Network Working Group
Request for Comments: 976

Mark. R. Horton
Bell Laboratories

February 1986

UUCP Mail Interchange Format Standard

Status of This Memo

In response to the need for maintenance of current information about the status and progress of various projects in the ARPA-Internet community, this RFC is issued for the benefit of community members. The information contained in this document is accurate as of the date of publication, but is subject to change. Subsequent RFCs will reflect such changes.

This document defines the standard format for the transmission of mail messages between machines in the UUCP Project. It does not address the format for storage of messages on one machine, nor the lower level transport mechanisms used to get the data from one machine to the next. It represents a standard for conformance by hosts in the UUCP zone. Distribution of this memo is unlimited.

1. Introduction

This document is intended to define the standard format for the transmission of mail messages between machines in the UUCP Project. It does not address the format for storage of messages on one machine, nor the lower level transport mechanisms used to get the data from one machine to the next. We assume remote execution of the rmail command (or equivalent) as the UUCP network primitive operation.

The general philosophy is that, if we were to invent a new standard, we would make ourselves incompatible with existing systems. There are already too many (incompatible) standards in the world, resulting in ambiguities such as a!b@c.d which is parsed a!(b@c.d) in the old UUCP world, and (a!b)@c.d in the Internet world. (Neither standard allows parentheses, and in adding them we would be compatible with neither. There would also be serious problems with the shell and with the UUCP transport mechanism.)

Having an established, well documented, and extensible family of standards already defined by the ARPA community, we choose to adopt these standards for the UUCP zone as well. (The UUCP zone is that subset of the community connected by UUCP which chooses to register with the UUCP project. It represents an administrative entity.) While the actual transport mechanism is up to the two hosts to arrange, and might include UUCP, SMTP, MMDF, or some other facility, we adopt RFC-920 (domains) and RFC-822 (mail format) as UUCP zone standards. All mail transmitted between systems should conform to

those two standards. In addition, should the ARPA community change these standards at a later time, we intend to change our standards to remain compatible with theirs, given a reasonable time to upgrade software.

This document specifies an interpretation of RFC-822 and RFC-920 in the UUCP world. It shows how the envelope should be encoded, and how UUCP routing is accomplished in an environment of mixed implementations.

2. Basics

Messages can be divided into two parts: the envelope and the message. The envelope contains information needed by the mail transport services, and the message contains information useful to the sender and receiver. The message is divided into the header and the body. Sometimes an intermediate host will add to the message (e.g. a Received line) but, except in the case of a gateway which must translate formats, it is not expected that intermediate hosts will change the message itself. In the UUCP world, the envelope consists of the "destination addresses" (normally represented as the argument or arguments to the rmail command) and the "source path" (normally represented in one or more lines at the beginning of the message beginning either "From " or ">From ", sometimes called "From_ lines".) The RFC-822 header lines (including "From:" and "To:") are part of the message, as is the text of the message body itself.

UUCP uses short host names, such as "ucbvax", at and below the transport layer. We refer to these names as "6 letter names", because all implementations of UUCP consider at least the first 6 letters significant. (Some consider the first 7 or the first 14 significant, but we must use the lowest common denominator.) UUCP names may be longer than 6 characters, but all such names much be unique in their first 6 letters. RFC-920 domain names, such as "ucbvax.Berkeley.EDU", are called "domain names." The two names are different. Upper and lower case are usually considered different in 6 letter names, but are considered equivalent in domain names. Names such as "ucbvax.UUCP", consisting of a 6 letter name followed by ".UUCP", previously were domain style references to a host with a given 6 letter name. Such names are being phased out in favor of organizational domain names such as "ucbvax.Berkeley.EDU"

2.1 Hybrid Addresses

There are (among others) two major kinds of mailing address syntax used in the UUCP world. The a!b!c!user ("bang paths") is used by older UUCP software to explicitly route mail to the destination. The user@domain ("domain") syntax is used in conformance to RFC-822. Under most circumstances, it is possible to look at a given address and determine which sort of address it is. However, a hybrid address with a ! to the left of an @, such as a!b@c, is ambiguous: it could be interpreted as (a!b)@c.d or a!(b@c.d). Both interpretations can be useful. The first interpretation is required by RFC-822, the second is a de-facto standard in the UUCP software.

Because of the confusion surrounding hybrid addresses, we recommend that all transport layer software avoid the use of hybrid addresses at all times. A pure bang syntax can be used to disambiguate, being written c.d!a!b in the first case above, and a!c.d!b in the second. We recommend that all implementations use this "bang domain" syntax unless they are sure of what is running on the next machine.

In conformance with RFC-822 and the AT&T Message Transfer Architecture, we recommand that any host that accepts hybrid addresses apply the (a!b)@c.d interpretation.

2.2 Transport

Since SMTP is not available to much of the UUCP domain, we define the method to be used for "remote execution" based transport mechanisms. The command to be "remotely executed" should read

```
rmail user@domain ...
```

with the message on the standard input of the command. The "user@domain" argument must conform to RFC-920 and RFC-822. More than one address argument is allowed, in order to save transmission costs for multiple recipients of the same message.

An alternative form that may be used is

```
rmail domain!user
```

where "domain" contains at least one period and no !'s. This is to be interpreted exactly the same as user@domain, and can be used to transport a message across old UUCP hosts without fear that they might change the address. The "user" string can contain any characters except "@". This character is forbidden because it is unknown what an intermediate host might do to it. (It is also

recommended that the "%" character be avoided, since some hosts treat "%" as a synonym for "@".) However, to route across hosts that don't understand domains, the following is possible

```
rmail a!b!c!domain!user
```

A "domain" can be distinguished from a 6 letter UUCP site name because a domain will contain at least one period. (In the case of single level domains with no periods, a period should be added to the end, e.g. Mark.Horton@att becomes "att.!Mark.Horton". A translator from ! to @ format should remove a trailing dot at the end of the domain, if one is present.) We don't expect this to happen, except for local networks using addresses like "user@host".

A simple implementation can always generate domain!user syntax (rather than user@domain) since it is safe to assume that gateways are class 3 (Classes are explained in section 3.5).

2.3 Batch SMTP

Standard conforming implementations may optionally support a protocol called "Batch SMTP". SMTP (Simple Mail Transfer Protocol) is the ARPA community standard mail transfer protocol (RFC-821). It is also used on BITNET and Mailnet. While SMTP was designed to be interactive, it is possible to batch up a series of commands and send them off to a remote machine for batch execution. This is used on BITNET, and is appropriate for UUCP. One advantage to BSMTP is that the UNIX shell does not get involved in the interpretation of messages, so it becomes possible to include special characters such as space and parentheses in electronic messages. (Such characters are expected to be popular in X.400 addresses.)

To support BSMTP on UNIX, a conforming host should arrange that mail to the user "b-smtp" is interpreted as Batch SMTP commands. (We use b-smtp instead of bsmtp because bsmtp might conflict with a login name.) Since many mail systems treat lines consisting of a single period as an "end of file" flag, and since SMTP uses the period as a required end of file flag, and to strip off headers, we put an extra "#" at the beginning of each BSMTP line. On a sendmail system, an easy way to implement this is to include the alias

```
b-smtp: "|egrep '^#' | sed 's/^#//' | /usr/lib/sendmail -bs"
```

which will feed the commands to an SMTP interpreter. A better solution would appropriately check for errors and send back an error message to the sender.

An example BSMTP message from seismo.CSS.GOV to cbosgd.ATT.COM is shown here. This sample is the file shipped over the UUCP link for in put to the command "rmail b-smtp". Note that the RFC- 822 message is between the DATA line and the period line. The envelope information is passed in the MAIL FROM and RCPT TO lines. The name of the sending system is in the HELO line. The actual envelope information (above the # lines) is ignored and need not be present.

```
From foo!bar Sun Jan 12 23:59:00 1986 remote from seismo Date:
Tue, 18 Feb 86 13:07:36 EST
From: mark@ucbvax.Berkeley.EDU
Message-Id: <8602181807.AA10228@mark@ucbvax.Berkeley.EDU> To:
b-smtp@cbosgd.ATT.COM

#HELO seismo.CSS.GOV
#MAIL FROM:<mark@ucbvax.Berkeley.EDU>
#RCPT TO:<mark@cbosgd.ATT.COM>
#DATA
#Date: Tue, 18 Feb 86 13:07:36 EST
#From: mark@ucbvax.Berkeley.EDU
#Message-Id: <8602181807.AA10228@mark@ucbvax.Berkeley.EDU> #To:
mark@cbosgd.ATT.COM
#
#This is a sample message.
#.
#QUIT
```

2.4 Envelope

The standard input of the command should begin with a single line

```
From domain!user date remote from system
```

followed immediately by the RFC-822 format headers and body of the message. It is possible that there will be additional From_ lines preceding this line - these lines may be added, one line for each system the message passes through. It is also possible that the "system" fields will be stacked into a single line, with many !’s in the "user" string. The ">" character may precede the "From". In general, this is the "envelope" information, and should follow the same conventions that previous UUCP mail has followed. The primary difference is that, when the system names are stacked up, if previously the result would have been a!b!c!mysys!me, the new result will be a!b!c!mysys!domain!me, where domain will contain at least one period, and "mysys" is often the 6 letter UUCP name for the same

system named by "domain". If the "domain!" is redundant, it may be omitted from the envelope, either in the source path or in the destination address.

The receiving system may discard extra "From_" lines if it folds the information into a a single From_ line. It passes the path!domain!user along as the "envelope" information containing the address of the sender of the message, and possibly preserves the forwarding date and system in a newly generated header line, such as Received or Sent-By. (Adding Received using this information is discouraged, since the line appears to have been added on a different system than the one actually adding it. That other system may have actually included a Received line too! The Sent-By line is similar to Received, but the date need not be converted into RFC-822 format, and the line is not claimed to have been added by the system whose name is mentioned.)

If the receiving system passes the message along to another system, it will add a "From_" line to the front, giving the same user@domain address for the sender, and its own name for the system. If the receiving system stores the message in a local mailbox, it is recommended that a single "From_" line be generated at the front of the message, keeping the date (in the same format, since certain mail reading programs are sensitive to this format), and not using the "remote from system" syntax.

Note - if an intermediate system adds text such as "system!" to the front of a "user@domain" syntax address, either in the envelope or the body, this is a violation of this standard and of RFC-822.

2.5 Routing

In order to properly route mail, it is sometimes necessary to know what software a destination or intermediate machine is running, or what conventions it follows. We have tried to minimize the amount of this information that is necessary, but the support of subdomains may require that different methods are used in different situations. For purposes of predicting the behavior of other hosts, we divide hosts into three classes. These classes are:

Class 1 old-style UUCP ! routing only. We assume that the host understands local user names:

```
rmail user
```

and bang paths

rmail host1!host2!user

but we assume nothing more about the host. If we have no information about a host, we can treat it as class 1 with no problems, since we make no assumptions about how it will handle hybrid addresses.

Class 2 Old style UUCP ! routing, and 4.2BSD style domain parsing. We assume the capabilities of class 1, plus the ability to understand

rmail user@domain

if the "domain" is one outside the UUCP zone which the host knows about. Class 2 hosts do not necessarily understand domain!user or have routers. Hosts in non-

UUCP RFC-920 domains are considered class 2, even though they may not understand host!user.

Class 3 All class 1 and 2 features are present. In addition, class 3 hosts must be able to route UUCP mail for hosts that are not immediately adjacent and also understand the syntax

rmail domain!user

as described above. All gateways into UUCP must be class 3.

This document describes what class 3 hosts must be able to process. Classes 1 and 2 already exist, and will continue to exist for a long time, but are viewed as "older systems" that may eventually be upgraded to class 3 status.

3. Algorithm

The algorithm for delivering a message to an address "user@domain" over UUCP links can be summarized as follows:

a. If the address is actually of the form @domain1:user@domain2, the "domain" used for the remainder should be "domain1" instead of "domain2", and the bang form reads domain1!domain2!user.

b. Determine d: the most specific part of "domain" that is recognized locally. This part will be a suffix of "domain". This can be done by scanning through a table with entries that go from specific to general, comparing entries with "domain" to see if the entries are at the tail of "domain". For example, with the address "mark@osgd.cb.att.com", if the local host recognizes "uucp" and "att.com", d would be "att.com". The final entry in the table will be the null string, matching any completely unrecognized domain.

c. Look in the found table entry for g: the name of the "gateway", and for r: a UUCP !-style route to reach g. G is not necessarily directly connected to the local host, but should be viewed as a gateway into the d domain. (The values of g and r for a given d may be different on different hosts, although g will often be the same.)

d. Look at the beginning of r to find the "next hop" host n. N will always be directly connected to the local host.

e. Determine, if possible, the class of g and n.

f. Create an appropriate destination string s to be interpreted by n. (See below.)

g. Pass the message off to n with destination information s.

In an environment with other types of networks that do not use UUCP ! parsing, the table will probably contain additional information, such as which type of link to use. The path information may be replaced in other environments by information specific to the network.

The first entries in the table mentioned in part (b) are normally very specific, and allow well known routes to be constructed directly instead of routing through the domain tree. The domain tree should be reserved for cases where no better information is available, or where traffic is very light, or where the default route is the best available. If a better route is available, that information can be put in the table. If a host has any significant amount of traffic sent to a second host, it is normally expected that the two hosts will set up a direct UUCP link and make an entry in their tables to send mail directly, even if they are in separate domains. Routing tables should be constructed to try to keep paths short and inexpensive for as much traffic as possible.

Here are some hints for the construction of the destination string n (step f above.) The "envelope recipient" information (the argument(s) to rmail) may be in either domain ! form (host.com!user) or domain @ form (user@host.com) as long as the sending site is sure the next hop is class 3. If the next hop is not class 3, or the sending site is not sure, the ! form should be used, if possible, since it is hard to predict what the next hop would do with a hybrid address.

If the gateway is known to be class 3, domain ! form may be used, but if the sending site is not sure, and the entire destination string was matched in the lookup (rather than some parent domain), the 6 letter ! form should be used: r!user, for example: dumbhost!host!user. If the gateway appears to actually be a gateway for a subdomain, e.g. because a parent domain was matched, (such as the address user@host.gateway.com, where host.gateway.com was not found but gateway.com was) it can be assumed to be at class 3. This allows routes such as dumbhost!domain!host.domain.com!user to be used with a reasonable degree of safety. If a direct link exists to the destination host, the user@domain syntax or the domain!user syntax may be used.

All hosts conforming to this standard are class 3, and all subdomain gateways must be class 3 hosts.

4. Example

Suppose host A.D.COM sends mail to host C.D.COM. Let's suppose that the 6 letter names for these hosts are aname and dname, and that the intermediate host to be routed through has name bname.

The user on A types

```
mail user@c.d.com
```

The user interface creates a file such as

```
Date:  9 Jan 1985   8:39 EST
From: myname@A.D.COM (My Name)
Subject: sample message
To: user@c.d.com

This is a sample message
```

and passes it to the transport mechanism with a command such as

```
sendmail user@c.d.com < file
```

The transport mechanism looks up a route to c.d.com. It does not find c.d.com in its database, so it looks up d.com, and finds that the path is bname!dname!%s, and that c.d.com is a class 3 host. Plugging in c.d.com!user, it gets the path bname!dname!c.d.com!user. (If it had found c.d.com with path bname!cname!%s, it would have omitted the domain from the resulting path: bname!cname!user, since it is not sure whether the destination host is class 1, 2, or 3.)

It prepends a From_ line and passes it to uux:

```
uux - bname!rmail dname!c.d.com!user < file2
```

where file2 contains

```
From A.D.COM!user Wed Jan  9 12:43:35 1985 remote from aname Date:
9 Jan 1985   8:39 EST
From: myname@A.D.COM (My Name)
Subject: sample message
To: user@c.d.com

This is a sample message
```

(Note the blank line at the end of the message - at least one blank line is required.) This results in the command

```
rmail dname!c.d.com!user
```

running on B. B prepends its own from line and passes the mail along:

```
uux - dname!rmail c.d.com!user < file3
```

where file3 contains

```
From nuucp Wed Jan  9 12:43:35 1985 remote from bname >From
A.D.COM!user Wed Jan  9 11:21:48 1985 remote from aname Date:  9
Jan 1985   8:39 EST
From: myname@A.D.COM (My Name)
Subject: sample message
To: user@c.d.com

This is a sample message
```

The command

```
rmail c.d.com!user
```

is run on C, which stacks the From_ lines

```
From bname!aname!A.D.COM!user Wed Jan  9 12:43:35 1985 Date:  9
Jan 1985   8:39 EST
From: myname@A.D.COM (My Name)
Subject: sample message
To: user@c.d.com

This is a sample message
```

and stores the message locally, probably in this same format.

5. Summary

Hosts conforming to this standard should accept all of the following forms:

```
rmail localuser               (no !%@ in user)
rmail hosta!hostb!user        (no !%@ in user)
rmail user@domain             (only . in domain)
rmail domain!user             (at least 1 . in domain)
rmail domain.!user            (in case domain has no dots)
```

The "envelope" portion of the message ("From_" lines) should conform to existing conventions, using ! routing. The "heading" portion of the message (the Word: lines such as Date:, From:, To:, and Subject:) must conform to RFC-822. All header addresses must be in the @ form. The originating site should ensure that the addresses conform to RFC-822, since no requirement is placed on forwarding sites or gateways to transform addresses into legal RFC-822 format. (Such forwarding sites and gateways should NOT, however, change a legal RFC-822 address such as user@domain into an illegal RFC-822 address such as gateway!user@domain, even if forwarding to a class 1 UUCP host.)

6. References

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

[2] Crocker, D., "Standard for the Format of ARPA Internet Text Messages", RFC-822, Department of Electrical Engineering, University of Delaware, August, 1982.

[3] Postel, J., and J. K. Reynolds, "Domain Requirements", RFC-920, USC/Information Sciences Institute, October, 1984.