Proposed Standard for Message Encapsulation

STATUS OF THIS MEMO

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

Introduction, Scope, and Motivation

    The services that a user agent (UA) can offer are varied. Although
    all outgoing mail may be thought of as going through a single posting
    slot to connect to the message transport system (MTS), it is possible
    to consider a message draft being posted as described by one of the
    following four types of postings:

        Originate - a new message is composed from scratch, which, to the
        knowledge of the UA, is unrelated to any message previously
        handled by the user.

        Reply - a message is composed as a reply to a message previously
        received by the user. In most circumstances, the UA aids the user
        in composing the reply by constructing the header portion of the
        message draft, using components extracted from the received
        message headers.

        Forward - one more more messages previously received by the user
        are formatted by the UA as a part of the body portion of the
        draft. In this sense, a "digest" for an interest group may be
        considered as forwarding. Similarly, an argument may be made that
        "blind-carbon-copies" should also be handled in this fashion.

        Distribute - a message previously received by the user is
        re-posted to the MTS. The draft being re-posted is identical to
        the original message with the exception that certain "ReSent-XXX"
        headers are appended to the headers portion of the draft, and the
        "Return-Path" header is reset to reference the re-sender's
        address. (See [RFC-821] for a discussion of the Return-Path
        header.)

    Most user agents support the first two of these activities, many
    support the first three, and a few support all four.

    This memo concerns itself only with the third type, which is message
    forwarding. (For a brief treatment of the semantics of message
    components with respect to replies, see [RFC-822].) In many ways,

forwarding can be thought of as encapsulating one or more messages
inside another. Although this is useful for transfer of past
correspondence to new recipients, without a decapsulation process
(which this memo terms "bursting"), the forwarded messages are of
little use to the recipients because they can not be distributed,
forwarded, replied-to, or otherwise processed as separate individual
messages.

   NOTE: RFC-822 mistakenly refers to distribution as forwarding
   (section 4.2). This memo suggests below, that these two
   activities can and should be the same.

In the case of an interest group digest, a bursting capability is
especially useful. Not only does the ability to burst a digest
permit a recipient of the digest to reply to an individual digested
message, but it also allows the recipient to selectively process the
other messages encapsulated in the digest. For example, a single
digest issue usually contains more than one topic. A subscriber may
only be interested in a subset of the topics discussed in a
particular issue. With a bursting capability, the subscriber can
burst the digest, scan the headers, and process those messages which
are of interest. The others can be ignored, if the user so desires.

This memo is motivated by three concerns:

   In order to burst a message it is necessary to know how the
   component messages were encapsulated in the draft. At present
   there is no unambiguous standard for interest group digests. This
   memo proposes such a standard for the ARPA-Internet. Although
   interest group digests may appear to conform to a pseudo-standard,
   there is a serious ambiguity in the implementations which produce
   digests. By proposing this standard, the authors hope to solve
   this problem by specifically addressing the implementation
   ambiguity.

   Next, there is much confusion as to how "blind-carbon-copies"
   should be handled by UAs. It appears that each agent in the
   ARPA-Internet which supports a "bcc:" facility does so
   differently. Although this memo does not propose a standard for
   the generation of blind-carbon-copies, it introduces a formalism
   which views the "bcc:" facility as a special case of the
   forwarding activity.

   Finally, both forwarding and distribution can be accomplished with
   the same forwarding procedure, if a distributed message can be
   extracted as a separate individually processable message. With a
   proper bursting agent, it will be difficult to distinguish between

a message which has been distributed and a message which has been
extracted from a forwarded message. This memo argues that there is
no valuable distinction to be made, between forwarding and
distribution, and that in the interests of simplicity,
distribution facilities should not be generally available to the
ordinary users of a message system. However, this memo also
argues that such facilities should be available to certain trusted
entities within the MTS.

   NOTE: this memo does not propose that the distribution facility
   be abolished. Rather it argues the case forcefully in the hope
   that other interested parties in the ARPA-Internet will join
   this discussion.

Message Encapsulation

   This memo proposes the following encapsulation protocol: two agents
   act on behalf of the user, a forwarding agent, which composes the
   message draft prior to posting, and a bursting agent which decomposes
   the message after delivery.

   Definitions: a draft forwarding message consists of a header portion
   and a text portion. If the text portion is present, it is separated
   from the header portion by a blank line. Inside the text portion a
   certain character string sequence, known as an "encapsulation
   boundary", has special meaning. Currently (in existing
   digestification agents), an encapsulation boundary (EB) is defined as
   a line in the message which starts with a dash (decimal code 45,
   "-"). Initially, no restriction is placed on the length of the
   encapsulation boundary, or on the characters that follow the dash.

   1. The Header Portion

   This memo makes no restriction on the header portion of the draft,
   although it should conform to the RFC-822 standard.

   2. The Text Portion

   The text of the draft forwarding message consists of three parts: an
   initial text section, the encapsulated messages, and the final text
   section.

      2.1. The Initial Text Section

      All text (if any) up to the first EB comprises the initial text
      section of the draft. This memo makes no restrictions on the

format of the initial text section of the draft. In the case of a
digest, this initial text is usually the "table of contents" of
the digest.

2.2. The Final Text Section

All text (if any) after the last EB composes the final text
section of the draft. This memo makes no restrictions on the
format of the final text section of the draft. In the case of a
digest, this final text usually contains the sign-off banner for
the digest (e.g., "End of FOO Digest").

2.3. Encapsulated Messages

Each encapsulated message is bounded by two EBs: a pre-EB, which
occurs before the message; and, a post-EB, which occurs after the
message. For two adjacent encapsulated messages, the post-EB of
the first message is also the pre-EB of the second message.
Consistent with this, two adjacent EBs with nothing between them
should be treated as enclosing a null message, and thus two or
more adjacent EBs are equivalent to one EB.

Each encapsulated message consists of two parts: a headers portion
and a text portion. If the text portion is present, it is
separated from the header portion by a blank line.

    2.3.1. The Header Portion

    Minimally, there must be two header items in each message being
    forwarded, a "Date:" field and a "From:" field. This differs
    from RFC-822, which requires at least one destination address
    (in a "To:" or "cc:" field) or a possibly empty "Bcc:" field.
    Any addresses occuring in the header items for a message being
    forwarded must be fully qualified.

    2.3.2. The Text Portion

    This memo makes no restrictions on the format of the text
    portion of each encapsulated message. (Actually, this memo
    does restrict the format of the text portion of each
    encapsulated message, but these restrictions are discussed
    later.)

  Before summarizing the generation/parsing rules for message
encapsulation, two issues are addressed.

Compatibility with Existing User Agents

   The above encapsulation protocol is presently used by many user
   agents in the ARPA-Internet, and was specifically designed to
   minimize the amount of changes to existing implementations of
   forwarding agents in the ARPA-Internet.

   However, the protocol is not exactly like the pseudo-standard used by
   those forwarding agents that compose digests. In particular, the
   post-EB of all messages encapsulated in a digest is preceeded and
   followed by by a blank line. In addition, the first message
   encapsulated in a digest has a pre-EB that is followed by a blank
   line, but usually isn't preceeded by a blank line (wonderful).

   This memo recommends that implementors of forwarding agents wishing
   to remain compatible with existing bursting agents consider
   surrounding each EB with a blank line. It should be noted that blank
   lines following a pre-EB for an encapsulated message must be ignored
   by bursting agents. Further, this memo suggests that blank lines
   preceeding a post-EB also be ignored by bursting agents.

      NOTE: This recommendation is made in the interest of
      backwards-compatibility. A forwarding agent wishing to strictly
      adhere to this memo, should not generate blank lines surrounding
      EBs.

Character-Stuffing the Encapsulation Boundary

   It should be noted that the protocol is general enough to support
   both general forwarding of messages and the specific case of digests.
   Unfortunately, there is one issue of message encapsulation which
   apparently is not addressed by any forwarding agent (to the authors'
   knowledge) in the ARPA-Internet: what action does the forwarding
   agent take when the encapsulation boundary occurs within a the text
   portion of a message being forwarded? Without exception, this
   circumstance is ignored by existing forwarding agents.

   To address this issue, this memo proposes the following
   character-stuffing scheme: the encapsulation boundary is defined as a
   line which starts with a dash. A special case is made for those
   boundaries which start with a dash and are followed by a space
   (decimal code 32, " ").

      During forwarding, if the forwarding agent detects a line in the
      text portion of a message being forwarded which starts with the
      encapsulation boundary, the forwarding agent outputs a dash
      followed by a space prior to outputting the line.

     During bursting, if the bursting agent detects an encapsulation
     boundary which starts with a dash followed by a space, then the
     bursting agent does not treat the line as an encapsulation
     boundary, and outputs the remainder of the line instead.

   This simple character-stuffing scheme permits recursive forwardings.

Generation/Parsing Rules for Message Encapsulation

   The rules for forwarding/bursting are described in terms of regular
   expressions. The first author originally derived simple finite-state
   automata for the rules, but was unable to legibly represent them in
   this memo. It is suggested that the implementors sketch the automata
   to understand the grammar.

   The conventions used for the grammar are simple. Each state is
   followed by one or more alternatives, which are separated by the "|"
   character. Each alternative starts with a character that is received
   as input. (CRLF, although two characters is treated as one character
   herein.) The last alternative for a state is the character "c",
   which represents any character not specified in the preceeding
   alternatives. Optionally following the input character is an output
   string enclosed by curly-braces. Following this is the state that
   the automata enters. The reader should note that these grammars are
   extremely simple to implement (and, in most cases, can be implemented
   quite efficiently).

   When the forwarding agent encapsulates a message, it should apply the
   following finite-state automaton. The initial state is S1.

       S1 ::   CRLF {CRLF} S1
             | "-" {"- -"} S2
             | c {c} S2

       S2 ::   CRLF {CRLF} S1
             | c {c} S2

   This simply says that anytime a "-" is found at the beginning of a
   line, a "- " is output prior to outputting the line.

When the bursting agent decapsulates the text portion of a draft, it
should apply the following finite-state automaton. The initial state
is S1.

```
S1 ::    "-" S3
       | CRLF {CRLF} S1
       | c {c} S2

S2 ::    CRLF {CRLF} S1
       | c {c} S2

S3 ::    " " S2
       | c S4

S4 ::    CRLF S5
       | c S4

S5 ::    CRLF S5
       | c {c} S2
```

Although more complicated than the grammar used by the forwarding
agent to encapsulate a single message, this grammer is still quite
simple. Let us make the simplifying assumption that both the initial
and final text sections of the draft are messages in addition to the
encapsulated messages.

To begin, the current message being burst is scanned at state S1. All
characters are output until the EB is found (state S3). If "- " is
found, the automaton enters state S2 and characters from the current
message are continued to be output. Finally, a true EB is found
(state S4). As the automaton traverses from state S3 to S4, the
bursting agent should consider the current message ended. The
remainder of the EB is discarded (states S4 and S5). As the
automaton traverses from state S5 to S2, the bursting agent should
consider a new message started and output the first character. In
state S2, all characters are output until the EB is found.

Blind Carbon Copies

Many user agents support a blind-carbon-copy facility. With this
facility a draft has two types of addressees: visible and blind
recipients. The visible recipients are listed as addresses in the
"To:" and "cc:" fields of the draft, and the blind recipients are
listed as addresses in the "Bcc:" fields of the draft. The basis of
this facility is that copies of the draft which are delivered to the
recipients list the visible recipients only.

One method of achieving this is to post a single draft, which lacks
any "Bcc:" fields, and, during posting, to interact with the MTS in
such a way that copies are sent to both the visible and blind
recipients.

Unfortunately, a key problem with this arrangement is that the blind
recipients can accidently reply to the draft in such a way that the
visible recipients are included as addressees in the reply. This is
socially unacceptable! To avoid this problem, the message which the
visible recipients receive must be different than the message which
the blind recipients receive.

A second method is to post two drafts. The first, which goes to the
visible recipients, is simply the draft without any "Bcc:" fields.
The second, which goes to the blind recipients, is simply the draft
with some string prepended to any "To:" and "cc:" field. For example,
the user agent might prepend "BCC-" to these fields, so that the
blind recipients get a draft with "BCC-To:" and "Bcc-cc:" fields and
no "To:" or "cc:" fields. Unfortunately, this is often very confusing
to the blind recipients. Although accidental replies are not
possible, it is often difficult to tell that the draft received is
the result of a blind-carbon-copy.

The method which this memo suggests is to post two drafts, a visible
draft for the visible recipients, and a blind draft for the blind
recipients. The visible draft consists of the original draft without
any "Bcc:" fields. The blind draft contains the visible message as a
forwarded message. The headers for the blind draft contain the
minimal RFC-822 headers and, if the original draft had a "Subject:"
field, then this header field is also included. In addition, the
user agent might explicitly show that the blind draft is the result
of a blind-carbon-copy, with a "Bcc" header or prior to the first
encapsulating boundary in the body.

Message Distribution

The main purpose of message distribution (often called redistribution
or resending) is to provide to a secondary recipient, perhaps not
included among the original addressees, with a "true original" copy
that can be treated like an original in every respect.

Such distribution is most often done by discussion group moderators
who use automated agents to simply repost received messages to a
distribution list. The better automatic distribution agents insert a
new "Return-Path" header field to direct address failure notices to
the discussion group address list maintainer, rather than to the
original author. This form of distribution is encouraged because it

most simply serves to deliver messages to discussion group recipients
as processable originals. It is performed by trusted pseudo-MTS
agents.

A second kind of distribution is that done by individuals who wish to
transfer a processable copy of a received message to another
recipient. This second form is discouraged in various new standards
for message transfer. These include the NBS Standard for Mail
Interchange [FIPS-98], and the recent CCITT draft MHS (Mail Handling
Systems) X.400 standards [X.400]. In place of direct reposting of
received messages as though they are new drafts, the recommendation
is to forward the received message in the body of a new draft from
which is can be extracted by its secondary recipient for further
processing.

It is in support of this recommendation that this standard for
encapsulation/decapsulation is proposed.

References

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

    [RFC-821]   J.B. Postel. "Simple Mail Transfer Protocol",
                 USC/Information Sciences Institute. (August, 1982).

    [FIPS-98]   National Bureau of Standards. "Specification for
                 Message Format for Computer Based Message Systems."
                 (January, 1983).

    [X.400]   Consultative Committee on International Telephone and
                 Telegraph. "DRAFT Recommendation X.400. Message
                 Handling Systems: System Model-Service Elements."

Authors' Addresses

    Marshall T. Rose

        Department of Computer and Information Sciences
        University of Delaware
        Newark, DE 19716

        MRose@UDel.ARPA

    Einar A. Stefferud

        Network Management Associates, Inc.
        17301 Drey Lane
        Huntington Beach, CA 92647

        Stef@UCI.ARPA