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

STATUS OF THIS MEMO

   This RFC suggests a proposed protocol for the Internet community, and
   requests discussion and suggestions for improvements. Distribution
   of this memo is unlimited.

ACKNOWLEDGMENT

   This RFC is the outgrowth of a series of IAB Privacy Task Force
   meetings and of internal working papers distributed for those
   meetings. I would like to thank the following Privacy Task Force
   members and meeting guests for their comments and contributions at
   the meetings which led to the preparation of this RFC: David
   Balenson, Curt Barker, Matt Bishop, Danny Cohen, Tom Daniel, Charles
   Fox, Morrie Gasser, Steve Kent (chairman), John Laws, Steve Lipner,
   Dan Nessett, Mike Padlipsky, Rob Shirey, Miles Smid, Steve Walker,
   and Steve Wilbur.

1. Executive Summary

   This RFC defines message enciperment and authentication procedures,
   as the initial phase of an effort to provide privacy enhancement
   services for electronic mail transfer in the Internet. Detailed key
   management mechanisms to support these procedures will be defined in
   a subsequent RFC. As a goal of this initial phase, it is intended
   that the procedures defined here be compatible with a wide range of
   key management approaches, including both conventional (symmetric)
   and public-key (asymmetric) approaches for encryption of data
   encrypting keys. Use of conventional cryptography for message text
   encryption and/or integrity check computation is anticipated.

   Privacy enhancement services (confidentiality, authentication, and
   message integrity assurance) are offered through the use of
   end-to-end cryptography between originator and recipient User Agent
   processes, with no special processing requirements imposed on the
   Message Transfer System at endpoints or at intermediate relay
   sites. This approach allows privacy enhancement facilities to be
   incorporated 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.

2. Terminology

   For descriptive purposes, this RFC uses some terms defined in the OSI
   X.400 Message Handling System Model per the 1984 CCITT
   Recommendations. This section replicates a portion of 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 User
   Agent. A User Agent (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
   sender and recipient UAs. No privacy enhancements are offered for
   message fields which are added or transformed by intermediate relay
   points.

   Authentication and integrity facilities are always applied to the
   entirety of a message's text. No facility for confidentiality
   service without authentication is provided. Encryption facilities
   may be applied selectively to portions of a message's contents; this
   allows less sensitive portions of messages (e.g., descriptive fields)
   to be processed by a recipient's delegate in the absence of the

recipient's personal cryptographic keys. In the limiting case, where
the entirety of message text is excluded from encryption, this
feature can be used to yield the effective combination of
authentication and integrity services without confidentiality.

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 sender to be cognizant
       of whether a message's intended recipient implements privacy
       enhancements, in order that encoding and possible
       encipherment will not be performed on a message whose
       destination is not equipped to perform corresponding inverse
       transformations.

   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
       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 offer compatibility 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 an electronic mail privacy
       enhancement be available to the broadest possible user
       community, the selected mechanism 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
enhanced privacy 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 privacy-enhanced mail with a higher assurance
    level than a strictly UA-based approach would allow.

6. The defined mechanisms support privacy protection of
    electronic mail addressed to mailing lists.

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 modifications
throughout the Internet, three basic restrictions are imposed on the
set of measures to be considered in this RFC:

1. Measures will be restricted to implementation at endpoints
    and will be amenable to integration at the user agent (UA)
    level or above, rather than necessitating 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. 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.

As a result of these restrictions, the following facilities can be
provided:

    1. disclosure protection,

    2. sender authenticity, and

    3. message integrity measures,

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 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.

An important goal is that privacy enhancement mechanisms impose a
minimum of burden on the users they serve. In particular, this goal
suggests eventual automation of the key management mechanisms
supporting message encryption and authentication. In order to
facilitate deployment and testing of pilot privacy enhancement
implementations in the near term, however, compatibility with
out-of-band (e.g., manual) key distribution must also be supported.

A message's sender will determine whether privacy enhancements are to
be performed on a particular message. Therefore, a sender must be
able to determine whether particular recipients are equipped to
process privacy-enhanced mail. In a general architecture, these
mechanisms will be based on server queries; thus, the query function
could be integrated into a UA to avoid imposing burdens or
inconvenience on electronic mail users.

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.

   A two-level keying hierarchy is used to support privacy-enhanced
   message 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 quantities (MICs). DEKs are generated individually
          for each transmitted message; no predistribution of DEKs is
          needed to support privacy-enhanced message transmission.

      2. Interchange Keys (IKs) are used to encrypt DEKs for
          transmission within messages. An IK may be a single
          symmetric cryptographic key or, where asymmetric
          (public-key) cryptography is used to encrypt DEKs, the
          composition of a public component used by an originator and
          a secret component used by a recipient. Ordinarily, the
          same IK will be used for all messages sent between a given
          originator-recipient pair over a period of time. Each
          transmitted message includes a representation of the DEK(s)
          used for message encryption and/or authentication,
          encrypted under an individual IK per named recipient. This
          representation is associated with sender and recipient
          identification header fields, which enable recipients to
          identify the IKs used. With this information, the recipient
          can decrypt the transmitted DEK representation, yielding
          the DEK required for message text decryption and/or MIC
          verification.

   When privacy enhancement processing is to be performed on an outgoing
   message, a DEK is generated [1] for use in message encryption and a
   variant of the DEK is formed (if the chosen MIC algorithm requires a
   key) for use in MIC computation. An "X-Sender-ID:" field is included
   in the header to provide one identification component for the IK(s)
   used for message processing. An IK is selected for each individually
   identified recipient; a corresponding "X-Recipient-ID:" field,
   interpreted in the context of a prior "X-Sender-ID:" field, serves to
   identify each IK. Each "X-Recipient-ID:" field is followed by an
   "X-Key-Info:" field, which transfers the DEK and computed MIC. The
   DEK and MIC are encrypted for transmission under the appropriate IK.

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 system to be
decrypted on a different type. 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 encryption and MIC computation processes.

For encryption purposes, the canonical representation is padded as
required by the encryption algorithm. The padded canonical
representation is encrypted (except for any regions explicitly
excluded from encryption). The canonically encoded representation is
encoded, after encryption, 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.

The output of the encoding procedure is combined with a set of header
fields carrying cryptographic control information. The result is
passed to the electronic mail system to be encapsulated as the text
portion of a transmitted message.

When a privacy-enhanced message is received, the cryptographic
control fields within its text portion provide the information
required for the authorized recipient to perform MIC verification and
decryption of the received message text. First, the printable
encoding is converted to a bitstring. The MIC is verified.
Encrypted portions of the transmitted message are decrypted, and the
canonical representation is converted to the recipient's local form,
which need not be the same as the sender's local form.

4.2 Encryption Algorithms and Modes

   For purposes of this RFC, the Block Cipher Algorithm DEA-1, defined
   in ISO draft international standard DIS 8227 [2] shall be used for
   encryption of message text. The DEA-1 is equivalent to the Data
   Encryption Standard (DES), as defined in FIPS PUB 46 [3]. When used
   for encryption of text, the DEA-1 shall be used in the Cipher Block
   Chaining (CBC) mode, as defined in ISO DIS 8372 [4]. The CBC mode
   definition in DIS 8372 is equivalent to that provided in FIPS PUB 81
   [5]. A unique initializing vector (IV) will be generated for and
   transmitted with each privacy-enhanced electronic mail message.

An algorithm other than DEA-1 may be employed, provided that it
satisfies the following requirements:

    1. It must be a 64-bit block cipher, enciphering and
        deciphering in 8-octet blocks.

    2. It is usable in the ECB and CBC modes defined in DIS
        8372.

    3. It is able to be keyed using the procedures and
        parameters defined in this RFC.

    4. It is appropriate for MIC computation, if the selected
        MIC computation algorithm is eCcryption-based.

    5. Cryptographic key field lengths are limited to 16 octets
        in length.

Certain operations require that one key be encrypted under another
key (interchange key) for purposes of transmission. This encryption
may be performed using symmetric cryptography by using DEA-1 in
Electronic Codebook (ECB) mode. A header facility is available to
indicate that an associated key is to be used for encryption in
another mode (e.g., the Encrypt-Decrypt-Encrypt (EDE) mode used for
key encryption and decryption with pairs of 64-bit keys, as described
by ASC X3T1 [6], or public-key algorithms).

Support of public key algorithms for key encryption is under active
consideration, and it is intended that the procedures defined in this
RFC be appropriate to allow such usage. Support of key encryption
modes other than ECB is optional for implementations, however.
Therefore, in support of universal interoperability, interchange key
providers should not specify other modes in the absence of a priori
information indicating that recipients are equipped to perform key
encryption in other modes.

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 transport facilities. An encryption mechanism must also be
compatible with the local conventions of the computer systems which
it interconnects. In our approach, a canonicalization step is
performed to abstract out local conventions and a subsequent encoding

step is performed to conform to the characteristics of the underlying mail transport medium (SMTP). The encoding conforms to SMTP constraints, established to support interpersonal messaging. SMTP's rules are also used independently in the canonicalization process. RFC-821's [7] Section 4.5 details SMTP's transparency constraints.

To encode 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 use a different native representation to delimit lines. For example, the <CR><LF> sequences delimiting lines in mail inbound to UNIX(tm) 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 [8] server. 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 privacy-enhanced mail across an
   environment in which intermediate conversions may occur encodes mail
   in a fashion which is uniformly representable across the set of
   privacy-enhanced UAs regardless of their systems' native character
   sets. This encoded form is used to represent mail text from sender
   to recipient, but the encoding is not applied to enclosing mail
   transport headers or to encapsulated headers inserted to carry
   control information between privacy-enhanced UAs. The encoding's
   characteristics are such that the transformations anticipated between
   sender and recipient UAs will not prevent an encoded message from
   being decoded properly at its destination.

   A sender may exclude one or more portions of a message from
   encryption processing. Authentication processing is always applied
   to the entirety of message text. Explicit action is required to
   exclude a portion of a message from encryption processing; by
   default, encryption is applied to the entirety of message text. The
   user-level delimiter which specifies such exclusion is a local
   matter, and hence may vary between sender and recipient, but all
   systems should provide a means for unambiguous identification of
   areas excluded from encryption processing.

   An outbound privacy-enhanced message undergoes four transformation
   steps, described in the following four subsections.

4.3.2.1 Step 1: Local Form

   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

   The entire message text, including both those portions subject to
   encipherment processing and those portions excluded from such
   processing, is converted to the universal canonical form,
   equivalent to the inter-SMTP representation [9] as defined in
   RFC-821 and RFC-822 [10] (ASCII character set, <CR><LF> line
   delimiters). The processing required to perform this conversion is
   minimal on systems whose native character set is ASCII. Since a
   message is converted to a standard character set and representation
   before encryption, it can be decrypted and its MIC can be verified
   at any destination system before any conversion necessary to
   transform the message into a destination-specific local form is
   performed.

4.3.2.3 Step 3: Authentication and Encipherment

   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.

   Padding is applied to the canonical form as needed to perform
   encryption in the DEA-1 CBC mode, as follows: The number of octets
   to be encrypted is determined by subtracting the number of octets
   excluded from encryption from the total length of the encapsulated
   text. Octets with the hexadecimal value FF (all ones) are appended
   to the canonical form as needed so that the text octets to be
   encrypted, along with the added padding octets, fill an integral
   number of 8-octet encryption quanta. No padding is applied if the
   number of octets to be encrypted is already an integral multiple of
   8. The use of hexadecimal FF (a value outside the 7-bit ASCII set)
   as a padding value allows padding octets to be distinguished from
   valid data without inclusion of an explicit padding count indicator.

   The regions of the message which have not been excluded from
   encryption are encrypted. To support selective encipherment
   processing, an implementation must retain internal indications of the
   positions of excluded areas excluded from encryption with relation to
   non-excluded areas, so that those areas can be properly delimited in
   the encoding procedure defined in step 4. If a region excluded from
   encryption intervenes between encrypted regions, cryptographic state
   (e.g., IVs and accumulation of octets into encryption quanta) is
   preserved and continued after the excluded region.

4.3.2.4 Step 4: Printable Encoding

   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). This encoding step is
   performed for all privacy-enhanced messages.

   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.)
   Two additional characters, "=" and "*", are used to signify special
   processing functions. The character "=" is used for padding within
   the printable encoding procedure. The character "*" is used to
   delimit the beginning and end of a region which has been excluded

from encipherment processing. The encoding function's output is
delimited into text lines (using local conventions), with each line
containing 64 printable characters.

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, either at the end of a message or (when the
selective encryption facility is invoked) at the end of an encrypted
region or an excluded region. In other words, a full encoding
quantum is always completed at the end of a message and before the
delimiter "*" is output to initiate or terminate the representation
of a block excluded from encryption. 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.

In summary, the outbound message is subjected to the following
composition of transformations:

        Transmit_Form = Encode(Encipher(Canonicalize(Local_Form)))

The inverse transformations are performed, in reverse order, to
process inbound privacy-enhanced mail:

        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 sender and recipient
systems without loss of information.

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	(1)	*

(1) The character "*" is used to delimit portions of an encoded
message to which encryption processing has not been applied.

Printable Encoding Characters
Table 1

4.4 Encapsulation Mechanism

Encapsulation of privacy-enhanced messages within an enclosing layer
of headers interpreted by the electronic mail transport system offers
a number of advantages in comparison to a flat approach in which
certain fields within a single header are encrypted and/or carry
cryptographic control information. 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. Further, privacy enhancement processing can be
applied recursively. As far as the MTS is concerned, information
incorporated into cryptographic authentication or encryption
processing will reside in a message's text portion, not its header
portion.

The encapsulation mechanism to be used for privacy-enhanced mail is
derived from that described in RFC-934 [11] which is, in turn, based
on precedents in the processing of message digests in the Internet
community. To prepare a user message for encrypted or authenticated
transmission, it will be transformed into the representation shown in
Figure 1.

Enclosing Header Portion
        (Contains header fields per RFC-822)

Blank Line
        (Separates Enclosing Header from Encapsulated Message)

Encapsulated Message

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

   Encapsulated Header Portion
       (Contains encryption control fields inserted in plaintext.
       Examples include "X-IV:", "X-Sender-ID:", and "X-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
       (Separates Encapsulated Header from subsequent encoded
       Encapsulated Text Portion)

   Encapsulated Text Portion
       (Contains message data encoded as specified in Section 4.3;
       may incorporate protected copies of "Subject:", etc.)

   Post-Encapsulation Boundary (Post-EB)
       -----PRIVACY-ENHANCED MESSAGE BOUNDARY-----

                        Message Encapsulation
                             Figure 1

As a general design principle, sensitive data is protected by
incorporating the data within the encapsulated text rather than by
applying measures selectively to fields in the enclosing header.
Examples of potentially sensitive header information may include
fields such as "Subject:", with contents which are significant on an
end-to-end, inter-user basis. The (possibly empty) set of headers to
which protection is to be applied is a user option. It is strongly
recommended, however, that all implementations should replicate

copies of "X-Sender-ID:" and "X-Recipient-ID:" fields within the
encapsulated text and include those replicated fields in encryption
and MIC computations.

If a user wishes disclosure protection for header fields, they must
occur only in the encapsulated text and not in the enclosing or
encapsulated header. If disclosure protection is desired for a
message's subject indication, it is recommended that the enclosing
header contain a "Subject:" field indicating that "Encrypted Mail
Follows".

If an authenticated version of header information is desired, that
data can be replicated within the encapsulated text portion in
addition to its inclusion in the enclosing header. For example, a
sender wishing to provide recipients with a protected indication of a
message's position in a series of messages could include a copy of a
timestamp or message counter field within the encapsulated text.

A specific point regarding the integration of privacy-enhanced mail
facilities with the message encapsulation mechanism is worthy of
note. The subset of IA5 selected for transmission encoding
intentionally excludes the character "-", so encapsulated text can be
distinguished unambiguously from a message's closing encapsulation
boundary (Post-EB) without recourse to character stuffing.

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-
perrecipient method. The choice depends on the information available
to the sender and on the sender's preference.

If a message's sender addresses a message to a list name or alias,
use of an IK associated with that name or alias as a entity (IK-
perlist), 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. For the case of public-key
cryptography, the secret component of the composite IK must be
available to all list members. This alternative will be the normal
case for messages sent via remote exploder sites, as a sender to such
lists may not be cognizant of the set of individual recipients.
Unfortunately, it implies an undesirable level of exposure for the
shared IK or component, 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 sender to the list for
authentication purposes.

If, in contrast, a message's sender is equipped to expand the
destination mailing list into its individual constituents and elects
to do so (IK-per-recipient), the message's DEK and 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 X-Key-Info
field, not for the message text which is, in general, much larger.
Although more IKs are involved in processing under the IK-
perrecipient 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.

4.6 Summary of Added Header and Control Fields

   This section summarizes the syntax and semantics of the new
   encapsulated header fields to be added to messages in the course of
   privacy enhancement processing. In certain indicated cases, it is
   recommended that the fields be replicated within the encapsulated
   text portion as well. Figure 2 shows the appearance of a small
   example encapsulated message using these fields. The example assumes
   the use of symmetric cryptography; no "X-Certificate:" field is
   carried. In all cases, hexadecimal quantities are represented as
   contiguous strings of digits, where each digit is represented by a
   character from the ranges "0"-"9" or upper case "A"-"F". Unless
   otherwise specified, all arguments are to be processed in a
   casesensitive fashion.

   Although the encapsulated header fields resemble RFC-822 header
   fields, they are a disjoint set and will not in general be processed
   by the same parser which operates on enclosing header fields. The
   complexity of lexical analysis needed and appropriate for
   encapsulated header field processing is significantly less than that
   appropriate to RFC-822 header processing. For example, many
   characters with special significance to RFC-822 at the syntactic
   level have no such significance within encapsulated header fields.

   When the length of an encapsulated header field is longer than the
   size conveniently printable on a line, whitespace may be used between
   the subfields of these fields to fold them in the manner of RFC-822,
   section 3.1.1. Any such inserted whitespace is not to be interpreted
   as a part of a subfield.

```
     -----PRIVACY-ENHANCED MESSAGE BOUNDARY-----
     X-Proc-Type: 2
     X-IV: F8143EDE5960C597
     X-Sender-ID: linn@ccy.bbn.com:::
     X-Recipient-ID: linn@ccy.bbn.com:ptf-kmc:3:BMAC:ECB
     X-Key-Info: 9FD3AAD2F2691B9A,B70665BB9BF7CBCD
     X-Recipient-ID: privacy-tf@venera.isi.edu:ptf-kmc:4:BMAC:ECB
     X-Key-Info: 161A3F75DC82EF26,E2EF532C65CBCFF7

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

                        Example Encapsulated Message
                                 Figure 2

4.6.1 X-Certificate Field

   The X-Certificate encapsulated header field is used only when
   public-key certificate key management is employed. It transfers a
   sender's certificate as a string of hexadecimal digits. The
   semantics of a certificate are discussed in Section 5.3,
   Certificates. The certificate carried in an X-Certificate field is
   used in conjunction with all subsequent X-Sender-ID and X-RecipientID
   fields until another X-Certificate field occurs; the ordinary case
   will be that only a single X-Certificate field will occur, prior to
   any X-Sender-ID and X-Recipient-ID fields.

   Due to the length of a certificate, it may need to be folded across
   multiple printed lines. In order to enable such folding to be
   performed, the hexadecimal digits representing the contents of a
   certificate are to be divided into an ordered set (with more
   significant digits first) of zero or more 64-digit groups, followed
   by a final digit group which may be any length up to 64-digits. A
   single whitespace character is interposed between each pair of groups
   so that folding (per RFC-822, section 3.1.1) may take place; this
   whitespace is ignored in parsing the received digit string.

4.6.2 X-IV Field

   The X-IV encapsulated header field carries the Initializing Vector
   used for message encryption. Only one X-IV field occurs in a
   message. It appears in all messages, even if the entirety of message
   text is excluded from encryption. Following the field name, and one
   or more delimiting whitespace characters, a 64-bit Initializing
   Vector is represented as a contiguous string of 16 hexadecimal

   digits.

4.6.3 X-Key-Info Field

   The X-Key-Info encapsulated header field transfers two items: a DEK
   and a MIC. One X-Key-Info field is included for each of a message's
   named recipients. The DEK and MIC are encrypted under the IK
   identified by a preceding X-Recipient-ID field and prior X-Sender-ID
   field; they are represented as two strings of contiguous hexadecimal
   digits, separated by a comma. For DEA-1, the DEK representation will
   be 16 hexadecimal digits (corresponding to a 64-bit key); this
   subfield can be extended to 32 hexadecimal digits (corresponding to a
   128-bit key), if required to support other algorithms. MICs are also
   represented as contiguous strings of hexadecimal digits. The size of
   a MIC is dependent on the choice of MIC algorithm as specified in the
   X-Recipient-ID field corresponding to a given recipient.

4.6.4 X-Proc-Type Field

   The X-Proc-Type encapsulated header field identifies the type of
   processing performed on the transmitted message. Only one X-ProcType
   field occurs in a message. It has one subfield, 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 "2".

4.6.5 X-Sender-ID Field

   The X-Sender-ID encapsulated header field provides the sender's
   interchange key identification component. It should be replicated
   within the encapsulated text. The interchange key identification
   component carried in an X-Sender-ID field is used in conjunction with
   all subsequent X-Recipient-ID fields until another X-Sender-ID field
   occurs; the ordinary case will be that only a single X-Sender-ID
   field will occur, prior to any X-Recipient-ID fields.

   The X-Sender-ID field contains (in order) an Entity Identifier
   subfield, an (optional) Issuing Authority subfield, an (optional)
   Version/Expiration subfield, and an (optional) IK Use Indicator
   subfield. The optional subfields are omitted if their use is
   rendered redundant by information carried in subsequent X-RecipientID
   fields; this will ordinarily be the case where symmetric cryptography
   is used for key management. The subfields are delimited by the colon
   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.

Note that multiple X-Sender-ID fields may occur within a single
encapsulated header. All X-Recipient-ID fields are interpreted in
the context of the most recent preceding X-Sender-ID field; it is
illegal for an X-Recipient-ID field to occur in a header before an
X-Sender-ID has been provided.

4.6.6 X-Recipient-ID Field

The X-Recipient-ID encapsulated header field provides the recipient's
interchange key identification component. One X-Recipient-ID field
is included for each of a message's named recipients. It should be
replicated within the encapsulated text. The field contains (in
order) an Entity Identifier subfield, an Issuing Authority subfield,
a Version/Expiration subfield, a MIC algorithm indicator subfield,
and an IK Use Indicator subfield. The subfields are delimited by the
colon character (":"), optionally followed by whitespace.

The MIC algorithm indicator is an ASCII string, selected from the
values defined in Appendix A of this RFC. Section 5.2, Interchange
Keys, discusses the semantics of the other subfields and specifies
the alphabet from which they are chosen. All X-Recipient-ID
fields are interpreted in the context of the most recent preceding
XSender-ID field; it is illegal for an X-Recipient-ID field to
occur in a header before an X-Sender-ID has been provided.

5. Key Management

Several cryptographic constructs are involved in supporting the
privacy-enhanced message processing procedure. While (as noted in
the Executive Summary section of this RFC), key management mechanisms
have not yet been fully defined, a set of fundamental elements are
assumed. Data Encrypting Keys (DEKs) are used to encrypt message
text and in the message integrity check (MIC) computation process.
Interchange Keys (IKs) are used to encrypt DEKs for transmission with
messages. In an asymmetric key management architecture, certificates
are used as a means to provide entities' public key 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 for computation of message integrity check quantities (MICs). It
is strongly recommended that DEKs be generated and used on a one-time
basis. A transmitted message will incorporate a representation of
the DEK encrypted under an appropriate interchange key (IK) for each
the authorized recipient.

DEK generation can be performed either centrally by key distribution
centers (KDCs) or by endpoint systems. Dedicated KDC systems may be
able to implement better 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 a message's originator and recipient. Moreover, decentralized
DEK generation at endpoints reduces the frequency with which senders
must make real-time queries of (potentially unique) servers in order
to send mail, enhancing communications availability.

When symmetric cryptography 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 senders. 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-sender secret IK components need not necessarily be
shared with a KDC.

5.2 Interchange Keys (IKs)

   Interchange Keys (IKs) are used to encrypt Data Encrypting Keys. 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 privacy-enhanced
   electronic mail using conventional cryptography, they must first
   share a common interchange key. When symmetric cryptography is used,
   the interchange key consists of a single component. When asymmetric
   cryptography is used, an originator and recipient must possess an
   asymmetric key's public and secret components, as appropriate. This
   pair of components, when composed, constitute an interchange key.

   While this RFC does not prescribe the means by which interchange keys
   are provided to appropriate parties, it is useful to note that 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 an IA generates and
   distributes an IK, associated control information is provided to
   direct how that IK is to be used. In order to select the appropriate
   IK to use in message encryption, a sender 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 component
representations, corresponding to multiple intended recipients, each
recipient must be able to determine which IK component is intended
for it. Moreover, if no corresponding IK component is available in
the recipient's database when a message arrives, the recipient must
be able to determine which IK component to request and to 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 privacy-enhanced message is transmitted, an indication of the
IK components used for DEK encryption must be included. To this end,
the "X-Sender-ID:" and "X-Recipient-ID:" encapsulated header fields
provide 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. Indicator of IK usage mode (IK use indicator subfield).

        4. Version/Expiration subfield.

The colon character (":") is used to delimit the subfields within an
"X-Sender-ID:" or "X-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 X-Recipient-ID: field is as follows:

            X-Recipient-ID: linn@ccy.bbn.com:ptf-kmc:2:BMAC:ECB

This example field indicates that IA "ptf-kmc" has issued an IK
component for use on messages sent to "linn@ccy.bbn.com", that the IA

has provided the number 2 as a version indicator for that IK
component, that the BMAC MIC computation algorithm is to be used for
the recipient, and that the IK component is to be used in ECB mode.

5.2.1 Subfield Definitions

The following subsections define the subfields of "X-Sender-ID:" and
"X-Recipient-ID:" fields.

5.2.1.1 Entity Identifier Subfield

An entity identifier 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. IA
identifiers must be assigned in a manner which assures uniqueness.
This can be done on a centralized or hierarchic basis.

5.2.1.3 Version/Expiration Subfield

A version/expiration subfield is constructed as an IKsubfld. The
version/expiration subfield format may 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.

5.2.1.4 MIC Algorithm Identifier Subfield

The MIC algorithm identifier, which occurs only within X-Recipient-ID
fields, is used to identify the choice of message integrity check
algorithm for a given recipient. Appendix A of this RFC specifies
the defined values for this subfield.

5.2.1.5 IK Use Indicator Subfield

   The IK use indicator subfield is an optional facility, provided to
   identify the encryption mode in which an IK component is to be used.
   Currently, this subfield may assume the following reserved string
   values: "ECB", "EDE", "RSA256", "RSA512", and "RSA1024"; the default
   value is "ECB".

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.

5.3 Certificates

   In an asymmetric key management architecture, a certificate binds an
   entity's public key component to a representation of the entity's
   identity and other attributes of the entity. A certificate's issuing
   authority signs the certificate, vouching for the correspondence
   between the entity's identity, attributes, and associated public key
   component. Once signed, certificate copies may be posted on multiple
   servers in order to make recipients' certificates directly accessible
   to originators at dispersed locations. This allows privacy-enhanced
   mail to be sent between an originator and a recipient without prior
   placement of a pairwise key at the originator and recipient, greatly
   enhancing mail system flexibility. The properties of a certificate's
   authority-applied signature make it unnecessary to be concerned about
   the prospect that servers, or other entities, could undetectably
   modify certificate contents so as to associate a public key with an
   inappropriate entity.

   Per the 1988 CCITT Recommendations X.411 [12] and X.509 [13], a
   subject's certificate is defined to contain the following parameters:

             1. A signature algorithm identifier, identifying the
                 algorithm used by the certificate's issuer to compute the
                 signature applied to the certificate.

2. Issuer identification, identifying the certificate's
    issuer with an O/R name.

3. Validity information, providing date and time limits
    before and after which the certificate should not be
    used.

4. Subject identification, identifying the certificate's
    subject with an O/R name.

5. Subject's public key.

6. Algorithm identifier, identifying the algorithm with
    which the subject's public key is to be used.

7. Signature, an asymmetrically encrypted, hashed version of
    the above parameters, computed by the certificate's
    issuer.

The Recommendations specify an ASN.1 encoding to define a
certificate. Pending further study, it is recommended that
electronic mail privacy enhancement implementations using asymmetric
cryptography for key management employ this encoding for
certificates. Section 4.2.3 of RFC-987 [14] specifies a procedure
for mapping RFC-822 addresses into the O/R names used in X.411/X.509
certificates.

6. User Naming

6.1 Current Approach

Unique naming of electronic mail users, as is needed in order to
select corresponding keys correctly, is an important topic and one
requiring significant study. A logical association exists between
key distribution and name/directory server functions; their
relationship is a topic deserving further consideration. These
issues have not been fully resolved at this writing. The current
architecture relies on association of IK components with user names
represented in a universal form ("user@host"), relying on the
following properties:

1. The universal form must be specifiable by an IA as it
    distributes IK components and known to a UA as it processes
    received IK components and IK component identifiers. If a
    UA or IA uses addresses in a local form which is different
    from the universal form, it must be able to perform an
    unambiguous mapping from the universal form into the local
    representation.

2. The universal form, when processed by a sender UA, must have
    a recognizable correspondence with the form of a recipient
    address as specified by a user (perhaps following local
    transformation from an alias into a universal form).

It is difficult to ensure these properties throughout the Internet.
For example, an MTS which transforms address representations between
the local form used within an organization and the universal form as
used for Internet mail transmission may cause property 2 to be
violated.

6.2 Issues for Consideration

The use of flat (non-hierarchic) electronic mail user identifiers,
which are unrelated to the hosts on which the users reside, may offer
value. Personal characteristics, like social security numbers, might
be considered. Individually-selected identifiers could be registered
with a central authority, but a means to resolve name conflicts would
be necessary.

A point of particular note is the desire to accommodate multiple
names for a single individual, in order to represent and allow
delegation of various roles in which that individual may act. A
naming mechanism that binds user roles to keys is needed. Bindings
cannot be immutable since roles sometimes change (e.g., the
comptroller of a corporation is fired).

It may be appropriate to examine the prospect of extending the
DARPA/DoD domain system and its associated name servers to resolve
user names to unique user IDs. An additional issue arises with
regard to mailing list support: name servers do not currently perform
(potentially recursive) expansion of lists into users. ISO and CSNet
are working on user-level directory service mechanisms, which may
also bear consideration.

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 prototype
implementation. This implementation is a standalone program which is
invoked by a user, and lies above the existing UA sublayer. In the
UNIX(tm) 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 (interacting with the program in order to provide
address information and other data required to perform privacy
enhancement processing), which in turn generates output suitable for
transmission via the UA. When a user receives a privacy-enhanced
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. This cache is,
effectively, a simple database. IK components are selected for
transmitted messages based on the sender's identity and on recipient
names, and corresponding "X-Sender-ID:" and "X-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
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 "X-Sender-ID:"
and "X-Recipient-ID:" fields.

8. Areas For Further Study

The procedures defined in this RFC are sufficient to support pilot
implementation of privacy-enhanced electronic mail transmission among
cooperating parties in the Internet. Further effort will be needed,
however, to enhance robustness, generality, and interoperability. In
particular, further work is needed in the following areas:

    1. User naming techniques, and their relationship to the domain
        system, name servers, directory services, and key management
        functions.

    2. Standardization of Issuing Authority functions, including
        protocols for communications among IAs and between User
        Agents and IAs.

    3. Specification of public key encryption algorithms to encrypt
        data encrypting keys.

    4. Interoperability with X.400 mail.

We anticipate generation of subsequent RFCs which will address these
topics.

9. References

   This section identifies background references which may be useful to
   those contemplating use of the mechanisms defined in this RFC.

      ISO 7498/Part 2 - Security Architecture, prepared by ISO/TC97/SC
      21/WG 1 Ad hoc group on Security, extends the OSI Basic Reference
      Model to cover security aspects which are general architectural
      elements of communications protocols, and provides an annex with
      tutorial and background information.

      US Federal Information Processing Standards Publication (FIPS PUB)
      46, Data Encryption Standard, 15 January 1977, defines the
      encipherment algorithm used for message text encryption and
      Message Authentication Code (MAC) computation.

      FIPS PUB 81, DES Modes of Operation, 2 December 1980, defines
      specific modes in which the Data Encryption Standard algorithm may
      to be used to perform encryption.

      FIPS PUB 113, Computer Data Authentication, May 1985, defines a
      specific procedure for use of the Data Encryption Standard
      algorithm to compute a MAC.

A. Message Integrity Check Algorithms

   This appendix identifies the alternative algorithms which may be used
   to compute Message Integrity Check (MIC) values, and assigns them
   character string identifiers to be incorporated in "X-Recipient-ID:"
   fields to indicate the choice of algorithm employed for individual
   message recipients.

   MIC algorithms which utilize DEA-1 cryptography are computed using a
   key which is a variant of the DEK used for message text encryption.
   The variant is formed by modulo-2 addition of the hexadecimal
   quantity F0F0F0F0F0F0F0F0 to the encryption DEK.

A.1 Conventional MAC (MAC)

   A conventional MAC, denoted by the string "MAC", is computed using
   the DEA-1 algorithm in the fashion defined in FIPS PUB 113 [15]. Use
   of the conventional MAC is not recommended for multicast messages.
   The message's encapsulated text is padded at the end, per FIPS PUB
   113, with zero-valued octets as needed in order to form an integral
   number of 8-octet encryption quanta. These padding octets are

inserted implicitly and are not transmitted with a message. The
result of a conventional MAC computation is a single 64-bit value.

A.2 Bidirectional MAC (BMAC)

A bidirectional MAC, denoted by the string "BMAC", yields a result
which is transferred as a single 128-bit value. The BMAC is computed
in the following manner: First, the encapsulated text is padded at
the end with zero-valued octets as needed in order to form an
integral number of 8-octet encryption quanta. These padding octets
are inserted implicitly and are not transmitted with a message. A
conventional MAC is computed on the padded form, and the resulting
64-bits form the high-order 64-bits of the BMAC result.

The low-order 64-bits of the BMAC result are also formed by computing
a conventional MAC, but the order of the 8-octet encryption quanta is
reversed for purposes of computation. In other words, the first
quantum entered into this computation is the last quantum in the
encapsulated text, and includes any added padding. The first quantum
in the text is the last quantum processed as input to this
computation. The octets within each 8-octet quantum are not
reordered.

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.

    [2] Information Processing Systems: Data Encipherment: Block
         Cipher Algorithm DEA 1.

    [3] Federal Information Processing Standards Publication 46,
         Data Encryption Standard, 15 January 1977.

    [4] Information Processing Systems: Data Encipherment: Modes of
         Operation of a 64-bit Block Cipher.

    [5] Federal Information Processing Standards Publication 81,
         DES Modes of Operation, 2 December 1980.

    [6] Addendum to the Transport Layer Protocol Definition for
         Providing Connection Oriented End to End Cryptographic Data
         Protection Using a 64-Bit Block Cipher, X3T1-85-50.3, draft
         of 19 December 1985, Gaithersburg, MD, p. 15.

[7]   Postel, J., Simple Mail Transfer Protocol (RFC-821), August
      1982.

[8]   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.

[9]   Use of the SMTP canonicalization procedure at this stage
      was selected since it is widely used and implemented in the
      Internet community, not because SMTP interoperability with
      this intermediate result is required; no privacy-enhanced
      message will be passed to SMTP for transmission directly
      from this step in the four-phase transformation procedure.

[10] Crocker, D., Standard for the Format of ARPA Internet Text
      Messages (RFC-822), August 1982.

[11] Rose, M. T. and Stefferud, E. A., Proposed Standard for
      Message Encapsulation (RFC-934), January 1985.

[12] CCITT Recommendation X.411 (1988), "Message Handling
      Systems: Message Transfer System: Abstract Service
      Definition and Procedures".

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

[14] Kille, S. E., Mapping between X.400 and RFC-822 (RFC-987),
      June 1986.

[15] Federal Information Processing Standards Publication 113,
      Computer Data Authentication, May 1985.