# The Rand MH Message Handling System:

# Tutorial

Marshall T. Rose[†]
Jerry N. Sweet[‡]

Wed May 21 21:04:08 PDT 1986

ABSTRACT

This document introduces the UCI version of the Rand MH system to novice users. In particular, this tutorial discusses how to read, send, reply to, and review mail; aspects of the MH user profile affecting these activities; and other reference works on MH.

Although this document is based on the standard MH user manual[MROSE85A], this document is meant to supplement, not supersede, that lengthier work.

Comments concerning this documentation should be addressed to the Internet mailbox Bug-MH@ICS.UCI.EDU.

Computer Mail: [†] MRose@NRTC.NORTHROP.COM, [‡] JSweet@ICS.UCI.EDU.

# The Rand MH Message Handling System:

# Tutorial


## Acknowledgements

The MH system described herein is based on the original Rand MH system. It has been extensively developed (perhaps too much so) by Marshall Rose and John Romine at the University of California, Irvine. Einar Stefferud, Jerry Sweet, and Terry Domae provided numerous suggestions to improve the UCI version of MH.

Parts of this document are taken from a Rand tutorial [SPayn85] by Sue Payne.


## Disclaimer

The Regents of the University of California issue the following disclaimer concerning the UCI version of MH:

> "Although each program has been tested by its contributor, no warranty, express or implied, is made by the contributor or the University of California, as to the accuracy and functioning of the program and related program material, nor shall the fact of distribution constitute any such warranty, and no responsibility is assumed by the contributor or the University of California in connection herewith."

## Scope

This document assumes that you have no knowledge of MH. However, to use MH you should have some familiarity with the UNIX[1] operating system, particularly with the way commands are given, how files are named, the jargon (e.g. *shell*, *argument*, *home directory*, *pathname*), and how to use a text editor (such as *ex*, *vi*, or *emacs*).

This tutorial covers only basic material. For additional information about MH, consult the *User's Manual* [MRose85a]. Other documents of possible interest to you include *The UCI BBoards Facility* [MRose84] and the *MH Administrator's Guide* [MRose85b].

[1] UNIX is a trademark of AT&T Bell Laboratories.

## How To Use This Tutorial

Different typefaces and symbols are used in this document to denote the kinds of things you (the user) must type on your keyboard.

1. The names of programs are given in *text italics*:

   *comp*

2. Arguments to programs are given in `typewriter style`, delimited by single-quotes:

   `'msgs'`

3. UNIX pathnames are given in *slanted roman*:

   */usr/uci/*

4. Text giving a full example is presented in `typewriter style`:

   `comp␣-editor␣vi`

   The "␣" glyph is used to indicate an explicit space (the kind you make with the space bar on your keyboard).

## Introduction

With MH you can send messages to other people on your system and read messages that other people send to you. Depending on how things have been set up on your system, it may be possible for you to send messages to people on remote systems. You can also reply to messages that you have received, review them, organize them in *folders*, and delete them.

MH differs from other mail programs in that it is composed of many small programs instead of just one very large program. Among new users this sometimes causes some confusion along the lines of "what program do I run?" With MH, you use the shell to invoke one program at a time. This means that when you handle mail, the entire power of the shell is at your disposal in addition to the facilities that MH provides. In the beginning, this may not make much sense or may not seem important. However, we have found that as new users of MH gain experience, they find this style of interface to be very useful.

## Summary

The most minimal list of MH commands that you can get by with is:

*inc* - incorporate mail (get new mail)

*show* - show the first message

*next* - show the next message

*prev* - show the previous message

*comp* - compose a new message to send

*repl* - reply to a received message

*Comp* and *repl* give enough prompting possibly to get you along. However, it is suggested that you take the time to peruse this tutorial before leaping into things.

## Messages and Folders

A message takes the form of a memorandum, and is composed of two major parts: a *header*, which contains such information as ``To'' and ``From'' addresses, ``Subject'', ``Date'', etc.; and the *body*, which is the actual text of the message. Each *component* in the header starts with a keyword followed by a colon and additional information. For example, in the message:

```
Date:  10 Oct 84 17:41:14 EDT (Wed)
To:  News@udel-dewey
Subject:  UCI Software Talk
From:  UCI Portal (agent:  Marshall Rose) <uci@udel-dewey>

This is the text.
```

there are four header items, and one line of text in the body. Note that a blank line separates the body from the headers.

MH stores a message as an ordinary file in a UNIX directory. This directory is called a *folder*. If you choose to keep and organize your messages, you may create as many folders as you wish. There is no limit as to the number of messages in a folder. Typically messages are numbered from 1 up. All of your personal folders, along with some other information that MH needs to know, are kept in a special directory called *Mail* under your home directory. Normally, MH manages these files and directories automatically, so you needn't muck around with them directly unless you really want to.

You won't have any folders until somebody sends mail to you, as a rule. If you are anxious to try out MH, but no one has sent you mail yet, try sending mail to yourself to start out with.

## Reading New Mail

When you are notified that you have mail (usually when you log in), perhaps with the message

```
You have mail.
```

then you know that messages are waiting in your *maildrop*. To read these messages, you first have to *incorporate* the mail into your "in-box" by typing the command:

```
inc
```

This incorporates the new mail from your mail drop to your in-box, which is a folder named (naturally enough) `'+inbox'`. As *inc* incorporates your new mail, it generates a *scan listing* of the mail:

```
Incorporating new mail into inbox...

2+  10/10  WESTINE%USC-ISIF  RFC 916 Now Available    <<A new Request for Co
3   10/10  G B Reilly        Gosling EMACS manual     <<Marshall, I am lookin
4   10/11  WESTINE%USC-ISIF  Internet Monthly Report
```

Each time *inc* is invoked, any new messages are added to the end of your ``'+inbox'' folder.

To read the first message, use the *show* command:

```
show
```

This displays the current message. To read each subsequent message, use the *next* command:

```
next
```

If you want to back up, the command *prev* shows the previous message. Another way to read your messages is to name them all at once:

```
show␣all
```

This command displays them all, one after the other. The `'all'` argument to *show* above might also be replaced with `'next'` or `'prev'`, as in

```
show␣next
show␣prev
```

which are respectively equivalent to the *next* and *prev* commands.

If you have had occasion to type *inc* more than once, then you will find that ``show␣all'' is showing not only the new messages, but also the old messages that you've already seen. Therefore, you might find it better to use

```
show␣cur-last
```

instead. This command displays messages from the current message (`cur`) to the last message (`last`). Each time *inc* is invoked, it makes the first new message the current message. It should be noted here that the name `all` given in a previous example is equivalent to the *message range* `first-last`, where `first` is the name of the first message in `+inbox`. Also, ``show'' by itself is equivalent to

```
show␣cur
```

As mentioned earlier, with the UNIX shell as your interface to MH, it becomes easy to list a message on a line printer or to another file. For example,

```
show␣all␣|␣lpr
```

lists all the messages in the current folder to the line printer.

To summarize, the preceding has introduced these important concepts: *folders* (in particular, the `+inbox` folder), *messages*, *message names* (e.g. `prev`, `next`, `cur`, `last`), and *message ranges* (e.g. `cur-last`, `all`). More will be said about folders and messages in succeeding sections.

## Sending Messages

To send a message, you compose a message *draft*, either by replying to a message that someone sent to you, or by creating a draft from scratch. The *send* command is used **after** completing the final draft of a message, in the same way that you mail a paper letter only after you are finished writing it. This is a common source of confusion among new MH users who may have had experience with other mail systems.

This section discusses how to originate messages and how to reply to messages that were previously received, along with a word or two about addresses.

### *Originating Messages*

To create a message draft from scratch, use the *comp* program. You will be prompted for the header components and then the body of the message. If you make a mistake, you may correct it later with a text editor. The draft will be sent only if you give an explicit *send* command, so you do not have to worry about the draft getting away from you prematurely.

To start, you simply type:

```
comp
```

**To:** First, the prompt `'To:'` appears. Here you type the address of the person to whom you wish the message sent. If this person is on the same computer system as you, then that person's login ID should serve as the address (e.g. `'mrose'` or `'jsweet'`).

Here we digress briefly to discuss addresses. A full discussion of addresses is beyond the scope of this tutorial, but it should be mentioned that there are other kinds of addresses besides login IDs. To send messages to people on remote systems, the usual way is to type `'login-id@host'` in the `'To:'` component, as in `'MRose@UCI-ICSA'`. Examples of `'host'` names at UCI include `'uci-icsa'`, `'uci-icse'`, and `'uci-cip1'`. Upper and lower case letters may be used interchangeably. Sometimes a person's last name (e.g. `'Rose'`, `'Sweet'`) can be used instead of a login ID, but this cannot be relied upon in a world without unique surnames.

**cc:** After you have given an address to the `'To:'` prompt, you are prompted for the `'cc:'` ("carbon copy"–an archaism) address. It is customary, but not required, to put your own address here so that you get a copy of the message when it is sent.

To put more than one address in the `'To:'` and `'cc:'` components, just use a comma (",") between each address on a line.

**Subject:** The third prompt is for the `'Subject:'` component. Here a line of any descriptive text will do. Once you have typed a line of text, a dashed line is printed, and you are then expected to type the body of the message. End the body with EOT (usually CTRL-D).

An example of a complete message draft, as it appears on your screen, might be:

```
To:  News
cc:  farber, mrose
Subject:  UCI Software Talk
--------
A presentation on the UCI software suite, including
the Rand/UCI Mail Handling System (MH), will be given
in CS220 on October 31st at 2:30 PM. Refreshments
will be served afterward.

/mtr
^D
```

(The "^D" does not appear in the draft.)

At this point, you are asked

```
What␣now?
```

This is known as being at `What now?` level. For now, there are probably only four options that will interest you:

*edit* - edit the draft

*list* - list the draft on your screen

*quit* - quit, without sending the draft

*send* - send the draft, then quit

All of these options take various arguments, but only *edit* really needs an argument.

**Edit:** The *edit* option will let you edit the draft before sending it. If your favorite text editor is *vi*, then you would use the *edit* option as:

```
edit␣vi
```

Just specifying *edit* with no argument will only let you append text to the body of the message draft. Another editor (e.g. *vi*, *ex*, *emacs*) should really be run to finish the draft up. When you leave the editor, you will come back to the `What now?` level, where you can re-edit the draft, send it, list it, or simply quit without sending the draft at all.

Caution: while in the editor, you should not delete colons in the headers or change the spelling of `'To:'`, `'cc:'`, or `'Subject:'`; and do not leave blank lines between these lines. Feel free to change the addresses that you typed previously, or to add these lines if they are missing. Do not delete the dashes that separate the header lines from the text of the message. You should not add additional header lines unless you understand precisely what you are doing. This means particularly that you should not type or fill in a `'From:'` line. When the message is sent, the system automatically adds this line. Also, you should not type a `'Date:'` line in the header. When the message is sent, the system automatically adds the current date and time.

**Quit:** If you *quit* without sending the draft, the draft is saved in a file called *Mail/draft* under your home directory. This file can be recalled later using the `'-use'` argument to *comp*:

```
comp␣-use
```

The `What now?` level will permit you to do further editing and to send the final draft when you are ready.

**Send:** When it is time to send the draft on its way, use the *send* option by itself. If there are any problems with the draft (for example, if one or more of the people whom you specified in the `'To:'` and `'cc:'` components do not exist) then you will be notified at this time.

*Replying to Messages*

To reply to a message, use the *repl* command. For example,

```
repl
```

creates a reply to the current message. You may also reply to a specific message (other than the current one) by giving a *message number* (e.g. `'1'`, `'4'`, etc.) or a *message name* (e.g. `'first'`, `'last'`, `'prev'`):

```
repl␣prev
```

We haven't really introduced message numbers yet. They will be discussed in the next section.

The process of replying to a message is very similar to composing a message from scratch (see the previous section), but *repl* conveniently constructs and displays the header of the reply draft for you. You need only type in the text of the reply. An EOT (usually CTRL-D) indicates that you are done typing. If you make a mistake, you may correct it later with a text editor. The draft will be sent only if you give an explicit *send* command, so you do not have to worry about the draft getting away from you prematurely.

An example of a complete reply draft, as it appears on your screen might be:

```
To:  MRose
cc:  JSweet
Subject:  Re:  UCI Software Talk
In-reply-to:  Your message of 10 Oct 84 18:15:08 PDT (Wed).
--------
I'll be there.
-jns
^D
```

(The "^D" does not appear in the draft.)

At this point, you are asked

```
What␣now?
```

This is known as being at `What now?` level. Refer to the previous section regarding how to edit, display, or send the draft at this point.

As with *comp*, if you *quit* without sending the reply draft, the draft is saved in a file called *Mail/draft* under your home directory. This file can be recalled later using the `'-use'` argument to *comp*:

```
comp -use
```

The `What now?` level will permit you to do further editing and to send the final draft when you are ready.

## Scanning Messages

The scan listing created by *inc* shows the *message number*, the date on which the message was sent, the sender, and the subject of the message. If there is sufficient space remaining on the line, the beginning of the text of the message is displayed as well, preceded by two left angle brackets (" <<"). An example of a scan listing is:

```
1+  10/10  WESTINE%USC-ISIF  RFC 916 Now Available    <<A new Request for Co
2   10/10  G B Reilly        Gosling EMACS manual     <<Marshall, I am lookin
3   10/11  WESTINE%USC-ISIF  Internet Monthly Report
```

Note that all messages have message numbers.

To generate your own scan listing, use the *scan* program. Typing simply

```
scan
```

will list all the messages in the current folder. To scan a subset of these messages, you can specify the numbers of the messages that you consider interesting, e.g.,

```
scan 2 3
```

Message names may be specified in addition to discrete message numbers. The built-in message names recognized by MH are:

all: all messages in the folder (`'first-last'`)

first: the first message in the folder

last: the last message in the folder

prev: the message immediately before the current message

cur: the current message

next: the message immediately after the current message

Message ranges may be specified in addition to discrete message numbers or names by separating the beginning and final message numbers with a dash ("-"). For example,

```
scan 5-10
```

scans messages 5 through 10 inclusive. A range of messages may also be specified by separating a beginning message number and a relative number of messages with a colon (":"). For example,

```
scan last:3
```

scans the last three messages in the folder. Similarly,

```
scan first:3
```

scans the first three messages in the folder;

```
scan next:3
```

scans the next three messages;

```
scan cur:3
```

scans the three messages beginning from the current message;

```
scan 100:4
```

scans four messages beginning from message number 100.

To summarize, the important concepts that have been discussed in the section are: *message ranges*, *message numbers*, and *message names*. When an MH command is described as taking a `'msg'` argument, it accepts either a message name or a message number. Most MH commands are described as taking `'msgs'` arguments, meaning that more than one message or message range is accepted.

## Deleting Messages

To delete a message, use the *rmm* program. By default, *rmm* deletes the current message, but you can give *rmm* a list of messages to be removed as well. There is no corresponding "*unrmm*" program, but clever users with a need will find out how to change the way *rmm* works so that it simply moves messages to another folder (say, `'+wastebasket'`).

## Filing Messages

The possibility of having folders other than ‘‘+inbox’’ has been mentioned previously. The methods for moving messages between folders and manipulating folders are discussed here.

The *refile* command moves messages from a *source folder* to one or more *destination folders*. By default, the current message is moved from the *current folder* (typically ‘+inbox’) to another folder specified as an argument to *refile*. For example,

```
refile␣+todo
```

moves the current message from the current folder to the folder ‘‘+todo’’. To move messages from a folder other than the current folder, use the ‘-src +folder’ switch, as in

```
refile␣-src␣+todo␣last␣+save␣+notes
```

which moves the last message in the ‘‘+todo’’ folder to the folders ‘‘+save’’ and ‘‘+notes’’. Note that this operation is a *move*, not a *copy*; it removes the message from the source folder. To keep a copy in the source folder as well, use the ‘-link’ switch

```
refile␣-link␣-src␣+todo␣last␣+save␣+notes
```

Whenever a folder argument is given to an MH command, that folder becomes the *current folder*. To find out which folder is current, use the command

```
folder
```

The *inc* command sets the current folder back to ‘+inbox’ by default. To find out about all of a user's folders, use the command

```
folders
```

Since folders can contain other folders, the command

```
folders␣-recurse
```

will recursively examine each folder for you.

To set the current folder, without doing anything else, use the *folder* program with a folder argument. Hence,

```
folder␣+inbox
```

makes ‘‘+inbox’’ the current folder.

After a using *rmm* and *refile* on a folder a number of times, there tend to be gaps in the numbering sequence. To compress the numbers for the all messages in a folder, use

```
folder␣-pack
```

## The Profile

You can customize the MH environment by editing your *.mh_profile* file. Although there are lots of options, here are the most useful:

Editor: lists the default editor that *comp* and *repl* should use. The default is

```
editor:␣prompter
```

but another editor might be preferred.

*editor*-next: lists the editor that should be used after the last edit with *editor*. Hence, if you have a profile entry

```
prompter-next:␣vi
```

after editing a draft with *prompter*, and being at `What now?` level, you could say ``edit'' (instead of ``edit vi'') to continue to edit the draft with *vi*.

Msg-Protect: Whenever MH creates a message (for example, with *inc*), this is the octal protection mode that the message is created with. The default is

```
Msg-Protect:␣644
```

This protection mode permits all other users on the system to read your messages. To maintain privacy, the mode 600 should be used. Note that changing the mode in the profile does not change the modes of messages that have been created already. Use the UNIX command *chmod* to change the modes of your existing messages.

Folder-Protect: Whenever MH creates a folder (for example, with *refile*), this is the octal mode that the folder is created with. The default is

```
Folder-Protect:␣711
```

This mode permits other users on the system to make access to specific messages in your folders. To maintain stricter privacy, the mode 700 should be used.

*program*: Each MH program that reads user's *.mh_profile* file looks for an entry beginning with its own name to determine its initial defaults. For example, if you want the default editor for *repl* to be *emacs*, the line

```
repl:␣-editor␣emacs
```

is sufficient. Command line arguments tend to override profile settings. Given the profile setting for *repl* above, if you invoked *repl* with

```
repl␣-editor␣vi
```

*repl* would use the *vi* editor instead of *emacs*.

signature: When MH posts mail for you, it looks for this profile entry for your "real world" name. For example,

```
signature:␣Marshall␣Rose
```

The contents of the ‘‘`signature:`’’ entry in the profile should be a simple phrase, with no embedded periods (e.g. "Marshall T. Rose").

Note that your profile resembles the header portion of a message. Be sure that it is properly formatted by placing a colon after each entry name, and keep each entry on a single line.

## Conventions

Now let's summarize the conventions that MH programs use:

1. Any MH command that deals with messages can be given a ‘`+folder`’ argument to say which folder to use. However, only one ‘`+folder`’ argument may be given per command in most cases.

2. If an MH command accepts a ‘`msgs`’ argument, then any number of messages can be given to the command. The MH command will expand all the ranges and process each message, starting with the lowest numbered one and working its way to the message with the highest number.

3. If an MH command accepts a ‘`msg`’ argument, then at most one message can be given.

4. Switches (options) to MH commands start with a dash. Unlike the standard UNIX convention, each switch consists of more than one character, for example ‘`-header`’. To minimize typing, only a unique abbreviation of the switch need be typed; thus for ‘`-header`’, ‘`-hea`’ is probably sufficient, depending on the other switches accepted by the command.

5. All MH commands have a '-help' switch, which *must* be spelled out fully. When an MH command encounters the '-help' switch, it prints out the syntax of the command, the switches that it accepts, and version information. In the list of switches, parentheses indicate required characters. For example, all '-help' switches will appear as '-(help)', indicating that no abbreviation is accepted.

6. Many MH switches have both on and off forms, such as '-format' and '-noformat'. In these cases, the last occurrence of the switch on the command line determines the setting of the option.

7. All MH commands that read your MH profile operate the same way: first, the profile is consulted for an entry matching the name with which the command was invoked; second, if such an entry was found, then the command immediately uses the arguments listed; third, any arguments on the command line are then interpreted. Since most switches have both on and off forms, it's easy to customize the default options for each MH command in the *.mh_profile*, and to override those defaults on the command line.

### Online Documentation

Each MH program has its own UNIX manual entry. For example, to get information about *comp*, type

```
man␣comp
```

The manual entry for *mh*(1) lists all MH commands, while the manual entry for *mh-chart*(1) lists the syntax and switches for all MH commands.

In addition, here are a few other manual entries might be found useful:

*mh-alias*(5) to find out how aliases in MH work;

*mh-mail*(5) to find out how MH stores and interprets messages (this manual entry explains all of the standard header components);

*mh-profile*(5) to find out about the MH user-environment.

The manual pages for MH are in the standard UNIX format, but contain additional sections unique to MH. Here's a summary of the sections one might find in an MH manual entry:

NAME command name and one-line description.

SYNOPSIS syntax of the command.
All commands accept a '-help' switch.

DESCRIPTION semantics of the command.

FILES files used by the command
Almost always this includes *.mh_profile*.

PROFILE COMPONENTS entries in the *.mh_profile* used by the command; these do not include the profile entry for the command itself.

SEE ALSO other UNIX manual entries (usually MH programs) that are related to this command.

DEFAULTS default arguments for the command
If the command takes a '`+folder`' argument, this defaults to the current folder. If the command takes a '`msg`' argument, this defaults to the current message. If the command takes a '`msgs`' argument, this defaults to the current message or all messages, depending on which one makes more sense.

CONTEXT changes to your MH context made by the command.

HINTS Helpful hints discussing the easy way to do things.

HISTORY A historical perspective on why MH works the way it does.

BUGS Too embarrassing to mention.
Just kidding.

Obviously, not all MH manual entries may have all of these sections.

### Reporting Problems

If problems are encountered with an MH program, the problems should be reported to the local maintainers of MH. When doing this, the name of the program should be reported, along with the version information for the program. To find out what version of an MH program is being run, invoke the program with the '`-help`' switch. In addition to listing the syntax of the command, the program will list information pertaining to its version. This information includes the version of MH, the host it was generated on, the date the program was loaded, and the configuration options in effect when MH was generated. For example,

```
version:  MH 6.1 #1[UCI] (gremlin) of Wed Nov 6 01:13:53 PST 1985
options:  [BSD42] [MHE] [NETWORK] [SENDMTS] [MMDFII] [SMTP] [POP]
```

The ''`6.1␣#1[UCI]`'' indicates that the program is from the UCI mh.6 version of MH. The program was generated on the host ''`gremlin`'' on ''`Wed Nov 6 01:13:53 PST 1985`''. It's usually a good idea to send the output of the '`-help`' switch along with your report.

If there is no local MH maintainer, try the address `Bug-MH`. If that fails, use the Internet mailbox `Bug-MH@UCI.ARPA`.

## More on MH

There are myriad aspects of MH that this tutorial hasn't touched upon. Here are a few to whet your appetite:

1. user-defined sequences
   Define *meaningful* message names and shorten type-in considerably (see *pick*(1) for details).

2. draft folders
   Maintain a folder of drafts so that more than one draft can be edited at a time, and allow a draft to be edited over several UNIX sessions independently of other drafts (see the **Advanced Features** section of the MH user's manual for details).

3. draft pushing
   Post a draft in the background and immediately free your terminal for other activities (see the **Advanced Features** section of the MH user's manual for details).

4. aliases
   Maintain one or more alias files containing the addresses of the people frequently (or infrequently) sent to. This lets you shorten type-in of addressees and saves you from looking up their addresses all the time. (see *mh-alias*(5) for details).

## References

[MRose84] M.T. Rose. The Rand MH Message Handling System: The UCI BBoards Facility. Department of Computer and Information Sciences, University of Delaware (October, 1984).

[MRose85a] M.T. Rose, J.L. Romine. The Rand MH Message Handling System: User's Manual. UCI Version. Department of Information and Computer Science, University of California, Irvine (January, 1985).

[MRose85b] M.T. Rose. The Rand MH Message Handling System: Administrator's Guide. UCI Version, MH Classic. Northrop Corporation, Research and Technology Center (July, 1985).

[SPayn85] S. Payne MH5: Electronic Mail. Rand Note #N-2281-RCC. The Rand Computation Center, Rand, 1700 Main St., Santa Monica, CA 90406-2138 (May, 1985).

# Contents


| | Page |
|---|---|
| Acknowledgements | 1 |
| Disclaimer | 1 |
| Scope | 1 |
| How To Use This Tutorial | 2 |
| Introduction | 2 |
| Summary | 3 |
| Messages and Folders | 3 |
| Reading New Mail | 4 |
| Sending Messages | 5 |
| Originating Messages | 5 |
| Replying to Messages | 8 |
| Scanning Messages | 9 |
| Deleting Messages | 10 |
| Filing Messages | 11 |
| The Profile | 12 |
| Conventions | 13 |
| Online Documentation | 14 |
| Reporting Problems | 15 |
| More on MH | 16 |
| References | 17 |


This document (version #2.8) was TEXset April 12, 1990 with DISS.STY v103.