LIST

proc

version7

p r o c

version 7

CREN
corporation
for
research
and
educational
networking

1112 16th street nw
suite 600
washington, dc 20036
202 · 872 · 4200

© Copyright 1994 CREN. All rights reserved

UNIX® is a registered trademark of X Open.
ULTRIX® is a registered trademark of Digital Equipment Corporation.
AIX® is a registered trademark of IBM Corporation.

Table of Contents

1. Introduction 6
2. Overview 7
3. Installation. 8
4. Configuration 10
    - Converting from VM Listserv 10
    - Converting from ListProc 6.0 10
    - The Config File 12
    - Remote Lists 25
    - Server Environment Variables 26
    - System Mail Alias 27
5. Defining a List 28
    - Creating a Mailing List 28
    - Configuring a Mailing List 28
    - Setting the List Aliases 39
6. List Management 40
    - Archiving Files 42
    - Subscribing to Files 46
7. ILP 48
8. General Tips 50
    - News-Mail Gateways 50
    - Peering 51
    - Anonymous FTP 52
    - Gopher and WAIS 53
    - Mailers 54
    - Alias and Ignore 54
    - Listproc Release Notes 57

Appendix A: Installation Script. 59
Appendix B: Sample Config File 60
Appendix C: Catmail Utility 61
Appendix D: Interactive ListProcessor 62
Appendix E: ListProc Files 65

INDEX 70

1. Introduction

The CREN Site Manual is a preliminary installation and management document for the CREN ListProc Version 7.0 sites. It walks you through an installation and list configuration / management, but it is not meant to be a comprehensive user document. User and owner documentation will be available separately.

CREN has set up a mailing list to support ListProc Version 7.0 sites. To subscribe, send e-mail to ListProc@list.cren.net and type in the body of the mail:

```
sub cren-listproc Firstname Lastname
```

This is a private, hidden list. We will maintain archives that subscribers can access. All subscription requests are owner approved.

If at all possible, direct problems, bugs and questions to the mailing list so everyone can benefit from your experience. We will summarize any private e-mail and/or telephone conversations and post them to the mailing list unless you specifically request otherwise. Since ListProc runs on different systems, please indicate in your questions and comments the hardware platform, hardware configuration (amount of RAM/ hard disk), operating system and operating system release you are running. Also include any relevant error messages and /or log entries.

© Copyright 1994 CREN. All rights reserved.

2. Overview

The target audience of this document is system maintainers and list owners.

ListProc(essor) is an automated mailing list and archive manager for UNIX® systems. Users may add themselves to lists, send and receive messages, subscribe to files from archives and set various parameters to interact with the mailing list. List owners edit or moderate lists, control subscribers or hide lists from public scrutiny, just to name a few options. ListProc is intended to be easy to maintain, support and use. It can also be integrated with other data retrieval and presentation tools (such as gopher, anonymous ftp, http, and WAIS).

There are six levels of ListProc Users:

Regular Users
: subscribe to lists, query the server for available lists and archives, get files and post to lists.

Subscription Managers
: are responsible for subscribing and unsubscribing users from the list. A subscription manager does not have to be subscribed to the list itself or have access to the list's archives.

Errors To Recipients
: get RFC-822 error messages and can remove subscribers from the list. Errors To Recipients do not have to be subscribed to the list or have access to the lists postings or archives.

Moderators
: approve and/or edit postings going to a mailing list if it is moderated.

List Owners
: oversee the functions of particular lists, configure list parameters, deal with bounced mail, moderate or otherwise manage the list.

System *or* **Server Manager**
: is responsible for overall server configuration and operation, placement of archives, monitoring logs and getting server error messages.

Not all of those categories have to be defined. In most cases, the Subscription Manager, the Errors-To recipient and the Moderator will be the List Owner by default. These additional options allow a List Owner to distribute the responsibilities of list management.

There are two levels of list control:

- Owner controlled lets the list owner change every aspect of the list configurations that are authorized for a list owner. This even includes adding other owners and removing himself or herself as a list owner.
- Manager controlled includes everything for owner controlled except the ability to add and remove owners, change the list password and archive list traffic. (Manager control guarantees that the list owner meets required criteria and does not change without your institution's knowledge.)

3. Installation.

A file listproc.tar.Z will be available to sites that want to run CREN's ListProc. Sites must provide CREN with an ftp account on their system where we can place upgrades and documentation for the client.

This file contains all the necessary binaries to install and run ListProc-7.0.

To begin, create an account called *server* on your system. It is important that the account be called server to function properly, to allow for uniform access and to provide for future server to server communications. This consistent behavior must be maintained. ListProc will run as server. Place the tar file in server's home directory, make sure the file is owned by server and then *su* to server.

On CREN's system (SUN Sparc LX running SOLARIS 2.3 and another LX running SUNOS 4.1.3_U1 Path Level 8) the server is running under a csh, although any shell should work, with a standard search path to system executables defined and no aliases. It is important to define a standard search path as ListProc will need various UNIX utilities in the course of normal operations. It is also important not to define any command aliases (i.e. *rm -i*) because ListProc expects certain behavior from the utilities in question.

The provided utility *systest* determines if it can find the needed system utilities.

After you have extracted the distribution, you will have two files: **install.listproc** and **src.tar**. The first file is a shell script that you should make executable:

```
chmod 750 install.listproc
```

or preface with /bin/sh (or whatever the path to your bourne shell is).

At this point you should define the environment variable LPDIR (ListProc DIRectory). This variable, if defined, determines the server's executable files directory. If you define it, put it in server's .login or .cshrc. It defines the home directory to the server and is set as follows:

- csh `% setenv LPDIR 'pwd'`
- sh `% set LPDIR = 'pwd'; export LPDIR`

Other shells and operating systems may vary.

If LPDIR is not defined, then all executables determine the LPDIR by their invocation. LPDIR is optional and required only if you are installing ListProc in a directory different from the one which contains the distribution archives.

While the example above assumes you are untar'ing the file in the directory where you will perform the final installation, this does not have to be the case. You must set LPDIR to where you will do the final installation. At this point type:

```
install.listproc
```

and the **src.tar** file will be extracted. This file should have been extracted with file permissions set to execute. If not, the following will also work:

```
/bin/sh install.listproc
```

You are now ready to run setup. On the command line type:

```
setup
```

or

```
/bin/sh setup
```

Setup will look at your system and make some recommendations. It will recommend a umask setting of 077 if that is not currently set, and you will receive a warning. **This is important as you do not want anyone but server to have access to the servers directory and files. Passwords are maintained in the configuration files for the server and various lists.** It will inform you that ListProc will be installed in the current directory or wherever LPDIR environment variable is defined to. In addition, setup will accept an optional command line argument, which is the directory for installation.

At the end of the installation process you will be asked for the system's root password. This is only necessary if you want ListProc to listen for interactive connections at a port below 1024. ListProc's default port (which is officially registered) is 372. Unix TCP/IP implementations define the concept of priveleged ports are those below 1024. Only processes running under root's uid can listen on those ports.

If you cannot or do not want to give the root password, you have two options:

1. Do not allow interactive connections to ListProc. The -i flag to *serverd* allows interactive connections.

2. Have ListProc listen on a port above 1024 using the -p flag to *serverd*. (see Configuration)

For a more detailed explanation, see *Internetworking with TCP/IP Volumes I-II* by Douglas Comer.

It is imperative that the installation occur while the maintainer is operating under the uid server. All files must have the proper ownership in order for ListProc to work properly.

Once the system is installed, you must customize the config file to suit your particular site parameters and add a list maintainer server definition to the owners file (see *The Config File* in Chapter 4). To begin the server, simply type:

```
start
```

at the prompt.

During normal operations, you will want to start Listproc from one of the system startup scripts. Due to the wide variability and implementations, a discussion of startup scripts is beyond the scope of this manual. It is important to start ListProc while effectively running as *server*. A line similar to the following will suffice:

```
su –c server /path/to/server/start > /dev/null
```

A sample installation session is included in Appendix A

Converting from VM Listserv

If you are converting from VM Listserv, two conversion tools are available in the support directory of the ListProc distribution. These tools are:

lsv_conv_list.pl
: This utility converts a VM Listserv subscribers file into ListProc format. It preserves most of the user options although some user options have no equivalent.

lsv_conv_logs.pl
: This utility converts VM Listserv logs into a format suitable for ListProc archives. It changes the logs to Unix mbox format and places them, using *farch*, in the archive directory.

Both tools are perl scripts and require that you first ftp (or otherwise transfer) the VM listserv list header and subscribers file to the host on which ListProc resides.

Converting from ListProc 6.0

If you are converting your lists from ListProc 6.0, a conversion tool is available in the support directory of the ListProc distribution. This tool is:

convert
: Perl script that will take as input your ListProc 6.0 config and owners file and generate a suitable init command that can be sent to ListProc and a file containing the necessary aliases to insert in your system alias file.

tconfig
: A bourne shell script to create config files for each list.

This tool requires that the installer have read access to the previous version of ListProc's owners and config file. It is an interactive script and will do one list at a time.

If you are running an earlier version of ListProc, contact info@cren.net for further information. To convert from ListProc's before 6.0, you must first upgrade to ListProc 6.0c. You can obtain a copy of ListProc 6.0x via anonymous ftp from cs.bu.edu. In the transition from 5.5 to 6, the structure of the subscribers file underwent fundamental changes which first must be addressed.

When you upgrade, as with any other system related activity, it is advisable to backup the current ListProc directory tree. It is also important that all the file ownership and permissions are done correctly. Unless otherwise indicated, you must do all the activities below relating to the upgrade as uid server or ListProc.

1. Stop the currently running ListProc server. This is done from wherever you have placed ListProc's binaries with the command:

```
start -k
```

2. If you have not backed up the existing ListProc source tree, now would be an ideal time to do this. While the upgrade procedure is non-destructive, we advise keeping a backup.

3. While you are upgrading the server, you want to prevent any new mail or requests from coming into ListProc. You can do this in one of two ways:

   · Stop your SMTPSERVER (i.e. sendmail). This will disrupt all mail to/from your site. No mail should be lost since sites trying to send you mail will simply queue it up. This is the recommended method since it ensures that no mail is bounced or lost. (Requires root access).

   · Remove all the list/ListProc aliases from the system mailer's alias file (/etc/aliases or /usr/lib/aliases). This will cause mail destined for a list or ListProc to bounce with a "No such local user" error but will not disrupt general mail destined to/from your site.

4. The archive (listproc.tar) should be extracted. This will create two files, src.tar and install.tar). At this point you should follow the installation instructions for installing from scratch.

5. While running the install utility, ListProc will move some of your existing files to Oxxxxx, for example, if you have an existing owners file, ListProc will move it to Oowners and insert its own owners file. You will get a message when running install.listproc informing you of all the files ListProc's installation procedure is moving. ListProc will also move your archives directory to Oarchives. It is important that you restore the archives directory prior to continuing because the ListProc setup script will re-index your archives. You should also restore the original owners file because ListProc 7.0 will use it.

6. At this point you should go into the newly created config file and change your site's default options as needed. Please refer to the section below on Config for more detailed information.

7. Run the setup script. Depending on the number and size of archives that you have, this could take a while (hours in some cases).

8. After the setup script finishes, you must now do some general housekeeping. We have provided two scripts to help you do this. These scripts actually make changes in each list's sub directory $HOMEDIR/lists/LIST-ALIAS/… and generate a new config file for you.

tconfig

This is a bourne shell script which must be run as the ListProc's uid. This version of ListProc requires a config file in each lists' directory. This file contains list-specific configuration information that is under the control of the list owner. This script simply "touches" this file in each list sub directory so that the server can run the first time through.

convert

This is an interactive perl script. It takes your owners file and your original config file and generates two files: **listname-init** and **listname.alias**. The first file is a config command (see Config below) which is used to configure the list to the old parameters. It is ready to send to ListProc in this form. The second file is the system aliases for the mailing list which must replace the old aliases. (This part requires root access). Convert will also append a correct *list* command to the end of the new config file (if you use convert more than once on the same list, you must edit the config file and remove the duplicate list statement). The convert script has many variables which must be customized to your site. The comments at the beginning of the script list the variables that you must change.

9. After you remove all the old aliases from the system alias file and add the new ones, you must run the **newaliases** command. (This is a system command, not a ListProc command and it requires root access).

10. Start ListProc. ListProc must be running when you send it all the listname.init files that you created during the upgrade process. These files must be sent, for this initial step, from the ListProc's manager's account. You can do this via e-mail or using the interactive client (see ILP). If you use e-mail, you must restart the system mailer (root access) before this step, otherwise you can restart the system mailer after this step.

Pitfalls

- ListProc 6 allowed certain characters in the list password field which are not allowed under 7.0. Try to avoid using the quotation mark, apostrophe, and the comma (" ' ,).

- Unless your site is running the script *re_from*: sh, ListProc uses the envelope address instead of the **From:**, this will cause some users to not be recognized by ListProc since ListProc 7.0 uses the **From:** address.

The Config File

At this point the basic system is installed and we are ready to set system configuration parameters. The config file will, in all likelihood, only need to be setup by editing during initial installation and whenever there are system changes that negate an entry in it (i.e. if you upgrade your operating system or your mailer, you will probably have to revisit the config file). The system comes with a default config file that you will customize for your site.

A sample config file is given in Appendix B.

Each line in the configuration file is described below.

organization

The Name of your Organization

This line defines the Organization that users will see on interactive login to the server. It is a free text string and does not need to be quoted.

multiple_recipients ##

This number means that the connection will send one copy of the data for every multiple_recipients defined. This defines how many recipients ListProc sends to your MTA per connection. When ListProc sends mail, it connects by default to your smtpserver at port 25 on the localhost, unless the parameter mta_host (explained on page 20) is set.

The higher the number, the better the efficiency, but this number is specific to your system and mailer specific. We recommend a value of 25 which works fine with sendmail. With ZMAILER, a value of 100 is acceptable.

list list-name list-address *list-owner list-password* -flags

This is the directive that defines a mailing list. The list parameter is obsolete in ListProc 7.0 and is automatically added by the *init* command. This line, along with the system aliases must be removed manually to decommission a list but should never be added manually. The list command is explained here to make the site manager aware of its syntax and debug any problems that might arise, but you should not manipulate it directly.

This command will be explained later. The syntax is:

list
is the command itself

list-name
the name of the mailing list

list-address
the fully qualified e-mail address of the owner

list-owner*
the fully qualified address of one of the list owners or an alias which points to the list owners (owner-listname@…)

list-password*
The initial password for the list owners to use to configure and access the list

* The *list-owner* and *list-password* are no longer required under ListProc 7.0 but are maintained for backwards compatibility with prior versions of ListProc. These parameters are defined through the *initialize* and *configuration* directives explained in the next section.

With the *initialize* command you no longer need to specify the list directive in the config file. We explain it here to help site managers understand the config file and backwards compatibility.

The following flags are available to use with this version of ListProc on the list command:

-e:
Echo Reports to the screen. If you have a modern syslog facility, this flag should not be used. If used and syslog is enabled, no echoing to the screen will occur.

-D:
Turn on extensive debugging information. Used for error tracing.

server listproc@host.domain -flags

This command defines the e-mail address of the server . The syntax is:

server
The command itself

server-address
The e-mail address of the server

Flags
-b <REQUEST>: This flag instructs the server to
batch all requests to off peak hours. For example, if getting statistics puts a large load on your system, you can postpone all statistics requests to non-prime hours with the following:

```
server ListProc@host.domain –b statistics
```

Defining peak/off peak hours is discussed later in this section.

-r <COMMAND>: This flag allows the site administrator to disable a particular command while the system load is above a configured number. For example:

```
server ListProc@host.domain –r statistics
```

This disables the *statistics* command for the entire site if the number of users logged in exceeds the number specified in the restriction parameter (defined below).

-d <Command>: This flag removes a particular command from the available set for use. It effectively disables the command on this server. For example, to disable the *statistics* command, use following parameter:

```
server ListProc@host.domain –d statistics
```

unix_cmd list_alias password cmd_alias 'unix-command [args]' #Comment

This request may be used to allow a particular list's subscribers to execute the above defined UNIX command aliased to name. The list's users who will be able to execute this command will be given the password ahead of time. The optional arguments, *arg*, may be literal arguments to the command as well as the special sequences $n, where n is a digit from 1 - 9. Their meaning is that of the bourne shell and they will be substituted by the user's actual arguments. For example:

```
unix_cmd list-alias password swap `/bin/echo
$2 $1' #Syntax swap a1 a2
```

will simply swap the user's first and second arguments. The user would submit a request such as:

```
run list-alias password swap arg1 arg2.
```

Since no security checks are made except to ensure that the user arguments do not contain: ', |, :, < or >, we encourage you to review carefully the functioning of this argument. It is included for backwards compatibility with ListProc 6.0, but we discourage using it.

The comment field is used in response to the user command:

```
run listname
```

This queries for run commands available.

Again, since **NO SECURITY CHECKS ARE MADE FOR THE RUN COMMAND**, use it with extreme caution. You are allowing a remote user to spawn a process on your system via e-mail with this command. CREN does not recommend its use. The run command can be disabled via the -d flag in the server directive (see above) with a config line such as:

```
server listproc@host.domain -d run
```

batch starthour endhour

This command defines the hours (inclusively) between which requests will be batched. For example:

```
batch 8 20
```

Will batch all requests between 8 am and 8 pm. Commenting this line out in the config file disables this feature. These requests are defined on the server line (see above) of the config file. For example to batch statistics requests, the following server line would work:

```
server listproc@host.domain -b statistics
```

Any user who sends a statistics request to ListProc during the pre-defined prime hours, will receive a message telling them that the request will be batched for processing at a later time.

serverd [-1] [-e] [-l load] [-i duration] [-p port] [-c max connections]

This command defines what flags are passed to the server deamon, the principle ListProc deamon. The available flags are:

-i #seconds:
Go interactive and listen on a privileged port for incoming connections. This requires that serverd is owned by root and suid. The duration field specifies the maximum number of seconds an interactive session will last. In order to go interactive, in addition to the -i flag, you need the following entry in your /etc/services file:

```
ulistproc  372/tcp
```

-c #
Specify the maximum number of interactive connections, the default is 5.

-p
Specify the listening port. The default is 372.

-l
Postpone filling requests if the system load goes above that specified on the command line.

-e
Echo messages to the screen. Not valid if syslog is defined.

A typical serverd line might look like this:

```
serverd –l 5 –i 1800 –c 10
```

Simply put, postpone filling requests if the system load rises above 5, go interactive and allow interactive users a maximum of 1800 seconds per session (20 minutes), and allow a maximum of 10 interactive users.

Note: AIX users should use *smit* to modify system files.

restriction #

This command specifies that certain requests will not be satisfied when the number of users logged into the system rises above #. This number refers to all users, not just interactive ListProc sessions. You specify the commands that are disabled using the -r flag to server (see server above). If you wanted to restrict statistics requests when more than 50 users were logged onto your system, you could use the following server and restriction lines in your config file:

```
restrictions 50

server listproc@host.domain –r statistics
```

These two lines can appear in either order.

manager <user@host.domain>

The e-mail address of the server manager. This could be an alias to allow for multiple managers.

password <secret>

The manager password is defined here. It should not be the same as any actual user account password.

comment server #Site Comment

This defines a Comment field for all outgoing postings and messages. The field appears in the **From:** line, for example:

```
comment CREN ListProcessor Version 7.0
```

frequency # of minutes

This specifies in minutes how often the server process checks for incoming mail.

ListProc provides a site with two different algorithms for checking mail. One favors server request. The first algorithm is chosen by including a positive integer after the keyword *frequency*.

```
frequency 1
```

In this scenario ListProc will search for the first list's mail, then ListProc's mail, then sleep one minute. Then it checks the second list's mail spool, check ListProc's mail spool, then sleep one minute. It follows this procedure until all lists are scanned. At that point it starts again.

The second algorithm is chosen with the following line:

```
frequency 0 1
```

With this algorithm, ListProc scans all lists' mail spools, then ListProc's mail spool, then sleeps for one minute. As the number of lists on your server increases, this option becomes more desirable. In this example the first variable must be a zero (0).

limit message # of bytes

This rejects messages longer than # of bytes. The default configuration is 65535 bytes, but you can and should tailor it to your system needs. A notification is sent back to the sender along with the first few lines of his/her original message for reference.

With the advent of MIME, messages of 1+ Megabytes are not uncommon. If your users want to send and receive multimedia documents on a mailing list, you must set this to a higher number. (1 - 2 Megabytes).

limit files # bytes

This limits the size of the archive files that can be sent out to # of bytes. When archive files reach this limit, they are automatically split up into subparts prior to mailing. When a user requests a global list of lists it too will be split up if it exceeds this limit.

precedence mail-precedence

This is the **Precedence:** field to use for outgoing list mail. It is usually set to bulk, junk, first-class or none. Vacation programs are *supposed* to not respond to bulk mail if they receive it. It is hoped that most vacation programs are well behaved in this regard. It has also been reported that certain DECNET gateways will not forward bulk mail. We recommend setting this to bulk. Each site can reconfigure this as it sees fit. If you comment out this option, ListProc will not use a Precedence: field. This command is correctly used as follows:

```
precedence bulk
```

ucb_mail [path to mailer]

This defines a path to your BSD mailer if it is available on your system. It is usually /bin/mailx, /usr/ucb/mail(x).

fax fax_program options

This defines the fax program to use for fax request (if any). The program should also be able to accept the file to fax from its standard input; otherwise you will need a script to use the fax program appropriately. If this directive is left off or commented out, fax requests are turned off.

To use this feature, you must have a second party fax driver that can receive input from ListProc and a modem that can send faxes.

header list-name { [Header Line] }

ListProc 7.0 retains all headers and has configuration options to manipulate it. This feature is maintained for backward compatibility with previous releases. This option specifies which precious header lines from the original sender's message to preserve during distribution. The list has to be defined before this line is reached and precious header lines are to specified within the enclosing { and }, possibly spanning multiple lines. For example, to preserve all **Errors-To** headers on list test, use the following:

```
header test {Errors-To: }
```

You probably would not want to preserve this particular header. This was used in release 6.0c and before because of how previous versions of ListProc behaved. ListProc 7.0 attempts to:

- Not alter any user supplied headers to the extent possible. (If you have your list set to always set reply to the list, then any user supplied **Reply-To:** header will of course be lost). ListProc will preserve and pass long headers (even X-headers) to the list.
- It will take original **To:** and **CC:** headers and convert them into **X-To:** and **X-cc:** to let the list know if there were other recipients of this message. We do not recommend setting this configuration line unless you find a real need for it; ListProc's header handling should be sufficient. This option also takes regular expressions, so that:

```
header test { .+ }
```

will preserve all headers.

option gate_mail/post_mail

These two options deal with newsgroup - mail gateways.

gate_mail:
This option forces the system to gate messages to news using the gateways e-mail address. Some popular packages are **mail2news** and **gatemail**, both widely available on the Internet. Listproc does not directly use these but simply sends mail to the address specified for further action (see News)

post_mail
This option forces the system to post messages to news groups (using inews), using the groups' names (e.g. misc.test). To use this, make sure that inews resides in /usr/lib/news, or that the config directive *inews* has been defined. This is the preferred way to do this, for more information in news - list gateways, see the section, Admin-Tips.

inews

This directive tells ListProc where to find the **inews** program for news - mail postings. ListProc will look in /usr/lib/inews by default. The syntax is:

```
inews /path/to/inews
```

option ignore_invalid_requests

This option instructs the server to ignore all unrecognized user requests while still processing valid ones. By default, the system stops processing requests when it encounter the first invalid one, at which point the users gets an error message and all subsequent requests are flushed.

mailmethod method-name [arguments]

System is the recommended method. It provides a unified approach to sending mail and has been tested on a wide variety of systems. In addition, the full set of the SMTP protocol is implemented, it allows list-owners and the manager to be notified of mail delivery problems. Also, with this method, mail is queued when it cannot be delivered. This method requires host TCP/IP connectivity and Internet support. An SMTP mail agent must exist on your system. If you needs to institute any of the other available mail methods, please talk to the CREN off line.

By using the system mail method, if the smtpserver is down or goes down while ListProc is distributing mail, the mail gets queued. The system manager will be notified and can take action. All mail is queued in $LPDIR/mqueue and the queued files have numeric names, 1001, 1002. The manager should check the files before sending them out because queued mail may indicate problems other than the smtpserver going down, (i.e. malformed headers). To send the queue once the problem has been resolved, the manager must run $LPDIR/pqueue <file-name> (i.e. $LPDIR/pqueue 1001)

telnet

To be used when system is inappropriate for your system and telnet is available on your system. The behavior is similar.

env-var

Usually followed by <logname>/bin/rmail, or <logname>/usr/lib/sendmail -ha (-ha is required only to be used when the previous methods are inappropriate.

sendmail

Path/to/sendmail -flags. ListProc will prepare the message body and envelope and pass it, using the shell, to sendmail. In this case **server** must be a trusted user (see *Mailers* chapter 8) and there is also a limit to the size of the command line to 2K (although some shells might allow more) which limits your headers (see multiple-recipients).

deliver_files [option]

deliver_files hourly 30
deliver_files daily 8:30
deliver_files weekly Monday
deliver_files monthly

This option controls the frequency and time that files are delivered to subscribers. This applies only if the are accessible through AFD/FUI.

mta_host [localhost] [port]

This parameter defines the MTA host and the connection port for sending outgoing mail. The default host is localhost; the default port is 25.

helo_arg [localhost]

This argument defines the HELO argument for the SMTP transaction to the mta_host. Mail Transfer Agents such as ZMAILER complain when HELO arguments do not match their view of the network . This allows a site to tailor that argument. The default is localhost.

syslog FACILITY

This allows a site to turn on syslog reporting. FACILITY are the following syslog facilities:

LOG_USER
LOG_DAEMON
LOG_LOCAL0-7
LOG_MAIL

To log to a file called /var/log/list-log using the LOCAL2 facility, the following line would be defined in your config file.

```
syslog LOG_LOCAL2
```

You must have this facility defined in your /etc/syslogd.conf. ListProc logs to level info and debug. For BSD 4.3 syslog systems, the following line is adequate.

```
local2.info;local2.debug     /var/log/list-log
```

This uses LOG_LOCAL2 as the syslog facility which is used in the example above.

Turning on syslog disables reporting.

mailer_daemon 'addr1|addr2…'

This option allows the site administrator to specify additional blackout addresses. The argument is a | separated list of regular expressions.

You will notice the root is in this list. Sites which use ListProc have automated servers that posted mail to a mailing list as uid **root**. These sites required the flexibility of allowing the root account to post to a mailing list. If you do not want **root** to be able to post, leave this in. ListProc defines (by default) the following suspicious address:

MAILER_DAEMON "<MAILER&~.+@.+\
\.MAILER.+>|SMTP@|
\<DA?EMON&~DEMON\
\.CO\
\.UK>|POST.*MAST|WPUSER|\
\$EMD|
\VMMAIL|MAIL.*SYSTEM|MAISER|MAIL.+AGENT|
TCPMAIL|BITMAIL|\
MAILMAN|MAIL_SYSTEM|^LISTSERV| ^LISTPROC| ^SERVER|
^LISTMAN|^CRENLIST"

These addresses are compiled into the code and cannot be changed.

Though it seems a little cryptic, it is actually easy to understand. The first line takes MAILER_DAEMON as a suspicious address and goes on to define variations on MAILER and the word SMTP in combination as suspicious addresses. For a explanation of regular expressions see the chapter on regular expressions in *Programming Perl* by Larry Wall and Randal L. Schwartz.

susp_subject 'subj1|subj2...'

This option allows the site administrator to specify invalid suspicious subjects. This is a list of | separated regular expressions. ListProc also comes preconfigured with a variety of suspicious subjects which cannot be changed. These are defined below:

```
DELIVERY[ \t]+ERROR|\
DELIVERY[ \t]+REPORT|\
DELIVERY[ \t]+PROBLEM|\
WARNING[ \t]+FROM[ \t]+UUCP|\
USER[ \t]+UNKNOWN|\
UNDELIVER.+[ \t]MAIL|\
UNDELIVER.+[ \t]MESSAGE|\
PROBLEMS.+DELIVERING.+MAIL|\
PROBLEMS.+DELIVERING.+MESSAGE|\
CAN[ \t]*['NO]*T.+DELIVER.+MAIL|\
UNABLE.+DELIVER.+MAIL|\
UNABLE.+DELIVER.+MESSAGE|\
MAIL.+NOT.+DELIVERABLE|\
MESSAGE.+NOT.+DELIVERABLE|\
FAILED[ \t]+MAIL|\
FAILED[ \t]+MESSAGE|\
MAIL[ \t]+FAILED|\
MAIL[ \t]+RETURNED|\
RETURNED[ \t]+MAIL|\
MAIL.*[ \t]ERROR|\
MAIL[ \t]+RECEIVED|\
MESSAGE[ \t]+RECEIVED|\
MESSAGE[ \t]+DELIVER|\
MAIL[ \t]+DELIVER|\
INTERCEPTED[ \t]+MAIL|\
WAITING[ \t]+MAIL|\
READ[ \t]+RECEIPT|\
RECEIPT[ \t]+NOTIFICATION|\
STATUS.+SIGNAL[ \t]+[0-9]+|\
^ERROR[ \t]+CONDITION[ \t]+RE:|\
^NO[ \t]+REQUESTS[ \t]+FOUND|\
AUTO[ \t]+REPLY|\
AUTOMATIC[ \t]+REPLY|\
AUTOMATICALLY[ \t]+GENERATED|\
AUTOANSWERED|\
WAITING.+MAIL|\
ON[ \t]+VACATION|\
VIA[ \t]+VACATION|\
CONCIERGE[ \t]+NOTICE|\
AWAY[ \t]+FROM.+MAIL|\
CAN[ \t]*['NO]*T.+ANSWER|\
CAN[ \t]*['NO]*T.+REPLY|\
OUT[ \t]+OF[ \t]+TOWN|\
OUT[ \t]+OF[ \t]+OFFICE
```

These try to correspond to the wide variety of error messages and automated responders, vacation program, etc. on the network. These are built into the code. The config file allows a site to expand further and tailor this set of suspicious subjects.

ignored_requests 'req1|req2|…'
This option allows the site manager to define requests that can be safely ignored and should not generate an error message. i.e.

```
ignore_requests 'this is a test|please ignore'
```

The argument is a | separated list of regular expressions.

erroranalysis <level> <grace-period>
ListProc performs error analysis on SMTP bounced mail. It attempts to determine the type of error and the originating user. It will auto-delete if possible. The error analysis takes two parameters:

level
1 or 9 where nine is of more stringent analyses

grace-period
the number of seconds to wait between the first bounce and auto deleting the offending user.

An example would be:

```
error_analysis 9 604800
```

Level 9 analysis will auto-delete more cases and types of error messages than level one. For a more conservative automatic deletion policy, select level 1.

ListProc will notify the list owner of all auto-deletes.

In addition, ListProc will postpone mail to users when the bounce is of an interim type such as:

```
Disk Quota Exceeded
```

In this case, the user's subscription mail mode will be reset to Postpone and the list-owner will be notified of the change.

sense_request 'req1|req2|…'
This option allows a site manager to filter out requests to a list, (i.e. subscribe <list-name><fname><lname>. If any of the regular expressions appear in the first line of the message, the user will get an error message stating that a request was sent to the list and should instead be sent to the listproc address. Sites can accept CREN's default or they may tighten or loosen the list as necessary. See listing on the next page.

```
(^[ \t]*Q?U?I?E?T?[ \t]*ADD[ \t]+([A-Z0-9.@-]+)[ \t]+.+[ \t]+.+)|&
(^[ \t]*ALIA?S?[ \t]+([A-Z0-9.@-]+)[ \t]+.+[ \t]+.+)|&
(^[ \t]*APPR?O?V?E?[ \t]+([A-Z0-9.@-]+)[ \t]+.+[ \t]+[0-9]+[ \t]*$)|&
(^[ \t]*CONF?I?G?U?R?A?T?I?O?N?[ \t]+([A-Z0-9.@-]+)[ \t]+.+[ \t]+)|&
(^[ \t]*Q?U?I?E?T?[ \t]*DELE?T?E?[ \t]+([A-Z0-9.@-]+)[ \t]+.+[ \t]+.+)|&
(^[ \t]*DISC?A?R?D?[ \t]+([A-Z0-9.@-]+)[ \t]+.+[ \t]+[0-9]+[ \t]*$)|&
(^[ \t]*EDIT[ \t]+([A-Z0-9.@-]+)[ \t]+.+[ \t]+[A-Z]+[ \t]*.*$)|&
(^[ \t]*FREE[ \t]+([A-Z0-9.@-]+)[ \t]*.*$)|&
(^[ \t]*GET[ \t]+[A-Z0-9/.-]+.+)|&
(^[ \t]*HOLD[ \t]+([A-Z0-9.@-]+)[ \t]*.*$)|&
(^[ \t]*SENDM?E?[ \t]+[A-Z0-9/.-]+.+)|&
(^[ \t]*VIEW[ \t]+[A-Z0-9/.-]+.+)|&
(^[ \t]*HELP[ \t]*[A-Z0-9]*$)|&
(^[ \t]*IGNO?R?E?[ \t]+([A-Z0-9.@-]+)[ \t]+.+[ \t]+.+)|&
(^[ \t]*INDEX[ \t]*[A-Z0-9/.-]*.*)|&
(^[ \t]*INFO?R?M?A?T?I?O?N?[ \t]*([A-Z0-9.@-]*)[ \t]*$)|&
(^[ \t]*INIT?I?A?L?I?Z?E?[ \t]+([A-Z0-9.@-]+)[ \t]*.*$)|&
(^[ \t]*NEW-L?I?S?T?[ \t]+([A-Z0-9.@-]+)[ \t]*.*$)|&
(^[ \t]*JOIN?[ \t]+([A-Z0-9.@-]+).+)|&
(^[ \t]*LIST?S?[ \t]*$)|&
(^[ \t]*LOCK[ \t]+([A-Z0-9.@-]+)[ \t]*.*$)|&
(^[ \t]*PUT[ \t]+([A-Z0-9.@-]+)[ \t]+.+)|&
(^[ \t]*PURG?E?[ \t]+([A-Z0-9.@-]+))|&
(^[ \t]*RECI?P?I?E?N?T?S?[ \t]+([A-Z0-9.@-]+)[ \t]*.*$)|&
(^[ \t]*REVI?E?W?[ \t]+([A-Z0-9.@-]+)[ \t]*.*$)|&
(^[ \t]*RELE?A?S?E?[ \t]*$)|&
(^[ \t]*VERS?I?O?N?[ \t]*$)|&
(^[ \t]*REPO?R?T?S?[ \t]+([A-Z0-9.@-]+)[ \t]+.+$)|&
(^[ \t]*RUN[ \t]+([A-Z0-9.@-]+))|" "&
(^[ \t]*SEAR?C?H?T[ \t]+([A-Z0-9.@-]+))|&
(^[ \t]*Q?U?I?E?T?[ \t]*SET[ \t]+([A-Z0-9.@-]+))|&
(^[ \t]*QUER?Y?[ \t]+([A-Z0-9.@-]+))|&
(^[ \t]*STAT?I?S?T?I?C?S?[ \t]+([A-Z0-9.@-]+))|&
(^[ \t]*SYST?E?M?[ \t]+([A-Z0-9.@-]+)[ \t]+.+[ \t]+#.+)|&
(^[ \t]*SUBS?C?R?I?B?E?[ \t]*([A-Z0-9.@-]*))|&
(^[ \t]*SIGN?O?F?F?[ \t]*([A-Z0-9.@-]*))|&
(^[ \t]*UNLOCK[ \t]+([A-Z0-9.@-]+)[ \t]*.*$)|&
(^[ \t]*UNSUBS?C?R?I?B?E?[ \t]*([A-Z0-9.@-]*))|&
(^[ \t]*SHUTDOWN[ \t]+.+$)|&
(^[ \t]*RESTART[ \t]+.+$)|&
(^[ \t]*EXEC?U?T?E?[ \t]+.+#.+)|&
(^[ \t]*WHIC?H?[ \t]*$)|&
(^[ \t]*REMOVE[ \t]+M[EY])|&
(^[ \t]*DELETE[ \t]+M[EY])|&
(^[ \t]*ADD[ \t]+M[EY])|&
(^[ \t]*PLEASE[ \t]+REMOVE[ \t]+)|&
(^[ \t]*PLEASE[ \t]+DELETE[ \t]+)|&
(^[ \t]*PLEASE[ \t]+U?N?SUBS?C?R?I?B?E?[ \t]+)|&
(^[ \t]*PLEASE[ \t]+SIGN[ \t]+)|&
(^[ \t]*PLEASE[ \t]+ADD[ \t]+)
```

default_arch_dir <dir>

This instruction specifies the default directory where archived files reside. The manager controls this and is the only person who can override it for individual lists. The system default, if this variable is not defined, is:

```
$LPDIR/archives/list-alias
```

The ListProc must have write permissions in the archive directory. ListProc will always maintain its indices under $LPDIR/archives, regardless of where the actual logs reside. You must have this directory on your system.

Remote Lists

ListProc maintains knowledge of remote lists in a file in its home directory called **remote.lists**. By installing this file locally, a user can subscribe, unsubscribe, change list options all through his/her local ListProc. This list is currently installed and updated manually, but CREN plans to automate this procedure for an interim release of ListProc.

A list of remote ListProc lists in the proper format can be picked up via anonymous ftp from ftp.cren.net in /archives/listproc/remote.lists. Or you may send mail to listproc@list.cren.net with the message:

```
get listproc remote.list.
```

This listing contains both VM Listserv and ListProc remote lists.

To register your mailing list with CREN, send e-mail to list-master@ListProc.net, and include the following information:

- The name of the mailing list
- The address of the server for subscription requests
- The address of the mailing list
- The IP address or hostname of the machine running ListProc
- A one line description of the mailing list
- The port where the remote server listens for connections (if different from the default).

The ListProc maintained by CREN will be updated with all the registered remote lists, as well as the VM Listserv remote list. You can point your users to listproc@listproc.net for a complete global list, as well as subscribing and unsubscribing. CREN will collect list information from registered sites and compile and update this list nightly.

The syntax of the remote.list file is:

remote list-name list-address server-address [remote-server port] #list description

The remote directive allows a local site to be aware of remote lists. Notice that the host name of the remote server is required if you wish to allow remote connections; otherwise e-mail will be used to forward all requests. It will return and search these lists in response to the user *list* command. It will also forward, *subscribe*, *unsub*, etc. end user commands to the remote list. An example of the remote command is:

```
remote i-unix i-unix@list.cren.net ListProc@cren.org
# Unix Disc. List
```

The remote-server and port directives are optional and specify the fully qualified hostname and port number of the remote server for live connections (Yes this can differ from the e-mail address). If no port number is specified, the default port is used.

The remote list file allows your end users one stop shopping. They can query, search and subscribe/unsubscribe from mailing lists simply by sending mail to your local server. You can also point your users to listproc@list.cren.net, which will have an up-to-date list of available mailing lists.

Server Environment Variables

Several Environment variables can be defined for the server. Some of these have been discussed before. There will be system specific differences in how to define these variables which are beyond the scope of this manual.

- File Creation Mask

You must have a umask of 077 for listproc. This is critical for ListProc to maintain file security. This cannot be stressed enough.

- LPDIR

This variable defines the default directory for ListProc.

- MD5

If this variable is defined, ListProc will use the MD5 algorithm for checksums instead of the system call sum. MD5 provides greater width in calculating checksums which in term is one of the methods used for loop detection. MD5 is available from the anonymous ftp server at info.cren.net in /pub/software/MD5 Please note that MD5 must be in the server's search path.

- ULISTPROC_ARCHIVES_UMASK=[MASK Value]

This variable establishes separate file permissions for the ListProc archives directories and files. This needs to be other than 700 if you are to provide anonymous ftp/gopher or other server access to the archives. This mask affects only the archives. A mask of 022 creates archives with permissions 644 which is needed for gopher, wais and anonymous ftp access.

To set any of the aliases, do the following:

for csh:

```
setenv ULISTPROC_ARCHIVES_UMASK 022
```

for bourne shell:

```
set ULISTPROC_ARCHIVES_UMASK=022;
export ULISTPROC_ARCHIVES_UMASK
```

System Mail Alias

For the ListProc to receive and process mail, you must set up a system-wide alias for it. Usually this file is /etc/aliases or /etc/mail/aliases, although it can reside in different places. At this point, root access is needed.

The alias is a pipe to a process called *catmail*. A detailed description of *catmail* is provided in Appendix C. The server alias should be as follows:

```
ListProc: "|/home/server/catmail -r -f"
```

Do not forget to run the system command *newaliases* to rebuild the alias database.

As stated before, everything except a list parameter should be defined in this file. A default file is provided for you to look at and use as a template to customize to your site.

Your mailer must allow you to launch programs. Some versions of sendmail, as a security precaution, do not allow programs to be executed by the mailer. They implement this via some form of restricted shell. It is important that you or your sysadmin familiarize yourself with your particular implementation. A detailed explanation of mailers is beyond the scope of this manual. For sendmail documentation, see *sendmail* by Bryan Costales with Eric Allman and Neil Rickert.

5. Defining a List

In defining lists, both the site manager and the list-owner have responsibilities. The site manager is the only person who can create a mailing list. Also, root access is required because you must add system wide mail aliases for both the server and list to receive mail from the Network.

Creating a Mailing List

The principle tool in defining lists is the *initialize new-list* command. This is how a manager creates a list on the system. Only the manager can issue this command. The syntax of this command is:

```
initialize <manager password> <list alias> &
<list address> <configuration-options>

new-list <manager password> <list alias> &
<list address> <configuration-options>
```

You can define a new mailing list by specifying its name, full address and configuration options. For a detailed description of configuration options please see the next section, **Configuring a Mailing List**. *Initialize* uses the same options and syntax for options as does the *configuration* command. You must define at least one parameter for a mailing list to be created; for example, this will not work:

```
init passwd test test@host.domain
```

This example will work:

```
init passwd test test@host.domain archive, &
owners foobar, archive, hidden, owner-subscriptions, &
auto-delete, password test1, digest daily 12:00, &
reply-to-list-always, message-limit 50
```

Note that the & is the continuation character for ListProc commands. It must be the last character on the line other than the carriage return.

The minimum required to create a list is as follows:

```
init password test test@host.domain owners
foo@host.domain
```

Based on this command, ListProc will create a list which is owner-controlled, has no archives, handles headers using the reflector option, is visible, anyone can post, anyone can subscribe, has no digests or message limit, and uses the list-name, in uppercase, as the list password.

Configuring a Mailing List

List owners can change list parameters except those marked with an asterisk (*) at any time without restarting the server daemon. This provides owners with a wide range of options and flexibility.

When owners submit a series of configuration parameters and one or more

contains an error, all valid configuration parameters will be applied and the owner(s) will be notified.

Once the site maintainer/manager creates the mailing list, the defined list-owners then initialize the list parameters using the *config* command. This command has many options as listed below.

```
configuration <list> <password> [<option> [args] [,
<option> [args]…]]
```

Configuration gets or sets a list's configuration options. Without any of the options below, you get the current settings. The following command:

```
configuration list-name list-password
```

returns the list's current settings.

Options are a comma-separated list of one or more of the following keywords (options marked with ◆ can be issued by the manager only; **<>** specify required arguments, [] specify optional arguments, and | separates choices; options can be abbreviated to the shortest non-ambiguous strings, indicated by all caps):

- ◆ **ARChive [dir] [filespec] [arch-name] [password] [messages|digests]**
  This turns on archiving for said list by the manager, optionally specifying a destination directory for files, a file specification, an archive name, password or whether messages or digests are to be archived. By default, every list's messages are stored in **default_arch_dir** as defined in the config file, as logyymm, with the archive named after the list, and no password. Filespecs can contain any lower case characters of the alphabet, numerals, the underscore and one or more of the following special character sequences:

  %a contents of Archive-Name header line (cannot be used if archiving digests)

  %c the current message count

  %d day of month (01 - 31)

  %h month name (Jan - Dec)

  %j julian date (001 - 366)

  %m month (01 - 12)

  %n issue number (extracted from Volume # Number # line)

  %v volume number (extracted from Volume # Number # line)

  %w week number of month (a - f)

  %y year (00 - 99)

  %1 first word of first non-blank line of file

  %# digest number (if archiving digests)

  %% the character %

A dash (-) can replace any of the arguments above as a place holder for the default parameters.

For example, the system default is defined as log%y%m. %a expects the **Archive-Name:** header line in the header section or anywhere in the message body. (That header line specifies the archive name to store to. This is how various "source" news groups archive messages.)

%v and %n are usually specified together and expect the first line of the body of the message to be something like this:

```
Volume 3 Number 45
```

This is useful when archiving an electronic newsletter or magazine.

```
arc /ftp/list-archives/test log%y%m messages
```

stores all list archives in message format instead of digest in a directory called /ftp/list-archives/test as a file named according to the month and year. If it were December 1994, the file would be named log9412.

```
arc /ftp/list-archives/misc %a
```

will store archives only if the keyword *archive-name:* appears as a header or on a separate line in the body of the message with the specified file name. If you have

```
archive-name: census-data-1994
```

the file name would be *census-data-1994*.

ARCHIVE [password] [messages|digests]
Turn on archiving for said list by the owners (see above).

NO-ARCHIVE
Turn off archiving of messages or digests.

- **COMPRESSED-ARCHIVES**
  Keep logs in compressed format.

- **UNCOMPRESSED-ARCHIVES**
  Keep logs in uncompressed format.

ARCHIVES-TO-ALL
Allow anyone to get the list's index of logs, and the logs themselves.

ARCHIVES-TO-OWNERS
Only owners have access to the list's index of logs and the logs themselves.

ARCHIVES-TO-SUBSCRIBERS
Only owners and subscribers have access to the list's index of logs and the logs themselves.

AUTO-DELETE-SUBSCRIBERS
Turn on auto-deletion of subscribers whose addresses cause delivery problems.

ListProc will attempt to determine the source of the bounced mail and the severity of the error. If ListProc determines that the error is unrecoverable (i.e. No Such Local User), it will auto-delete the subscriber. If ListProc detects that the error is transitory (i.e. Disk Quota Exceeded), ListProc will set the user's mail option to *postpone*. In both cases the list owner will be notified of these actions.

NO-AUTO-DELETE-SUBSCRIBERS
Turn off auto-deletion.

CLOSED-SUBSCRIPTIONS
Do not accept any more subscriptions. This is different from owner subcriptions (see below) because this command will simply not accept any subscription requests. The person requesting subscription will be notified that no subscriptions are being accepted for the list. This command is useful when temporarily reorganizing or moving a list or making administrative changes.

OPEN-SUBSCRIPTIONS
Open up subscriptions again. This command is issued after the closed-subscriptions (see above). This and *closed-subscriptions* are useful when temporarily reorganizing or moving a list or making administrative changes.

OWNER-SUBSCRIPTIONS
Subscriptions are monitored by the owners or the subscription managers who get a chance to approve them or not. (see below).

WIDE-OPEN-LIST
Open up subscriptions to anyone, make the list visible when users issue a list command and allow anyone to review the list, get statistics on usage and get access to the list's archives. This option also allows anyone (whether subscribed or not) to post to the list.

A wide open list is equivalent to:
- visible
- send-by-all
- open-subscriptions
- archives-to-all
- review-by-all
- reflector.

COMMENT <string>
Specify the one-liner content of header line **To:** line. For example, if the following comment is defined:

```
config happy-list password comment Happy Happy Joy
Joy Mailing List
```

The list from header will appear as such:

```
To: Happy Happy Joy Joy Mailing List <happy-
list@joy.com>
```

To have the ListProc not put any comment in the **To:** line, define a null comment such as:

```
config happy-list password comment ""
```

The **To:** line will now appear as such:

```
To:  <happy-list@joy.com>
```

The following characters are not allowed in the comment field: [^<>'*?|,\n] Some characters will need to be escaped to go into the comment field, characters such as quotes and colon.

NO-COMMENT
If a comment line was previously defined, undefine it, and define no comment field; this is the default for a new list. In this case, ListProc will place the *familiar* Multiple Recipients of List in the **To:** field of all list mailings. The **To:** line will now appear as such:

```
To: Multiple Recipients of List <happy-list@joy.com>
```

DEFAULT <mailmode> <value> [<mailmode> <value>]…
address variable | fixed
mail ack | noack | postpone | digest
password <string>
conceal yes | no
*preference CCoption (*The preference field applies to list owners defined after this change is put in) The initial list owner is given the default preference of CCERRORS, see below*)

Set the default subscription options; any of the above values can be the empty string "" or '' which stands for the system default. The system defaults are:

address fixed
mail ack
system generated password (as opposed to a default)
conceal no

The meaning of each option is explained below:

address variable
A list subscriber can change the address he/she is subscribed from by sending the following command to ListProc:

```
set list-name address current-password new-address
```

address fixed
A list subscriber must unsubscribe and resubscribe from the new address in order to change the address from which he/she is subscribed and can thus issue commands.

mail ack
A list subscriber will receive a copy of his/her posting to the list.

mail noack
A list subscriber will not receive a copy of his/her posting.

mail digest
If the digest option is set for this list, the subscriber will only receive the digest, not individual postings.

mail postpone
The subscriber will not receive any postings or digest.

A subscriber can change any options with the following command to ListProc. The following sets mail to digest:

```
set list-name mail digest
```

password
This option sets a default user password upon initial subscription. If not set, a random system generated password is assigned to each user. The need for a password arises from the interactive use of ListProc. The password is the only means to authenticate a user. The quotation mark, comma, and apostrophe (" , ') cannot be used in valid passwords. A list owner can change listproc to generate random passwords for subscribers with the following command if the list were set to generate a standard password previously:

```
conf list-name list-password default mail password ""
```

conceal
This determines whether or not the user's name and e-mail address will be shown during a review command. (The ability for other subscribers/non-subscribers to see who is subscribed to the list).

preferences
These are list owner preferences which the list owner can set by sending a set command. For example, for a list owner to change his/her options from the default CCERRORS to -CCERRORS, he/she would send the following set command to ListProc:

```
set list-name preferences -CCERRORS
```

Any preference that you set can be unset by including a minus sign (-) before the preference. You can use the following command to check your preferences:

```
set list-name
```

The following are valid for list owner(s):

CCERRORS
The list owner will be copied on every listproc error message generated.

CCSET
The list owner(s) will be copied on every set command that a subscriber issues (i.e. when changing mail options to digest).

CCSUBSCRIBE
The list owner(s) gets copied on all subscription requests.

CCRECIPIENTS
The list owner gets copied on all get requests for archives.

CCINFORMATION
The list owner(s) get copied on all info requests.

CCSTATISTICS
The list owner(s) get copied on all statistics requests.

CCPRIVATE
The owner(s) is notified when a message is rejected because the list is set up so that only subscribers can post, and a non subscriber attempts to post. It is good to debug addresses.

CCRUN
The manager gets copied on all run requests.

CCIGNORE
The manager is notified when a message is ignored for whatever reason.

CREVIEW
The list owner(s) gets copied on all review requests.

CCALL
The list owner(s) and/or ListProc Maintainer gets copied on all of the above which are applicable. Be cautioned that this one especially can generate a lot of mail.

The following are valid for ListProc Managers:

CCGET
Copies of all get requests are sent to the manager.

CCINDEX
Copies of all index requests are sent to the manager.

CCLISTS
Copies of all list request are sent to the manager.

CCRELEASE
Copies all release requests.

CCHELP
Copies of all help requests are sent to the manager.

CCALL
See above.

DELIVERY-ERRORS-TO <address> [address]…
Specify the recipients of delivery error messages. Delivery error messages are mailer error messages, as opposed to error messages generated by ListProc. The address(es) specified on this option will have the power to remove users (hopefully only as a result of bounced mail). By default, this is the list owner. Each new address is separated by a space.

REMOVE-ERRORS-TO <address> [address]…
Remove address(es) from the above list. Each new address is separated by a space.

DIGEST <frequency> [when] [lines bytes]
daily hh:mm
weekly Sunday | … | Saturday
monthly

Turn on digesting of messages and distribute them at the specified time or day (monthly digests are distributed on the first day of the month). The hh:mm is hour and minute in 24 hour or military time, so to send off digests at one minute after midnight daily the following would be the config option:

```
config list-name password digest daily 00:01
```

If you specify the number of lines or size in bytes (or both), digests will be distributed as soon as these limits are exceeded. Thus it is possible to distribute multiple digests within a specified timeframe. If changing these limits only, [when] can be substituted with a minus sign (-) as a place holder, one (-) (covers all cases). For example, to send out digests when 50,000 lines are met, the following would do the trick.

```
config list-name password digest monthly 5000 0
```

Notice how the time is monthly, if the list is relatively busy, the prevailing trigger for digests will be the size not the time. Also notice how the byte count is zero, again making the predominant trigger the number of lines, in this case 5000 lines.

NO-DIGESTS
Turn off collections of digests.

DISABLE <request> [request]…
Disable use of the specified request(s). For example, to disable the ability of anyone to review the list, the following command would be sent:

```
conf list-name password disable review
```

ENABLE <request> [request]…
Enable use of the specified request(s).

SET-DISABLE <mode> [value] [<mode> [value]]…
Modes are as follows:
mail ack | noack | postpone | digest
conceal yes | no
password

Disable the specified SET value(s). In other words, do not allow individual subscribers to reset these options from those provided in the list defaults. If the list owner wants to maintain a policy of no hidden users, the following configuration option would achieve that:

```
conf list-name password set-disable conceal yes
```

SET-ENABLE <mode> [value] [<mode> [value]]…
mail ack | noack | postpone | digest
conceal yes | no
password

Enable the specified SET value(s). This is the opposite of SET-DISABLE. To reset the ability of users to conceal themselves, send the following command:

```
conf list-name password set-enable conceal yes
```

FORWARD-REJECTS
Forward all ListProc-generated error messages to the owners. ListProc may generate an error message if someone attempts to post an article that contains a suspicious subject or if a non-subscriber posts to a list which is set to accept postings from subscribers only .

DONT-FORWARD-REJECTS
Forward all ListProc-generated error messages to the senders. This is the default.

KEEP-RESENT-LINES
Keep Resent: header lines.

DONT-KEEP-RESENT-LINES
Don't keep Resent: header lines.

HIDDEN-LIST
Hide the list from "lists" and "index" requests. ListProc makes certain assumptions about hidden lists and sets the following other options automatically. It restricts access to archives, review and statistics to subscribers and owners only. If you want your hidden list to exhibit other characteristics, you must explicitly set the corresponding options.

VISIBLE-LIST
Turn on visibility for lists and index requests, but do not affect the rest of the options.

- **MANAGER-CONTROLLED**
  Only the manager can remove owners and change the list password. This lets site managers temporarily take control of a list while changes are being made in ownership or policy or to enforce a policy that the owner be known and a member of the institution. When a list is manager controlled, an owner can't change the owner, archive, or password options.

- **OWNER-CONTROLLED**
  Reset normal operation. This is the default.

MAX-MESSAGES-PER-DAY <number>
MESSAGE-LIMIT <number>
Specify the maximum number of messages to be processed in one day (can be reset with the FREE request). The list owner will be notified when this limit has been reached. We recommend setting the message limit as a last defense against any loops that ListProc does not catch. If you have a busy list, set this to a high value.

NO-MESSAGE-LIMIT
Turn off the MAX-MESSAGES-PER-DAY and MESSAGE-LIMIT restriction.

MODERATED-EDIT [address]…
Make the list moderated by providing the ability to edit messages, and optionally specify moderator addresses; otherwise messages will be sent to the owners for approval.

MODERATED-NO-EDIT [address]…
Make the list moderated by letting the moderators simply approve or discard submitted messages, and optionally specify moderator addresses; otherwise messages will be sent to the owners for approval.

UNMODERATED
No postings are moderated.

REMOVE-MODERATORS <address> [address]
Remove moderator address(es). Each new address is separated by a space.

OWNERS <address> [address]…
Specify new owner addresses. Each new address is separated by a space.

REMOVE-OWNERS <address> [address]…
Remove the specified owners. Each new address is separated by a space.

PASSWORD <string>
Reset the list's password to the new string.

SUBSCRIPTION-MANAGERS <address> [address]…
Specify who will approve subscriptions when the owners decide not to monitor subscriptions (default is the owners). This applies to lists where the subscription option is set to owner approved. A subscription manager has no privileges beyond approving subscriptions, and adding users to the list. Subscription managers change list options or approve postings if the list is edited. Each new address is separated by a space.

REMOVE-SUBSCRIPTION-MANAGERS <address> [address]…
Remove such addresses.

REMOVE-ALL-SUBSCRIPTION-MANAGERS
Short cut for the above, it removes all the managers.

REPLY-TO-LIST
Set the **Reply-To:** header line to the list's address unless a **Reply-To:** is present in the sender's message, in which case it is preserved.

REPLY-TO-LIST-ALWAYS
Set the **Reply-To:** header line to the list's address all the time.

REPLY-TO-SENDER
Set the **Reply-To:** header line to the From: address unless a **Reply-To:** is present in the sender's message, in which case it is preserved.

REPLY-TO-SENDER-ALWAYS
Set the **Reply-To:** header line to the From: address all the time.

REPLY-TO-OMITTED
Do not put a **Reply-To:** header line, unless a **Reply-To:** is present in the sender's message, in which case it is preserved.

REVIEW-TO-ALL
REVIEW-BY-ALL
Open review to everyone.

REVIEW-TO-OWNERS and **REVIEW-BY-OWNERS**
Restrict review to owners only.

REVIEW-TO-SUBSCRIBERS
REVIEW-BY-SUBSCRIBERS
Restrict review to subscribers and owners only.

STATISTICS-TO-ALL
STATISTICS-BY-ALL
STATS-TO-ALL
STATS-BY-ALL
Open statistics to everyone.

STATISTICS-TO-OWNERS
STATISTICS-BY-OWNERS
STATS-TO-OWNERS
STATS-BY-OWNERS
Restrict statistics to owners only.

STATISTICS-TO-SUBSCRIBERS
STATISTICS-BY-SUBSCRIBERS
STATS-TO-SUBSCRIBERS
STATS-BY-SUBSCRIBERS
Restrict statistics to subscribers and owners only.

SEND-BY-ALL
Open up postings to everyone, including non-subscribers.

SEND-BY-OWNERS
Restrict postings to owners only (one-way list). This does not imply a moderated list.

SEND-BY-OWNERS-CONFIRM
Restrict postings to owners only, in addition, the owner must place the following line in the body of the message:

```
confirm: list-owners-password
```

substitute the word subscribers-password with the list password. ListProc will remove this line prior to distributing mail.

SEND-BY-SUBSCRIBERS
Restrict postings to subscribers and owners only.

SEND-BY-SUBSCRIBERS-CONFIRM
Restrict postings to subscribers and owners only. Additionally, the following line is required in the body of the message:

```
confirm: subscribers-password
```

where *subscribers-password* would be substituted with the user's password for that list. ListProc will remove this line prior to distributing the messages.

REFLECTOR
This option makes the list processor's headers appear as a sendmail reflector. It will pass **To:** and **CC:** as both received, and no comment, and will exclude a pre-set **Reply to:** (i.e. You cannot have *reflector* and *reply-to-list*). This is the default and must be explicitly turned off when creating a new list.

NO-REFLECTOR
Turns off the reflector option. Owner example:

```
configuration test pass1 archive - messages, &
archives-to-owners, review-by-all, auto-delete, &
owner-subscriptions, comment Test List, &
default mail ack conceal no password XyX &
preference CCSUBSCRIBE preference CCUNSUBSCRIBE, &
set-disable mail digest conceal yes password, &
delivery-errors-to root@your-site server@your-site, &
digest daily 15:30, message-limit 100, &
moderated-edit you@your-site, owners foo@bar.com, &
subscr foo@your-site, reply-to-list-always
```

Manager example:

```
conf test pass1 archive /tmp - - - digests, &
compressed-archives, message-limit 100
```

Setting the List Aliases

Three mailer aliases must be defined for each list, these are:

- list-name
- owner-list-name
- listname-request

These aliases are piped to *catmail* (see appendix C for a detailed description of catmail). The following examples show how a typical list would be setup:

```
list-name: "|/LPDIR/catmail -L LIST-ALIAS -f"

owner-listname: "|/LPDIR/catmail -L LIST-ALIAS -f -e"

listname-request: "|/LPDIR/catmail -L LIST-ALIAS -f -o"
```

Please remember that LPDIR should be replaced with the full path to server's home directory, where the binaries exist.

Note: AIX users may want to use *smit* to set the mailer's alias database. The location of the mail aliases file and format will vary depending on your O/S and MTA

6 List Management

The configuration command is one of the primary tools for list management, dealing specifically with list and server parameters. A number of other tools allow a list owner(s) to perform actions on behalf of subscribers, delete and undelete subscribers, etc. When references are made to password below, they specifically refer to the list password, not the site maintainer or the end user.

Commands sent to ListProc are limited to 1024 characters in length. If you have a longer command, you must split it up.

[quiet] add <list> <password> <address> <name>
[quiet] add <list> <password> {<address> <name>} {<address> <name>}…
Add the specified user(s) to the list. If more than one user is added, you need to enclose each one in {}. If quiet, the user(s) will not be notified.

approve <list> <password> <tag> [tag]…
Approve the message identified by the tag number for posting to the specified moderated list — applies to moderated-no-edit lists only.

[quiet] delete <list> <password> <address> [address]…
Delete the specified address(es)/users from the list. If quiet, the user(s) will not be notified.

discard <list> <password> <tag> [tag]…
Discard the message identified by the tag number sent to the specified moderated list. This applies to moderated-no-edit lists only.

edit <list> <password> <file> [-nolock]
Obtain the specified file for editing; file can be one of: subscribers | aliases | news | peers | ignored | info | welcome i.e. to retrieve and edit the lists info file for list test, use the following command:

```
edit test test-password info
```

Without *-nolock*, the file will be automatically locked and no more list-specific requests that may alter the file will be processed, from anybody else, until the list is *UNLOCKed*, or the file is *PUT* back.

free <list> <password>
Resume delivery of a held list; it resets the message-limit to zero. If the manager held the list, only he can free it.

hold <list> <password>
Suspend distribution of messages.

lock <list> <password>
Suspend execution of list-specific requests and queue them up for later processing. Owners may still issue such requests unless the list is locked by the manager.

put <list> <password> <keyword> [args]

Manipulate addresses and change system files. Keywords and args can be:

keyword:
alias | ignore | subscribers | aliases | news | peers | ignored | info | welcome

args:
e-mail address for alias | ignore

When putting a file, its contents start immediately after the request and span the entire message. For example, to put a short info file for the test list:

```
put test test-password info

This list is set up to test ListProc.
```

That short sentence would now become the info file for the list.

Alias and *ignore* allow you to set up a list so that user A can post and perform other actions from alternate e-mail addresses. They are especially handy for admins who wish to use multiple e-mail accounts to administer the list but only one account to receive mail, error messages, etc…

For example:

```
put test test-password alias marco@foo.bar
marco@foo.zee
```

This would treat the account marco@foo.zee as equivalent to the owner/subscriber account marco@foo.bar.

The ignored file allows you to "server off" troublesome user accounts. It is similar to aliases. For example, if a listowner wishes to prevent pesky@pain.domain from participating in this forum, you would add this entry to the ignored file.

```
put test test-password ignore pesky@pain.domain
```

There is also a global alias and ignore file which the site administrator controls and which overrides all individual list ignore and alias files.

alias list-name list-password expr1 expr2
ignore list-name list-password expr

These commands work like the put command above but are much easier to remember and use. See *Alias and Ignore* in Chapter 7 for a more detailed explanation. Expr# are any regular expressions. For example, if you want to ignore all postings from pesky@anyhost.domain, the following example highlights how to do this:

```
ignore test-list test-list-owner-password
^pesky@*.domain
```

reports <list> <password>
Obtain all reports about the specified local list.

[quiet] set <list> [<option> <arg[s]>] for <address> [address]…
Same as 'set' above but applies changes to specified address(es). If quiet, the user(s) will not be notified.

system <list> <password> <user-address> #user-request
Issue any request on another user's behalf.

unlock <list> <password>
Resume execution of list-specific requests, including those queued up while the list was locked. All owners may unlock a list, unless it's locked by the manager.

purge <manager password> <address> [address]…
The purge command allow the ListProc manager to remove a particular user or group of user-id's from all lists on the local server. This would be particularly useful at the end of a semester when student accounts are erased.

List subscribers can also purge themselves from all lists on a particular server by issuing the purge command, *purge* **<***any list password***>**.

Archiving Files

ListProc provides a utility called *farch* to archive files. *Farch* is used to archive files under the ListProcessor system or remove files already archived, with support provided for private archives. Archives reside under LPDIR/archives as subdirectories; the default is **ListProc**, and it is also the master archive. Archives are hierarchically structured, and each archive has at least two files in its directory—**INDEX** and **DIR**.

INDEX
The index contains a list of all subarchives including itself. Each archive is on it's own line. The format is:

```
archive-name full-path-to-archive-directory [password]
```

The first entry should be the archive itself followed by each first level subarchive; then each second level subarchive, starting with the first first-level-archive's subarchives, followed by the second first-level-archive's subarchives, etc. No sibling archives may have the same name, although there is no such restriction for archives on different levels, or "cousin" archives. The latter archives are distinguished by the relative path needed to access them.

An archive name may be start with the special character % to designate it for AFD/FUI files.

DIR

This is a list of files available from that archive. Each file is on it's own line. The format is:

```
file nparts size-of-each-part+ full-path-to-directory
```

```
[description [\]] [multiple description lines [\]]*
```

The descriptive message may span several lines and each line (except the last) should end with a '\':

The number of parts the file is split into may be -1 to indicate that this is a binary file.

farch reports on the action taken on each input file: whether it was split (and how many parts), whether it was uuencoded, whether all files have been tar'red into a single one, the archive this file was archived under, and the directory the output file(s) (if any) has/have been placed. It also reports when the input files are tarred.

Options

The following command line options are recognized:

-n Do not split files when archiving them. The default is to split them with each part not larger than the specified size (see the -s option below).

-b Input files are binary; they are uuencoded before being archived.

-B Input files are binary; they are neither uuencoded nor split.

-s<size> Specify the maximum size in kilobytes of each of the subparts (default is 64).

-d<dir> Specify the directory in which the output files are to be placed; if left out, it defaults to the specified archive's directory (LPDIR/archives/listproc if the archive is left out as well). If the directory (and all of its subdirectories, if any) does not exist, it will be created.

-p<password> When a new archive is to be created, and that archive is to be private, specify the access password.

-D<description> Put description (most likely surrounded by single quotes) in the DIR file as explanatory comment about the archived file. This option does not make much sense when archiving more than one file at the same time.

-t<file> Input files are tar'red into a file which is then archived (the -b flag is automatically turned on). Turning on the -B flag will prevent the uuencoding of the tar file. Note that whatever the path to the file may be, it will be moved to the specified directory as noted by the -d flag.

-u Update existing files while archiving; preserve the current comment fields, unless -D is explicitly used.

-r Remove the specified file(s) from the specified archive (see below), or the default one; it has higher priority than all of the options above.

-a archive | path-to-archive
Specify the archive that the input files will be archived under (default is listproc). Archives on different levels of the hierarchy may have the same name and in this case a path-to-archive may be specified to distinguish between them; path-to-archive has the form archive[/archive[/archive...]]. If any of the archives in the path do not exist, they will be created with the necessary files.

-A archive | path-to-archive
Same as -a, but it marks the archive ready for AFD/FUI files. To switch this mode off, simply specify the -a flag and vice versa.

-Z Turn off file compression.
File names in the DIR file have to be unique and *farch* will not archive duplicate files.

Archiving a File

Farch by default splits the input files if necessary (or the tar file, if any), into files of maximum size as specified. Each part will contain as many complete lines from the original input file as possible, without exceeding the specified size. Binary files (including the tar file) are uuencoded before they are archived unless the -B option is specified, and all archived files are compressed, if possible.

Adding a New Archive

automatically
Simply specify the new archive as argument to the -a flag. All necessary DIR and INDEX files as well as all required directories will be created, and all parent archives will be updated.

manually

1. Create a directory under LPDIR/archives or under an existing archive's directory (beware of the hierarchy structure), naming the directory after the archive. The archive's name should be unique among its siblings on that level in the hierarchy (besides, *mkdir* will not let specify a name that already exists).

2. Edit the master archive's index file **LPDIR/archives/listproc/INDEX** and add the new archive along with its path (and optional password — see below) at the proper place in the file, making sure you adhere to the rules about hierarchy outined above. The first line of every **INDEX** file should be the archive itself. Also edit (add to) every ancestor archive's **INDEX** file, if the new archive is not the default, again making sure you adhere to the same rules.

3. Create a new INDEX file in the new archive's directory and put an entry for itself.

4. Create an empty DIR file in the new archive's directory.

Private Archives

Private archives are archives that require a password for obtaining indices and/or files from them. To make an archive private, simply append a password (case does not matter) after its full path specification in every **INDEX** file the archive is defined (its own plus all ancestors' — the same password has to be used in all of these files). Then, that password has to be made known to all users who are to be granted access to this archive. Note that all files in the same private archive can be obtained with the same single password.

Examples

Archive src/data and src/data2 under archive listproc (the default), using a maximum file size of 1K; the output file(s) will be placed in /tmp/tmp:

```
% farch -s 1 -d /tmp/tmp src/data src/data2
```

Archive /etc/hosts under archive unix (that is listproc/unix) — the -n flag is used to avoid writing split parts to /etc which is doomed to fail:

```
% farch -n -a unix -d /etc -p private /etc/hosts
```

Archive /etc/password under archive pub/unix (that is listproc/pub/unix):

```
% farch -n -a pub/unix -d /etc /etc/passwd
```

Tar and archive all files in /usr/src to the default archive and place the tar file under /tmp/tmp:

```
% farch -t LPDIR/source -d /tmp/tmp /usr/src/*.c
```

Descriptive messages about these files are added manually into the archive's DIR file. However, the -D option can be used to specify a string as follows:

```
% farch -D 'ListProcessor system files' -t
ulistproc.tar -n -d /usr/server /usr/server/*
```

Upgrading

If you are upgrading from:

6.0 Functionality and flags are the same.

5.5 The -B, -p, -r and -D options are new; functionality is the same.

5.41 or less

Every **DIR** file should now include the size of each of the parts of every file, placed between the number of parts and the path name.

The -a option has been extended to accept paths to archives.

Restrictions

Archive names and input files must use only lower case characters of the alphabet.

Warnings

Input files that are to be tar'red should only include relative path names; otherwise the end user may not be able to extract them.

The tar file should not be any of the input files.

see also server(1)

Subscribing to Files

```
afd <action> {<archive> [/password] [files]}
[{<archive> [/password] [files]}]

fui <action> {<archive> [/password] [files]}
[{<archive> [/password] [files]}]
```

Automatic File Delivery (AFD) and File Update Information (FUI). Add/delete yourself or other addresses to archives and files, and be notified when these files are updated. "action" can be: *add*, *delete* or *query*. Only the manager may manipulate other addresses and he need not specify archive passwords if issuing these requests via ilp. The system-wide password may be used if sending e-mail. User addresses may be listed after the archive specification(s).

AFD ADD will add your address to the specified file(s) of the specified archives notify lists, and when these files are updated a copy will be sent to you. You may add yourself to an archive only instead of specified files, and you will be sent any updated files from that archive.

AFD DELETE will delete your address from the notify lists of the specified archives an/or files.

AFD QUERY will tell you which archives and/or files you have subscribed to.

FUI is similar to *AFD*, but instead you are notified when a file is updated, instead of being sent a new copy of the file.

Examples, regular user:

```
afd add { doc /pass1 ref.ps user.ps } { cooking }
```

This will add your address to archive **cooking's** notify list and you will be sent files **recipes** as updated, and to archive doc's notify list, but only for files "ref.ps" and "user.ps" — any other files updated will not be sent to you. This example assumes that doc is a private archive protected by password *pass1*.

```
afd delete { doc ref.ps }
```

This will remove your address from archive doc's notify list for file ref.ps.

```
fui query { doc }
```

This will return you a list of files you are subscribed to in archive doc.

Examples, manager:

```
quiet fui add { listproc } foo@bar.com user@foo.com
```

This will add the two addresses to listproc's notify list, but will not send an acknowledgment to them.

```
fui review { listproc file1 } { cooking }
```

Get a list of users subscribed to the specified archives/files.

7 ILP

Ilp stands for Interactive ListProcessor It allows you to log into the ListProc and issue commands, change the list parameters and see the results immediately. See Appendix D for additional information on Ilp.

Ilp requires that the site manager, list owner and regular users have IP connectivity. It is a command line interface that you enable simply by typing:

```
ilp hostname port
```

To connect to CREN's ListProcessor, the command line would be as follow:

```
ilp yukon.cren.org
```

You do not have to specify a port address since we are using the standard port for ilp. Ilp will present the following prompt:

```
Hit ^\ (control-backslash) to abort at any time.

CREN interactive ListProcessor, (c) 1993-94 by CREN.
All rights reserved.
Version 7.0 Mon Jul 18 14:23:30 1994
Maximum connection duration is 1800 seconds.

Welcome!

The set of requests you may issue during this session
depends on the privileges
assigned by the system. You may log in as a:

-Subscriber: provide your e-mail address and password
as subscribed to one of the local lists.

-Owner: simply provide your e-mail address and your
list's password.

-Manager: provide your e-mail address and the
system's password.

-Other: simply hit return at the prompt below.

e-mail address [none]:
```

At the e-mail address prompt, you type in the e-mail address by which you are known to the system. This could be a list owner's address, a site manager's or subscriber's address. If you enter no e-mail address, you will have the minimum privileges on the system. The next prompt:

```
Password:
```

is your access password. Again this could be your subscriber's, list owner's or site manager's password. If you do not enter a password, you will have minimal privileges on the system. This password will not be echoed to the screen, and **you do not need to type the password again after initial authentication**. For all future commands, you can substitute a - for the password, i.e.

```
conf list-name -

conf list-name password
```

are equivalent.

Ilp should be made publicly available. For more detailed information on how to use the interactive capabilities of ListProc, see Appendix D.

As a list owner, it provides a handy way of getting information about your list and pipes and it allows redirect. The angle brackets (< >) indicate the directional flow of the information. For example to find out if jdoe@foo.bar is subscribed to the list, the following will work:

```
review list-name | grep -i jdoe

conf list-name - > list-name.config
```

The first command will return only those subscribers where the string *jdoe* (UPPER and lower case) appears. The second command will return the list configuration (if you have logged in as a list owner) and redirect the output to a file, thus allowing you to edit this file and send it back to ListProc with some configuration changes as such:

```
conf list-name - < list-name.config
```

News–Mail Gateways

ListProc provides mechanisms for setting up one and two-way mail to news gateways. A one-way gateway means that postings to a particular newsgroup will be forwarded to the mailing list but postings to the mailing list will not be forwarded to the news group. A two-way gateway means that postings to/from the mailing list/news group will travel across the gateway.

There are two prerequisites to setting up a News-Mail gateway :

- You must have access to a news server (the machine that will act as the news gateway) that is running *CNews* or *INNd*. The machine where the newsgroup resides must have adequate permissions to Post articles to the server (for two way gateways)
- You must be running *inews* or have access to mail-news gateway running appropriate software like *mail2news* or *gatenews*. Listproc does not use these gateways directly but simply sends e-mail to them.

Your config file must be set up for the appropriate entry, either option post_mail or gate_mail depending on whether or not you are using *inews* or *gateway* software.

Using *inews*, the procedure for setting up a news gateway is rather clear. The *inews* package comes with the nntp source, which is available via anonymous ftp from a number of sites, including ftp.uu.net. You must compile *inews* for a client machine if the machine on which the mailing lists are running is not your news server. If ListProc is running on your news server, *inews* is on this machine.

ListProc provides a news command that resides in ListProc's home directory and that the server manager must invoke. The news command subscribes the newsgroup to your list, but uses a special file to do this: $LPDIR/LIST-ALIAS/.news

In ListProc's home directory, the ListProc maintainer must invoke the news command to activate the gateway.

```
Usage: news <LIST_ALIAS> <news group> <news e-mail
address> <news mode>
```

where:

LIST_ALIAS
: a local list alias, in UPPERCASE, that is to be connected to the news group.

news group
: the newsgroup name (e.g. alt.sex)

news e-mail address
: full e-mail address of the news source.

news mode

receive if the local list will only be receiving articles, or *send_receive* if the local list will post to the news group as well.

The news e-mail address is only used if using gateway software, in this case, the e-mail address of the gateway software must be placed here for ListProc to know where to mail to article. If using *inews*, use news@newsserver.domain. This field is effectively ignored and this address is just a marker.

To gateway the newsgroup alt.winsock to a list called winsock-l@yukon.cren.org, issue the following news command:

```
news WINSOCK-L alt.winsock news-gw@info.cren.net &
send_receive
```

This establishes a two-way gateway to the newsgroup. It also will set up the following .news file in $LPDIR/lists/WINSOCK-L/.news

```
news@info.cren.net NOACK alt.winsock
```

Also, the following line must be added to the news server's **sys** file (when using C-News, with INNd, the equivalent file must be modified).

```
mail.to.winsock-l@yukon.cren.org:alt.winsock/
all::news-gw winsock-l@yukon.cren.org
```

The above command will mail all posted articles to the mailing list and must be put into place regardless of whether or not the gateway is configured for *send_receive* or *receive*. It must go in the news server's sys file (usually /usr/lib/news/sys).

The news articles that get mailed to the mailing list maintain the original **From:** lines. This is typically associated with the person who originally posted the article to the news group. This means that you must have your list configured to be *send-by-all*, i.e. both subscribers and non-subscribers have to be allowed to send to the newsgroup. An alternative is to use or create a small wrapper program. The *news-gw*, referenced above is one which sets a *Resent-* from line to be the gateway address. This is defined as news-gw@info.cren.net.

ListProc respects *Resent-* headers and this can be used advantageously for news-gateways.

Peering

Peer lists are mailing lists that are subscribed to one or more mailing lists; they handle local distribution of messages just like any other list and also messages that originate in one peer list are propagated to the other peers for distribution. Only one copy of the message is sent to the peer. Peers have mechanisms built in to avoid looping.

A peer list is added by the ListProc manager by typing the *peer* command:

```
peer <LIST_ALIAS> <remote alias> <remote peer e-mail
address> <remote peer ListProc e-mail address>
```

where:

LIST_ALIAS
: a local list alias, capitalized, that is to be connected to the new peer. This is different than the list's e-mail alias which is placed in the system mailers' **alias** file.

remote alias
: the peer's list alias on the remote machine

remote peer e-mail address
: full e-mail address of the peer list to be added

remote peer ListProcessor e-mail address
: full e-mail address of the server that handles the remote peer

All participating sites must be in sync when peering. They must all issue the peer command with the appropriate parameters.

The peer command creates a special file in $LPDIR/lists/LIST-ALIAS/.peers

Here is an example of establishing a connection between two peers, lists peer1@foo.bar and peer2@bar.foo.

The manager at foo.bar will issue the following peer command:

```
peer peer1 peer2 peer2@bar.foo ListProc@bar.foo
```

The manager at bar.foo will issue the following peer command:

```
peer peer2 peer1 peer1@foo.bar ListProc@foo.bar
```

Once these two commands are issued, the connection is automatically set up. Peered lists should make sure that only one of them posts to the same news group(s) and that only one of them receives articles from the same news group(s) / gateway(s). Lists handled by the same server cannot be mutual peers. Also, peered lists should not be mutual subscribers. The peer command places them in the **.peers** file mentioned above.

Anonymous FTP

Anonymous ftp can be set up with ListProc's archives fairly effortlessly. When you do this, create an anonymous ftp directory according to your system's instructions (see the system manual pages on ftpd for an explanation of the necessary files and directories). Make sure that the ListProc archives that are to be made available this way are within the anonymous ftp area and the ListProc account (server) has write permissions in this area. Also, it is important that the ListProc archives file creation mask be set up as follows:

```
ULISTPROC_ARCHIVES_UMASK=022
```

Otherwise, no one will be able to retrieve the files. The archive directory must also have permissions 755.

CREN has its archive directory as the top of the anonymous ftp tree so that each archive is a directory immediately below that. This can be tailored by site.

Gopher and WAIS

Like anonymous ftp, gopher can be set up to provide access to your list archives. Like anonynmous ftp, you must place the archives you want to make available via gopher in the directory tree that your gopher files are running. (It can be and often is identical to your anonymous ftp directory hierarchy). ListProc's account (server) must have read-write permissions to this directory.

When using gopher, it is important that you do not compress archives. By maintaining uncompressed archives, your gopher server will automatically pick up that the archives are Unix mailboxes and present them to the user as a list of individual mail items. (Gopher to cren.org to see an example of this.) Every new posting on a list where there is archiving is automatically added to the gopher menu. If you compress archives, this type of presentation is not possible.

It is also possible to WAIS index the mail logs. To do this, you must compile and install your gopher server with the WAIS capability as described in the gopher installation instructions. For an example of how this is done, again gopher to cren.org.

Unlike gopher, WAIS is not automatically done and must be updated via the cron. WAIS has flags to automatically pick up mailboxes and index them appropriately. The following is an example:

```
cd /archive_dir/.archivename
find /archive_dir/archive-name -type f -name "log*" -
print | grep - v '/\.' | waisindex -a -d archiveindex
-t mail_or_rmail -stdin
```

In this example, first go to a directory called **.archivename**—the period in front of the name makes this directory invisible to gopher. This directory is where all the waisindex information is stored. It must be read-writeable by the ListProc account. It can be set up with permissions 600. Then use the find command to pass WAIS a list of your archives. The -t flag to WAIS tells the server that this list being passed to it is a mail_digest (i.e. UNIX mbox). The -a flag is used to reindex only those files which have changed. A full indexing should be done approximately once per week.

For every archive that you want to make WAIS searchable, you should run this nightly out of the ListProcessor's user-id cron entry to reindex the archives.

There is another caveat with WAIS and gopher. For security reasons, the gopher server runs chrooted. The problem with WAIS Indices is that they store the full pathname inside the inverted index. Thus, if your documents are in:

```
/usr/local/archives/listname
```

that entire path will be kept in the index.

However, when using *chroot()*, a whole portion of the path is chopped off and the directory /usr/local/archives effectively becomes root, '/'.

To circumvent this problem, you can use a symbolic link to make the removed directory structure inside the archives directory. In the case above, you would do the following:

```
cd /usr/local/archives
mkdir usr
cd usr
mkdir local
cd local
ln -s / archives
```

Then, when the pathname /usr/local/archives/listname is looked up, the symbolic link will put you in the correct place. Also, so that your gopher users do not see this new directory, add the following line to the **gopherd.conf file**[1]:

```
ignore: usr
```

Mailers

Your choice of a Mail Transfer Agent is a critical determinant in the overall performance of your ListProcessor. At the very least you should install the latest UCB *sendmail* (8.6.9) and the Berkeley *db library*. The new *sendmail* with its expanded database capabilities and limited multi-queuing will give you some strong performance enhancements. Both packages are available via anonymous ftp from info.cren.net in /pub/software.

CREN is currently using *ZMAILER*, developed at the University of Toronto. We are running it on a SPARC LX, with 40 Mbytes of RAM. We are currently doing between 20k and 50k smtp deliveries per day. While *ZMAILER* is a preferred mailer for high volume mail hosts and List machines, it is resource intensive. A copy of *ZMAILER* 2.2 is available via anonymous ftp from info.cren.net in <directory>. This mailer has its rewrite rules optimized for smtp only hosts. ZMAILER is a large and complex system but it seems to pay off for large sites.

Another alternative is *MMDF*; its multi-queued nature will give you enhanced performance over sendmail.

We urge you to discuss the various mailers, gopher, ftp, WAIS and their interface to ListProc on the mailing list. We intend the list to assist site managers in setting up and integrating ListProc with the various Internet resources available.

Alias and Ignore

ListProc provides two powerful and flexible tools in the **.alias** and **.ignore** files. These files let you rewrite incoming addresses and deny users, hosts or entire domains access to your server for whatever reasons you decide as the site manager.

The two levels of **.aliases** and **.ignore** are installation wide and list specific. The site manager is the only person with access to the **.aliases** and **.ignore** files in the top level of ListProc's home directory. The expressions in these files will affect every list on the server. In addition, each list owner can manipulate these files in the lists directory, and they affect only the one specific list.

[1] Thanks to Paul Lindner, Gophermaster at the University of Minnesota for that solution.

Alias

It is possible that a peer/subscriber/newsgroup's/gateway's mail may arrive differently than subscribed, which may raise subscription issues and prevent the person from posting or changing subscription options.

For example, when using the conversion script provided to change your lists from VM Listserv to ListProc, BITNET only addresses are rewritten as follows:

```
user%node.bitnet@interbit.cren.net
```

This is fine, except that when that user sends mail to the list or ListProc, there is no way to predetermine which gateway the mail will traverse. In the end, the address, although in the form above in the subscribers file, may appear to the ListProc as:

```
user%node.bitnet@pucc.princeton.edu
```

To get the ListProcessor to accept that address as equivalent to what is in the subscriber's file, the following alias will work:

```
^(.+)%(.+)\.bitnet@.+ \1%\2.bitnet@interbit.cren.net
```

This alias should go in the top level **.alias** file if you convert VM Listserv lists to ListProc and leave the preassigned gateway as interbit.cren.net. If you change the gateway, change it in the alias or add it, since aliases can be added.

Another useful top level alias deals with organizations that have clusters of many machines. When mailing into the same domain, some fully qualify and others do not. The following alias will accept all non-fully qualified e-mail address as the equivalent cren.net address:

```
^(.+)@([^.])$ \1@\2.cren.net
```

In the case above, the ([^.]) accepts anything but the period, so that user@foo.bar is not converted to user@foo.bar.cren.net.

The *alias* command allows the site manager and list owners to add new aliases to the **.alias** file

For each new list, the system adds the following alias:

```
^@.*:(.*)@(.*) \1@\2
```

This removes source routing.

Warning: This command can have disastrous results if it is set up improperly. Be sure to test and verify your information when using *Alias*.

Ignore

As described before, the system's home directory as well as the every list's subdirectory contains a **.ignored** file which filters out messages sent by certain users. The default file contains entries for server, bin and sys. You may want to add entries for every list alias that is defined on your system. A lists's **.ignored** file also contains an entry of its alias and full e-mail address as well as the server

account's full e-mail address.

Entries in this file may also be regular expressions as shown below.

Alias and Ignore Command

alias <list> <password> <new-address> <address-as-subscribed
ignore <list> <password> <address>

Alias an existing subscriber to a new address, or ignore postings from the specified address. Both requests are synonyms of the more generic *put* request (put <list> <password> alias <address> <address> and put <list> <password> ignore <address>).

<address> and <new-address> can be regular expressions.

Examples:

```
alias ermis ermis1 uucp!bar.com!foo foo@bar.com

alias ermis ermis1 ^@.*:(.*)@(.*\..*) \1@\2
```

The latter will remove source routing from all addresses and convert them to the user@domain syntax; for example:

```
@GATEWAY:USER@HOST.DOMAIN
```

will be converted to USER@HOST.DOMAIN

```
alias ermis ermis1 [^!@]*!([^!@.]*)!([^!@]*)@.*
\2@\1.uucp
```

This will convert GATE!HOP!USER@UUCP.SOME.COM to USER@HOP.UUCP

```
ignore ermis ermis1 foo@bar.com

ignore ermis ermis1 ~<.*\.com|.*\.edu>
```

The latter restricts postings to .com and .edu domains only.

For additional information on regular expressions, you can anonymous FTP zoo.toronto.edu.

9.0 Listproc Release Notes

List owners and site maintainers need to be aware of certain limitations ListProc has on its behavior . These limitations involve the types of characters allowed in certain fields to the length of a command and debugging a ListProc error.

- When you submit a command to ListProc, the total length of that command cannot exceed 1024 characters. ListProc ignores anything beyond 1024 characters.
- List mail that does not get delivered due to a mailer unavailable, or too many sockets open will be queued by ListProc. The queued mail is placed in $LPDIR/mqueue and has a numeric file name. Listproc provides a shell script that will automatically scan this directory and attempt to resend this queued mail. This script must be run as uid server with the following syntax:

  ```
  queued <time between scans in seconds> &

  queued 3600 &
  ```

  The above example would check the **mqueue** directory once per hour for queued mail. You must edit this script and change all the references to ListProc's home directory to match your site's configuration.
- In debugging ListProc, the site maintainers should be aware of certain files given the two scenarios below:
  - The syslog options is enabled. In this case, the first place to go is the syslog file where ListProc is placing diagnostic messages. Next is the $LPDIR/.warning. Normally the $LPDIR/.warning is used by ListProc to record the output of system utility programs that it uses (i.e. *grep*) but valuable diagnostic messages can be found there.
  - No syslog enabled. In this case Listproc will write process related output in a series of files under $LPDIR. These files can be identified because they either start with .rep or .report, (i.e. *.rep.serverd.a* is the diagnostic for the serverd process). There is a series of these all under $LPDIR. In addition, the $LPDIR/.warning file exists here as with the previous option.
- ListProc maintains a series of files which are useful for debugging but which must be culled on a regular basis otherwise they will grow out of bounds.

  For each list, ListProc maintains a copy of all incoming mail under $LPDIR/lists/LIST-ALIAS/mbox. These files are useful for debugging and verifying user requests, postings and errors, but can get larger very fast.

  ListProc maintains a top level mbox ($LPDIR/mbox). This contains a copy of all mail sent to ListProc. It can be very useful for debugging and verifying user requests but can grow very large in a short amount of time.

  The file $LPDIR/.warning can grow very large in a short amount of time.

- When using a mail method (as specified in ListProc's top level config file) other than the system mail method, there is a 2k limit on mail header size. This effectively means that you should not bundle in more than 50 recipients per message. This limit is due to the parameter limits on most user shells, which pass the mailer program the recipient list.
- ListProc cannot have a tilda (~) as part of its path to its home directory, i.e. (/user/~listproc). This causes the *chown* system call to fail on many systems.
- When configuring lists, the comma (,) cannot be used as part of a list or list manager's password.
- When running an interactive serverd, remember that server must have the *setuid* bit set and be owned by root. This is because serverd listens by default on port 372 (which is registered). All ports below 1024 are privileged ports which can only be open by root. You can get around this by having ListProc listen on a port above 1024. Beware that having your server listen on a non standard port will make it harder for end users to find you.
- When using any of the ListProc 7.0 clients to log into the server, the e-mail address asked at login is case sensitive for all users except the site maintainer. The ListProc manager must enter the e-mail address exactly as it is configured, including case.

Appendix A. Installation Script.

ListProcessor system setup script:

```
                                Version 7.0
                          Copyright (c) 1993-94 by
                the Corporation for Research and Educational Networking
                                     (CREN)
The system will now be installed under /home/server; if you want to install it
under another directory, you may redefine the LPDIR environment variable now,
or specify another directory in the command line.

Proceed [y]?
Remaking ./setup
Remaking ./distrib/setup
Creating directory /home/server/lists
Creating directory /home/server/mqueue
Creating directory /home/server/tmp

Now serverd will have to be setuid to root if it is to run in
interactive mode and use a privileged port. Please enter the root pass-
word below if so:
Password:

NOTICE: the canned archives/ilp directory contains everything a user
will need for live connections; you may wish to copy it to your own
archives now.
ListProcessor System ready.
```

Appendix B: Sample Config File

```
organization MY_ORG_NAME
multiple_recipients 30
list venus venus elias venus1
server ListProc #-r review -b get
list herc herc elias herc1 -r
list ermis ermis elias ermis1
unix_cmd ermis pass1 dir '/bin/ls $*' #Syntax: dir [args]
unix_cmd ermis pass1 swap '/bin/echo $2 $1' #Syntax: swap <arg1> <arg2>
unix_cmd ermis pass1 who '/bin/who' #Syntax: who
unix_cmd venus pass1 from '/bin/from' #Syntax: from
batch 8 20     # batch requests between 8 am and 8 pm
serverd -i 1800
restriction 4  # certain requests may not be serviced above this # of users
manager tasos  # e-mail address of the manager of the system
password ellines # password for the 'restart' and 'shutdown' requests
comment server       #Taso's ListProcessor
frequency 5    # how often to read mail (in seconds >= 0)
limit message 65536    # reject messages longer than 65536 bytes
precedence bulk # Usually bulk, junk, first-class or none
option bsd_mail      # define it if BSD mail is available
#option post_mail # Post to news; other choice is gate_mail
option gate_mail
option ignore_invalid_requests
mailmethod system
#mailmethod env_var LOGNAME /bin/rmail
#mailmethod sendmail /usr/ucblib/sendmail -ba
#mailmethod env_var LOGNAME /bin/rmail
#mailmethod env_var LOGNAME /usr/ucblib/sendmail -ba
#mailmethod telnet
remote pen_s pen_s@remote-host1 listserv@remote-host1 127.0.0.1 #Sex issues
remote pen_s-2 pen_s-2@remote-host2 listserv@remote-host2 localhost 372 #Fruit punch
issues!!
remote cook cook@foo.com listserv@foo.com playboy #Cooking oil issues
# Define a new host for sending mail
mta_host localhost
# More mailer daemons
mailer_daemon '[      ]*test$|^uucp|mrgate|mmdf|^smt.*|^mal@|^admin'
# More suspicious subjects
susp_subject 'remove[      ]+me|remove[  ]+my|&
delete[       ]+me|delete[  ]+my|add[     ]+me|add[     ]+my|&
^[     ]*test[       ]*$|^[ ]*test.+ignore|please[     ]+ignore'
# Turn on BSD_MAIL and define the path to it
ucb_mail /bin/mailx
# Set SMTP HELO argument
helo_arg aspirin
# Turn on error analysis, set values
error_analysis 9 259200
# Turn on syslog reporting, set facility
#syslog LOG_USER
# Ignore these extra requests:
ignore_requests 'this[     ].+[   ]test|please[ ]ignore'
```

Appendix C: Catmail Utility

This is a utility that appends a mail message supplied by *sendmail* in *stdin* to the appropriate file, depending on the command line options supplied. The application has to be *setuid*. If the incoming message is for a moderated list and not from the moderator, append an INTERNAL_NOTIFY request to the requests file so that ListProc will later notify the list owner, soliciting his approval for posting the message.

The utility is to be used primarily in the host's **aliases** file when setting up aliases for various lists. To append to the system's requests file, it may be used as follows:

```
ListProc: "|LPDIR/catmail -r -f"
```

To append to a list's mail file:

```
list_alias: "|LPDIR/catmail -L LIST_ALIAS -f"
```

To append to a list's errors file:

```
owner-list_alias: "|LPDIR/catmail -L LIST_ALIAS -f -e"
```

To send to owners:

```
list_alias-request: "|LPDIR/catmail -L LIST_ALIAS -f -
o"
```

COMMAND LINE OPTIONS:

-L The input is appended to the specified list's "mail" file.

-l Same as -L.

-e Append to the list's errors file instead.

-o Send message to the owners

-r The input is appended to LPDIR/requests.

-f The input is reformatted while appended.

Appendix D: Interactive ListProcessor

How to connect and use the Interactive List Processor

ListProcessor may be reached for interactive (direct, without using e-mail) sessions if both this system and your host are on Internet. Live sessions look a lot like ftp sessions, in the sense that the user may issue all of the valid requests he would send over e-mail, plus download files from the archives. The server has also the capability of connecting to remote live UNIX ListProcessors for requests that refer to remote lists.

To connect to an Ilp, you will need a program called *ilp*; this should be available from this site's archives. Issue an *index* request and *get* all files from the designated archive. X windows, PC and Macintosh clients are also available from CREN, in binary format. These clients offer a graphical user interface and this discussion applies to the text based Ilp client only.

You will need to compile the Ilp client before you use it; once compiled, you may use it as follows:

```
ilp yukon.cren.org [port]
```

All compliant Interactive ListProcessors listen at port 372 for connections by default, but they may use any other port, in which case you will have to specify which one. For example, on host csa.bu.edu a sample server is running for testing purposes; to connect to it you would issue the following command:

```
ilp csa.bu.edu 8000
```

Log in with e-mail address *test* and password *new-user* to obtain subscriber privileges (more on this later). Notice that you may also specify an IP address instead of the DNS name of the host. If using domain names, you probably have to be running a name server on your local network.

Once a connection is established, you will be asked for an e-mail address and a password; these will be used to establish the privileges (valid set of requests) upon login. A mismatch or no e-mail address reduces these privileges to a minimum (casual user).

Casual users may only issue *help*, *information*, *recipients* and *statistics* for non-private lists, *lists*, *index*, *get*, *view*, *search* and *release* requests.

Subscriber privileges also include the *set*, *run*, *unsubscribe*, and *which* requests.

Owners may, in addition, issue all of their administrative requests.

To gain subscriber privileges you must be subscribed to at least one list, use your e-mail address as shown by a *recipients* request, and provide the password that you have set before. If you are subscribed to more than one list, any password from the ones that you use for each list will do (assuming that your subscription address is the same for all those lists). However, if you belong to a private list, you should use the password as set for that list to have access to *recipients* and *statistics* listings.

To gain owner privileges, please contact this system's manager (you may get the

manager's address via a *release* request).

To end a live session, just type `quit` or `exit`. To review your privileges at any time, type '?' or *privileges*. Connections are limited to a certain amount of time; to get the time left on your connection, type *timeleft*.

Requests may be continued on multiple lines by use of '&' followed by a new-line character. The output of every request (if it is not an error message) may be redirected to a file by using a '>' followed by a file name.

Example:

```
REQUEST> index > index
```

or by using '>>' in which case the output is appended to the file. File transfer also takes place when you *get* files from an archive. For example, if you

```
REQUEST> get ListProc info
```

the output will be saved under file *info*. You choose another file name by redirecting:

```
REQUEST> get ListProc info > index
```

The keywords *binary* and *ascii* specify the transfer type; the default is *binary*, i.e. files are transferred as they are; in ascii mode, new-line characters are translated to carriage-return/line-feed pairs.

Input may also be redirected from a file by using '<' followed by a file name; for example:

```
REQUEST> < batched.requests
```

In this case, each line will be interpreted as a separate request and replies will be produced. Here is an example batch file:

```
index > index
get ListProc info >> index
quit
```

A particularly useful case of input redirection is the *put* request:

```
REQUEST> put ermis ermis1 welcome < prepared-welcome-
file
```

Note that for requests requiring a password, you may replace the password with a dash (-) since you provided the password at login; this of course works only if you use the same password for all lists you are subscribed to or own. Then, the previous request could have been written as:

```
REQUEST> put ermis - welcome < prepared-welcome-file
```

Moreover, the output of requests not redirected may be piped to UNIX command using the '|' character. For example:

```
REQUEST> review ermis | more
```

will pipe the output of the *review* request to the UNIX command. As another example, to look for a particular subscriber in a list, one may do:

```
REQUEST> review venus | grep -i Name
```

As a last note, requests for remote lists are serviced by contacting the remote servers, if possible. Live requests are subject to the usual restrictions (system loads) — requests that are designated by the manager for batch processing will be rejected, and you will have to submit your request via e-mail.

Appendix E: ListProc Files

HOMEDIR/.aliases
Aliases of email addresses of users having trouble getting replies to requests.

HOMEDIR/.awk
awk program for formatting the statistics request.

HOMEDIR/.grep
script used for counting the number of messages sent by a subscriber.

HOMEDIR/.ignored
List of unwanted senders.

HOMEDIR/.lock.pqueue
Lock file for pqueue (see queue(1)).

HOMEDIR/.lock.serverd
Lock file for serverd(1).

HOMEDIR/.message.ids
A database of recent message id's used to detect mail loops.

HOMEDIR/.queue.id
Next id to be assigned to next undeliverable message placed in the mqueue/ directory.

HOMEDIR/.reply.listser
The reply code given to serverd(1) during a live session.

HOMEDIR/.rep.server.acc
Archived listproc(1) reports.

HOMEDIR/.rep.serverd.acc
Archived serverd(1) reports.

HOMEDIR/.rep.start.acc
Archived start(1) reports.

HOMEDIR/.report.catmail
Current catmail(1) report.

HOMEDIR/.report.daemon
Current serverd(1) report.

HOMEDIR/.report.list
Error messages from list(1).

HOMEDIR/.report.server
Current listproc(1) report.

HOMEDIR/.report.start
Last start(1) action (system started or shut down).

HOMEDIR/.stats
Script used for the statistics request.

HOMEDIR/.sums
Database of checksums of public messages.

HOMEDIR/.tag.id
Next id to be assigned to next moderated message.

HOMEDIR/.warning
Log of system calls' exit statuses.

HOMEDIR/batch
Batched user requests.

HOMEDIR/catmail
Incoming message redirection application.

HOMEDIR/config
The configuration file.

HOMEDIR/farch
File archiving utility.

HOMEDIR/ilp
Interactive ListProcessor client; to be used to connect to the interactive server.

HOMEDIR/list
The mailing list program.

HOMEDIR/listproc
The system's request server.

HOMEDIR/lost+found
Requests that could not be appended to

HOMEDIR/requests.

HOMEDIR/mbox
An archive of all requests sent to date to ListProcessor.

HOMEDIR/news
Script used to add a news group to a list.

HOMEDIR/owners
List of list owners' addresses used for authentication of list maintenance requests.

HOMEDIR/peer
Script used to add a peer list.

HOMEDIR/pqueue
The mail queue processing program (see queue(1)).

HOMEDIR/priv.hosts
List of hosts that may connect to the interactive part of the system.

HOMEDIR/queued
The mail queue processing daemon (see queue(1)).

HOMEDIR/received
File containing the messages received from sendmail during the last message distribution -- the system mailmethod has to be in use.

HOMEDIR/requests
Newly arrived user requests.

HOMEDIR/requests.live
Repository for requests coming in from live connections.

HOMEDIR/sent
File containing the messages sent to sendmail during the last message distribution—the system mailmethod has to be in use.

HOMEDIR/serverd
The system's daemon.

HOMEDIR/setup
System setup script.

HOMEDIR/start
The system's start program (start the deamon and kill any existing running daemons

HOMEDIR/tlock
Checks for any locked files.

HOMEDIR/ulock
Script to remove all lock files.

HOMEDIR/unwanted.hosts
List of hosts that may not connect to the interactive part of the system.

HOMEDIR/welcome.live
Welcoming message upon live connection establishment.

HOMEDIR/archives/listproc/DIR
Directory of all files available from the master archive (listproc).

HOMEDIR/archives/listproc/INDEX
The master archive index.

HOMEDIR/doc/catmail.1
Man page source for catmail(1).

HOMEDIR/doc/catmail.nr
Formatted man page for catmail(1).

HOMEDIR/doc/farch.1
Man page source for the farch(1) utility.

HOMEDIR/doc/farch.nr
Formatted man page for the farch(1) utility.

HOMEDIR/doc/ilp.1

Man page source for the ilp(1) utility.

HOMEDIR/doc/ilp.nr
Formatted man page for ilp(1).

HOMEDIR/doc/list.1
Man page source for list(1).

HOMEDIR/doc/list.nr
Formatted man page for list(1).

HOMEDIR/doc/queue.1
Man page source for pqueue and queued(1).

HOMEDIR/doc/queue.nr
Formatted man page for pqueue and queued(1).

HOMEDIR/doc/serverd.1
Man page source for serverd(1).

HOMEDIR/doc/serverd.nr
Formatted man page for serverd(1).

HOMEDIR/doc/start.1
Man page source for start(1).

HOMEDIR/doc/start.nr
Formatted man page for start(1).

HOMEDIR/help/*
General and topic-specific help files available upon a help request. The TOPICS is a special file where topics are defined.

HOMEDIR/lists/LISTALIAS/.aliases
Aliases of email addresses of subscribers.

HOMEDIR/lists/LISTALIAS/.digest.toc
All the subject lines from the digest. It will be written at the head of the digest when it is sent.

HOMEDIR/lists/LISTALIAS/.digest.msg
All the messages of the digest, separated by dashed lines.

HOMEDIR/lists/LISTALIAS/.digest.time
The time that the last digest was sent, represented as the value returned by time(2). It is used to calculate when it is time to send out the next digest.

HOMEDIR/lists/LISTALIAS/.headers
An archive of who sent each message; used upon a statistics request.

HOMEDIR/lists/LISTALIAS/.ignored
A list of unwanted senders.

HOMEDIR/lists/LISTALIAS/.info
Text sent upon an information request.

HOMEDIR/lists/LISTALIAS/.limits

Used by serverd(1) when placing various limits for the list.

HOMEDIR/lists/LISTALIAS/.message.ids
A database of recent message id's used to detect mail loops.

HOMEDIR/lists/LISTALIAS/.news
A list of all news groups connected to this list.

HOMEDIR/lists/LISTALIAS/.peers
A list of all peer lists for this mailing list, along with the email addresses of their servers.

HOMEDIR/lists/LISTALIAS/.rep.list.acc
Archived reports from list(1).

HOMEDIR/lists/LISTALIAS/.report.list
Current report from list(1).

HOMEDIR/lists/LISTALIAS/.restricted
List of subscribers with alternate recipient files.

HOMEDIR/lists/LISTALIAS/.subscribers
A list of all subscribers along with preferences and settings.

HOMEDIR/lists/LISTALIAS/.un.digest
When a digest is constructed, it will be stored in this file until it has been sent to all digest subscribers.

HOMEDIR/lists/LISTALIAS/.welcome
Text sent on a subscribe request.

HOMEDIR/lists/LISTALIAS/archive
Archive of all distributed messages since the file was last created.

HOMEDIR/lists/LISTALIAS/lost+found
Mail that could not be appended to the 'mail' or 'moderated' files (see next entries and catmail(1)).

HOMEDIR/lists/LISTALIAS/mail
Newly arrived public messages to be distributed.

HOMEDIR/lists/LISTALIAS/mbox
An archive of all messages sent to date.

HOMEDIR/lists/LISTALIAS/moderated
Newly arrived public messages that need to be edited before distributed.

HOMEDIR/lists/LISTALIAS/removed.users
Entries from the .subscribers file for users who have been automatically removed from the list.

HOMEDIR/lists/LISTALIAS/removed.alias
Entries from the .aliases file for users who have been automatically removed from the list.

HOMEDIR/mqueue/*
Queued messages for later delivery.

Index

A
Add 7, 9, 12, 24, 28, 40, 41, 45, 46, 47, 48, 54, 56, 61
A10, 11, 12, 13, 15, 16, 25, 27, 28, 39, 41, 50, 51, 52, 54, 55, 56, 57
Alias and Ignore 54
Anonymous FTP 52
Appendix A. Installation Script. 59
Appendix B: Sample Config File 60
Appendix C: Catmail Utility 61
Appendix D: Interactive ListProcessor 62
Archives 6, 7, 8, 10, 11, 25, 26, 30, 31, 33, 36, 39, 42, 43, 44, 45, 46, 47, 52, 53, 54, 59, 62
Archiving Files 42

C
CNews 50
Comment 15, 17, 18, 31, 32, 39, 43, 60
Configuration 10
configure 7, 12, 13
Configuring a Mailing List 28
Converting from ListProc 6.0 10
Converting from VM Listserv 10
Creating a Mailing List 28

D
Defining a List 28
Delete 23, 24, 28, 30, 31, 39, 40, 46, 60
Digest 28, 29, 32, 33, 35, 39

E
Environment 8, 9, 26, 59

F
FTP 7, 8, 10, 25, 26, 50, 52, 53, 54, 56, 62

G
General Tips 50
Gopher 7, 26, 53, 54
Gopher and WAIS 53

H
headers 18, 19, 20, 39, 51
How to connect and use the Interactive List Processor 62

I
Ignore 19, 41, 54, 55, 56, 60
ILP 12, 46, 48, 49, 59, 62
Initialize 13, 14, 28, 29
install 8, 11, 53, 54, 59
Installation. 8
Interactive 10, 12, 16, 33, 48, 49, 58, 59, 62
Introduction 6

L
List Management 40
ListProc Release Notes 57
Lists 7, 9, 10, 11, 17, 25, 26, 28, 36, 37, 40, 42, 46, 48, 50, 51, 52, 54, 55, 57, 58, 59, 61, 62, 63, 64

M
Mail 6, 7, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 25, 26, 27, 28, 31, 32, 33, 34, 35, 39, 41, 46, 48, 50, 51, 52, 53, 54, 55, 56, 57, 58, 60, 61, 62, 64
Mailer 12, 13, 18, 21, 34, 39, 54, 57, 58, 60
Manager 7, 13, 16, 17, 19, 23, 25, 28, 29, 34, 36, 37, 39, 40, 42, 46, 47, 48, 50, 51, 52, 54, 55, 58, 60, 62, 64
Mask 26, 52
mta_host 13, 20, 60
multiple_recipients 13, 60

N
News 19, 30, 40, 41, 50, 51, 52, 60
News–Mail Gateways 50
nntp 50

O
Organization 12, 60
Overview 7

P
Password 7, 12, 13, 15, 17, 28, 29, 30, 31, 32, 33, 35, 36, 37, 39, 40, 41, 42, 43, 44, 45, 46, 48, 49, 56, 58, 59, 60, 62, 63
Peer 51, 52, 55
Peering 51
permissions 8, 10, 25, 26, 50, 52, 53
Pitfalls 12
purge 42

R
Remote Lists 25

S
sendmail 13, 20, 39, 54, 60, 61
Server 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 19, 21, 20, 25, 26, 27, 28, 39, 40, 41, 42, 45, 46, 50, 52, 53, 54, 55, 56, 57, 58, 59, 60, 62
Server Environment Variables 26
serverd 16, 57, 58, 59, 60
Setting the List Aliases 39
subject 36, 64
Subscribing to Files 46
syslog 14, 16, 20, 21, 57, 60
System Mail Alias 27

T
The Config File 12

W
WAIS 7, 53, 54