An MH Primer

Matt Midboe <matt@garply.com >
Revision: 43184

FreeBSD is a registered trademark of the FreeBSD Foundation.

Motif, OSF/1, and UNIX are registered trademarks and IT DialTone and The Open Group are trademarks of The Open Group in the United States and other countries.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this document, and the FreeBSD Project was aware of the trademark claim, the designations have been followed by the "™" or the "®" symbol.

v1.0, 16 January 1996 by .

Abstract

This document contains an introduction to using MH on FreeBSD

Table of Contents

1. Introduction 1
2. Reading Mail 2
3. Folders and Mail Searching 6
4. Sending Mail 9

1. Introduction

MH started back in 1977 at the RAND Corporation, where the initial philosophies behind MH were developed. MH is not so much a monolithic email program but a philosophy about how best to develop tools for reading email. The MH developers have done a great job adhering to the KISS principle: Keep It Simple Stupid. Rather than have one large program for reading, sending and handling email they have written specialized programs for each part of your email life. One might liken MH to the specialization that one finds in insects and nature. Each tool in MH does one thing, and does it very well.

Beyond just the various tools that one uses to handle their email MH has done an excellent job keeping the configuration of each of these tools consistent and uniform. In fact, if

you are not quite sure how something is supposed to work or what the arguments for some command are supposed to be, then you can generally guess and be right. Each MH command is consistent about how it handles reading the configuration files and how it takes arguments on the command line. One useful thing to remember is that you can always add a `-help` to the command to have it display the options for that command.

The first thing that you need to do is to make sure that you have installed the MH package on your FreeBSD machine. If you installed from CDROM you should be able to execute the following to load MH:

```
# pkg_add -/cdrom/packages/mh-6.8.3.tgz
```

You will notice that it created a `/usr/local/lib/mh` directory for you as well as adding several binaries to the `/usr/local/bin` directory. If you would prefer to compile it yourself then you can anonymous ftp it from ftp.ics.uci.edu or louie.udel.edu.

This primer is not a full comprehensive explanation of how MH works. This is just intended to get you started on the road to happier, faster mail reading. You should read the manual pages for the various commands. You might also want to read the comp.mail.mh newsgroup. Also you can read the FAQ for MH. The best resource for MH is Jerry Peek's MH & nmh: Email for Users & Programmers.

2. Reading Mail

This section covers how to use `inc`, `show`, `scan`, `next`, `prev`, `rmm`, `rmf`, and `msgchk`. One of the best things about MH is the consistent interface between programs. One thing to keep in mind when using these commands is how to specify message lists. In the case of `inc` this does not really make any sense but with commands like `show` it is useful to know.

A message list can consist of something like *`23 20 16`* which will act on messages 23, 20 and 16. This is fairly simple but you can do more useful things like *`23-30`* which will act on all the messages between 23 and 30. You can also specify something like *`cur:10`* which will act on the current message and the next 9 messages. The *`cur`*, *`last`*, and *`first`* messages are special messages that refer to the current, last or first message in the folder.

2.1. `inc`, `msgchk`—read in your new email or check it

If you just type in **`inc`** and hit return you will be well on your way to getting started with MH. The first time you run `inc` it will set up your account to use all the MH defaults and ask you about creating a `Mail` directory under your HOME directory. If you have mail waiting to be downloaded you will see something that looks like:

```
  29  01/15 Doug White         Re: Another Failed to boot ↺
problem<<On Mon, 15 J
```

```
  30  01/16 -"Jordan K. Hubbar  Re: FBSD 2.1<<> Do you want a ↺
library instead of
  31  01/16 Bruce Evans        Re: location of bad144 table<<>> >It ↺
would appea
  32  01/16 -"Jordan K. Hubbar  Re: video is up<<> Anyway, mrouted ↺
won't run, ev
  33  01/16 Michael Smith      Re: FBSD 2.1<<Nate Williams stands ↺
accused of sa
```

This is the same thing you will see from a `scan` (see Section 2.3, "`scan`—shows you a scan of your messages"). If you just run `inc` with no arguments it will look on your computer for email that is supposed to be coming to you.

A lot of people like to use POP for grabbing their email. MH can do POP to grab your email. You will need to give `inc` a few command line arguments.

```
% inc --host mail.pop.org --user username --norpop
```

That tells `inc` to go to *`mail.pop.org`* to download your email, and that your username on their system is *`username`*. The `-norpop` option tells `inc` to use plain POP3 for downloading your email. MH has support for a few different dialects of POP. More than likely you will never ever need to use them though. While you can do more complex things with `inc` such as audit files and scan format files this will get you going.

The `msgchk` command is used to get information on whether or not you have new email. `msgchk` takes the same `-host` and `-user` options that `inc` takes.

2.2. `show`, `next` and `prev`—displaying and moving through email

`show` is to show a letter in your current folder. Like `inc`, `show` is a fairly straightforward command. If you just type **`show`** and hit return then it displays the current message. You can also give specific message numbers to show:

```
% show 32 45 56
```

This would display message numbers 32, 45 and 56 right after each other. Unless you change the default behavior `show` basically just does a `more` on the email message.

`next` is used to move onto the next message and `prev` will go to the previous message. Both commands have an implied `show` command so that when you go to the next message it automatically displays it.

2.3. `scan`—shows you a scan of your messages

`scan` will display a brief listing of the messages in your current folder. This is an example of what the `scan` command will give you.

```
  30+ 01/16 Jordan K. Hubbar   Re: FBSD 2.1<<> Do you want a ↺
library instead of
  31  01/16 Bruce Evans        Re: location of bad144 table<<>> >It ↺
would appea
  32  01/16 Jordan K. Hubbar   Re: video is up<<> Anyway, mrouted ↺
won't run, ev
  33  01/16 Michael Smith      Re: FBSD 2.1<<Nate Williams stands ↺
accused of sa
```

Like just about everything in MH this display is very configurable. This is the typical default display. It gives you the message number, the date on the email, the sender, the subject line, and a sentence fragment from the very beginning of the email if it can fit it. The + means that message is the current message, so if you do a `show` it will display that message.

One useful option for scan is the `-reverse` option. This will list your messages with the highest message number first and lowest message number last. Another useful option with `scan` is to have it read from a file. If you want to scan your incoming mailbox on FreeBSD without having to `inc` it you can do `scan -file /var/mail/username .` This can be used with any file that is in the mbox format.

2.4. `rmm` and `rmf`—remove the current message or folder

`rmm` is used to remove a mail message. The default is typically to not actually remove the message but to rename the file to one that is ignored by the MH commands. You will periodically need to go through and physically delete the "removed" messages.

The `rmf` command is used to remove folders. This does not just rename the files but actually removes the from the hard drive so you should be careful when you use this command.

2.5. A typical session of reading with MH

The first thing that you will want to do is `inc` your new mail. So at a shell prompt just type in `inc` and hit return.

```
% inc
Incorporating new mail into inbox...

  36+ 01/19 Stephen L. Lange   Request...<<Please remove me as ↺
contact for pind
  37  01/19 Matt Thomas        Re: kern/950: Two PCI bridge chips ↺
fail (multipl
  38  01/19 Amancio Hasty Jr   Re: FreeBSD and VAT<<>>> Bill Fenner ↺
said: > In
%
```

This shows you the new email that has been added to your mailbox. So the next thing to do is `show` the email and move around.

```
% show
Received: by sashimi.wwa.com (Smail3.1.29.1 #2)
        id m0tdMZ2-001W2UC; Fri, 19 Jan 96 13:33 CST
Date: Fri, 19 Jan 1996 13:33:31 --0600 (CST)
From: -"Stephen L. Lange" <stvlange@wwa.com>
To: matt@garply.com
Subject: Request...
Message-Id: <Pine.BSD.3.91.960119133211.824A-100000@sashimi.wwa.com>
Mime-Version: 1.0
Content-Type: TEXT/PLAIN; charset=US-ASCII


Please remove me as contact for pindat.com

% rmm
% next
Received: from localhost (localhost [127.0.0.1]) by ↺
whydos.lkg.dec.com (8.6.11/8
.6.9) with SMTP id RAA24416; Fri, 19 Jan 1996 17:56:48 GMT
Message-Id: <199601191756.RAA24416@whydos.lkg.dec.com>
X-Authentication-Warning: whydos.lkg.dec.com: Host localhost didn't ↺
use HELO pro
tocol
To: hsu@clinet.fi
Cc: hackers@FreeBSD.org
Subject: Re: kern/950: Two PCI bridge chips fail (multiple ↺
multiport ethernet
 boards)
In-Reply-To: Your message of -"Fri, 19 Jan 1996 00:18:36 +0100."
             <199601182318.AA11772@Sysiphos>
X-Mailer: exmh version 1.5omega 10/6/94
Date: Fri, 19 Jan 1996 17:56:40 +0000
From: Matt Thomas <matt@lkg.dec.com>
Sender: owner-hackers@FreeBSD.org
Precedence: bulk


This is due to a typo in pcireg.h (to
which I am probably the guilty party).
```

The `rmm` removed the current message and the `next` command moved me on to the next message. Now if I wanted to look at ten most recent messages so I could read one of them here is what I would do:

```
% scan last:10
  26  01/16 maddy             Re: Testing some stuff<<yeah, well, ↺
Trinity has
  27  01/17 Automatic digest   NET-HAPPENINGS Digest -- 16 Jan 1996 ↺
to 17 Jan 19
  28  01/17 Evans A Criswell   Re: Hey dude<<>From ↺
matt@tempest.garply.com Tue
  29  01/16 Karl Heuer         need configure/make volunteers<<The ↺
FSF is looki
```

```
  30  01/18 Paul Stephanouk    Re: [alt.religion.scientology] Raw ↺
Meat (humor)<
  31  01/18 Bill Lenherr       Re: Linux NIS Solaris<<--- On Thu, ↺
18 Jan 1996 1
  34  01/19 John Fieber        Re: Stuff for the email section?<<On ↺
Fri, 19 Jan
  35  01/19 support@foo.garpl  [garply.com #1138] parlor<<Hello. ↺
This is the Ne
  37+ 01/19 Matt Thomas        Re: kern/950: Two PCI bridge chips ↺
fail (multipl
  38  01/19 Amancio Hasty Jr   Re: FreeBSD and VAT<<>>> Bill Fenner ↺
said: > In
%
```

Then if I wanted to read message number 27 I would do a **`show 27`** and it would be displayed. As you can probably tell from this sample session MH is pretty easy to use and looking through emails and displaying them is fairly intuitive and easy.

3. Folders and Mail Searching

Anybody who gets lots of email definitely wants to be able to prioritize, stamp, brief, debrief, and number their emails in a variety of different ways. MH can do this better than just about anything. One thing that we have not really talked about is the concept of folders. You have undoubtedly come across the folders concept using other email programs. MH has folders too. MH can even do sub-folders of a folder. One thing you should keep in mind with MH is that when you ran `inc` for the first time and it asked you if it could create a `Mail` directory it began storing everything in that directory. If you look at that directory you will find a directory named `inbox`. The `inbox` directory houses all of your incoming mail that has not been thrown anywhere else.

Whenever you create a new folder a new directory is going to be created underneath your MH `Mail` directory, and messages in that folder are going to be stored in that directory. When a new email message comes, it is thrown into your `inbox` directory with a file name that is equivalent to the message number. So even if you did not have any of the MH tools to read your email you could still use standard UNIX® commands to munge around in those directories and just more your files. It is this simplicity that really gives you a lot of power with what you can do with your email.

Just as you can use message lists like *`23 16 42`* with most MH commands there is a folder option you can specify with just about every MH command. If you do a `scan +freebsd` it will scan your `freebsd` folder, and your current folder will be changed to `freebsd`. If you do a `show +freebsd 23 16 42`, `show` is going to switch to your `freebsd` folder and display messages 23, 16 and 42. So remember that *`+folder`* syntax. You will need to make sure you use it to make commands process different folders. Remember you default folder for mail is `inbox` so doing a `folder +inbox` should always get you back to your mail. Of course, in MH's infinite flexibility this can be changed but most places have probably left it as `inbox`.

3.1. `pick`—search email that matches certain criteria

`pick` is one of the more complex commands in the MH system. So you might want to read the pick(1) man page for a more thorough understanding. At its simplest level you can do something like

```
% pick --search pci
15
42
55
56
57
```

This will tell `pick` to look through every single line in every message in your current folder and tell you which message numbers it found the word `pci` in. You can then `show` those messages and read them if you wish or `rmm` them. You would have to specify something like `show 15 42 55-57` to display them though. A slightly more useful thing to do is this:

```
% pick --search pci --seq pick
5 hits
% show pick
```

This will show you the same messages you just did not have to work as hard to do it. The `-seq` option is really an abbreviation of `-sequence` and `pick` is just a sequence which contains the message numbers that matched. You can use sequences with just about any MH command. So you could have done an `rmm pick` and all those messages would be removed instead. You sequence can be named anything. If you run pick again it will overwrite the old sequence if you use the same name.

Doing a `pick -search` can be a bit more time consuming than just searching for message from someone, or to someone. So `pick` allows you to use the following predefined search criteria:

`-to`
: search based upon who the message is to

`-cc`
: search based on who is in the `Cc:` list

`-from`
: search for who sent the message

`-subject`
: search for emails with this subject

`-date`
: find emails with a matching date

--component
: search for any other component in the header. (i.e. `--reply-to` to find all emails with a certain reply-to in the header)

This allows you to do things like

```
% pick --to freebsd-hackers@FreeBSD.org --seq hackers
```

to get a list of all the email send to the FreeBSD hackers mailing list. `pick` also allows you to group these criteria in different ways using the following options:

- … `-and` …
- … `-or` …
- `-not` …
- `-lbrace` … `-rbrace`

These commands allow you to do things like

```
% pick --to freebsd-hackers --or --cc freebsd-hackers
```

That will grab all the email in your `inbox` that was sent to freebsd-hackers or cc'd to that list. The brace options allow you to group search criteria together. This is sometimes very necessary as in the following example

```
% pick --lbrace --to freebsd-hackers --and
  --not --cc freebsd-questions --rbrace --and --subject pci
```

Basically this says "pick (to freebsd-hackers and not cc'd on freebsd-questions) and the subject is pci". It should look through your folder and find all messages sent to the freebsd-hackers list that are not cc'd to the freebsd-questions list and contain "pci" in the subject line. Ordinarily you might have to worry about something called operator precedence. Remember in math how you evaluate from left to right and you do multiplication and division first and addition and subtraction second? MH has the same type of rules for `pick`. It is fairly complex so you might want to study the manual page. This document is just to help you get acquainted with MH.

3.2. `folder`, `folders`, `refile`—three useful programs for folder maintenance

There are three programs which are primarily just for manipulating your folders. The `folder` program is used to switch between folders, pack them, and list them. At its simplest level you can do a `folder +newfolder` and you will be switched into *`newfolder`*. From there on out all your MH commands like `comp`, `repl`, `scan`, and `show` will act on that `newfolder` folder.

Sometimes when you are reading and deleting messages you will develop "holes" in your folders. If you do a `scan` you might just see messages 34, 35, 36, 43, 55, 56, 57, 80. If you do a `folder -pack` this will renumber all your messages so that there are no holes. It does not actually delete any messages though. So you may need to periodically go through and physically delete `rmm`'d messages.

If you need statistics on your folders you can do a `folders` or `folder -all` to list all your folders, how many messages they have, what the current message is in each one and so on. This line of stats it displays for all your folders is the same one you get when you change to a folder with `folder +foldername`. A `folders` command looks like this:

```
                Folder      # of messages (  range  -); cur  msg  ↺
(other files)
              announce  has    1 message  (   1-   1).
                drafts  has   no messages.
             f-hackers  has   43 messages (   1-  43).
           f-questions  has   16 messages (   1-  16).
                inbox+ has   35 messages (   1-  38); cur=  37.
                lists  has    8 messages (   1-   8).
             netfuture  has    1 message  (   1-   1).
                   out  has   31 messages (   1-  31).
              personal  has    6 messages (   1-   6).
                  todo  has   58 messages (   1-  58); cur=   1.

                     TOTAL=  199 messages in 13 folders.
```

The `refile` command is what you use to move messages between folders. When you do something like `refile 23 +netfuture` message number 23 is moved into the `netfuture` folder. You could also do something like `refile 23 +netfuture/latest` which would put message number 23 in a subfolder called `latest` under the `netfuture` folder. If you want to keep a message in the current folder and link it you can do a `refile -link 23 +netfuture` which would keep 23 in your current `inbox` but also list in your `netfuture` folder. You are probably beginning to realize some of the really powerful things you can do with MH.

4. Sending Mail

Email is a two way street for most people so you want to be able to send something back. The way MH handles sending mail can be a bit difficult to follow at first, but it allows for incredible flexibility. The first thing MH does is to copy a components file into your outgoing email. A components file is basically a skeleton email letter with stuff like the `To:` and `Subject:` headers already in it. You are then sent into your editor where you fill in the header information and then type the body of your message below the dashed lines in the message. When you leave the editor, the `whatnow` program is run. When you are at the `What now?` prompt you can tell it to `send`, `list`, `edit`, `push`, and `quit`. Most of these commands are self-explanatory. So the message sending process involves copying

a component file, editing your email, and then telling the whatnow program what to do with your email.

4.1. comp, forw, reply—compose, forward or reply to a message to someone

The comp program has a few useful command line options. The most important one to know right now is the -editor option. When MH is installed the default editor is usually a program called prompter which comes with MH. It is not a very exciting editor and basically just gets the job done. So when you go to compose a message to someone you might want to use comp -editor /usr/bin/vi or comp -editor /usr/local/bin/pico instead. Once you have run *comp* you are in your editor and you see something that looks like this:

```
To:
cc:
Subject:
--------
```

You need to put the person you are sending the mail to after the To: line. It works the same way for the other headers also, so you would need to put your subject after the Subject: line. Then you would just put the body of your message after the dashed lines. It may seem a bit simplistic since a lot of email programs have special requesters that ask you for this information but there really is no point to that. Plus this really gives you excellent flexibility.

```
To:freebsd-rave@FreeBSD.org
cc:
Subject:And on the 8th day God created the FreeBSD core team
--------
Wow this is an amazing operating system. Thanks!
```

You can now save this message and exit your editor. You will see the What now? prompt and you can type in **send** or **s** and hit return. Then the FreeBSD core team will receive their just rewards. As I mentioned earlier, you can also use other commands at the What now? prompt. For example you can use quit, if you do not want to send the message.

The forw command is stunningly similar. The big difference being that the message you are forwarding is automatically included in the outgoing message. When you run forw it will forward your current message. You can always tell it to forward something else by doing something like forw 23 and then message number 23 will be put in your outgoing message instead of the current message. Beyond those small differences forw functions exactly the same as comp. You go through the exact same message sending process.

The repl command will reply to the current message, unless you give it a different message to reply to. repl will do its best to go ahead and fill in some of the email headers already. So you will notice that the To: header already has the address of the recipient

in there. Also the `Subject:` line will already be filled in. You then go about the normal message composition process and you are done. One useful command line option to know here is the `-cc` option. You can use *`all`*, *`to`*, *`cc`*, *`me`* after the `-cc` option to have `repl` automatically add the various addresses to the `Cc:` list in the message. You have probably noticed that the original message is not included. This is because most MH setups are configured to do this from the start.

4.2. `components`, and `replcomps`—components files for `comp` and `repl`

The `components` file is usually in `/usr/local/lib/mh` . You can copy that file into your MH Mail directory and edit to contain what you want it to contain. It is a fairly basic file. You have various email headers at the top, a dashed line and then nothing. The `comp` command just copies this `components` file and then edits it. You can add any kind of valid RFC822 header you want. For instance you could have something like this in your `components` file:

```
To:
Fcc: out
Subject:
X-Mailer: MH 6.8.3
X-Home-Page: http://www.FreeBSD.org/
-------
```

MH would then copy this components file and throw you into your editor. The `components` file is fairly simple. If you wanted to have a signature on those messages you would just put your signature in that `components` file.

The `replcomps` file is a bit more complex. The default `replcomps` looks like this:

```
%(lit)%(formataddr %<{reply-to}%?{from}%?{sender}%?{return-path}%>)\
%<(nonnull)%(void(width))%(putaddr To: -)\n%>\
%(lit)%(formataddr{to})%(formataddr{cc})%(formataddr(me))\
%<(nonnull)%(void(width))%(putaddr cc: -)\n%>\
%<{fcc}Fcc: %{fcc}\n%>\
%<{subject}Subject: Re: %{subject}\n%>\
%<{date}In-reply-to: Your message of -"\
%<(nodate{date})%{date}%|%(pretty{date})%>."%<{message-id}
            %{message-id}%>\n%>\
--------
```

It is in the same basic format as the `components` file but it contains quite a few extra formatting codes. The `%(lit)` command makes room for the address. The `%(formataddr)` is a function that returns a proper email address. The next part is `%<` which means if and the `{reply-to}` means the reply-to field in the original message. So that might be translated this way:

```
%<if {reply-to} the original message has a reply-to
then give that to formataddr, %? else {from} take the
from address, %? else {sender} take the sender address, %?
```

```
else {return-path} take the return-path from the original
message, %> endif.
```

As you can tell MH formatting can get rather involved. You can probably decipher what most of the other functions and variables mean. All of the information on writing these format strings is in the MH-Format manual page. The really nice thing is that once you have built your customized `replcomps` file you will not need to touch it again. No other email program really gives you the power and flexibility that MH gives you.