# Privacy Enhancement for Internet Electronic Mail:
# Part I: Message Encryption and Authentication Procedures

John Linn
February 1993

## Status of this Memo

This RFC specifies an IAB standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

## Acknowledgements

This document is the outgrowth of a series of meetings of the Privacy and Security Research Group (PSRG) of the IRTF and the PEM Working Group of the IETF. I would like to thank the members of the PSRG and the IETF PEM WG, as well as all participants in discussions on the "pem-dev@tis.com" mailing list, for their contributions to this document.

## 1. Executive Summary

This document defines message encryption and authentication procedures, in order to provide privacy-enhanced mail (PEM) services for electronic mail transfer in the Internet. It is intended to become one member of a related set of four RFCs. The procedures defined in the current document are intended to be compatible with a wide range of key management approaches, including both symmetric (secret-key) and asymmetric (public-key) approaches for encryption of data encrypting keys. Use of symmetric cryptography for message text encryption and/or integrity check computation is anticipated. RFC 1422 specifies supporting key management mechanisms based on the use of public-key certificates. RFC 1423 specifies algorithms, modes, and associated identifiers relevant to the current RFC and to RFC 1422. RFC 1424 provides details of paper and electronic formats and procedures for the key management infrastructure being established in support of these services.

Privacy enhancement services (confidentiality, authentication, message integrity assurance, and non-repudiation of origin) are offered through the use of end-to-end cryptography between originator and recipient processes at or above the User Agent level. No special processing requirements are imposed on the Message Transfer System at endpoints or at intermediate relay sites. This approach allows privacy enhancement facilities to be incorporated selectively on a site-by-site or user-by-user basis without impact on other Internet entities. Interoperability among heterogeneous components and mail transport facilities is supported.

The current specification's scope is confined to PEM processing procedures for the RFC-822 textual mail environment, and defines the Content-Domain indicator value "RFC822" to signify this usage. Follow-on work in integration of PEM capabilities with other messaging environments (e.g., MIME) is anticipated and will be addressed in separate and/or successor documents, at which point additional Content-Domain indicator values will be defined.

## 2. Terminology

For descriptive purposes, this RFC uses some terms defined in the OSI X.400 Message Handling System Model per the CCITT Recommendations. This section replicates a portion of (1984) X.400's Section 2.2.1, "Description of the MHS Model: Overview" in order to make the terminology clear to readers who may not be familiar with the OSI MHS Model.

In the [MHS] model, a user is a person or a computer application. A user is referred to as either an originator (when sending a message) or a recipient (when receiving one). MH Service elements define the set of message types and the capabilities that enable an originator to transfer messages of those types to one or more recipients.

An originator prepares messages with the assistance of his or her User Agent (UA). A UA is an application process that interacts with the Message Transfer System (MTS) to submit messages. The MTS delivers to one or more recipient UAs the messages submitted to it. Functions performed solely by the UA and not standardized as part of the MH Service elements are called local UA functions.

The MTS is composed of a number of Message Transfer Agents (MTAs). Operating together, the MTAs relay messages and deliver them to the intended recipient UAs, which then make the messages available to the intended recipients.

The collection of UAs and MTAs is called the Message Handling System (MHS). The MHS and all of its users are collectively referred to as the Message Handling Environment.

## 3. Services, Constraints, and Implications

This RFC defines mechanisms to enhance privacy for electronic mail transferred in the Internet. The facilities discussed in this RFC provide privacy enhancement services on an end-to-end basis between originator and recipient processes residing at the UA level or above. No privacy enhancements are offered for message fields which are added or transformed by intermediate relay points between PEM processing components.

If an originator elects to perform PEM processing on an outbound message, all PEM-provided security services are applied to the PEM message's body in its entirety; selective application to portions of a PEM message is not supported. Authentication, integrity, and (when asymmetric key management is employed) non-repudiation of origin services are applied to all PEM messages; confidentiality services are optionally selectable.

In keeping with the Internet's heterogeneous constituencies and usage modes, the measures defined here are applicable to a broad range of Internet hosts and usage paradigms. In particular, it is worth noting the following attributes:

1. The mechanisms defined in this RFC are not restricted to a particular host or operating system, but rather allow interoperability among a broad range of systems. All privacy enhancements are implemented at the application layer, and are not dependent on any privacy features at lower protocol layers.

2. The defined mechanisms are compatible with non-enhanced Internet components. Privacy enhancements are implemented in an end-to-end fashion which does not impact mail processing by intermediate relay hosts which do not incorporate privacy enhancement facilities. It is necessary, however, for a message's originator to be cognizant of whether a message's intended recipient implements privacy enhancements, in order that encoding and possible encryption will not be performed on a message whose destination is not equipped to perform corresponding inverse transformations. (Section 4.6.1.1.3 of this RFC describes a PEM message type ("MIC-CLEAR") which represents a signed, unencrypted PEM message in a form readable without PEM processing capabilities yet validatable by PEM-equipped recipients.)

3. The defined mechanisms are compatible with a range of mail transport facilities (MTAs). Within the Internet, electronic mail transport is effected by a variety of SMTP [2] implementations. Certain sites, accessible via SMTP, forward mail into other mail processing environments (e.g., USENET, CSNET, BITNET). The privacy enhancements must be able to operate across the SMTP realm; it is desirable that they also be compatible with protection of electronic mail sent between the SMTP environment and other connected environments.

4. The defined mechanisms are compatible with a broad range of electronic mail user agents (UAs). A large variety of electronic mail user agent programs, with a corresponding broad range of user interface paradigms, is used in the Internet. In order that electronic mail privacy enhancements be available to the broadest possible user community, selected mechanisms should be usable with the widest possible variety of existing UA programs. For purposes of pilot implementation, it is desirable that privacy enhancement processing be incorporable into a separate program, applicable to a range of UAs, rather than requiring internal modifications to each UA with which PEM services are to be provided.

5. The defined mechanisms allow electronic mail privacy enhancement processing to be performed on personal computers (PCs) separate from the systems

on which UA functions are implemented. Given the expanding use of PCs and the limited degree of trust which can be placed in UA implementations on many multi-user systems, this attribute can allow many users to process PEM with a higher assurance level than a strictly UA-integrated approach would allow.

6. The defined mechanisms support privacy protection of electronic mail addressed to mailing lists (distribution lists, in ISO parlance).

7. The mechanisms defined within this RFC are compatible with a variety of supporting key management approaches, including (but not limited to) manual pre-distribution, centralized key distribution based on symmetric cryptography, and the use of public-key certificates per RFC 1422. Different key management mechanisms may be used for different recipients of a multicast message. For two PEM implementations to interoperate, they must share a common key management mechanism; support for the mechanism defined in RFC 1422 is strongly encouraged.

In order to achieve applicability to the broadest possible range of Internet hosts and mail systems, and to facilitate pilot implementation and testing without the need for prior and pervasive modifications throughout the Internet, the following design principles were applied in selecting the set of features specified in this RFC:

1. This RFC's measures are restricted to implementation at endpoints and are amenable to integration with existing Internet mail protocols at the user agent (UA) level or above, rather than necessitating modifications to existing mail protocols or integration into the message transport system (e.g., SMTP servers).

2. The set of supported measures enhances rather than restricts user capabilities. Trusted implementations, incorporating integrity features protecting software from subversion by local users, cannot be assumed in general. No mechanisms are assumed to prevent users from sending, at their discretion, messages to which no PEM processing has been applied. In the absence of such features, it appears more feasible to provide facilities which enhance user services (e.g., by protecting and authenticating inter-user traffic) than to enforce restrictions (e.g., inter-user access control) on user actions.

3. The set of supported measures focuses on a set of functional capabilities selected to provide significant and tangible benefits to a broad user community. By concentrating on the most critical set of services, we aim to maximize the added privacy value that can be provided with a modest level of implementation effort.

Based on these principles, the following facilities are provided:

1. disclosure protection,

2. originator authenticity,

3. message integrity measures, and
4. (if asymmetric key management is used) non-repudiation of origin,

but the following privacy-relevant concerns are not addressed:

1. access control,
2. traffic flow confidentiality,
3. address list accuracy,
4. routing control,
5. issues relating to the casual serial reuse of PCs by multiple users,
6. assurance of message receipt and non-deniability of receipt,
7. automatic association of acknowledgments with the messages to which they refer, and
8. message duplicate detection, replay prevention, or other stream-oriented services

## 4. Processing of Messages

### 4.1 Message Processing Overview

This subsection provides a high-level overview of the components and processing steps involved in electronic mail privacy enhancement processing. Subsequent subsections will define the procedures in more detail.

#### 4.1.1 Types of Keys

A two-level keying hierarchy is used to support PEM transmission:

1. *Data Encrypting Keys (DEKs)* are used for encryption of message text and (with certain choices among a set of alternative algorithms) for computation of message integrity check (MIC) quantities. In the asymmetric key management environment, DEKs are also used to encrypt the signed representations of MICs in PEM messages to which confidentiality has been applied. DEKs are generated individually for each transmitted message; no predistribution of DEKs is needed to support PEM transmission.
2. *Interchange Keys (IKs)* are used to encrypt DEKs for transmission within messages. Ordinarily, the same IK will be used for all messages sent from a given originator to a given recipient over a period of time. Each transmitted message includes a representation of the DEK(s) used for message encryption and/or MIC computation, encrypted under an individual IK per

named recipient. The representation is associated with Originator-ID and Recipient-ID fields (defined in different forms so as to distinguish symmetric from asymmetric cases), which allow each individual recipient to identify the IK used to encrypt DEKs and/or MICs for that recipient's use. Given an appropriate IK, a recipient can decrypt the corresponding transmitted DEK representation, yielding the DEK required for message text decryption and/or MIC validation. The definition of an IK differs depending on whether symmetric or asymmetric cryptography is used for DEK encryption:

2a. When symmetric cryptography is used for DEK encryption, an IK is a single symmetric key shared between an originator and a recipient. In this case, the same IK is used to encrypt MICs as well as DEKs for transmission. Version/expiration information and IA identification associated with the originator and with the recipient must be concatenated in order to fully qualify a symmetric IK.

2b. When asymmetric cryptography is used, the IK component used for DEK encryption is the public component [8] of the recipient. The IK component used for MIC encryption is the private component of the originator, and therefore only one encrypted MIC representation need be included per message, rather than one per recipient. Each of these IK components can be fully qualified in a Recipient-ID or Originator-ID field, respectively. Alternatively, an originator's IK component may be determined from a certificate carried in an "Originator-Certificate:" field.

### 4.1.2 Processing Procedures

When PEM processing is to be performed on an outgoing message, a DEK is generated [1] for use in message encryption and (if a chosen MIC algorithm requires a key) a variant of the DEK is formed for use in MIC computation. DEK generation can be omitted for the case of a message where confidentiality is not to be applied, unless a chosen MIC computation algorithm requires a DEK. Other parameters (e.g., Initialization Vectors (IVs)) as required by selected encryption algorithms are also generated.

One or more Originator-ID and/or "Originator-Certificate:" fields are included in a PEM message's encapsulated header to provide recipients with an identification component for the IK(s) used for message processing. All of a message's Originator-ID and/or "Originator- Certificate:" fields are assumed to correspond to the same principal; the facility for inclusion of multiple such fields accomodates the prospect that different keys, algorithms, and/or certification paths may be required for processing by different recipients. When a message includes recipients for which asymmetric key management is employed as well as recipients for which symmetric key management is employed, a separate Originator-ID or "Originator-Certificate:" field precedes each set of recipients.

In the symmetric case, per-recipient IK components are applied for each individually named recipient in preparation of ENCRYPTED, MIC- ONLY, and MIC-CLEAR messages. A corresponding "Recipient-ID- Symmetric:" field, interpreted in the context of the most recent preceding "Originator-ID-Symmetric:" field, serves to identify each IK. In the asymmetric case, per-recipient IK components are applied only for ENCRYPTED messages, are independent of originator-oriented header elements, and are identified by "Recipient-ID-Asymmetric:" fields. Each Recipient-ID field is followed by a "Key-Info:" field, which transfers the message's DEK encrypted under the IK appropriate for the specified recipient.

When symmetric key management is used for a given recipient, the "Key-Info:" field following the corresponding "Recipient-ID- Symmetric:" field also transfers the message's computed MIC, encrypted under the recipient's IK. When asymmetric key management is used, a "MIC-Info:" field associated with an "Originator-ID-Asymmetric:" or "Originator-Certificate:" field carries the message's MIC, asymmetrically signed using the private component of the originator. If the PEM message is of type ENCRYPTED (as defined in Section 4.6.1.1.1 of this RFC), the asymmetrically signed MIC is symmetrically encrypted using the same DEK, algorithm, encryption mode and other cryptographic parameters as used to encrypt the message text, prior to inclusion in the "MIC-Info:" field.

#### 4.1.2.1 Processing Steps

A four-phase transformation procedure is employed in order to represent encrypted message text in a universally transmissible form and to enable messages encrypted on one type of host computer to be decrypted on a different type of host computer. A plaintext message is accepted in local form, using the host's native character set and line representation. The local form is converted to a canonical message text representation, defined as equivalent to the inter-SMTP representation of message text. This canonical representation forms the input to the MIC computation step (applicable to ENCRYPTED, MIC- ONLY, and MIC-CLEAR messages) and the encryption process (applicable to ENCRYPTED messages only).

For ENCRYPTED PEM messages, the canonical representation is padded as required by the encryption algorithm, and this padded canonical representation is encrypted. The encrypted text (for an ENCRYPTED message) or the unpadded canonical form (for a MIC-ONLY message) is then encoded into a printable form. The printable form is composed of a restricted character set which is chosen to be universally representable across sites, and which will not be disrupted by processing within and between MTS entities. MIC-CLEAR PEM messages omit the printable encoding step.

The output of the previous processing steps is combined with a set of header fields carrying cryptographic control information. The resulting PEM message is passed to the electronic mail system to be included within the text portion of a transmitted message. There is no requirement that a PEM message comprise the entirety of an MTS message's text portion; this allows PEM-protected information to be accompanied by (unprotected) annotations. It is also permissible for multiple PEM messages (and associated unprotected text, outside the PEM message boundaries) to be represented within the encapsulated text of a higher-level PEM message. PEM message signatures are forwardable when asymmetric key management is employed; an authorized recipient of a PEM message with confidentiality applied can reduce that message to a signed but unencrypted form for forwarding purposes or can re-encrypt that message for subsequent transmission.

When a PEM message is received, the cryptographic control fields within its encapsulated header provide the information required for each authorized recipient to perform MIC validation and decryption of the received message text. For ENCRYPTED and MIC-ONLY messages, the printable encoding is converted to a bitstring. Encrypted portions of the transmitted message are decrypted. The MIC is validated. Then, the recipient PEM process converts the canonical representation to its appropriate local form.

#### 4.1.2.2 Error Cases

A variety of error cases may occur and be detected in the course of processing a received PEM message. The specific actions to be taken in response to such conditions are local matters, varying as functions of user preferences and the type of user interface provided by a particular PEM implementation, but certain general recommendations are appropriate. Syntactically invalid PEM messages should be flagged as such, preferably with collection of diagnostic information to support debugging of incompatibilities or other failures. RFC 1422 defines specific error processing requirements relevant to the certificate-based key management mechanisms defined therein.

Syntactically valid PEM messages which yield MIC failures raise special concern, as they may result from attempted attacks or forged messages. As such, it is unsuitable to display their contents to recipient users without first indicating the fact that the contents' authenticity and integrity cannot be guaranteed and then receiving positive user confirmation of such a warning. MIC-CLEAR messages (discussed in Section 4.6.1.1.3 of this RFC) raise special concerns, as MIC failures on such messages may occur for a broader range of benign causes than are applicable to other PEM message types.

### 4.2 Encryption Algorithms, Modes, and Parameters

For use in conjunction with this RFC, RFC 1423 defines the appropriate algorithms, modes, and associated identifiers to be used for encryption of message text with DEKs.

The mechanisms defined in this RFC incorporate facilities for transmission of cryptographic parameters (e.g., pseudorandom Initializing Vectors (IVs)) with PEM messages to which the confidentiality service is applied, when required by symmetric message encryption algorithms and modes specified in RFC 1423.

Certain operations require encryption of DEKs, MICs, and digital signatures under an IK for purposes of transmission. A header facility indicates the mode in which the IK is used for encryption. RFC 1423 specifies encryption algorithm and mode identifiers and minimum essential support requirements for key encryption processing.

RFC 1422 specifies asymmetric, certificate-based key management procedures based on CCITT Recommendation X.509 to support the message processing procedures defined in this document. Support for the key management approach defined in RFC 1422 is strongly recommended. The message processing procedures can also be used with symmetric key management, given prior distribution of suitable symmetric IKs, but no current RFCs specify key distribution procedures for such IKs.

## 4.3 Privacy Enhancement Message Transformations

### 4.3.1 Constraints

An electronic mail encryption mechanism must be compatible with the transparency constraints of its underlying electronic mail facilities. These constraints are generally established based on expected user requirements and on the characteristics of anticipated endpoint and transport facilities. An encryption mechanism must also be compatible with the local conventions of the computer systems which it interconnects. Our approach uses a canonicalization step to abstract out local conventions and a subsequent encoding step to conform to the characteristics of the underlying mail transport medium (SMTP). The encoding conforms to SMTP constraints. Section 4.5 of RFC 821 [2] details SMTP's transparency constraints.

To prepare a message for SMTP transmission, the following requirements must be met:

1. All characters must be members of the 7-bit ASCII character set.
2. Text lines, delimited by the character pair $< CR >< LF >$, must be no more than 1000 characters long.
3. Since the string $< CR >< LF > . < CR >< LF >$ indicates the end of a message, it must not occur in text prior to the end of a message.

Although SMTP specifies a standard representation for line delimiters (ASCII $< CR >< LF >$ ), numerous systems in the Internet use a different native representation to delimit lines. For example, the $< CR >< LF >$ sequences delimiting lines in mail inbound to UNIX systems are transformed to single $< LF > s$ as mail is written into local mailbox files. Lines in mail incoming to record-oriented systems (such as VAX VMS) may be converted to appropriate records by the destination SMTP server [3]. As a result, if the encryption process generated $< CR > s$ or $< LF > s,$ those characters might not be accessible to a recipient UA program at a destination which uses different line delimiting conventions. It is also possible that conversion between tabs and spaces may be performed in the course of mapping between inter-SMTP and local format; this is a matter of local option. If such transformations changed the form of transmitted ciphertext, decryption would fail to regenerate the transmitted plaintext, and a transmitted MIC would fail to compare with that computed at the destination.

The conversion performed by an SMTP server at a system with EBCDIC as a native character set has even more severe impact, since the conversion from EBCDIC into ASCII is an information-losing transformation. In principle, the transformation function mapping between inter-SMTP canonical ASCII message representation and local format could be moved from the SMTP server up to the UA, given a means to direct that the SMTP server should no longer perform that transformation. This approach has a major disadvantage: internal file (e.g., mailbox) formats would be incompatible with the native forms used on the systems where they reside. Further, it would require modification to SMTP servers, as mail would be passed to SMTP in a different representation than it is passed at present.

### 4.3.2 Approach

Our approach to supporting PEM across an environment in which intermediate conversions may occur defines an encoding for mail which is uniformly representable across the set of PEM UAs regardless of their systems' native character sets. This encoded form is used (for specified PEM message types) to represent mail text in transit from originator to recipient, but the encoding is not applied to enclosing MTS headers or to encapsulated headers inserted to carry control information between PEM UAs. The encoding's characteristics are such that the transformations anticipated between originator and recipient UAs will not prevent an encoded message from being decoded properly at its destination.

Four transformation steps, described in the following four subsections, apply to outbound PEM message processing:

#### 4.3.2.1 Step 1: Local Form

This step is applicable to PEM message types ENCRYPTED, MIC-ONLY, and MIC-CLEAR. The message text is created in the system's native character set, with lines delimited in accordance with local convention.

#### 4.3.2.2 Step 2: Canonical Form

This step is applicable to PEM message types ENCRYPTED, MIC-ONLY, and MIC-CLEAR. The message text is converted to a universal canonical form, similar to the inter-SMTP representation [4] as defined in RFC 821 [2] and RFC 822 [5]. The procedures performed in order to accomplish this conversion are dependent on the characteristics of the local form and so are not specified in this RFC.

PEM canonicalization assures that the message text is represented with the ASCII character set and "$<CR><LF>$" line delimiters, but does not perform the dot-stuffing transformation discussed in RFC 821, Section 4.5.2. Since a message is converted to a standard character set and representation before encryption, a transferred PEM message can be decrypted and its MIC can be validated at any type of destination host computer. Decryption and MIC validation is performed before any conversions which may be necessary to transform the message into a destination-specific local form.

#### 4.3.2.3 Step 3: Authentication and Encryption

Authentication processing is applicable to PEM message types ENCRYPTED, MIC-ONLY, and MIC-CLEAR. The canonical form is input to the selected MIC computation algorithm in order to compute an integrity check quantity for the message. No padding is added to the canonical form before submission to the MIC computation algorithm, although certain MIC algorithms will apply their own padding in the course of computing a MIC.

Encryption processing is applicable only to PEM message type ENCRYPTED. RFC 1423 defines the padding technique used to support encryption of the canonically-encoded message text.

#### 4.3.2.4 Step 4: Printable Encoding

This printable encoding step is applicable to PEM message types ENCRYPTED and MIC-ONLY. The same processing is also employed in representation of certain specifically identified PEM encapsulated header field quantities as cited in Section 4.6. Proceeding from left to right, the bit string resulting from step 3 is encoded into characters which are universally representable at all sites, though not necessarily with the same bit patterns (e.g., although the character "E" is represented in an ASCII-based system as hexadecimal 45 and as hexadecimal C5 in an EBCDIC-based system, the local significance of the two representations is equivalent).

A 64-character subset of International Alphabet IA5 is used, enabling 6 bits to be represented per printable character. (The proposed subset of characters is represented identically in IA5 and ASCII.) The character "=" signifies a special processing function used for padding within the printable encoding procedure.

To represent the encapsulated text of a PEM message, the encoding function's output is delimited into text lines (using local conventions), with each line except the last containing exactly 64 printable characters and the final line containing 64 or fewer printable characters. (This line length is easily printable and is guaranteed to satisfy SMTP's 1000-character transmitted line length limit.) This folding requirement does not apply when the encoding procedure is used to represent PEM header field quantities; Section 4.6 discusses folding of PEM encapsulated header fields.

The encoding process represents 24-bit groups of input bits as output strings of 4 encoded characters. Proceeding from left to right across a 24-bit input group extracted from the output of step 3, each 6-bit group is used as an index into an array of 64 printable characters. The character referenced by the index is placed in the output string. These characters, identified in Table 1, are selected so as to be universally representable, and the set excludes characters with particular significance to SMTP (e.g., ".", "$<CR>$", "$<LF>$" ).

Special processing is performed if fewer than 24 bits are available in an input group at the end of a message. A full encoding quantum is always completed at the end of a message. When fewer than 24 input bits are available in an input group, zero bits are added (on the right) to form an integral number of 6-bit groups. Output character positions which are not required to represent actual input data are set to the character "=". Since all canonically encoded output is an integral number of octets, only the following cases can arise: (1) the final quantum of encoding input is an integral multiple of 24 bits; here, the final unit of encoded output will be an integral multiple of 4 characters with no "=" padding, (2) the final quantum of encoding input is exactly 8 bits; here, the final unit of encoded output will be two characters followed by two "=" padding characters, or (3) the final quantum of encoding input is exactly 16 bits; here, the final unit of encoded output will be three characters followed by one "=" padding character.

**Table 1: Printable Encoding Characters**

| Value | Encoding | Value | Encoding | Value | Encoding | Value | Encoding |
|---|---|---|---|---|---|---|---|
| 0 | A | 17 | R | 34 | i | 51 | z |
| 1 | B | 18 | S | 35 | j | 52 | 0 |
| 2 | C | 19 | T | 36 | k | 53 | 1 |
| 3 | D | 20 | U | 37 | l | 54 | 2 |
| 4 | E | 21 | V | 38 | m | 55 | 3 |
| 5 | F | 22 | W | 39 | n | 56 | 4 |
| 6 | G | 23 | X | 40 | o | 57 | 5 |
| 7 | H | 24 | Y | 41 | p | 58 | 6 |
| 8 | I | 25 | Z | 42 | q | 59 | 7 |
| 9 | J | 26 | a | 43 | r | 60 | 8 |
| 10 | K | 27 | b | 44 | s | 61 | 9 |
| 11 | L | 28 | c | 45 | t | 62 | + |
| 12 | M | 29 | d | 46 | u | 63 | / |
| 13 | N | 30 | e | 47 | v | | |
| 14 | O | 31 | f | 48 | w | (pad) | = |
| 15 | P | 32 | g | 49 | x | | |
| 16 | Q | 33 | h | 50 | y | | |

#### 4.3.2.5 Summary of Transformations

In summary, the outbound message is subjected to the following composition of transformations (or, for some PEM message types, a subset thereof):

$$Transmit\_Form = Encode(Encrypt(Canonicalize(Local\_Form)))$$

The inverse transformations are performed, in reverse order, to process inbound PEM messages:

$$Local\_Form = DeCanonicalize(Decipher(Decode(Transmit\_Form)))$$

Note that the local form and the functions to transform messages to and from canonical form may vary between the originator and recipient systems without loss of information.

### 4.4 Encapsulation Mechanism

The encapsulation techniques defined in RFC-934 [6] are adopted for encapsulation of PEM messages within separate enclosing MTS messages carrying associated MTS headers. This approach offers a number of advantages relative to a flat approach in which certain fields within a single header are encrypted and/or carry cryptographic control information. As far as the MTS is concerned, the entirety of a PEM message will reside in an MTS message's text portion, not the MTS message's header portion. Encapsulation provides generality and segregates fields with user-to-user significance from those transformed in transit. All fields inserted in the course of encryption/authentication processing are placed in the encapsulated header. This facilitates compatibility with mail handling programs which accept only text, not header fields, from input files or from other programs.

The encapsulation techniques defined in RFC-934 are consistent with existing Internet mail forwarding and bursting mechanisms. These techniques are designed so that they may be used in a nested manner. The encapsulation techniques may be used to encapsulate one or more PEM messages for forwarding to a third party, possibly in conjunction with interspersed (non-PEM) text which serves to annotate the PEM messages.

Two encapsulation boundaries (EB's) are defined for delimiting encapsulated PEM messages and for distinguishing encapsulated PEM messages from interspersed (non-PEM) text. The pre-EB is the string "—–BEGIN PRIVACY-ENHANCED MESSAGE—–", indicating that an encapsulated PEM message follows. The post-EB is either (1) another pre-EB indicating that another encapsulated PEM message follows, or (2) the string "—–END PRIVACY-ENHANCED MESSAGE—–" indicating that any text that immediately follows is non-PEM text. A special point must be noted for the case of MIC-CLEAR messages, the text portions of which may contain lines which begin with the "-" character and which are therefore subject to special processing per RFC-934 forwarding procedures. When the string "- " must be prepended to such a line in the course of a forwarding operation in order to distinguish that line from an encapsulation boundary, MIC computation is to be performed prior to prepending the "- " string. Figure 1 depicts the encapsulation of a single PEM message.

This RFC places no a priori limits on the depth to which such encapsulation may be nested nor on the number of PEM messages which may be grouped in this fashion at a single nesting level for forwarding. A implementation compliant with this RFC must not preclude a user from submitting or receiving PEM messages which exploit this encapsulation capability. However, no specific requirements are levied upon implementations with regard to how this capability is made available to the user. Thus, for example, a compliant PEM implementation is not required to automatically detect and process encapsulated PEM messages.

In using this encapsulation facility, it is important to note that it is inappropriate to forward directly to a third party a message that is ENCRYPTED because recipients of such a message would not have access to the DEK required to decrypt the message. Instead, the user forwarding the message must transform the ENCRYPTED message into a MIC-ONLY or MIC-CLEAR form prior to forwarding. Thus, in order to comply with this RFC, a PEM implementation must provide a facility to enable a user to perform this transformation, while preserving the MIC associated with the original message.

If a user wishes PEM-provided confidentiality protection for transmitted information, such information must occur in the encapsulated text of an ENCRYPTED PEM message, not in the enclosing MTS header or PEM encapsulated header. If a user wishes to avoid disclosing the actual subject of a message to unintended parties, it is suggested that the enclosing MTS header contain a "Subject:" field indicating that "Encrypted Mail Follows".

**Figure 1: Encapsulated Message Format**

```
Encapsulated Message

    Pre-Encapsulation Boundary (Pre-EB)
    ----- BEGIN PRIVACY-ENHANCED MESSAGE -----

    Encapsulated Header Portion
        (Contains encryption control fields inserted in plaintext.
        Examples include "DEK-Info:" and "Key-Info:".
        Note that, although these control fields have line-oriented
        representations similar to RFC-822 header fields, the set
        of fields valid in this context is disjoint from those used
        in RFC-822 processing.)

    Blank Line [ < CR >< LF > ]
        (Separates Encapsulated Header from subsequent encoded
        Encapsulated Text Portion)

    Encapsulated Text Portion
        (Contains message data encoded as specified in Section 4.3;

    Post-Encapsulation Boundary (Post-EB)
    ----- END PRIVACY-ENHANCED MESSAGE -----
```

If an integrity-protected representation of information which occurs within an enclosing header (not necessarily in the same format as that in which it occurs within that header) is desired, that data can be included within the encapsulated text portion in addition to its inclusion in the enclosing MTS header. For example, an originator wishing to provide recipients with a protected indication of a message's position in a series of messages could include within the encapsulated text a copy of a timestamp or message counter value possessing end-to-end significance and extracted from an enclosing MTS header field. (Note: mailbox specifiers as entered by end users incorporate local conventions and are subject to modification at intermediaries, so inclusion of such specifiers within encapsulated text should not be regarded as a suitable alternative to the authentication semantics defined in RFC 1422 and based on X.500 Distinguished Names.) The set of header information (if any) included within the encapsulated text of messages is a local matter, and this RFC does not specify formatting conventions to distinguish replicated header fields from other encapsulated text.

### 4.5 Mail for Mailing Lists

When mail is addressed to mailing lists, two different methods of processing can be applicable: the IK-per-list method and the IK-per- recipient method. Hybrid approaches are also possible, as in the case of IK-per-list protection of a message on its path from an originator to a PEM-equipped mailing list exploder, followed by IK- per-recipient protection from the exploder to individual list recipients.

If a message's originator is equipped to expand a destination mailing list into its individual constituents and elects to do so (IK-per- recipient), the message's DEK (and, in the symmetric key management case, MIC) will be encrypted under each per-recipient IK and all such encrypted representations will be incorporated into the transmitted message. Note that per-recipient encryption is required only for the relatively small DEK and MIC quantities carried in the "Key-Info:" field, not for the message text which is, in general, much larger. Although more IKs are involved in processing under the IK-per- recipient method, the pairwise IKs can be individually revoked and possession of one IK does not enable a successful masquerade of another user on the list.

If a message's originator addresses a message to a list name or alias, use of an IK associated with that name or alias as a entity (IK-per-list), rather than resolution of the name or alias to its constituent destinations, is implied. Such an IK must, therefore, be available to all list members. Unfortunately, it implies an undesirable level of exposure for the shared IK, and makes its revocation difficult. Moreover, use of the IK-per-list method allows any holder of the list's IK to masquerade as another originator to the list for authentication purposes.

Pure IK-per-list key management in the asymmetric case (with a common private key shared among multiple list members) is particularly disadvantageous in the asymmetric environment, as it fails to preserve the forwardable authentication and non-repudiation characteristics which are provided for other messages in this environment. Use of a hybrid approach with a PEM-capable exploder is therefore particularly recommended for protection of mailing list traffic when asymmetric key management is used; such an exploder would reduce (per discussion in Section 4.4 of this RFC) incoming ENCRYPTED messages to MIC-ONLY or MIC-CLEAR form before forwarding them (perhaps re-encrypted under individual, per-recipient keys) to list members.

### 4.6 Summary of Encapsulated Header Fields

This section defines the syntax and semantics of the encapsulated header fields to be added to messages in the course of privacy enhancement processing.

The fields are presented in three groups. Normally, the groups will appear in encapsulated headers in the order in which they are shown, though not all fields in each group will appear in all messages. The following figures show the appearance of small example encapsulated messages. Figure 2 assumes the use of symmetric cryptography for key management. Figure 3 illustrates an example encapsulated ENCRYPTED message in which asymmetric key management is used.

**Figure 2: Example Encapsulated Message (Symmetric Case)**

```
----- BEGIN PRIVACY-ENHANCED MESSAGE -----
Proc-Type : 4, ENCRYPTED
Content-Domain : RFC822
DEK-Info : DES-CBC, F8143EDE5960C597
Originator-ID-Symmetric : linn@zendia.enet.dec.com, ,
Recipient-ID-Symmetric : linn@zendia.enet.dec.com, ptf-kmc, 3
Key-Info : DES-ECB, RSA-MD2,
           9FD3AAD2F2691B9A, B70665BB9BF7CBCDA60195DB94F727D3
Recipient-ID-Symmetric : pem-dev@tis.com, ptf-kmc, 4
Key-Info : DES-ECB, RSA-MD2,
           161A3F75DC82EF26,E2EF532C65CBCFF79F83A2658132DB47

LLrHB0eJzyhP+/fSStdW8okeEnv47jxe7SJ/iN72ohNcUk2jHEUSoH1nvNSIWL9M
8tEjmF/zxB+bATMtPjCUWbz8Lr9wloXIkjHUlBLpvXR0UrUzYbkNpk0agV2IzUpk
J6UiRRGcDSvzrsoK+oNvqu6z7Xs5Xfz5rDqUcMlK1Z6720dcBWGGsDLpTpSCnpot
dXd/H5LMDWnonNvPCwQUHt==
----- END PRIVACY-ENHANCED MESSAGE -----
```

Figure 4 illustrates an example encapsulated MIC-ONLY message in which asymmetric key management is used; since no per-recipient keys are involved in preparation of asymmetric-case MIC-ONLY messages, this example should be processable for test purposes by arbitrary PEM implementations.

Fully-qualified domain names (FQDNs) for hosts, appearing in the mailbox names found in entity identifier subfields of "Originator- ID-Symmetric:" and "Recipient-ID-Symmetric:" fields, are processed in a case-insensitive fashion. Unless specified to the contrary, other field arguments (including the user name components of mailbox names) are to be processed in a case-sensitive fashion.

In most cases, numeric quantities are represented in header fields as contiguous strings of hexadecimal digits, where each digit is represented by a character from the ranges "0"-"9" or upper case "A"-"F". Since public-key certificates and quantities encrypted using asymmetric algorithms are large in size, use of a more space-efficient encoding technique is appropriate for such quantities, and the encoding mechanism defined in Section 4.3.2.4 of this RFC, representing 6 bits per printed character, is adopted for this purpose.

**Figure 3:**
**Example Encapsulated ENCRYPTED Message (Asymm. Case)**

```
-----BEGIN PRIVACY-ENHANCED MESSAGE-----
Proc-Type: 4, ENCRYPTED
Content-Domain: RFC822
DEK-Info: DES-CBC, BFF968AA74691AC1
Originator-Certificate:
  MIIBlTCCAScCAWUwDQYJKoZIhvcNAQECBQAwUTELMAkGA1UEBhMCVVMxIDAeBgNV
  BAoTF1JTQSBEYXRhIFNlY3VyaXR5LCBJbmMuMQ8wDQYDVQQLEwZCZXRhIDExDzAN
  BgNVBAsTBk5PVEFSWTAeFw05MTA5MDQxODM4MTdaFw05MzA5MDMxODM4MTZaMEUx
  CzAJBgNVBAYTAlVTMSAwHgYDVQQKExdSU0EgRGF0YSBTZWN1cml0eSwgSW5jLjEU
  MBIGA1UEAxMLVGVzdCBVc2VyIDEwWTAKBgRVCAEBAgICAANLADBIAkEAwHZHl7i+
  yJcqDtjJCowzTdBJrdAiLAnSC+CnnjOJELyuQiBgkGrgIh3j8/x0fM+YrsyF1u3F
  LZPVtzlndhYFJQIDAQABMA0GCSqGSIb3DQEBAgUAA1kACKr0PqphJYw1j+YPtcIq
  iWlFPuN5jJ79Khfg7ASFxskYkEMjRNZV/HZDZQEhtVaU7Jxfzs2wfX5byMp2X3U/
  5XUXGx7qusDgHQGs7Jk9W8CW1fuSWUgN4w==
Key-Info: RSA,
  I3rRIGXUGWAF8js5wCzRTkdhO34PTHdRZY9Tuvm03M+NM7fx6qc5udixps2Lng0+
  wGrtiUm/ovtKdinz6ZQ/aQ==
Issuer-Certificate:
  MIIB3DCCAUgCAQowDQYJKoZIhvcNAQECBQAwTzELMAkGA1UEBhMCVVMxIDAeBgNV
  BAoTF1JTQSBEYXRhIFNlY3VyaXR5LCBJbmMuMQ8wDQYDVQQLEwZCZXRhIDExDTAL
  BgNVBAsTBFRMQ0EwHhcNOTEwOTAxMDgwMDAwWhcNOTIwOTAxMDc1OTU5WjBRMQsw
  CQYDVQQGEwJVUzEgMB4GA1UEChMXUlNBIERhdGEgU2VjdXJpdHksIEluYy4xDzAN
  BgNVBAsTBkJldGEgMTEPMA0GA1UECxMGTk9UQVJZMHAwCgYEVQgBAQICArwDYgAw
  XwJYCsnp6lQCxYykNlODwutF/jMJ3kL+3PjYyHOwk+/9rLg6X65B/LD4bJHtO5XW
  cqAz/7R7XhjYCm0PcqbdzoACZtIlETrKrcJiDYoP+DkZ8k1gCk7hQHpbIwIDAQAB
  MA0GCSqGSIb3DQEBAgUAA38AAICPv4f9Gx/tY4+p+4DB7MV+tKZnvBoy8zgoMGOx
  dD2jMZ/3HsyWKWgSF0eH/AJB3qr9zosG47pyMnTf3aSy2nBO7CMxpUWRBcXUpE+x
  EREZd9++32ofGBIXaialnOgVUn0OzSYgugiQ077nJLDUj0hQehCizEs5wUJ35a5h
MIC-Info: RSA-MD5, RSA,
  UdFJR8u/TIGhfH65ieewe2lOW4tooa3vZCvVNGBZirf/7nrgzWDABz8w9NsXSexv
  AjRFbHoNPzBuxwmOAFeA0HJszL4yBvhG
Recipient-ID-Asymmetric:
  MFExCzAJBgNVBAYTAlVTMSAwHgYDVQQKExdSU0EgRGF0YSBTZWN1cml0eSwgSW5j
  LjEPMA0GA1UECxMGQmV0YSAxMQ8wDQYDVQQLEwZOT1RBUlk=,66
Key-Info: RSA,
  O6BS1ww9CTyHPtS3bMLD+L0hejdvX6Qv1HK2ds2sQPEaXhX8EhvVphHYTjwekdWv
  7x0Z3Jx2vTAhOYHMcqqCjA==

qeWlj/YJ2Uf5ng9yznPbtD0mYloSwIuV9FRYx+gzY+8iXd/NQrXHfi6/MhPfPF3d
jIqCJAxvld2xgqQimUzoS1a4r7kQQ5c/Iua4LqKeq3ciFzEv/MbZhA==
----- END PRIVACY-ENHANCED MESSAGE -----
```

Encapsulated headers of PEM messages are folded using whitespace per RFC 822 header folding conventions; no PEM-specific conventions are defined for encapsulated header folding. The example shown in Figure 4 shows (in its "MIC-Info:" field) an asymmetrically encrypted quantity in its printably encoded representation, illustrating the use of RFC 822 folding.

In contrast to the encapsulated header representations defined in RFC 1113 and its precursors, the field identifiers adopted in this RFC do not begin with the prefix "X-" (for example, the field previously denoted "X-Key-Info:" is now denoted "Key-Info:") and such prefixes are not to be emitted by implementations conformant to this RFC. To simplify transition and interoperability with earlier implementations, it is suggested that implementations based on this RFC accept incoming encapsulated header fields carrying the "X-" prefix and act on such fields as if the "X-" were not present.

**Figure 4:**
**Example Encapsulated MIC-ONLY Message (Asymmetric Case)**

---

```
-----BEGIN PRIVACY-ENHANCED MESSAGE-----
Proc-Type : 4, MIC-ONLY
Content-Domain : RFC822
Originator-Certificate :
  MIIBlTCCAScCAWUwDQYJKoZIhvcNAQECBQAwUTELMAkGA1UEBhMCVVMxIDAeBgNV
  BAoTF1JTQSBEYXRhIFNlY3VyaXR5LCBJbmMuMQ8wDQYDVQQLEwZCZXRhIDExDzAN
  BgNVBAsTBk5PVEFSWTAeFw05MTA5MDQxODM4MTdaFw05MzA5MDMxODM4MTZaMEUx
  CzAJBgNVBAYTAlVTMSAwHgYDVQQKExdSU0EgRGF0YSBTZWN1cml0eSwgSW5jLjEU
  MBIGA1UEAxMLVGVzdCBVc2VyIDEwWTAKBgRVCAEBAgICAANLADBIAkEAwHZHl7i+
  yJcqDtjJCowzTdBJrdAiLAnSC+CnnjOJELyuQiBgkGrgIh3j8/x0fM+YrsyF1u3F
  LZPVtzlndhYFJQIDAQABMA0GCSqGSIb3DQEBAgUAA1kACKr0PqphJYw1j+YPtcIq
  iWlFPuN5jJ79Khfg7ASFxskYkEMjRNZV/HZDZQEhtVaU7Jxfzs2wfX5byMp2X3U/
  5XUXGx7qusDgHQGs7Jk9W8CW1fuSWUgN4w==
Issuer-Certificate :
  MIIB3DCCAUgCAQowDQYJKoZIhvcNAQECBQAwTzELMAkGA1UEBhMCVVMxIDAeBgNV
  BAoTF1JTQSBEYXRhIFNlY3VyaXR5LCBJbmMuMQ8wDQYDVQQLEwZCZXRhIDExDTAL
  BgNVBAsTBFRMQ0EwHhcNOTEwOTAxMDgwMDAwWhcNOTIwOTAxMDc1OTU5WjBRMQsw
  CQYDVQQGEwJVUzEgMB4GA1UEChMXUlNBIERhdGEgU2VjdXJpdHksIEluYy4xDzAN
  BgNVBAsTBkJldGEgMTEPMA0GA1UECxMGTk9UQVJZMHAwCgYEVQgBAQICArwDYgAw
  XwJYCsnp6lQCxYykNlODwutF/jMJ3kL+3PjYyHOwk+/9rLg6X65B/LD4bJHtO5XW
  cqAz/7R7XhjYCm0PcqbdzoACZtIlETrKrcJiDYoP+DkZ8k1gCk7hQHpbIwIDAQAB
  MA0GCSqGSIb3DQEBAgUAA38AAICPv4f9Gx/tY4+p+4DB7MV+tKZnvBoy8zgoMGOx
  dD2jMZ/3HsyWKWgSF0eH/AJB3qr9zosG47pyMnTf3aSy2nBO7CMxpUWRBcXUpE+x
  EREZd9++32ofGBIXaialnOgVUn0OzSYgugiQ077nJLDUj0hQehCizEs5wUJ35a5h
MIC-Info : RSA-MD5, RSA,
  jV2OfH+nnXHU8bnL8kPAad/mSQlTDZlbVuxvZAOVRZ5q5+Ejl5bQvqNeqOUNQjr6
  EtE7K2QDeVMCyXsdJlA8fA==

LSBBIG1lc3NhZ2UgZm9yIHVzZSBpbiB0ZXN0aW5nLg0KLSBGb2xsb3dpbmcgaXMg
YSBibGFuayBsaW5lOg0KDQpUaGlzIGlzIHRoZSBlbmQuDQo=
-----END PRIVACY-ENHANCED MESSAGE-----
```

---

### 4.6.1 Per-Message Encapsulated Header Fields

This group of encapsulated header fields contains fields which occur no more than once in a PEM message, generally preceding all other encapsulated header fields.

#### 4.6.1.1 Proc-Type Field

The "Proc-Type:" encapsulated header field, required for all PEM messages, identifies the type of processing performed on the transmitted message. Only one "Proc-Type:" field occurs in a message; the "Proc-Type:" field must be the first encapsulated header field in the message.

The "Proc-Type:" field has two subfields, separated by a comma. The first subfield is a decimal number which is used to distinguish among incompatible encapsulated header field interpretations which may arise as changes are made to this standard. Messages processed according to this RFC will carry the subfield value "4" to distinguish them from messages processed in accordance with prior PEM RFCs. The second subfield assumes one of a set of string values, defined in the following subsections.

##### 4.6.1.1.1 ENCRYPTED

The "ENCRYPTED" specifier signifies that confidentiality, authentication, integrity, and (given use of asymmetric key management) non-repudiation of origin security services have been applied to a PEM message's encapsulated text. ENCRYPTED messages require a "DEK-Info:" field and individual Recipient-ID and "Key- Info:" fields for all message recipients.

4.6.1.1.2 MIC-ONLY

The "MIC-ONLY" specifier signifies that all of the security services specified for ENCRYPTED messages, with the exception of confidentiality, have been applied to a PEM message's encapsulated text. MIC-ONLY messages are encoded (per Section 4.3.2.4 of this RFC) to protect their encapsulated text against modifications at message transfer or relay points.

Specification of MIC-ONLY, when applied in conjunction with certain combinations of key management and MIC algorithm options, permits certain fields which are superfluous in the absence of encryption to be omitted from the encapsulated header. In particular, when a keyless MIC computation is employed for recipients for whom asymmetric cryptography is used, "Recipient-ID-Asymmetric:" and "Key-Info:" fields can be omitted. The "DEK-Info:" field can be omitted for all "MIC-ONLY" messages.

#### 4.6.1.1.3 MIC-CLEAR

The "MIC-CLEAR" specifier represents a PEM message with the same security service selection as for a MIC-ONLY message. The set of encapsulated header fields required in a MIC-CLEAR message is the same as that required for a MIC-ONLY message.

MIC-CLEAR message processing omits the encoding step defined in Section 4.3.2.4 of this RFC to protect a message's encapsulated text against modifications within the MTS. As a result, a MIC-CLEAR message's text can be read by recipients lacking access to PEM software, even though such recipients cannot validate the message's signature. The canonical encoding discussed in Section 4.3.2.2 is performed, so interoperation among sites with different native character sets and line representations is not precluded so long as those native formats are unambiguously translatable to and from the canonical form. (Such interoperability is feasible only for those characters which are included in the canonical representation set.)

Omission of the printable encoding step implies that MIC-CLEAR message MICs will be validatable only in environments where the MTS does not modify messages in transit, or where the modifications performed can be determined and inverted before MIC validation processing. Failed MIC validation on a MIC-CLEAR message does not, therefore, necessarily signify a security-relevant event; as a result, it is recommended that PEM implementations reflect to their users (in a suitable local fashion) the type of PEM message being processed when reporting a MIC validation failure.

A case of particular relevance arises for inbound SMTP processing on systems which delimit text lines with local native representations other than the SMTP-conventional $<CR><LF>$. When mail is delivered to a UA on such a system and presented for PEM processing, the $<CR><LF>$ has already been translated to local form. In order to validate a MIC-CLEAR message's MIC in this situation, the PEM module must recanonicalize the incoming message in order to determine the inter-SMTP representation of the canonically encoded message (as defined in Section 4.3.2.2 of this RFC), and must compute the reference MIC based on that representation.

#### 4.6.1.1.4 CRL

The "CRL" specifier indicates a special PEM message type, used to transfer one or more Certificate Revocation Lists. The format of PEM CRLs is defined in RFC 1422. No user data or encapsulated text accompanies an encapsulated header specifying the CRL message type; a correctly-formed CRL message's PEM header is immediately followed by its terminating message boundary line, with no blank line intervening.

Only three types of fields are valid in the encapsulated header comprising a CRL message. The "CRL:" field carries a printable representation of a CRL, encoded using the procedures defined in Section 4.3.2.4 of this RFC. "CRL:" fields may (as an option) be followed by no more than one "Originator-Certificate:" field and any number of "Issuer-Certificate:" fields. The "Originator-Certificate:" and "Issuer-Certificate:" fields refer to the most recently previous "CRL:" field, and provide certificates useful in validating the signature included in the CRL. "Originator-Certificate:" and "Issuer-Certificate:" fields' contents are the same for CRL messages as they are for other PEM message types.

#### 4.6.1.2 Content-Domain Field

The "Content-Domain:" encapsulated header field describes the type of content which is represented within a PEM message's encapsulated text. It carries one string argument, whose value is defined as "RFC822" to indicate processing of RFC-822 mail as defined in this specification. It is anticipated that additional "Content-Domain:" values will be defined subsequently, in additional or successor documents to this specification. Only one "Content-Domain:" field occurs in a PEM message; this field is the PEM message's second encapsulated header field, immediately following the "Proc-Type:" field.

#### 4.6.1.3 DEK-Info Field

The "DEK-Info:" encapsulated header field identifies the message text encryption algorithm and mode, and also carries any cryptographic parameters (e.g., IVs) used for message encryption. No more than one "DEK-Info:" field occurs in a message; the field is required for all messages specified as "ENCRYPTED" in the "Proc-Type:" field.

The "DEK-Info:" field carries either one argument or two arguments separated by a comma. The first argument identifies the algorithm and mode used for message text encryption. The second argument, if present, carries any cryptographic parameters required by the algorithm and mode identified in the first argument. Appropriate message encryption algorithms, modes and identifiers and corresponding cryptographic parameters and formats are defined in RFC 1423.

### 4.6.2 Encapsulated Header Fields Normally Per-Message

This group of encapsulated header fields contains fields which ordinarily occur no more than once per message. Depending on the key management option(s) employed, some of these fields may be absent from some messages.

#### 4.6.2.1 Originator-ID Fields

Originator-ID encapsulated header fields identify a message's originator and provide the originator's IK identification component. Two varieties of Originator-ID fields are defined, the "Originator- ID-Asymmetric:" and "Originator-ID-Symmetric:" field. An "Originator-ID-Symmetric:" header field is required for all PEM messages employing symmetric key management. The analogous

"Originator-ID-Asymmetric:" field, for the asymmetric key management case, is used only when no corresponding "Originator-Certificate:" field is included.

Most commonly, only one Originator-ID or "Originator-Certificate:" field will occur within a message. For the symmetric case, the IK identification component carried in an "Originator-ID-Symmetric:" field applies to processing of all subsequent "Recipient-ID- Symmetric:" fields until another "Originator-ID-Symmetric:" field occurs. It is illegal for a "Recipient-ID-Symmetric:" field to occur before a corresponding "Originator-ID-Symmetric:" field has been provided. For the asymmetric case, processing of "Recipient-ID- Asymmetric:" fields is logically independent of preceding "Originator-ID-Asymmetric:" and "Originator-Certificate:" fields.

Multiple Originator-ID and/or "Originator-Certificate:" fields may occur in a message when different originator-oriented IK components must be used by a message's originator in order to prepare a message so as to be suitable for processing by different recipients. In particular, multiple such fields will occur when both symmetric and asymmetric cryptography are applied to a single message in order to process the message for different recipients.

Originator-ID subfields are delimited by the comma character (","), optionally followed by whitespace. Section 5.2, Interchange Keys, discusses the semantics of these subfields and specifies the alphabet from which they are chosen.

#### 4.6.2.1.1 Originator-ID-Asymmetric Field

The "Originator-ID-Asymmetric:" field contains an Issuing Authority subfield, and then a Version/Expiration subfield. This field is used only when the information it carries is not available from an included "Originator-Certificate:" field.

#### 4.6.2.1.2 Originator-ID-Symmetric Field

The "Originator-ID-Symmetric:" field contains an Entity Identifier subfield, followed by an (optional) Issuing Authority subfield, and then an (optional) Version/Expiration subfield. Optional "Originator-ID-Symmetric:" subfields may be omitted only if rendered redundant by information carried in subsequent "Recipient-ID- Symmetric:" fields, and will normally be omitted in such cases.

#### 4.6.2.2 Originator-Certificate Field

The "Originator-Certificate:" encapsulated header field is used only when asymmetric key management is employed for one or more of a message's recipients. To facilitate processing by recipients (at least in advance of general directory server availability), inclusion of this field in all messages is strongly recommended. The field transfers an originator's certificate as a numeric quantity, comprised of the certificate's DER encoding, represented in the header field with the encoding mechanism defined in Section 4.3.2.4 of this RFC. The semantics of a certificate are discussed in RFC 1422.

#### 4.6.2.3 MIC-Info Field

The "MIC-Info:" encapsulated header field, used only when asymmetric key management is employed for at least one recipient of a message, carries three arguments, separated by commas. The first argument identifies the algorithm under which the accompanying MIC is computed. The second argument identifies the algorithm under which the accompanying MIC is signed. The third argument represents a MIC signed with an originator's private key.

For the case of ENCRYPTED PEM messages, the signed MIC is, in turn, symmetrically encrypted using the same DEK, algorithm, mode and cryptographic parameters as are used to encrypt the message's encapsulated text. This measure prevents unauthorized recipients from determining whether an intercepted message corresponds to a predetermined plaintext value.

Appropriate MIC algorithms and identifiers, signature algorithms and identifiers, and signed MIC formats are defined in RFC 1423.

A "MIC-Info:" field will occur after a sequence of fields beginning with a "Originator-ID-Asymmetric:" or "Originator-Certificate:" field and followed by any associated "Issuer-Certificate:" fields. A "MIC-Info:" field applies to all subsequent recipients for whom asymmetric key management is used, until and unless overridden by a subsequent "Originator-ID-Asymmetric:" or "Originator-Certificate:" and corresponding "MIC-Info:".

### 4.6.3 Encapsulated Header Fields with Variable Occurrences

This group of encapsulated header fields contains fields which will normally occur variable numbers of times within a message, with numbers of occurrences ranging from zero to non-zero values which are independent of the number of recipients.

#### 4.6.3.1 Issuer-Certificate Field

The "Issuer-Certificate:" encapsulated header field is meaningful only when asymmetric key management is used for at least one of a message's recipients. A typical "Issuer-Certificate:" field would contain the certificate containing the public component used to sign the certificate carried in the message's "Originator-Certificate:" field, for recipients' use in chaining through that certificate's certification path. Other "Issuer-Certificate:" fields, typically representing higher points in a certification path, also may be included by an originator. It is recommended that the "Issuer- Certificate:" fields be included in an order corresponding to successive points in a certification path leading from the originator to a common point shared with the message's recipients (i.e., the Internet Certification Authority (ICA), unless a lower Policy Certification Authority (PCA) or CA is common to all recipients.) More information on certification paths can be found in RFC 1422.

The certificate is represented in the same manner as defined for the "Originator-

Certificate:" field (transporting an encoded representation of the certificate in X.509 [7] DER form), and any "Issuer-Certificate:" fields will ordinarily follow the "Originator- Certificate:" field directly. Use of the "Issuer-Certificate:" field is optional even when asymmetric key management is employed, although its incorporation is strongly recommended in the absence of alternate directory server facilities from which recipients can access issuers' certificates.

#### 4.6.4 Per-Recipient Encapsulated Header Fields

The encapsulated header fields in this group appear for each of an "ENCRYPTED" message's named recipients. For "MIC-ONLY" and "MIC- CLEAR" messages, these fields are omitted for recipients for whom asymmetric key management is employed in conjunction with a keyless MIC algorithm but the fields appear for recipients for whom symmetric key management or a keyed MIC algorithm is employed.

##### 4.6.4.1 Recipient-ID Fields

A Recipient-ID encapsulated header field identifies a recipient and provides the recipient's IK identification component. One Recipient-ID field is included for each of a message's named recipients. Section 5.2, Interchange Keys, discusses the semantics of the subfields and specifies the alphabet from which they are chosen. Recipient-ID subfields are delimited by the comma character (","), optionally followed by whitespace.

For the symmetric case, all "Recipient-ID-Symmetric:" fields are interpreted in the context of the most recent preceding "Originator- ID-Symmetric:" field. It is illegal for a "Recipient-ID-Symmetric:" field to occur in a header before the occurrence of a corresponding "Originator-ID-Symmetric:" field. For the asymmetric case, "Recipient-ID-Asymmetric:" fields are logically independent of a message's "Originator-ID-Asymmetric:" and "Originator-Certificate:" fields. "Recipient-ID-Asymmetric:" fields, and their associated "Key-Info:" fields, are included following a header's originator- oriented fields.

###### 4.6.4.1.1 Recipient-ID-Asymmetric Field

The "Recipient-ID-Asymmetric:" field contains, in order, an Issuing Authority subfield and a Version/Expiration subfield.

###### 4.6.4.1.2 Recipient-ID-Symmetric Field

The "Recipient-ID-Symmetric:" field contains, in order, an Entity Identifier subfield, an (optional) Issuing Authority subfield, and an (optional) Version/Expiration subfield.

#### 4.6.4.2 Key-Info Field

One "Key-Info:" field is included for each of a message's named recipients. In addition, it is recommended that PEM implementations support (as a locally-selectable option) the ability to include a "Key-Info:" field corresponding to a PEM message's originator, following an Originator-ID or "Originator-Certificate:" field and before any associated Recipient-ID fields, but inclusion of such a field is not a requirement for conformance with this RFC.

Each "Key-Info:" field is interpreted in the context of the most recent preceding Originator-ID, "Originator-Certificate:", or Recipient-ID field; normally, a "Key-Info:" field will immediately follow its associated predecessor field. The "Key-Info:" field's argument(s) differ depending on whether symmetric or asymmetric key management is used for a particular recipient.

##### 4.6.4.2.1 Symmetric Key Management

When symmetric key management is employed for a given recipient, the "Key-Info:" encapsulated header field transfers four items, separated by commas: an IK Use Indicator, a MIC Algorithm Indicator, a DEK and a MIC. The IK Use Indicator identifies the algorithm and mode in which the identified IK was used for DEK and MIC encryption for a particular recipient. The MIC Algorithm Indicator identifies the MIC computation algorithm used for a particular recipient. The DEK and MIC are symmetrically encrypted under the IK identified by a preceding "Recipient-ID-Symmetric:" field and/or prior "Originator- ID-Symmetric:" field.

Appropriate symmetric encryption algorithms, modes and identifiers, MIC computation algorithms and identifiers, and encrypted DEK and MIC formats are defined in RFC 1423.

##### 4.6.4.2.2 Asymmetric Key Management

When asymmetric key management is employed for a given recipient, the "Key-Info:" field transfers two quantities, separated by a comma. The first argument is an IK Use Indicator identifying the algorithm and mode in which the DEK is asymmetrically encrypted. The second argument is a DEK, asymmetrically encrypted under the recipient's public component.

Appropriate asymmetric encryption algorithms and identifiers, and encrypted DEK formats are defined in RFC 1423.

## 5. Key Management

Several cryptographic constructs are involved in supporting the PEM message processing procedure. A set of fundamental elements is assumed. Data Encrypting Keys (DEKs) are used to encrypt message text and (for some MIC computation algorithms) in the message integrity check (MIC) computation process. Interchange

Keys (IKs) are used to encrypt DEKs and MICs for transmission with messages. In a certificate-based asymmetric key management architecture, certificates are used as a means to provide entities' public components and other information in a fashion which is securely bound by a central authority. The remainder of this section provides more information about these constructs.

### 5.1 Data Encrypting Keys (DEKs)

Data Encrypting Keys (DEKs) are used for encryption of message text and (with some MIC computation algorithms) for computation of message integrity check quantities (MICs). In the asymmetric key management case, they are also used for encrypting signed MICs in ENCRYPTED PEM messages. It is strongly recommended that DEKs be generated and used on a one-time, per-message, basis. A transmitted message will incorporate a representation of the DEK encrypted under an appropriate interchange key (IK) for each of the named recipients.

DEK generation can be performed either centrally by key distribution centers (KDCs) or by endpoint systems. Dedicated KDC systems may be able to implement stronger algorithms for random DEK generation than can be supported in endpoint systems. On the other hand, decentralization allows endpoints to be relatively self-sufficient, reducing the level of trust which must be placed in components other than those of a message's originator and recipient. Moreover, decentralized DEK generation at endpoints reduces the frequency with which originators must make real-time queries of (potentially unique) servers in order to send mail, enhancing communications availability.

When symmetric key management is used, one advantage of centralized KDC-based generation is that DEKs can be returned to endpoints already encrypted under the IKs of message recipients rather than providing the IKs to the originators. This reduces IK exposure and simplifies endpoint key management requirements. This approach has less value if asymmetric cryptography is used for key management, since per-recipient public IK components are assumed to be generally available and per-originator private IK components need not necessarily be shared with a KDC.

### 5.2 Interchange Keys (IKs)

Interchange Key (IK) components are used to encrypt DEKs and MICs. In general, IK granularity is at the pairwise per-user level except for mail sent to address lists comprising multiple users. In order for two principals to engage in a useful exchange of PEM using conventional cryptography, they must first possess common IK components (when symmetric key management is used) or complementary IK components (when asymmetric key management is used). When symmetric cryptography is used, the IK consists of a single component, used to encrypt both DEKs and MICs. When asymmetric cryptography is used, a recipient's public component is used as an IK to encrypt DEKs (a transformation invertible only by a recipient possessing the corresponding private component), and the originator's private component is used to encrypt MICs (a transformation invertible by all recipients, since the originator's certificate provides all recipients with the public component required to perform MIC validation.

This RFC does not prescribe the means by which interchange keys are made available to appropriate parties; such means may be centralized (e.g., via key management servers) or decentralized (e.g., via pairwise agreement and direct distribution among users). In any case, any given IK component is associated with a responsible Issuing Authority (IA). When certificate-based asymmetric key management, as discussed in RFC [1422, is employed, the IA function is performed by a Certification Authority (CA).

When an IA generates and distributes an IK component, associated control information is provided to direct how it is to be used. In order to select the appropriate IK(s) to use in message encryption, an originator must retain a correspondence between IK components and the recipients with which they are associated. Expiration date information must also be retained, in order that cached entries may be invalidated and replaced as appropriate.

Since a message may be sent with multiple IK components identified, corresponding to multiple intended recipients, each recipient's UA must be able to determine that recipient's intended IK component. Moreover, if no corresponding IK component is available in the recipient's database when a message arrives, the recipient must be able to identify the required IK component and identify that IK component's associated IA. Note that different IKs may be used for different messages between a pair of communicants. Consider, for example, one message sent from A to B and another message sent (using the IK-per-list method) from A to a mailing list of which B is a member. The first message would use IK components associated individually with A and B, but the second would use an IK component shared among list members.

When a PEM message is transmitted, an indication of the IK components used for DEK and MIC encryption must be included. To this end, Originator-ID and Recipient-ID encapsulated header fields provide (some or all of) the following data:

1. Identification of the relevant Issuing Authority (IA subfield)
2. Identification of an entity with which a particular IK component is associated (Entity Identifier or EI subfield)
3. Version/Expiration subfield

In the asymmetric case, all necessary information associated with an originator can be acquired by processing the certificate carried in an "Originator-Certificate:" field; to avoid redundancy in this case, no "Originator-ID-Asymmetric:" field is included if a corresponding "Originator-Certificate:" appears.

The comma character (",") is used to delimit the subfields within an Originator-ID or Recipient-ID. The IA, EI, and version/expiration subfields are generated from a restricted character set, as prescribed by the following BNF (using notation as defined in RFC 822, Sections 2 and 3.3):

*IKsubfld* := *1 * ia-char*

*ia-char* := *DIGIT* | *ALPHA* | " ' " | "+" | "(" | ")" | "." |
" | " | "=" | "?" | "–" | "@" | "%" | "!" | " " " |
"_" | " < " | " > "

An example Recipient-ID field for the symmetric case is as follows:

*Recipient-ID-Symmetric: linn@zendia.enet.dec.com,ptf-kmc,2*

This example field indicates that IA "ptf-kmc" has issued an IK component for use on messages sent to "linn@zendia.enet.dec.com", and that the IA has provided the number 2 as a version indicator for that IK component.

An example Recipient-ID field for the asymmetric case is as follows:

**Recipient-ID-Asymmetric:**

MFExCzAJBgNVBAYTAlVTMSAwHgYDVQQKExdSU0EgRGF0YSBTZWN1cml0eSwgSW5j
LjEPMA0GA1UECxMGQmV0YSAxMQ8wDQYDVQQLEwZOT1RBUlk=,66

This example field includes the printably encoded BER representation of a certificate's issuer distinguished name, along with the certificate serial number 66 as assigned by that issuer.

### 5.2.1 Subfield Definitions

The following subsections define the subfields of Originator-ID and Recipient-ID fields.

#### 5.2.1.1 Entity Identifier Subfield

An entity identifier (used only for "Originator-ID-Symmetric:" and "Recipient-ID-Symmetric:" fields) is constructed as an IKsubfld. More restrictively, an entity identifier subfield assumes the following form:

$$< user > \ @ < domain - qualified - host >$$

In order to support universal interoperability, it is necessary to assume a universal form for the naming information. For the case of installations which transform local host names before transmission into the broader Internet, it is strongly recommended that the host name as presented to the Internet be employed.

#### 5.2.1.2 Issuing Authority Subfield

An IA identifier subfield is constructed as an IKsubfld. This RFC does not define this subfield's contents for the symmetric key management case. Any prospective IAs which are to issue symmetric keys for use in conjunction with this RFC must coordinate assignment of IA identifiers in a manner (centralized or hierarchic) which assures uniqueness.

For the asymmetric key management case, the IA identifier subfield will be formed from the ASN.1 BER representation of the distinguished name of the issuing organization or organizational unit. The distinguished encoding rules specified in Clause 8.7 of Recommendation X.509 ("X.509 DER") are to be employed in generating this representation. The encoded binary result will be represented for inclusion in a transmitted header using the procedure defined in Section 4.3.2.4 of this RFC.

### 5.2.1.3 Version/Expiration Subfield

A version/expiration subfield is constructed as an IKsubfld. For the symmetric key management case, the version/expiration subfield format is permitted to vary among different IAs, but must satisfy certain functional constraints. An IA's version/expiration subfields must be sufficient to distinguish among the set of IK components issued by that IA for a given identified entity. Use of a monotonically increasing number is sufficient to distinguish among the IK components provided for an entity by an IA; use of a timestamp additionally allows an expiration time or date to be prescribed for an IK component.

For the asymmetric key management case, the version/expiration subfield's value is the hexadecimal serial number of the certificate being used in conjunction with the originator or recipient specified in the "Originator-ID-Asymmetric:" or "Recipient-ID-Asymmetric:" field in which the subfield occurs.

### 5.2.2 IK Cryptoperiod Issues

An IK component's cryptoperiod is dictated in part by a tradeoff between key management overhead and revocation responsiveness. It would be undesirable to delete an IK component permanently before receipt of a message encrypted using that IK component, as this would render the message permanently undecipherable. Access to an expired IK component would be needed, for example, to process mail received by a user (or system) which had been inactive for an extended period of time. In order to enable very old IK components to be deleted, a message's recipient desiring encrypted local long term storage should transform the DEK used for message text encryption via re-encryption under a locally maintained IK, rather than relying on IA maintenance of old IK components for indefinite periods.

## 6. User Naming

Unique naming of electronic mail users, as is needed in order to select corresponding keys correctly, is an important topic and one which has received (and continues to receive) significant study. For the symmetric case, IK components are identified in PEM headers through use of mailbox specifiers in traditional Internet-wide form ("user@domain-qualified-host"). Successful operation in this mode relies on users (or their PEM implementations) being able to determine the universal-form names corresponding to PEM originators and recipients. If a PEM implementation operates in an environment where addresses in a local form differing from the universal form are used, translations must be performed in order to map between the universal form and that local representation.

The use of user identifiers unrelated to the hosts on which the users' mailboxes reside offers generality and value. X.500 distinguished names, as employed in the certificates of the recommended key management infrastructure defined in RFC 1422, provide a basis for such user identification. As directory services become more pervasive, they will offer originators a means to search for desired recipients which is based on a broader set of attributes than mailbox specifiers alone. Future work is anticipated in integration with directory services, particularly the mechanisms and naming schema of the Internet OSI directory pilot activity.

## 7. Example User Interface and Implementation

In order to place the mechanisms and approaches discussed in this RFC into context, this section presents an overview of a hypothetical prototype implementation. This implementation is a standalone program which is invoked by a user, and lies above the existing UA sublayer. In the UNIX system, and possibly in other environments as well, such a program can be invoked as a "filter" within an electronic mail UA or a text editor, simplifying the sequence of operations which must be performed by the user. This form of integration offers the advantage that the program can be used in conjunction with a range of UA programs, rather than being compatible only with a particular UA.

When a user wishes to apply privacy enhancements to an outgoing message, the user prepares the message's text and invokes the standalone program, which in turn generates output suitable for transmission via the UA. When a user receives a PEM message, the UA delivers the message in encrypted form, suitable for decryption and associated processing by the standalone program.

In this prototype implementation, a cache of IK components is maintained in a local file, with entries managed manually based on information provided by originators and recipients. For the asymmetric key management case, certificates are acquired for a user's PEM correspondents; in advance and/or in addition to retrieval of certificates from directories, they can be extracted from the "Originator-Certificate:" fields of received PEM messages.

The IK/certificate cache is, effectively, a simple database indexed by mailbox names. IK components are selected for transmitted messages based on the originator's identity and on recipient names, and corresponding Originator-ID, "Originator-Certificate:", and Recipient-ID fields are placed into the message's encapsulated header. When a message is received, these fields are used as a basis for a lookup in the database, yielding the appropriate IK component entries. DEKs and cryptographic parameters (e.g., IVs) are generated dynamically within the program.

Options and destination addresses are selected by command line arguments to the standalone program. The function of specifying destination addresses to the privacy enhancement program is logically distinct from the function of specifying the corresponding addresses to the UA for use by the MTS. This separation results from the fact that, in many cases, the local form of an address as specified to a UA differs from the Internet global form as used in "Originator-ID- Symmetric:" and "Recipient-ID-Symmetric:" fields.

## 8. Minimum Essential Requirements

This section summarizes particular capabilities which an implementation must provide for full conformance with this RFC.

RFC 1422 specifies asymmetric, certificate-based key management procedures to support the message processing procedures defined in this document; PEM implementation support for these key management procedures is strongly encouraged. Implementations supporting these procedures must also be equipped to display the names of originator and recipient PEM users in the X.500 DN form as authenticated by the procedures of RFC 1422.

The message processing procedures defined here can also be used with symmetric key management techniques, though no RFCs analogous to RFC 1422 are currently available to provide correspondingly detailed description of suitable symmetric key management procedures. A complete PEM implementation must support at least one of these asymmetric and/or symmetric key management modes.

A full implementation of PEM is expected to be able to send and receive ENCRYPTED, MIC-ONLY, and MIC-CLEAR messages, and to receive CRL messages. Some level of support for generating and processing nested and annotated PEM messages (for forwarding purposes) is to be provided, and an implementation should be able to reduce ENCRYPTED messages to MIC-ONLY or MIC-CLEAR for forwarding. Fully-conformant implementations must be able to emit Certificate and Issuer- Certificate fields, and to include a Key-Info field corresponding to the originator, but users or configurers of PEM implementations may be allowed the option of deactivating those features.

## 9. Descriptive Grammar

This section provides a grammar describing the construction of a PEM message.

**; PEM BNF representation, using RFC 822 notation.**

```
; imports field meta-syntax (field, field-name, field-body,
; field-body-contents) from RFC-822, sec. 3.2
; imports DIGIT, ALPHA, CRLF, text from RFC-822
```

; **Note :** algorithm and mode specifiers are officially defined in RFC [1115+]

```
<pemmsg> ::= <preeb>
             <pemhdr>
             [ CRLF <pemtext> ] ; absent for CRL message
             <posteb>

<preeb> ::= "-----BEGIN PRIVACY-ENHANCED MESSAGE-----" CRLF
<posteb> ::= "-----END PRIVACY-ENHANCED MESSAGE-----" CRLF | <preeb>

<pemtext> ::= <encbinbody>            ; for ENCRYPTED or MIC-ONLY messages
             | *(<text> CRLF)         ; for MIC-CLEAR
<pemhdr> ::= <normalhdr> | <crlhdr>

<normalhdr> ::= <proctype>
             <contentdomain>
             [ <dekinfo> ]            ; needed if ENCRYPTED
             ( 1 * (<origflds> * <recipflds>) )
             ; symmetric case -- recipflds included for all proc types
             | ( ( 1 * <origflds>) * (<recipflds>) )
             ; asymmetric case -- recipflds included for ENCRYPTED proc type

<crlhdr> ::= <proctype> 1 * (<crl> [<cert>] * (<issuercert>) )

<asymmorig> ::= <origid-asymm> | <cert>

<origflds> ::= <asymmorig> [<keyinfo>] *(<issuercert>) <micinfo>
             ; assymetric
             | <origid-symm> [ <keyinfo> ]    ; symmetric

<recipflds> ::= <recipid> <keyinfo>
```

**; definitions for PEM header fields**

```
<proctype> ::= "Proc-Type" ":" "4" "," <pemtypes> CRLF
<contentdomain> ::= "Content-Domain" ":" <contentdescrip> CRLF
<dekinfo> ::= "DEK-Info" ":" <dekalgid> [ "," <dekparameters> ] CRLF
<symmid> ::= <IKsubfld> "," [ <IKsubfld> ] "," [ <IKsubfld> ]
<asymmid> ::= <IKsubfld> "," <IKsubfld>
<origid-asymm> ::= "Originator-ID-Asymmetric" ":" <asymmid> CRLF
<origid-symm> ::= "Originator-ID-Symmetric" ":" <symmid> CRLF
<recipid> ::= <recipid-asymm> | <recipid-symm>
<recipid-asymm> ::= "Recipient-ID-Asymmetric" ":" <asymmid> CRLF
<recipid-symm> ::= "Recipient-ID-Symmetric" ":" <symmid> CRLF
<cert> ::= "Originator-Certificate" ":" <encbin> CRLF
<issuercert> ::= "Issuer-Certificate" ":" <encbin> CRLF
```

<micinfo> ::= ”MIC-Info” ”:” <micalgid> ”,” <ikalgid> ”,” <asymsignmic> CRLF
<keyinfo> ::=
”Key-Info” ”:” <ikalgid> ”,” <micalgid> ”,” <symencdek> ”,” <symencmic> CRLF | ; symmetric case
”Key-Info” ”:” <ikalgid> ”,” <asymencdek> CRLF ; asymmetric case

<crl> ::= ”CRL” ”:” <encbin> CRLF

<pemtypes> ::= ”ENCRYPTED” | ”MIC-ONLY” | ”MIC-CLEAR” | ”CRL”

<encbinchar> ::= ALPHA | DIGIT | ”+” | ”|” | ”=”

<encbingrp> ::= 4 * 4<encbinchar>

<encbin> ::= 1 * <encbingrp>

<encbinbody> ::= * ( 16 * 16 <encbingrp> CRLF ) [ 1 * 16<encbingrp> CRLF ]

<IKsubfld> ::= 1 * <ia-char>

; **Note :** ”,” removed from <ia-char> set, so that Orig-ID and Recip-ID fields
; can be delimited with commas (not colons), like all other fields

<ia-char> ::= DIGIT | ALPHA | “ ’ ” | ”+” | ”(” | ”)” | ”.” | ”|” | ”=” |
”?” | ”-” | ”@” | ”%” | ”!” | “ ” ” | ”_” | ”<” | ”>”

<hexchar> ::= DIGIT | ”A” | ”B” | ”C” | ”D” | ”E” | ”F” ; no lower case

; This specification defines one value (”RFC822”) for <contentdescrip>:
; other values may be defined in future in separate or successor documents

<contentdescrip> ::= ”RFC822”

**; The following items are defined in RFC [1115+]**

; <dekalgid>
; <dekparameters>
; <micalgid>
; <ikalgid>
; <asymsignmic>
; <symencdek>
; <symencmic>
; <asymencdek>

**NOTES:**

[1] Key generation for MIC computation and message text encryption may either be performed by the sending host or by a centralized server. This RFC does not constrain this design alternative. Section 5.1 identifies possible advantages of a centralized server approach if symmetric key management is employed.

[2] Postel, J., "Simple Mail Transfer Protocol", STD 10, RFC 821, August 1982.

[3] This transformation should occur only at an SMTP endpoint, not at an intervening relay, but may take place at a gateway system linking the SMTP realm with other environments.

[4] Use of a canonicalization procedure similar to that of SMTP was selected because its functions are widely used and implemented within the Internet mail community, not for purposes of SMTP interoperability with this intermediate result.

[5] Crocker, D., "Standard for the Format of ARPA Internet Text Messages", STD 11, RFC 822, August 1982.

[6] Rose, M. T. and Stefferud, E. A., "Proposed Standard for Message Encapsulation", RFC 934, January 1985.

[7] CCITT Recommendation X.509 (1988), "The Directory - Authentication Framework".

[8] Throughout this RFC we have adopted the terms "private component" and "public component" to refer to the quantities which are, respectively, kept secret and made publicly available in asymmetric cryptosystems. This convention is adopted to avoid possible confusion arising from use of the term "secret key" to refer to either the former quantity or to a key in a symmetric cryptosystem.

**Patent Statement**

This version of Privacy Enhanced Mail (PEM) relies on the use of patented public key encryption technology for authentication and encryption. The Internet Standards Process as defined in RFC 1310 requires a written statement from the Patent holder that a license will be made available to applicants under reasonable terms and conditions prior to approving a specification as a Proposed, Draft or Internet Standard.

The Massachusetts Institute of Technology and the Board of Trustees of the Leland Stanford Junior University have granted Public Key Partners (PKP) exclusive sub-licensing rights to the following patents issued in the United States, and all of their corresponding foreign patents:

Cryptographic Apparatus and Method
("Diffie-Hellman")............................... No. 4,200,770

Public Key Cryptographic Apparatus
and Method ("Hellman-Merkle").................... No. 4,218,582

Cryptographic Communications System and
Method ("RSA")................................... No. 4,405,829

Exponential Cryptographic Apparatus
and Method ("Hellman-Pohlig").................... No. 4,424,414

These patents are stated by PKP to cover all known methods of practicing the art of Public Key encryption, including the variations collectively known as El Gamal.

Public Key Partners has provided written assurance to the Internet Society that parties will be able to obtain, under reasonable, nondiscriminatory terms, the right to use the technology covered by these patents. This assurance is documented in RFC 1170 titled "Public Key Standards and Licenses". A copy of the written assurance dated April 20, 1990, may be obtained from the Internet Assigned Number Authority (IANA).

The Internet Society, Internet Architecture Board, Internet Engineering Steering Group and the Corporation for National Research Initiatives take no position on the validity or scope of the patents and patent applications, nor on the appropriateness of the terms of the assurance. The Internet Society and other groups mentioned above have not made any determination as to any other intellectual property rights which may apply to the practice of this standard. Any further consideration of these matters is the user's own responsibility.

## Security Considerations

This entire document is about security.

## Author's Address

John Linn, *E-mail:* 104-8456@mcimail.com