., "PKCS #10: Certification Request Syntax Version 1.5", RFC 2314, March 1998.

[RFC-822] Crocker, D., "Standard For The Format Of ARPA Internet Text Messages", STD 11, RFC 822, August 1982.

[SMIME-MSG] Dusse, S., Hoffman, P., Ramsdell, R., Lundblade, L., and L. Repka, "S/MIME Version 2 Message Specification", RFC 2311, March 1998.

[X.500] ITU-T Recommendation X.500 (1997) | ISO/IEC 9594-1:1997, Information technology - Open Systems Interconnection - The Directory: Overview of concepts, models and services

[X.501] ITU-T Recommendation X.501 (1997) | ISO/IEC 9594-2:1997, Information technology - Open Systems Interconnection - The Directory: Models

[X.509] ITU-T Recommendation X.509 (1997) | ISO/IEC 9594-8:1997, Information technology - Open Systems Interconnection - The Directory: Authentication framework

[X.520] ITU-T Recommendation X.520 (1997) | ISO/IEC 9594-6:1997, Information technology - Open Systems Interconnection - The Directory: Selected attribute types.

C. Compatibility with Prior Practice in S/MIME

S/MIME was originally developed by RSA Data Security, Inc. Many developers implemented S/MIME agents before this document was published. All S/MIME receiving agents SHOULD make every attempt to interoperate with these earlier implementations of S/MIME.

D. Acknowledgements

Significant contributions to the content of this memo were made by many people, including David Solo, Anil Gangolli, Jeff Thompson, and Lisa Repka.

E. Authors' Addresses

Steve Dusse
RSA Data Security, Inc.
100 Marine Parkway, #500
Redwood City, CA 94065 USA

Phone: (415) 595-8782
EMail: spock@rsa.com

Paul Hoffman
Internet Mail Consortium
127 Segre Place
Santa Cruz, CA 95060

Phone: (408) 426-9827
EMail: phoffman@imc.org

Blake Ramsdell
Worldtalk
13122 NE 20th St., Suite C
Bellevue, WA 98005

Phone: (425) 882-8861
EMail: blaker@deming.com

Jeff Weinstein
Netscape Communications Corporation
501 East Middlefield Road
Mountain View, CA 94043

Phone: (415) 254-1900
EMail: jsw@netscape.com

F. Full Copyright Statement