# Majordomo and MajorCool HOWTO


## John Archie



15 November 2000



This document is intended to guide a user through an installation of the Majordomo Mailing List Software and MajorCool. MajorCool is a utility for managing Majordomo lists via a CGI script; many people who are unfamiliar with Majordomo's text−based nature prefer the more user−friendly, web−based interface of MajorCool.

# Table of Contents

# 1. Introduction

This HOWTO is divided into several sections. The Sendmail portion is a general discussion about Majordomo and how Majordomo interfaces with Sendmail, as well as the various ways Majordomo can be set up and the consequences of such decisions. In contrast, the rest of the HOWTO is a tutorial guiding a user through a plain installation process of Majordomo. I recommend going over the generic installation process described in the sections after Sendmail, referencing the appropriate portions of the Sendmail section when necessary (the appropriate sections are mentioned in the appropriate places). Then, read the Sendmail section carefully and decide exactly how to configure your system. Finally, a List of Terms provides definitions for some of the more abstruse terms.

Also, if the official sites for downloading any of the software mentioned in this document are down, the tarballs can be found at my web site.

---

## 1.1. Credits

Thanks go out (in alphabetical order) to a few people for their invaluable help.

- Lee Archie for proofreading
- James Bruce and Bill Poston for the opportunity to set up my first permanent machine running Majordomo
- Joseph D. Sloan for reading the Sendmail portion and making helpful suggestions

---

## 1.2. References

Although I have tried to make this HOWTO as complete as possible, it is always a good idea to look at more than one source. Below is a list of the resources that I found helpful when trying to configure Majordomo for the first time.

Books:

- Bryan Costales with Eric Allman, *sendmail*. Cambridge: O'Reilly, 1997.
- Alan Schwartz, *Managing Mailing Lists*. Cambridge: O'Reilly, 1998.

Free resources:

- the documentation accompanying Sendmail especially `README.cf`
- the documentation accompanying Majordomo especially `INSTALL` and `NEWLIST`
- the `Majordomo-Users` Mailing List Archive.
- the documentation accompanying MajorCool

---

# 2. Sendmail

Since Majordomo is responsible for managing E–mail lists, Majordomo relies heavily on a MTA such as Sendmail. There are other MTA's such as Smail and Qmail out there; however, Sendmail is the oldest and most common. This section introduces the reader to the areas of Sendmail that are useful or necessary to configure when using Majordomo.

## 2.1. Aliases

The Sendmail aliases file (usually `/etc/aliases`) is used for making aliases for E–mail addresses. For example, once Majordomo is installed, usually an entry in the aliases file reads:

```
majordomo-owner:        jarchie
```

This entry means that all mail addressed to `majordomo-owner@host.com` will actually be sent to `jarchie@host.com`. Notice it is unnecessary to append the `@host.com` to `jarchie` because both users are on the same host. If it were desired to redirect the message to a different user on a different host, one would have to add the `@host.com` portion.

Another type of entry in the aliases file allows E–mail to be redirected to multiple addresses listed in a file:

```
testlist:               :include:/usr/local/majordomo-1.94.5/lists/testlist
```

This entry states that any message sent to `testlist@host.com` will be redirected to all the addresses listed in the file `/usr/local/majordomo-1.94.5/lists/testlist`. The `testlist` file might look something like this:

```
johnarchie@emeraldis.com
srobirds@yahoo.com
acreswell@geocities.com
```

Majordomo is able to add or remove addresses from a list by taking advantage of this feature. When a `subscribe` request is processed, the user's E–mail address is appended to the `testlist` file; when an `unsubscribe` request is processed, the user's E–mail address is removed from the `testlist` file. One can also add or remove addresses manually simply by editing the file with a text editor such as **vi**.

Since Majordomo needs to be able to process commands sent to it via E–mail, Sendmail must be able to execute the Majordomo program and pass the message to it. This is done by adding another type of entry to the `aliases` file:

```
majordomo:              "|/usr/local/majordomo-1.94.5/wrapper majordomo"
```

The program `/usr/local/majordomo-1.94.5/wrapper` is a wrapper (SUID `majordomo` and SGID `majordomo` or `daemon` depending on the configuration) that runs the Majordomo program. The quotation marks around the second part of the alias entry are used to tell Sendmail that this part of the entry is all one statement; the quotation marks would be unnecessary if there were not a space between `wrapper` and `majordomo`. The `|` is known as a "pipe"; it is used to tell Sendmail to send the E–mail to the wrapper via the standard input. (Since all the wrapper does here is to call **majordomo**, the E–mail is actually being sent to Majordomo.) The wrapper accepts one parameter––the parameter of the program it is supposed to execute. (Any parameters after the first will be passed to the program the wrapper is executing.) For security reasons, the wrapper only executes programs located in the Majordomo directory, `/usr/local/majordomo-1.94.5/`. This restriction prevents a programmer from using the wrapper to run programs that should not have Majordomo privileges. (For example, **wrapper /bin/vi** would allow any user to edit any Majordomo configuration file.) When a message is sent to `majordomo@host.com`, Sendmail starts up the wrapper which, in turn, starts up **majordomo**, and Sendmail sends the message to the **majordomo** script via the standard input. Majordomo then extracts the commands out of the message and responds appropriately.

# 2.2. Editing `sendmail.cf`

Due to its arcane syntax, `sendmail.cf` is perhaps the most feared of all configuration files. In the installation of `majordomo`, it is not absolutely necessary to edit `sendmail.cf`; however, a couple of features are extremely useful. Unless major changes have to be made to `sendmail.cf` (which, thankfully, Majordomo does not require), editing the file is not that difficult. All that need be done is adding extra lines to the file.

## 2.2.1. Another Aliases File

Creating a separate file for the Majordomo aliases, such as `/usr/local/majordomo-1.94.5/majordomo.aliases`, is often a good idea. This can be done rather easily by adding a line to the end of the `sendmail.cf` file

```
OA/usr/local/majordomo-1.94.5/majordomo.aliases
```

To have a `/usr/local/majordomo-1.94.5/majordomo.aliases`, Sendmail must be able to generate a database (`/usr/local/majordomo-1.94.5/majordomo.aliases.db`). The easiest way to accomplish this is to go ahead and create an empty database for Sendmail to overwrite.

```
[root@kes majordomo-1.94.5]# touch majordomo.aliases.db
[root@kes majordomo-1.94.5]# chmod 644 majordomo.aliases.db
```

Another method to get around this issue is simply to create the `majordomo.aliases` file in the `/etc/` directory, rather than the Majordomo home directory.

## 2.2.2. Undesirable Sendmail Security Features

For certain setups, some security measures that Sendmail uses can prevent Majordomo from working properly. Obviously, these security features must be turned off.

## 2.2.3. Sendmail Trusted Users

Sendmail is programmed to make it difficult for people to make "perfect" forgeries of E–mail. For example, when a user sends a message via SMTP, the source IP address is typically logged, and when a user sends a message by giving it directly to Sendmail and specifying the sender using **sendmail –f**, Sendmail puts a warning message in the header specifying the user who really sent the message. However, some programs need to be able to send messages masquerading as other users, and having this extra security line appended to the header is annoying. Sendmail handles this problem by having trusted users. In order for Majordomo's **resend** script to work properly, `majordomo` must be a Sendmail trusted user since the program needs to resend mail from other users.

One way to make Majordomo a trusted user is by adding the line

```
Tmajordomo
```

to the `sendmail.cf` file.

## 2.2.4. Sendmail Restricted Shell

If Sendmail is using **smrsh**, then the only programs that can be executed are those under the `/etc/smrsh/` directory. Perhaps the best solution to run the wrapper from the `aliases` file is to create a symbolic link from `/etc/smrsh/wrapper` to `/usr/local/majordomo-1.94.5/wrapper`.

```
[root@kes smrsh]# ln -s /usr/local/majordomo-1.94.5/wrapper wrapper
```

A second solution is to actually move the wrapper into the `/etc/smrsh/` directory.

```
[root@kes smrsh]# mv /usr/local/majordomo-1.94.5/wrapper ./
```

If security is not a major concern, the secure shell can be disabled. One fairly crude method is simply to delete `/usr/sbin/smrsh` and copy or link `/bin/sh` in its place.

```
[root@kes sbin]# rm -f smrsh
[root@kes sbin]# ln -s /bin/sh smrsh
```

A better (but more difficult) method is to edit `sendmail.cf`. Change the reference from `/usr/sbin/smrsh`

```
Mprog,          P=/usr/sbin/smrsh, F=lsDFMoqeu9, S=10/30, R=20/40, D=$z:/,
                T=X-Unix,
                A=sh -c $u
```

to `/bin/sh`

```
Mprog,          P=/bin/sh, F=lsDFMoqeu9, S=10/30, R=20/40, D=$z:/,
                T=X-Unix,
                A=sh -c $u
```

## 2.2.5. Group Write Permission

If you plan on having a non–root user add and manage mailing lists, you will need to make the `majordomo.aliases` file group writable. However, Sendmail does not allow this configuration for security reasons (see Section 2.4). To disable this security feature, add the line

```
O DontBlameSendmail=GroupWritableAliasFile
```

to the `sendmail.cf` file. Also, the lists directory must be group writable in order to add a list, but Sendmail will not allow this setup for similar security reasons. To disable this security feature, adding the line

```
O DontBlameSendmail=IncludeFileInGroupWritableDirPath
```

to the `sendmail.cf` configuration file is necessary.

## 2.3. Configuring `sendmail.cf` Using the M4 Configuration

For administrators who do not want to edit the `sendmail.cf` file directly, it is possible to use M4 to create the file; this section describes how to make the changes discussed in the previous section to the mc file instead of the cf file.

The purpose of the M4 configuration is to provide an easy way to create the `sendmail.cf` file. The idea is that the created mc file is easier to understand than the `sendmail.cf` file. By running the m4 preprocessor, a `sendmail.cf` file is generated:

```
[root@kes etc]# m4 /etc/sendmail.mc > /etc/sendmail.cf
```

### 2.3.1. Creating Another Aliases File

Add the line

```
define(`ALIAS_FILE',`/etc/aliases,/usr/local/majordomo-1.94.5/majordomo.aliases')
```

to the `sendmail.mc` file.

### 2.3.2. Making Majordomo a Trusted User

Add the line

```
define(`confTRUSTED_USERS',`majordomo')
```

to the `sendmail.mc` file.

### 2.3.3. Disabling Sendmail Secure Shell

Delete the line that reads

```
FEATURE(smrsh)
```

in the `sendmail.mc` file.

### 2.3.4. Disabling Security Features

To disable the group write permission security check on the aliases file, add the line

```
define(`confDONT_BLAME_SENDMAIL',`GroupWritableAliasFile')
```

to the `sendmail.mc` file.

To disable the path write permission security check for the include files, add the line

```
define(`confDONT_BLAME_SENDMAIL',`IncludeFileInGroupWritableDirPath')
```

To enable both of these options, use

```
define(`confDONT_BLAME_SENDMAIL',`GroupWritableAliasFile,IncludeFileInGroupWritableDirPath')
```

Adding the last statement is equivalent to writing

```
O DontBlameSendmail=GroupWritableAliasFile,IncludeFileInGroupWritableDirPath
```

in `sendmail.cf`, and this entry is the same as writing the entries on separate lines:

```
O DontBlameSendmail=GroupWritableAliasFile
O DontBlameSendmail=IncludeFileInGroupWritableDirPath
```

## 2.4. Sendmail Security Concerns

Security is inversely proportional to convenience; the only secure machine is one that cannot be accessed by anyone. When some of Sendmail's security features are disabled, a machine will inevitably become less secure. However, it is important to understand the basic security risks in order to determine if the convenience out weighs possible breaches of security.

### 2.4.1. Consequences of Unsafe Group Writes

If a user has write permission to access an aliases file, *she should be a trusted user*. By putting an entry into the aliases file (such as the one used to execute **wrapper**) a user can execute any program with the privileges of Sendmail (`daemon` or, in older versions, `root`). This gaffe would allow people to remove or change the permissions of files that belong to `daemon` (using the **rm** or **chmod** commands in the aliases file). To some extent, this possibility is avoided by using **smrsh**; however, one must still be careful as to what files are in the `/etc/smrsh/` directory.

Another important security issue is that the user who can access the aliases file can append or write to files that belong to `daemon` by using file redirection (a >> or > instead of a `|`). Even so, this breach too can be countered by adding a line to the `sendmail.cf` file limiting what files can be written to through the aliases file. Add the line

```
O SaveFileEnvironment=/path/to/safe/files
```

to the `sendmail.cf` file or add

```
define(`confSAFE_FILE_ENV',`/path/to/safe/files')
```

to the `sendmail.mc` file. However, this maneuver only leaves a thin layer of security between the user and `daemon`. A much better idea would be to have the aliases file only writable by root and to create an SUID root program to add and remove the Majordomo related aliases.

In the case of include or .forward files, commands or redirections are run as the user who owns the file. Therefore, if a file is group writable, a member of the group can execute commands as the user who owns the file. In other words, any user in the `majordomo` group could execute commands as Majordomo. However, since the `majordomo` user is created without a shell, commands or redirections will not be processed in include files owned by `majordomo`.

## 2.4.2. Consequences of Unsafe Group Writable Directory Paths

If a user has group write permission to a directory, for example `/etc/`, the user could simply move any file and create a new one in its place. An attack might go something like this

```
[mallory@kes etc]$ mv aliases ...
[mallory@kes etc]$ vi aliases
```

The user can then make her own aliases! This attack, however, could be prevented by Sendmail's security checking for unsafe group writable paths. Such an attack also would work with include and .forward files having unsafe paths.

In the case of Majordomo, the user in the `majordomo` group already has access to the include files, so this does not really compromise security. However, an administrator should be careful to prevent these undesirable unsafe group writable directory paths from occurring in the future because Sendmail will *not* check for them.

## 2.4.3. Protecting Subscribers' Privacy

Unfortunately, sophisticated spammers can expand mail lists using the **EXPN** SMTP command. For this reason, administrators should disable this feature when serving mailing lists. Add the line

```
O PrivacyOptions=noexpn
```

to the `sendmail.cf` file or

```
define(`confPRIVACY_FLAGS',`noexpn')
```

to the `sendmail.mc` file.

# 3. Majordomo

Majordomo is, of course, the piece of code that this document revolves around; it consists of a collection of Perl scripts with the sole purpose of managing mailing lists.

## 3.1. Preparing to Install

Download the gzipped source distribution of the latest version of Majordomo from Great Circle Associates and uncompress it

```
[jarchie@kes jarchie]$ tar zxvf majordomo-1.94.5.tar.gz
```

This will create a subdirectory with all of the files necessary to install Majordomo; this directory *cannot* be the same directory in which Majordomo is to be installed.

Majordomo must run under a specific UID and GID so when any of the scripts are run, they will run under Majordomo's UID. Thus, it is necessary to decide what UID and GID Majordomo should run under. Also, Majordomo must be a Sendmail trusted user (see Section 2.2.3).

Check the `/etc/passwd` and `/etc/group` files to find a UID and GID that are not taken. For this example, a UID of 16 and a GID of 16 was chosen. You have to decide on the location where the Majordomo scripts will reside; in this HOWTO, the directory `/usr/local/majordomo-1.94.5/` was chosen. If you are using a shadowed password file, add entries similar to

```
majordomo:x:16:16:Majordomo List Manager:/usr/local/majordomo-1.94.5:
```

to your `/etc/passwd` and add an appropriate entry to `/etc/shadow`.

```
majordomo:*:10883:0:88888:7:::
```

Use the other entries in these files as a guide for exactly what should be added. *These are only the values for my system.* If you are not using shadowed passwords, only an entry in the `/etc/passwd` file is necessary.

To create a Majordomo group, add a line similar to

```
majordomo:x:16:jarchie
```

to your `/etc/group` file. Appending your username to the end of the line will give you access to the Majordomo files that are group writable.

## 3.2. Editing the Installation Files

The `Makefile` contains all the information needed to install Majordomo; it is usually necessary to edit lines in the `Makefile` that refer to system specific settings so Majordomo will be able to install cleanly on your system. Most of the default settings are correct; however, the following settings, almost invariably, need to be changed on a per system basis.

```
[jarchie@kes majordomo-1.94.5]$ vi Makefile
```

The settings

```
PERL = /bin/perl
CC = cc
W_HOME = /usr/test/majordomo-$(VERSION)
MAN = $(W_HOME)/man
W_USER = 123
W_GROUP = 45
```

should be changed to something more appropriate for your system. For example, in my setup, the values were changed to

```
PERL = /usr/bin/perl
CC = gcc
W_HOME = /usr/local/majordomo-1.94.5
MAN = /usr/man
W_USER = 16
W_GROUP = 16
```

Also the `majordomo.cf` file must be created. An easy way to create this file is to copy the provided `sample.cf` file to `majordomo.cf` and edit it.

```
[jarchie@kes majordomo-1.94.5]$ cp sample.cf majordomo.cf
[jarchie@kes majordomo-1.94.5]$ vi majordomo.cf
```

Again, most of the settings are correct by default, but the following lines might need to be changed for your system from

```
$whereami = "example.com";
$whoami = "Majordomo\@$whereami";
$whoami_owner = "Majordomo-Owner\@$whereami";
     $homedir = "/usr/test/majordomo";
$digest_work_dir = "/usr/local/mail/digest";
$sendmail_command = "/usr/lib/sendmail";
```

to something more appropriate such as

```
$whereami = "kes.emeraldis.com";
$whoami = "majordomo\@$whereami";
$whoami_owner = "majordomo-owner\@$whereami";
     $homedir = "/usr/local/majordomo-1.94.5";
$digest_work_dir = "/usr/local/majordomo-1.94.5/digest";
$sendmail_command = "/usr/sbin/sendmail";
```

`$whoami` and `$whoami_owner` do not need to be changed for Majordomo to work; however, I changed them because I like to avoid typing capital letters. `$digest_work_dir` is a temporary directory where digest files should be placed; this directory should be assigned to wherever you want digests to be stored. If you do not plan to use digested lists, do not worry about this option. `$whereami`, `$homedir`, and `$sendmail_command` should be changed to appropriate values for your system. Unlike the `Makefile`, these options can always be changed after Majordomo is installed by editing `majordomo.cf` in the directory where Majordomo was installed. (The configuration file is simply copied during setup.)

---

## 3.3. Installing Majordomo

The next step is to compile the Majordomo wrapper. The wrapper is the only Majordomo component that needs to be compiled because everything else is a collection of perl scripts and, therefore, is not compiled.

```
[jarchie@kes majordomo-1.94.5]$ make wrapper
```

To install the Majordomo files, execute the commands

```
[root@kes majordomo-1.94.5]# make install
[root@kes majordomo-1.94.5]# make install-wrapper
```

The first command can be done as the Majordomo user (assuming `majordomo` can create or has access to `$home_dir`), but the second command needs to be done as `root` so the installation script can SUID root the Majordomo wrapper. (Since, `majordomo` was created without a login shell or password, if you want to execute the first command as `majordomo`, you will need to **su majordomo** as root in order to become `majordomo`.)

---

## 3.4. Creating the Majordomo Aliases

Sendmail aliases must be created for Majordomo so commands sent to Majordomo can be processed by **majordomo**, and an alias for the Majordomo owner must be created so people can E-mail you through the standard `owner-majordomo` address. Add the following entries to your aliases file (see Section 2.1).

```
majordomo:       "|/usr/local/majordomo-1.94.5/wrapper majordomo"
owner-majordomo: jarchie
majordomo-owner: jarchie
```

---

## 3.5. Testing the Configuration

As a regular user (*not* as `majordomo` *or* as `root`), run

```
[jarchie@kes jarchie]$ /usr/local/majordomo-1.94.5/wrapper config-test
```

This program can detect most problems in the Majordomo installation.

---

## 3.6. Creating Lists

To create a list, create a file with the name of the list in the Majordomo lists directory. For example, to create a list called test, create a test file as Majordomo

```
[root@kes /]# su majordomo
[majordomo@kes /]$ touch /usr/local/majordomo-1.94.5/lists/test
```

and add the related aliases

```
test:           :include:/usr/local/majordomo-1.94.5/lists/test
owner-test:     jarchie
test-request:   "|/usr/local/majordomo-1.94.5/wrapper request-answer test"
test-approval:  jarchie
```

## 3.7. Further Testing of the Configuration

Now test the operation of the list by issuing a **lists** command to Majordomo.

```
[jarchie@kes jarchie]$ echo lists | mail majordomo
```

It should only take a second for **majordomo** to reply with a message containing all the lists which are currently set up. Next, try issuing a **help** command.

```
[jarchie@kes jarchie]$ echo help | mail majordomo
```

Majordomo should reply with a list of all commands that Majordomo accepts. It might be a good idea to save the message for future reference.

To see if the aliases are working properly, try subscribing and unsubscribing yourself to the list.

```
[jarchie@kes jarchie]$ echo subscribe test | mail majordomo
```

You will receive an E−mail message containing instructions on how to confirm your subscription as well as a letter confirming that your command was successful. After sending back your confirmation, Majordomo should send back two letters−−one letter stating that your subscribe request was successful and another letter welcoming you to the test list. The owner of the list will also be sent a message stating that you have subscribed to the list.

To unsubscribe from a list, send a **unsubscribe** command

```
[jarchie@kes jarchie]$ echo unsubscribe test | mail majordomo
```

You should be sent back a letter stating that your command was successful.

## 3.8. Creating Better Aliases

For some lists, it may be desirable to have Majordomo process messages before they reach the list. For example, Majordomo has the **resend** script to automatically filter messages based on content (such as taboo words), to prevent people from sending Majordomo commands to the list, and other features. To use these options, it is necessary to use a better set of aliases such as

```
test:         "|/usr/local/majordomo-1.94.5/wrapper resend -l test test-list"
test-list:    :include:/usr/local/majordomo-1.94.5/lists/test
owner-test:   jarchie
test-owner:   jarchie
test-request:  "|/usr/local/majordomo-1.94.5/wrapper majordomo -l test"
```

The last entry allows someone simply to send a message to `test-request@kes.emeraldis.com` with the text `subscribe` rather than sending a letter to `majordomo@kes.emeraldis.com` with the text `subscribe test`. Also, note that if sendmail is using smrsh, the above aliases should reference the copy of the wrapper in the safe path−−usually `/etc/smrsh/wrapper`.

## 3.9. Debugging

It is common for Majordomo's permissions to be set incorrectly causing Majordomo to work improperly. Fortunately, Sendmail and Majordomo typically, give decent error messages indicating a problem. For example, the `lists` directory must be executable by the user sendmail setuids to, typically `mail` or `daemon`. If **sendmail** cannot execute `lists`, the permissions must be loosened.

```
[root@kes root]# chmod +x /usr/local/majordomo−1.94.5/lists
```

Another common problem is caused by the `lists` directory being group writable. To solve this problem, one can ether clear the group writable bit, or use the sendmail option `IncludeFileInGroupWritableDirPath` (see Section 2.2.5 and Section 2.4.1 for more details).

## 3.10. Majordomo Security Concerns

Majordomo is intended to run on a isolated system; there are a couple of well−known security holes in the scripts that allow any local user capable of executing **wrapper** to execute code as the `majordomo` user. If Majordomo must be run on a system providing users with shell access, then it is advisable to tighten up permissions on the wrapper. This can be done by clearing the world executable bit and **chgrp**ing the wrapper to the user that needs to run the Majordomo scripts. For example, if Sendmail and MajorCool are both being used to execute the wrapper use the commands

```
[root@kes root]# cp /usr/local/majordomo−1.94.5/wrapper /etc/smrsh/wrapper
[root@kes root]# chmod 4750 /usr/local/majordomo−1.94.5/wrapper
[root@kes root]# chown root:nobody /usr/local/majordomo−1.94.5/wrapper
[root@kes root]# chmod 4750 /etc/smrsh/wrapper
[root@kes root]# chown root:mail /etc/smrsh/wrapper
```

to secure the system. This will allow **sendmail** (while running under `mail`) to execute **/etc/smrsh/wrapper** while allowing the webserver's MajorCool (running under `nobody`) to execute **/usr/local/majordomo−1.94.5/wrapper**. This solution, however, will allow anyone with the UID or GID of `mail` or `nobody` to also obtain access to the `majordomo` account. To protect the `nobody` account, it is important not to allow normal users to make use of server side includes or cgi scripts unless those services do not run under `nobody`.

# 4. MajorCool

MajorCool is a web–based interface to Majordomo allowing users to add and delete themselves from lists and manage lists that they own. The installation is fairly straightforward; all that need be done is to unzip the files, edit one line in the **Configure** script, and execute the script.

## 4.1. Extracting MajorCool

The latest files can be downloaded from Conveyance Digital.

```
[jarchie@kes jarchie]$ mkdir majorcool
[jarchie@kes jarchie]$ mv majorcool.tar.gz ./majorcool/
[jarchie@kes jarchie]$ cd majorcool/
[jarchie@kes majorcool]$ tar zxvf majorcool.tar.gz
```

## 4.2. Edit the Configure Script

Open **Configure** and

```
[jarchie@kes majorcool]$ vi Configure
```

change the line that reads

```
PERLBIN="/usr/local/bin/perl"    # How to start a perl script
```

to the proper location of **perl**

```
PERLBIN="/usr/bin/perl"          # How to start a perl script
```

otherwise, MajorCool will not be installed properly.

## 4.3. Installing MajorCool

When running the **Configure** script, if the default choice for an option is okay, simply pressing Enter will accept the default.

```
[root@kes majorcool]# ./Configure
```

The **Configure** script will ask you to hit Enter a few times, and then it will ask for the location of Majordomo and some more questions about the setup of your Web server.

```
What is the installation directory of Majordomo?
    /usr/local/majordomo-1.94.5
Will place the MajorCool programs in /usr/local/majordomo-1.94.5.

What is the path to your Majordomo configuration file?
        [/usr/local/majordomo-1.94.5/majordomo.cf]:
```

```
Using configuration file name '/usr/local/majordomo-1.94.5/majordomo.cf'

Where would you like temp files created when MajorCool runs?
        [/tmp]:

MajorCool needs to install CGI programs, support files, and icons in
your Web server directories.

What is the root directory for your Web server?
        []: /var/www

Where is the cgi-bin directory for your Web server?
        [/var/www/cgi-bin]:
Will place the programs in /var/www/cgi-bin.

What is your server's URL for '/var/www/cgi-bin'?
        [/cgi-bin]:

Where is the image directory for your Web server?
        [/var/www/icons]:
Will place the icons in /var/www/icons.

What is your server's URL for '/var/www/icons'?
        [/images]: /icons

Where is the root directory for documents on your Web server?
        []: /var/www/html
```

The **Configure** script will ask other questions that are less critical. (The defaults are fine, but you might want to change a few settings to fit your preferences. Unlike some of the Web server questions, the meanings should be obvious from the context.) When the configuration file that the script generated from your answers is displayed, you should accept the new version.

```
Accept the new version? [yes|no|list|edit|diff]? y
```

The installation script will install the MajorCool files and run the **majordomo** cgi script which outputs the html file to the console. Check to see if the installation worked by viewing the **majordomo** cgi script from the web.

```
[jarchie@kes jarchie]$ lynx http://localhost/cgi-bin/majordomo
```

# 5. Frequently Asked Questions

Two questions occur often.

*5.1. Why does sendmail give the error,* `sh: wrapper not available for sendmail programs`*?*
*5.2. Why will Red Hat not process my mc file?*

**5.1.** Why does sendmail give the error, `sh: wrapper not available for sendmail programs`?

**smrsh** will only allow sendmail to execute certain files. See Section 2.2.4.

**5.2.** Why will Red Hat not process my mc file?

For some reason, Red Hat does not include the necessary files to process mc files. According to `/etc/mail/sendmail.mc`, these files should be in the `sendmail-cf` package; however, I was unable to find this package on the CD. To fix this problem, install Red Hat's sendmail SRPM, uncompress the sendmail tarball, and copy the necessary macro files.

```
[root@kes root]# rpm -i sendmail-8.11.0-8.src.rpm
[root@kes root]# cd /usr/src/redhat/SOURCES/
[root@kes SOURCES]# tar zxvf sendmail-8.11.0.tar.gz
[root@kes SOURCES]# cd sendmail-8.11.0
[root@kes sendmail-8.11.0]# cp -R cf /usr/lib/sendmail-cf
[root@kes sendmail-8.11.0]# cd ..
[root@kes SOURCES]# rm -rf sendmail-8.11.0
```

Also in the default `/etc/mail/sendmail.mc` file, there is a slight syntax error. (The beginning single quotes on one line slant in the wrong direction.) The line that reads

```
define('ALIAS_FILE','/etc/aliases')dnl
```

should be changed to

```
define(`ALIAS_FILE',`/etc/aliases')dnl
```

After these two changes, the new `sendmail.cf` should be generated properly.

# List of Terms

***digest***

a collection of new messages mailed to the members of an archived list as one message. A list is called digested when it is archived and, periodically, a digest is sent out.

***Group ID***
(GID)

an identification number assigned to files, directories, and processes to restrict access−−similar to UID except multiple people can be a member of a group. On Unix−type systems, groups can be set up (defined in the `/etc/group` file). When a user name is a member of a group, she can access files created with that GID (assuming permissions allow it).

***Mail Transfer Agent***
(MTA)

a program, such as Sendmail, responsible for passing mail from one location to another.

***Set Group ID***
(SGID)

a file attribute which allows a program to run with specific group privileges no matter who executes it.

***smrsh***

(SendMail Restricted SHell) the shell that Sendmail uses to execute programs. **smrsh** puts restrictions on the programs that can be run to make it safer than using a regular shell such as the Bourne Shell.

***Set User ID***
(SUID)

a file attribute which allows a program to run as a specific user no matter who executes it.

***User ID***
(UID)

an identification number assigned to files directories, and processes−−similar to GID except every user has a unique UID. Every process must run under a UID (the one−to−one relationship between the UID and user name is defined in `/etc/passwd`). The process' UID determines what the program can access. In general a regular user can change the permissions on files that she owns unless the UID is 0 (the `root` user). In that case, `root` can modify any files on the system.

***wrapper***

a program used to start another program; usually a wrapper is SUID or SGID so it can bestow privileges onto another program that the other program would not normally have.