USENET Installation Guide

Eric S. Raymond (News 3.0 (beta level 7) rev.)

1. Introduction

This document is intended to help a USENET site verify, install and maintain the network news software. Please email questions to Eric S. Raymond (...!{uunet,att}!snark!eric), such questions will help to point out areas that need to be addressed here.

The overall order of things to do is:

(a) The READ.ME file in the news sources directory includes a partial list of systems to which this code has been successfully ported. Read it first. If your system is one of those listed you should be able to compile and install the software without difficulty (but check the section on machine-specific nits for known problems peculiar to your system type). If it isn't listed, you'll want to pay particular attention to the section on porting and verifying the code below.

(b) If you're not a fully SVID/POSIX conformant site (in practice, if you're running a pre System V Release 3 UNIX) first run the Makeposix script in the src/D.posix subdirectory. This will create a makefile that can install some number of useful SVID/POSIX compatible library functions either directly into your C library or into a /lib/libposix.a or /usr/lib/libposix.a library.

(c) Pick or create an owning user and group for news. Then run the Configure shell script to set your configuration parameters. You will want to do ‘‘Configure all’’ for a first-time installation. Configure with no arguments can be used to tune your system later, and ‘‘Configure links’’ to set up news feeds to other systems.

(d) If you didn't previously allow Configure to compile the code, compile the software now using the *make* command. If you're the first site with your hardware/software combination you should do the port verification procedure described below -- and please drop the maintainers a note telling them how it went.

(e) Su to root and type ‘‘make installnews’’, if you haven't already during the port verification part of (b). This will copy the executables out to the right places and make directories containing most of the important data files. If you haven't already tested posting to local groups and the news readers during port verification, you should do it now.

(f) Find somebody to link up with. You need a network connection of some kind, for example Internet or UUCP. If you must use UUCP and have no connections, you must have at least a dialup (and preferably an autodialer), and find someone willing to call your machine. The USENET directory may be helpful in finding some other site geographically near yours to hook up to. Get your contact at the other end of the link to add you to their feed table. See below to determine which options you should ask them to put in their feed table entry for you (usually you'll want F and C for batching and compression).

(g) Post a message to the other site's to.*sysname* newsgroup, which should be set up to go only to the site you are linked to, as a test. Have the other person send a message to your system using the same mechanism. If this doesn't work, find the problem and fix it. please don't use misc.test unless there is no alternative. It is almost always possible to use test, or to.*sysname*, or some local.test group, instead of misc.test.

(h) Post formatted versions of the ‘‘manner’’ and ‘‘howto’’ files (in the doc directory) to your ‘‘general’’ newsgroup with a long expiration date. You can use **inews** or **postnews** to do this.

(i) If you are using UUCP as your transport layer, you will need to edit UUCP's L.cmds or Commands file to permit rnews. The install script tries to do this, but it may not always succeed. If you want to look exactly like pre-News 3.0 (beta level 7) Netnews to your neighbors you may want to allow unbatch, cunbatch and c7unbatch as well. On V7 systems it will be be necessary to either hack your uucp code to support the -z and -n options or define ZRETURN (see below).

(j) Use the ConfigureMap utility in the library directory to create a UUCP map entry and mail it to the map maintainers (ConfigureMap will calculate a submission address from what you told the news configurator about your mailer and net connections). You may want to post a copy to the newsgroup ‘‘news.newsites’’. The news administration scripts will mail you reminders to re-run ConfigureMap when appropiate.

(k) Help keep the net alive and healthy. Once you have been up and running a while, try and find a site that needs you to feed them -- and do it!

2. Installation

2.1. Configuration

Local configuration of the version News 3.0 (beta level 7) software has been drastically simplified from previous versions. The distribution includes a shell script called Configure that does some clever adaptive testing to determine your system’s capabilities, then asks you questions about your preferences. The information gathered is used to generate a makefile, a macro-defining header for the C components, and a number of shell scripts. Configure can also be used to add links to other systems to your feeds file.

Before configuring, select an owning user and group for the USENET software. Usually you’ll want to create a new ‘‘news’’ user and new ‘‘news’’ group by adding entries to /etc/passwd. Also, read the READ.ME in the news sources directory for information on specific machine/OS combinations that may need special care. Then, just run

```
Configure all
```

in the source directory. The config.h file can be edited afterward, if you insist, using information provided in the section entitled *How to mung the config file* in the ‘‘Hacker’s Guide to News 3.0 (beta level 7)’’.

Most of the Configure questions are self-explanatory. In general, questions you don’t understand on the first time through can safely be answered with a carriage return, selecting the default. You may wish to have a hardcopy of this document handy to refer to the following clarifications.

2.1.1. Site names

The FROMNAME question wants a fully-qualified domainist name such as ‘snark.uu.net’ or ‘ucbvax.berkeley.edu’. It will default to your node name followed by the fake domain .UUCP; if you use this, be sure to get registered in the UUCP project maps! This name is used for From lines in mail and news.

The PATHNAME is the name that will be prepended to Path lines to identify this site. The default for this (your node name) is almost always correct.

The TRUENAME question wants a full internal name of your site for use in notification mail to the administrator. In general this will be the same as FROMNAME, but if you’re hiding a network of machines behind one gateway you’ll want it to be different.

2.1.2. Mailer configuration

The questions in this section are mainly intended to find out how friendly your system is to RFC-822 mail format and Internet-style *person@site.domain* addressing.

If you have installed the UUCP project’s smail(1) package the Configure script will detect this fact and automatically set the software up to use Internet addresses, defining the SMARTHOST attrbute (see below).

The SMARTHOST question wants to know if you are either an Internet site or have installed something like the smail or uumail code for translating @ addresses to (theoretically) optimum UUCP paths. If you answer "%s" it will assume that your mailer can handle Internet-style person@site addresses, and DOMAINIST will be set.

You can also fill in the address of a neighbor that is willing to relay mail for you and has a mailer that keeps a uucp paths database (most backbone sites do this). The address should be a sprintf(3) format with %s in the position a user name would normally taake. This will enable your users to use Internet-style @-addresses in reply mail as though your machine were an Internet site itself.

The same convention applies in the ‘backbone path’ question. If you answer yes to it, news will try to mail submissions to moderated groups through the backbone. The advantage of this is that you don’t have to keep a local file of moderators up to date. The BACKBONE question will then want a path to your backbone site (this is likely to be the same as your Internet neighbor if you have one).

As with ordinary mail, you should get the OK of your neighbor’s postmaster person before raising their phone bills by using them as a backbone path or Internet relay.

2.1.3. Other questions

The ZRETURN switch suppresses error returns from rnews, so that it always returns a shell error value of 0. This is useful if you have any neighbors that can’t enable the −z option. It’s a good thing to try if they complain of nuisance mail coming back to them after they send news. Configure will pick a default on the dubious assumption that your neighbors are all running the same uucp as your site.

When you define SPOOLNEWS, news received from remote sites will be stashed away in a spool file rather than inserted and broadcast. Normal news processing won’t happen until the next expire run or rnews -U invocation, whichever comes first. This gives you a way to force most news processing out of prime time if your machine is heavily loaded (compile with SPOOLNEWS and run rnews -U out of crontab when convenient). This option is on by default and strongly recommended to avoid lots of rnews startup overhead.

SPOOLPOST enables the same behavior for locally-posted news. It is off by default.

On first installation (and especially when porting to a new machine) it is a good idea to anwer ’y’ to the DEBUG question. This will enable the -D options of checknews, the readers, rnews, sendbatch, expire, postnews and inews, which will be quite useful if you suspect any bugs.

The TMNCONVERT option is, unfortunately, less useful than it sounds. It causes the code to read and write 2.10.3 history formats, but won’t handle databases made with the old dbm(3) code. Use at your own risk.

On some System V Release 2 implementations the installation procedure may tell you you have to ‘‘schedule expire by hand’’. If this happens install a crontab line to run **expire**(8) one a day, to clean out old messages (expire also calls the **sendbatch**(8) utility before actually doing expirations in order to avoid losing postings to volatile groupings before they get shipped out).

If you’re running HoneyDanBer UUCP (also known as AT&T’s ‘Basic Networking Utilities’ for System V Release 3 and later versions) be aware that the Installation script will not be able to automatically ensure that you can run rnews when your news feed calls in. If rnews isn’t already enabled, you’ll see XQT DENIED messages in the /usr/spool/uucp/.Log/uuxqt file for your feed. To fix this, you need to have a MACHINE entry in your Permissions file that matches your feed’s name and lists rnews as a COMMANDS option. A typical such entry looks like this:

```
MACHINE=OTHER COMMANDS=rmail:rnews \\
READ=/usr/spool/uucppublic WRITE=/usr/spool/uucppublic \\
SENDFILES=yes REQUEST=yes
```

See your UUCP documentation for details.

3. How to port the code

When porting the news software to a new machine, the obvious first step is to get things to compile correctly. If you have to tweak anything please mail details to the maintainers so other people can use the fix (please see the coding guidelines in the Hacker’s Guide if you need to do this).

If you are on a small-address-space machine and your linker complains that you’ve exceeded the maximum code space size, you can undefine some symbols in config.h and defs.h to reduce the code size. PDP11 sites without the split I&D mod lose, but 80286 machines should be able to support everything except vnews.

In news.h you can undefine the following:

RECMDS	enables regular expression cmds
DELAYMARK	enables delayed message marking
BYSUBJECT	enables subject-following code
SUBJFILE	enables the subject index feature
RNESCAPES	enable rn-style escape expansions
NGUPDATE	enable on-the-fly update fetches of group data
INITCMDS	enables the kill commands feature
MACROS	enable user macros

Undefining RECMDS is probably your best bet, it is known to do the trick on the Intel 286. If you're on a heavily-loaded machine, undefining SUBJFILE may be worthwhile; it causes a fair amount of crunching during keyboard waits.

In config.h, you can undefine CACHEBITS and/or FEEDBITS. FEEDBITS enable pre-compilation of feed subscription patterns at rnews startup time; it can save a lot of time (by rendering many ngmatch() calls unnecessary) but costs extra space per newsgroup in the newsgroups array. CACHEBITS attempts to optimize further by keeping a compiled form of the subscription information in LIB/feedbits and recompiling it only when it's out of date with respect to LIB/active.

You're now ready to verify the functioning of the software. This should not be difficult but it will take some time; reserve yourself about two hours, preferably at a time when the system is relatively quiescent. It is also recommended that you read through all the instructions one time before doing them. All the test modules mentioned below have on-line help (type ‘‘?’’ to any * prompt) that describes their test commands.

The procedure below is designed for sites that already have stuff in their article trees. If you are not already a news site, just go ahead and install the software all the way (via ‘make installnews’ from your news source directory), and watch the error logs. If you've got problems, they'll show up quickly enough, and you can come back here and run these tests.

Before you start, if you already have news running, unplug your news feed dialins. Then change the name of your LIBDIR (usually /usr/lib/news) to something else like /usr/lib/oldnews (we'll call this OLDLIB below). This will protect you and your news users; if the port effort fails, you'll still be able to use the old news software after changing the directory back.

Stage 1: Set up your news library directory properly

1. Go root and run the InstallNews script. This will properly initialize a bunch of files in your new library directory.

2. Su to news (or whatever account you've given as NEWSUSR). Run ‘expire -sv 2’ now to generate a history and active files in the new formats. It may take a while. This will **not** expire any existing articles Your old history file will, of course, not be affected, as it's sitting in a different directory.

When you're done, you can leave the su; the next steps don't require write permissions on the news library directory.

3.1. Stage 2: checkout of front ends and active/.newsrc handlers

3. Make ‘‘newsdb’’. This is an interactive tester for the modules that handle the news database, {rd,wr}active.c, {rd,wr}history.c, and {rd,wr}newsrc.c. The first thing to do is type ‘a’ to the * prompt to get to the active file editing code, and exercise it. The active file code does a lot of malloc()s, that is what is most likely to cause problems. The best test is simply to let it read your current active file and dump it to stdout (use w). The ‘‘w’’ code includes most of the code used to update your real active file, so this verifies that you can read and write the format consistently. You might also want to pick a couple of newsgroups at random and dump detailed info on them with ’g <newsgroup>’. When you're done, type ’x’ to return to the debugger top level.

4. Type ‘r’ to the newsdb top-level prompt. Then you can test your interface to .newsrc files. Again, an R followed by w is your best test. A few mark and unmark commands will test the macros that twiddle seen-bits.

5. If you've gotten this far, checknews should work. Try doing ‘‘checknews x’’. If there's unread news you will see a list of message file locations. If it fails your problem is probably in newslib.a or portlib.a somewhere.

Again, the most likely thing to break is the memory-allocation routines. If they do, build the alist and slist testers (“make alist slist”) and try them out.

6. Now try out the readers. You should probably try readnews first; there’s more to go wrong in vnews. If you don’t see something reasonable, give them the option -D 4 to make them verbose. Mailnews is less important (your user population may not even want it); you can skip it for now. Be sure to do a “-” or two to test the backtracking.

7. Run rnews -D on some message input. If you have incoming batched news sitting around somewhere, that’s perfect; otherwise, the mkbatch script in the misc directory is provided to help you generate a test load. The -D flag exercises the logic of rnews (option processing, decompression, de-encoding, unbatching, insertion, and broadcasting) without actually modifying the news database or log files. It will diplay a bunch of useful debugging information (down to the uux commands that would be used for your newsfeeds) to stdout instead.

If rnews -D doesn’t show broadcasting happening correctly, and you’ve configured CACHEBITS on, try recompiling with CACHEBITS off. The CACHEBITS code relies on your architecture and compiler being able to cast (int *) and (char *) things back and forth correctly in order to do bit-packing; if you have funny (char *) formats (as on older word-addressed machines) this might not work.

8. If you send batches to any of the sites you feed, run sendbatch -D after you’ve done a few postings. You should see messages on stdout indicating broadcast of batched news to those sites. Like rnews -D, none of these sends will actually be done.

9. Run inews with the -D 2 option to generate a sample message. You should see a reasonable-looking RFC822 header and the text you entered come out to standard output. Ditto with postnews -D 2, but since the latter is interactive (and writes prompts to stdout) you’ll want to feed it to tee(1) and look at the argument file afterwards (rather than simply redirecting all standard output). The area to suspect if either of these fails is again probably the memory-allocation code.

3.2. Stage 3: Check that new article slots are created OK

For this you’ll need to su to news (or whatever account you’ve configured as the local news owner).

10. First, use the ‘a’ mode in newsdb to test that active.c can generate new article numbers correctly. The active.c code needs to open ADM/active for r+ and then write data to it in order to register new articles; some UNIXes have trouble with this. There are a couple of hacks in the code that should solve the problem on most systems, but if you’re on someone’s weird workalike you may have to do something else. This test will expose the problem if it’s there.

Your command sequence should look like this (stuff following ; is comments):

```
R                           ; read in the active file
g misc.test                 ; go to a group entry
n                           ; reserve a new article number
g                           ; check out the entry
W                           ; flush the change to disk
!grep misc.test ADM/active  ; check that it’s really there
```

where ADM is your news library directory, usually /usr/lib/news. You can do the g/b/G sequence for a couple of other groups (the readers will just skip article numbers without attached files).

3.3. Stage 4: Finish configuring your library directory

11. Now, edit the %active and %feeds files in the source directory to reflect your configuration.

If you’re a new site, you should add a name and description for any local groups you intend to carry to the %newsgroups file. The install process will then put these special groups in your active file. If you’ve already been running netnews, just copy your OLDLIB/newsgroups into LIBDIR/%newsgroups − and copy your OLDLIB/active file to LIBDIR/active.

If you had a OLDLIB/sys file, and you didn’t re-enter the feeds data while running Configure, copy it into %feeds, and edit it to move distribution prefixes into their own field. This will enable stripping of distribution prefixes before local insertion (we talk about how to override this feature below).

11. Type ‘h’ to the top level of newsdb to get to the history file editing code. You will need to do some ‘l’ commands to verify that you can add article entries to the database, the a ‘g’ to verify that you can find them by name. Use legal newsgroup names when doing this in order to avoid upsetting the history code. Don’t worry about deleting your ‘‘l’’ test entries, they won’t do any harm unless someone else happens to generate the same silly article IDs you made up.

If the history tester behaves brokenly, make ‘‘edbm’’ (no need to be su news for this). This is a tester for the key-content database routines in edbm.c. These do bit- and pointer-twiddles that may conceivably fail on some machines. If these break, you’ve got real problems. You can try fixing them, or hacking up an equivalent from dbm(3) if you’re a V7 site, or hacking up an equivalent from scratch otherwise. Let the maintainers about this if it happens.

3.4. Stage 5: Live local posting tests and cleanup

If you’ve gotten this far without unearthing a bug, you have verified that the news database handling layers behave sanely and that the readers and posting programs seem to function OK. The only thing you haven’t exercised that’s critical is article insertion and the actual uux or other transport layer used by rnews and sendbatch.

12. If everything checks out, try running inews, which is a fairly trivial shell around rnews (you’ll need to be su news for rnews to work). If either inews or rnews fails it will probably emit some kind of relatively informative message. If they don’t, you’ll have shown that you can update the news database (of course, you’ll want to check the spool directory to see that the message was actually posted and the headers are OK).

13. If inews/rnews works O.K., the essential parts are all working -- you can take the plunge and install everything for production. Keep the old binaries around if you have any; older postnews and the readers can still be used if the new ones fail. You can run ‘‘expire -s’’ to generate subject list files.

14. Next, try postnews. It should work if inews did.

15. After expire has run once and changed the active-file format, change rec.humor or something like it to be volatile and run it again with -v. This should disappear most of those bad jokes :-).

3.4.0.1. Stage 6: Fancy interfaces

Try out vnews. If it fails, try edvnews (which shareas all its code except the paged.c output manager with vnews) with the debugging turned on. If edvnews works, the problem is localized to the D.scrn library (and is more than likely lurking in the great dismal swamp marked ’TTY support’!).

If all these things worked, you’re ready to run live. Hook up the phone lines again and enjoy. If you feel a need to test further, keep the D transmission-debugging option in mind.

3.5. What USENET demands from the file system

You should place news in an area of the disk with enough free space to hold news you intend to keep on line. The total volume of news in all groups currently runs about 6MB per day, split among perhaps 3000 articles. If you expire news after the default 2 weeks, you will need about 100M bytes of disk space (plus some extra as a safety margin and to allow for increased traffic in the future.) If you only receive some of the newsgroups, or expire news after a different interval, or use the new volatile-groups feature, these figures can be adjusted accordingly.

4. Setting Up Links

There are two basic types of links for exchanging news: those that use mail and those that don’t. The ones that use mail are more indirect, yet more versatile while the ones that don’t are simpler. The default is without mail so that is discussed first.

4.1. Non-mail Links

The basic theory behind a non-mail link is that the *rnews* program is invoked on the remote system with the article being transmitted as the standard input. This is possible on some networks, but the most common implementation is via the **UUCP** network. Using the *uux(1C)* command, the command which is forked to the shell looks like:

```
uux − −r −z remotesys!rnews < article
```

This is the default transmission method. In order to set up such a link, obviously a **UUCP** link with the remote system must be in effect. In addition, *rnews* must be available and executable by *uuxqt* on the remote machine. In most cases, this means that *rnews* must be in /usr/bin so *uux* can find it. Also, /usr/src/cmd/uucp/uuxqt.c should be checked to make sure that rnews is an allowed command. The installation procedure does this automatically.

Other networks that allow remote execution include the Berknet, BLICN (usend), many Ethernets, and the NSC hyperchannel (nusend). It is important, however, that a spooling mechanism be available. Otherwise, if system A tries to send an article to system B via a remote execution command, and B is down, the article could be lost. Spooling arranges that the system will try again when B comes back up.

4.2. Mail Links

When using mail to transmit articles, pick the ‘‘M’’ option with an argument giving the name of the remote system’s contact person or news account (the *sendnews(8)* utility that used to do this is gone). The feed description

```
target:comp,news,sci:na,usa,pa:FM"rnews"::
```

for example, will mail news batches to target!rnews rather than trying to uux rnews on target. If the first character of the argument is @ or : the rest of it is still used as the name, but the mail address is formatted for Internet or Berknet respectively. Somehow, site ‘‘target’’ is expected to make sure that all mail to user ‘‘rnews’’ is fed as input to the program *uurec*(8). This program unpackages it and invokes rnews. The *sendnews(8)* utility that used to do the transmission side is gone.

The best way to get mail to rnews fed into *uurec* is to use sendmail or delivermail, if you are on a system running them. Create an alias in /usr/lib/aliases as follows:

```
rnews: "|/usr/lib/news/uurec"
```

and sendmail will handle it. If you do not have a facility for forwarding mail to a program, you can gimmick your mailer to watch for it (using *popen*(3S), this is easy) or, if you don’t want to do any programming, you can have *cron* invoke uurec every hour with /usr/spool/mail/rnews as stdin. This solution is messier because uurec must potentially deal with multiple messages, something that has never been tested.

The mailed form of the article(s), by the way, simply has an ‘‘N’’ prepended to every text line; uurec is pretty trivial.

4.3. Editing the feeds file to make links

To set up links to another site, run ‘‘Configure links’’ or edit the **feeds** file in ADM (this was the **sys** file in older version). This file is similar to the L.sys file of uucp. The ‘‘Configure links’’ command will take you through a series of questions and set up a new feeds file based on your answers (modified from the old feeds file if one exists). This generated file will be left at %feeds in the source directory; next time you run install.sh it will be copied over the old feeds file. The script detects and warns of missing UUCP links.

The ‘‘Configure links’’ method cannot be used to delete systems or edit link entries that span more than one line (the script will politely refuse to do the latter if you try), and it does not (for the sake of simplicity) set the rarely-used and somewhat esoteric U, H, S and Q options. The remainder of this section therefore describes the file as you’d see it in a text editor.

It is useful to know that a group subscription consists of a list of one or more group names or group wildcards. A group name is as it appears in a Newsgroups line, a dot-separated sequence of name parts. A group wildcard is similar except that it may contain the special segments "all" or "any" and may be prefixed by a "!". The "all" wildcard matches any sequence of segments; the "any" wilcard matches any single segment (thus for example the subscription "all.stuff" matches "foo.bar.stuff" and "baz.stuff" but "any.stuff" would only match the latter). A subscription preceded by "!" rejects the groups it would otherwise match -- they will be rejected unless the subscription contains a following list entry that matches them. Any subscription that matches a group "foo" will match any subgroup of foo (foo.bar, foo.baz etc.).

If you're in doubt about what a newsgroup subscription will match, run newsdb and select the 's' option at the top level, then select '?'. This will give you a menu of commands for experimenting with newsgroup subscriptions and feed file syntax in general.

Each feed file line contains five fields, separated by colons:

(1) The system name of a site to which you forward news. Normally all systems you have links to will be included. You should also have a line for your own system. If this field contains a slash, any list of comma-separated sites following it will be checked against the Path header of each article broadcast; if there is a match, transmission of that article will be suppressed (this is to enable you to economize on transmissions if you know what a downstream site's other feeds are).

(2) The newsgroups to be forwarded to them. This is a pattern in the sense of a subscription. Generally, you will list classes of newsgroups. A typical forwarding list for a new site would be

comp,news,misc,rec,sci,soc,talk,to.*sysname*

where *sysname* is the name of the remote system. In particular, you don't want to forward **all** or **any** since local newsgroups (those without dots) should not be sent. For the line describing your own system, this field describes the newsgroups your site will accept from remote sites. Thus, if another site insists on sending you a newsgroup you don't want, say, **rec.humor,** then include **!rec.humor** here.

(3) A distributions field giving abbreviations for distributions the site is in. All the wildcarding conventions described for groups work here also. A typical distribution list for a site in New Jersey might look like

usa,na,nj

(Of course if you are not in New Jersey, the USA or North America, you would remove those distributions and replace them with the ones appropriate for you). Actually, distributions are treated just like newsgroup prefixes (the previous field) except for being stripped off before news inserts articles in the spool directories. A site accepting the *nj* and *pa* (Pennsylvania) distributions, for example, might receive messages posted to *nj.wanted* and *pa.wanted*. Both would end up in SPOOL/wanted (though they'd be rebroadcast to other machines with the distribution prefixes on). Note that you can override this stripping behavior simply by locally creating *nj.wanted* and *pa.wanted* − thus, you can choose on a per-newsgroup basis whether or not distributions should be stripped.

(4) This field contains flags describing the connection. These flags are described in detail below.

(5) This field is the command to be run to send news to the remote site. The article will be on the standard input. Leaving this field blank means an ordinary uucp link is being used, that is, the command defaults to

```
uux − −r −z sysname!rnews
```

or (on System V Release 1)

```
uux − −r −n sysname!rnews
```

The − option tells uux to expect input on stdin. The −z option on System III and System V Release 2 shuts off the annoying message you would otherwise get mailed to you telling you that your article was broadcast OK. To avoid using it, change the source or put the uux command in the fifth field. The −n option shuts off all notifications on System V Release 1. The −r option tells uux not to start up a daemon right away. This turns out to ease the load on the system, at the expense of making news be transmitted a bit slower. The news will be sent when the next daemon is started, usually this means the next time mail is sent to or from your system. If this turns out to be unreasonably long, put a line in crontab to run

```
/usr/lib/uucp/uucico -r1
```

every hour or so.

Here are the control flags for the fourth field. Flags may be followed by an argument suffix, which is either a series of digits (interpreted as a decimal integer) or a double-quote enclosed string (double quotes within the string may be quoted with backslash, and the outer quotes are stripped when the argument is processed).

A Not currently used; reserved to indicate an alternate article format (X.400 or some such). Formerly used to indicate that the other site was running an A version of netnews, but support for A news has been *removed* from this release.

B indicates batched transmission. The Internet ID of each article will be appended to a batch file for later retransmission by **sendbatch**(8). The fifth field may give a transmission method, otherwise a default will be used. The sendbatch code will properly interpret the C, D, E, H, M, N, Q, S, and U transmission options. If V2.10.2 is on, sendbatch will emit 2.10.2-style batches (i.e. without the header line naming the unbatcher to be used). If digits follow the B they will be interpreted as the maximum transmission batch size for that system; the code will automatically split up larger batches.

C arranges for news sent to the given destination to be compressed. If the version at the other end is a TMN-netnews its rnews will detect this and automatically decompress the news; otherwise the system administrator there may have to make special arrangements. You must make sure you are running the same or compatible versions of compress. See the section below on setting up compressed and batched links. By default it is assumed that you are running 3.0 or 4.0 compress, and a string option is interpreted as command-line switches to be passed to it. If the -C option is given, a compress file compatible with V2 of compress will be generated. If you send news to systems that are running compress as distributed with 2.10.2 will need to add this flag until they upgrade. If the -bx flag is given, compress will only use an "x" bit compression instead of the default 16. A -b12 is necessary when sending to a pdp11 system.

E specifies that compressed news sent to that system must be sent in 7-bit form (the connection drops parity bits).

D When this switch is active on a connection, messages sent to that connection are intercepted just before being handed off to UUCP or mail and appended to the file xmitlog in the LIB directory along with the transmission command. This is useful for debugging links.

F causes article IDs to be appended to a filename given in the transmission field. Note that this is *not* the same as the old F option, which appended the filename of an article copy.

L prevents transmission unless the article was created on this site. If a number follows the L (e.g. L3), sites less than that number of hops away will be comsidered local. (It is recommended that you feed an L link to a backbone site, to ensure that your submissions will be more likely to get to the entire network, even in the event of a local problem. Please make sure that a mail link exists too, so you can get replies.)

N indicates that mail should be sent using the **ihave/sendme** protocol described below.

M Tells rnews to send news to the system via mail for handling with **uurec**(8). An argument is interpreted as the username to mail to. No argument sends the mail to rnews, unbatch, cunbatch or c7unbatch according to the other transmission option. The code sends it using your configured mailer (this option replaces the old sendnews command).

S says to execute the xmit command directly instead of forking a shell.

U arranges for the parameter to the optional %s in the command field to be filled in with a permanent file name from SPOOL instead of a temporary copy file name. Not meaningful used with F.

Q This enables you to set the maximum length of the UUCP queue for a target system that will allow batching to it. This feature can be used along with the SPOOLMIN define to prevent overrunning the available space on your spool device.

V Specifies the version of news running on the target machine. At the moment, only the argument "2.10" is meaningful (this will change). This is used by the transmit code to enable some archaisms for 2.10.X sites. These sites cannot handle links that have C without B or F without an ’uncompress’ header on the batch. More generally, they need news to be batched to one of the remote agents unbatch, cunbatch, or c7unbatch if compression or batching or 7-bit encoding is on; giving "V2.10" enables this.

X Tells the code that the system-list in this feeds file entry is a multicast group. Normally, a separate transmission would be queued for each system in the list. With X enabled, the software passes the (space-separated) list of systems to a single instance of the transmission command. If no transmission command is given and

UUCAST is on, the code will attempt to do a UUCP multicast by diddling UUCP control files. If the M option was specified, a multiple-recipient mailer call will be generated.

Here is a sample feeds file for a site ‘‘myvax’’ with connections to ‘‘yourvax’’, ‘‘theirvax’’, and ‘‘foovax’’. We assume that ‘‘myvax’’ and ‘‘foovax’’ exchange a local newsgroup class lng.all as well as the network wide newsgroups. News to ‘‘foovax’’ is batched. We also assume that myvax, yourvax and theirvax are in the USA, while foovax is in Canada.

```
myvax:news,comp,lng,to:na,usa::
yourvax,theirvax:news,comp,to.yourvax:na,usa::
foovax:news,comp,lng,to.downstream:na:B:
```

Note that the feed file format is *different* from the sys file format of previous versions; the old subscription field has been split into the new subscription and distribution fields. Also, the fifth field of batched-feed descriptions now specifies a transmission command, *not* a batch file location as in older versions; sendbatch computes the batch file location itself.

Important note: if you write a feeds file line that has only four identifiable fields, the file parsing code will interpret it in the old style as a four-field line without separate distribution field. This is an unashamed compatibility hack and may go away soon; don’t count on it.

4.4. Setting up compressed and batched links

Setting up links that gather news transmissions in batches and send them at preset times can save you a lot of connect time, as can using the data compression utilities provided with the distribution to reduce transmission time. You’ll need a working version of the Lempel-Ziv **compress** program. Use the one shipped with this news distribution, which is based on version 4.0; Your news neighbors should be running a compatible version of compress. Versions 3.0 and 4.0 are compatible with each other, but both are incompatible with versions 2.0 and before.

Update your **feeds** file. First, add the B and C flags to the other news system’s line. For instance, if your compressed-and-batched news feed is named frobozz, and its **sys** file entry looks like:

```
frobozz:comp,sci,soc,to.frobozz:na,usa,ca::
```

then add FC as the fourth (colon-separated) field:

```
frobozz:comp,sci,news,to.frobozz:na,usa,ca:BC:
```

Now the pathnames of articles to be sent will be stashed in a file BATCHDIR/<sysname>; in this case it would be BATCHDIR/frobozz. You can specify a batch file name in the fifth field via the F option, but this shouldn’t normally be necessary. Your completed **feeds** file line should therefore look something like:

```
frobozz:comp,sci,soc,to.frobozz:na,usa,ca:BC:
```

In /usr/lib/crontab, find or create a once-a-day entry to run **expire**(8), trim log files, and perhaps compile weekly statistics that you post to a local-area newsgroup one day a week. Expire runs **sendbatch(8)** automatically before doing expirations (this is necessary to make sure postings to locally volatile groups don’t get clobbered before being sent). If you want news batches to be shipped more than once a day, call sendbatch from crontab with a line like

```
/usr/lib/news/sendbatch frobozz
```

or

```
/usr/lib/news/sendbatch
```

(the latter form checks and sends batches for all systems, the former for frobozz only). **Sendbatch** reads the files mentioned in a system’s batchfile batches them, compresses them, sends them to the remote system, and arranges for remote processing.

Now your system will transmit compressed batches. The receiving side of the business is all handled by rnews (this is one of the News 3.0 (beta level 7) improvements!), though for backward compatibility we dress it up as cunbatch. You must have a **compress** in *LIBDIR* (wherever your makefile defines it), because **rnews** will eventually try to exec that copy.

If the other site is running a 2.11B or older netnews, tell the person at the other end of your newsfeed to use the old **csendbatch** program and a uux command of **cunbatch** or (if the line drops parity bits) **c7unbatch** to send you news.

If the other site is running a 3.x netnews, have them use sendbatch and use C and possibly E options. Once that's in place, watch your uucp LOGFILE and your news **log** and **errlog** files to ensure that news is being correctly received and unpacked on your system.

5. News library data files and programs

5.1. Data files

This section lists the files in ADM and comments briefly on what they do. Normally ADM is just an alias of LIBDIR, but if you've selected the SHARED option it will be a separate LIBDIR/.admin.

5.1.1. active

A list of active newsgroups (replaces the *.ngfile* of earlier versions). For details on the format, see the **news**(5) manual page. Automatically updated as new newsgroups come in. The order here is the order news is initially presented by **readnews**, so you can edit this file to put important newsgroups first. If you have **SORTACTIVE** defined, after the first time the user invokes readnews, it will be presented in the order of his .newsrc.

The active file should contain **ALL** active net-wide active newsgroups. It is important that they all be present, as they are used as a check for valid newsgroup names. You should use the **feeds** file to keep out unwanted newsgroups.

5.1.2. admin

This file contains a list of pairs of newsgroup subscriptions and control-flag fields (separated by whitespace). Each control-flag field sets administration parameters for groups that match the subscription on that line. These parameters may include group expiration time and flags which tell if the group is volatile, whether articles in it should be stored compressed and the like. See the **news**(5) manual page for format details.

5.1.3. aliases

This file is used to map bad newsgroup names to the correct ones. (For example, comp.unix-wizards is mapped into comp.unix.wizards). Each line consists of two fields separated by a space. If the first field is found in the newsgroup list of the incoming article, it is changed to the second field. This change takes place in the article before it is passed on to other systems, not just locally.

5.1.4. distributions

This is a list of distributions that are valid for your site. Each line has two fields separated by the first space on the line. The first field is the name of the distribution (e.g. usa, na, etc.). The second field is text describing the distribution. As distributed, this file is only correct for sites in the USA. You should examine this file and add or delete the appropriate distributions from the file distrib.proto in your source directory.

5.1.5. errlog

This file contains the ‘‘important’’ error messages found in the log file. These errors usually indicate that something was wrong with an article. This file should be watched closely. The **log** file contains much more verbose information and it is often difficult to detect errors in it.

5.1.6. feeds

A list of all your neighbors, which newsgroups they get, and how to send news to them. The format is documented in the news(5) manual page.

5.1.7. followups

This file defines default followup groups. Postnews uses it to fill in Newsgroup fields on followup articles and Followup-To fields on original articles. It is similar in format to the aliases file.

5.1.8. history

A list of every article that has come in to your system. Used to reject articles that come in for the second time (presumably via a different path). This file will grow but is cleaned out by the expire command. Normally this file contains information on all messages received, posted cancelled and expired during the last HISTEXP seconds (normally 4 weeks). If you have enabled DEBUG a version of this file will be maintained for human perusal, but the real information will be kept in

5.1.9. history.dir, history.dat, history.pag

the three database files maintained by the history of the code.

5.1.10. log

If present, a log of articles processed and error conditions is kept here. This file grows without limit unless cleaned out periodically, the newslogs script in misc can be invoked from /usr/lib/crontab daily or weekly to keep the log short.

5.1.11. moderators

This file contains a list of the moderators and their mailing addresses for each moderated newsgroup. Each line consists of two fields. the first is the name of the moderated group. The second is the mailing address of the group’s moderator(s) (thre may be one or more, separated by commas and/or whitespace). As distributed, they are almost certainly wrong. You will need to modify the paths so they work from your site.

5.1.12. newsgroups

This file is displayed by **postnews** when a user hits “?” in response to its request for newsgroups. It is updated automatically by the **checkgroups** control message.

5.1.13. notify

If this file is present, its contents will be taken as the name of the user to notify in case of a problem. If the file is empty, nobody will be notified. (This overrides the NOTIFY configuration option switch.) This is useful if one person administers several systems and does not want multiple copies of control message notifications.

5.1.14. readnews.help

A list of commands printed when an illegal command is typed to **readnews**.

5.1.15. recording

A list of newsgroup classes and filenames to display recordings for. The recording feature is analogous to the recordings played in some areas when you dial directory assistance, trying to be annoying and make you think twice. Recordings on certain newsgroups are intended to remind the user of the rules for the newsgroup, or, in the case of a company worried about letting proprietary information out, reminding authors that anything they say is seen outside the company and so proprietary information should not be included.

The file contains one line per recording. The line contains two fields, separated by a space. The first field is the newsgroup class (e.g. “news.all, rec.all”), the second field is the name of the file containing the recorded message. If the file name does not begin with a slash, it will be searched for in ADM. Sample recording files can be found in the misc directory.

5.1.16. vnews.help

This is the helpfile used by vnews.

5.2. Auxilliary programs

This section lists the programs in LIBDIR and comments briefly on what they do. Some are helpers called by interface programs, others used for cleanup and maintenance.

5.2.1. arbitron

This command is used to collect statistics on newsgroup usage patterns, optionally posting or mailing them to a collection point for inclusion in net-wide summaries. The standard installation procedure sets it up in cron to run monthly, sending one copy of the summary to the news administrator and another to the net's collection point for arbitron statistics.

5.2.2. caesar

A program to do caesar decoding of rotated text, on a line by line basis. The standard input is copied to the standard output, rotating each line according to a static single letter frequency table. If an integer argument is given (e.g. 13), every line is rotated by that argument, without regard to letter frequencies. This program is invoked by the ‘‘D’’ readnews command. It is also used by postnews with the ‘‘13’’ argument to encode selected material for posting.

5.2.3. checkgroups

checkgroups is a shell file to aid in automatic checking of the accuracy of your active file. It is executed by the *checkgroups* control message and mails a list of out of date newsgroups to the person defined by **NOTIFY** It also updates the **newsgroups** file that is used by **postnews** as a helpfile for newsgroup selection.

5.2.4. compress

This shell script is used for data compression by the compressed batching scheme. It averages 50% compression on a typical batch of news.

5.2.5. delay

This shell script generates a report on news propagation delays.

5.2.6. euuname

This shell script may be used to edit the information returned by net mapping messages.

5.2.7. expire

This program expires old articles and archives then if archiving is selected. It is typically run once a day from **cron**. It always does a run of **sendbatch**(8) before deleting messages so that postings to groups that are locally volatile get shipped before possibly being clobbered. It also runs an **rnews**-U to transmit news spooled up during the expire.

5.2.8. inews

This program provides a minimum front-end for **rnews** (see below). It is provided primarily for backward-compatibility with 2.10.3 and earlier versions.

5.2.9. newsarchive

A script useful for archiving news articles. You can use this as the command in a fake news entry to automatically archive articles on receipt. See the header of newsarchive.sh for further details.

5.2.10. newslogs

A program to be run weekly to keep the size of your logfile down. As distributed it keeps the last six days' log files around.

5.2.11. newsspace

This shell script generates a report on the disk space usage of newsgroups.

5.2.12. recnews

A script that simply hands mail off to inews for posting (inews now knows about mail headers, see inews(1)). This is useful primarily for automated posting of Internet mail from an Arpa-to-UUCP gateway, but can be used to permit users to post news by mailing to a pseudo-user. Sample lines in /usr/lib/aliases (if you run delivermail) to do this:

worldnews: "|/usr/lib/news/recnews"
: Allows you to mail to worldnews rather than using inews. Intended for humans to mail to.

foo-wizards: "|/usr/lib/news/recnews
: Causes mail to foo-wizards to be posted to comp.foo.wizards and the return address forged as "merlin@site" where "site" is this machine. User "merlin" (on the local machine) should be part of the master mailing list somewhere (on a different machine).

5.2.13. rmgroup

This shell script should be used to remove any groups that are no longer used.

5.2.14. rnews

This is the program that actually sends and receives news. All other programs interface eventually with it. It is not intended to be used directly by a human, except for moderators broadcasting accepted submissions that already include all RFC-850 required headers.

5.2.15. sendbatch

This utility is used to send a batch of news to a list of target systems. Typically, you'll run it from cron once or twice a day. It checks the feeds file entry of each target machine for the C compression flag, and if so applies compression to the batch just before sending it. In order for the C feature to work, the target machine must either be running News 3.0 (beta level 7) or have some version of the old **cunbatch** shell script available to unbatch it.

5.2.16. uurec

A program to receive news sent by mail using the M transmission option.

5.2.17. newsclean

This script removes stray files from the spool area and rebuilds the history file. It should be invoked periodically to keep the size of the history databases down.

6. Posting Methods

The basic method is **postnews**. This program will prompt you for the title, newsgroups, and distribution, then place you in the editor. (The system default EDITOR is used unless the environment variable EDITOR is set, overriding the system default.) The text should be typed after the blank line. The title and newsgroups are available for editing at the top of the buffer. Other header lines can be added, such as an expiration date or distribution. When you write out the file and exit from the editor, the article will be posted.

Another method is to use mail. This can only be done on systems that allow mail to a given name to be fed into an arbitrary program as input. This is easily done with the Berkeley delivermail or sendmail program, and not with any other mailer the author is familiar with. (It may be possible to painfully set this up with MMDF, provided the newsgroup name is no more than 8 characters long.) To use mail, set up an alias such as the following:

```
comp.ai: "|/usr/lib/news/recnews comp.ai"
```

Whenever a user sends mail to **comp.ai**, this starts up the given shell command which calls recnews with a single argument, the name of the newsgroup. You need to create one alias for each newsgroup, and to keep the list up to date as new newsgroups are created. Note that recnews is now a shell script and you can easily customize the filtering it does to suit your mailer conventions.

7. Permissions and security

7.1. Suid and sgid bits

The current intended state of affairs is that *rnews* runs suid NEWSUSR. The various interface programs such as *readnews* do not need to have special permissions, unless the site administration has elected to run with COMMUNIST on. In that case, news database files will reserve read permissions for programs that are suid or sgid news, and you'll need to make readers sgid news (the installation procedure does this for the standard set of readers automatically). Note that in News 3.0 (beta level 7) the inews program is also a front end in this sense and does not need NEWSUSR privileges.

7.2. Modes of Spool Directories

All the files should be writable by NEWSUSR. However, due to a glitch, you will probably have to make SPOOLDIR and its subdirectories mode 777 on V7 and some older AT&T versions. It could be 755 except for one problem. When a new newsgroup comes in, *rnews* will attempt to *mkdir* a new subdirectory of SPOOLDIR for the newsgroup. Since both inews and mkdir are suid, mkdir will use the real uid instead of NEWSUSER when checking for permissions, and if the directory isn't 777 the check will fail. Here are several alternatives if you don't want a 777 directory around:

7.2.1. Fix Real Uid

If inews is always run from cron or by root, the real uid can be arranged to be root or NEWSUSR. This is a poor solution since it makes the local creation of new newsgroup require super user permissions, and is a potential security hole. If this approach is taken, care must be taken to insure that the owner of the created directory is NEWSUSR.

7.2.2. Change the Kernel

The *rnews* code will do *setuid(geteuid())* before it forks the mkdir. If your system permits this call, there will be no problem. In particular, Berkeley 4.0BSD and later systems allow this. An alternative change to the kernel is to automatically stack uids: when an suid program is run, set the new real uid to the old effective uid.

7.2.3. Groups

You could have inews be sgid NEWSGRP and all files writable by the group. This approach has been tested and the problem turns out to be that the **mkdir** command uses the **access** system call to check permissions. Since **access** uses the real gid, you run into the same problem.

7.2.4. Another mkdir

You could create a version of mkdir that does less checking, and put it in a directory that can only be accessed by NEWSUSR (mode 700, owned by NEWSUSR). Have inews fork this mkdir.

7.3. Running with COMMUNIST on

If you run with COMMUNIST on, you can control posting *and reading* access to groups on a per-user basis. But before configuring for COMMUNIST mode, think twice. Then think again. USENET is built on a tradition of open information sharing, and the vast majority of sites need not and should not run COMMUNIST. Please help keep the net open and the net.culture healthy; don't run COMMUNIST unless you absolutely must.

In COMMUNIST mode, netnews uses group permissions to achieve news database security. All news database files (including article text) are created and maintained owned by NEWSUSR and NEWSGROUP, with permissions u=rwx,g=rx,o-rw (750). Thus only programs that are suid or sgid NEWSUSR can read news database files. Only programs that are suid NEWSUSR can write them. Readers are sgid NEWSUSR; rnews, expire and sendbatch are suid NEWSUSR.

Within the interfaces, security checks are done in the standard service layers that news readers and posters use to get at the news database. Essentially, if you're locked out of a group, the service layer forces your subscription bit for it off at startup time, so you never see it. The few reader commands that permit you to go to a not-previously-subscribed group have their own checks. It will be the site administrator's responsibility to ensure that homebrew readers use the standard service libraries (and thus incorporate the security checks).

Security restrictions are expressed in the ADM/authorized file. The ADM/fascist file of older versions had two colon-separated fields per line, the first a user or group name and the second a subscription specifying those groups for which said user or group has posting privileges. The ADM/authorized file has a new format including a field to specify *reading* privileges. See for details.

COMMUNIST mode has two minor disadvantages. One is that you have to newgroup news or su news or root to snoop the machine history and active files to track problems. The other is that users who want to roll their own readers (and aren't just front-ending the ednews tool) will need to get someone with root privileges to set their programs sgid NEWSUSR on each runtime generation.

For really super-tight posting security, run COMMUNIST, turn off 'other' execute privileges on rnews and have each newsfeed login start in the news-owning group (rnews must be group-executable for this to work, of course). This way, ordinary users won't be able to execute rnews directly.

8. Miscellaneous administrivia

8.1. Expiration dates

To get articles to expire automatically, put a line in crontab to run

/usr/lib/news/expire

every night (the standard installation procedure does this for you). This command deletes all expired news. The −a *newsgroups* option causes all expired news to be archived under /usr/spool/oldnews depending on which newsgroups are selected. (See **expire(8)** for details.

Sometimes news is not expired when it should be. Be sure to check that expire has permissions to unlink files, and that it is properly suid. You can manually invoke expire with the −v (verbose) option to find out what it's doing. Adding levels of verbosity (e.g. −v 6) will get more and more output.

8.2. Presentation Order

The order of the newsgroups listed in ADM/active is the order the newsgroups will be presented in initially. If **SORTACTIVE** is defined in defs.h, after the first time news will be presented in the order of the persons .newsrc. Initially this will be directory order, but you can edit important newsgroups like general to the top.

A recommended order to maintain your active file in is this:

general
local.general
misc.misc
local newsgroups, in alphabetical order
news newsgroups, in alphabetical order
comp,misc,sci,soc,rec & talk groups
test
all.test
to.all
control
junk

9. Control Messages

Some news systems will send you articles that are not for human consumption. They are messages to your news system called *control messages*. Such messages contain the **Control:** header. Older systems use newsgroups matching **all.all.ctl**, and this will still work, although the **Control:** header is preferred. Since the newsgroup name is used for distribution only, and is not checked to ensure it's in the active file, such newsgroup names can still be used. Messages are cancelled, however, with a **Control:** line in a message to the same newsgroup(s) as the original message.

A control message contains a command and zero or more arguments (much like a UNIX program). The subject of the article contains the command and arguments. The body of the article is usually ignored, although some messages can use it for additional text information.

9.1. ihave/sendme

Two control messages are **ihave** and **sendme**. These messages allow two participating sites to set up a link so that one site will tell the other site it has a given article and wait for a request before it actually sends it. The normal case is to send an entire article to a system, which consults the history file to see if the article has already been seen, and then throws it away if it's been seen before.

Note that, since most messages are short anyway, experience has indicated that for ordinary UUCP unbatched communication, all ihave/sendme does is triple the load and slow down forwarding. If you're using a batched link, on the other hand, N protocol may reduce your transmission volume. Try it and see.

Use of these control messages can cut down on this wasted transmission, but if you have a polled UUCP connection, they can slow down receipt of news due to polling delays. It is up to each connected pair of sites whether they want to use this protocol. The choice is controlled by the N flag in the feed description. In the case of a leaf node (one with only one neighbor) there is no advantage to this protocol. Even if both sites are able to initiate a connection (have dialers or the link is hardwired) the −r option on the uux can cause 2 hour or more delays in propagating news. Since this protocol can triple the number of messages generated, you should carefully evaluate your situation when deciding whether to use it. If transmission time and phone bills dominate your costs, and you are sending news to several sites, and large article bodies dominate the costs (rather than the headers and the time spent by UUCP negotiating transmission) it is probably worthwhile to use ihave/sendme. If your costs are dominated by CPU load from UUCP, or if you send news to a site that cannot get it from anywhere else, you probably do not want to use this protocol. The decision can be made independently for each site in your feeds file.

This pair works as follows: Site **mysite** receives article **<123@abc.UUCP>**. It enters it locally and then broadcasts it to its neighbors. One of its neighbors is site **yoursite** which has the N flag in the **feeds** file. So mysite sends an article on newsgroup to.yoursite.ctl with title mysite''.**``ihave**<123@abc.UUCP> This control message has two arguments - the first (**<123@abc.UUCP>**) is the article ID of the article in question, the second (**mysite**) is the name of the site sending the article. The name of the newsgroup and the feeds file control transmission of the article, normally the feed description will read something like

yoursite:comp,misc,sci,rec,soc,talk,to.yoursite:usa,pa,na:BN:

which will cause an article on to.yoursite.ctl to be transmitted.

Yoursite receives the message and looks to see if it has seen it before. If it has, it throws the message away and stops. If it hasn't, it sends a message on to.mysite.ctl with title yoursite''**''sendme**<123@abc.UUCP> which is transmitted to **mysite**. (The two arguments to sendme are the article ID requested and the site to send it to.) Then mysite gets this message and actually transmits the article to yoursite.

9.2. newgroup

This message has one argument, the name of a newsgroup to be created. This allows special action to be taken locally when a new newsgroup is created. It is generated by the −C option to inews. By default, the newsgroup is added to the active file and a directory is created, and mail is sent to the local contact advising that this has happened. See the routine ''c_newgroup()'' in control.c if you want something different to happen.

9.3. rmgroup

This message has one argument, the name of a newsgroup to be removed. It is used for network-wide cancellation of a newsgroup. If MANUALLY is not defined, it will remove the articles, directory, and active file line for the group. There is a shell script **rmgroup** that does essentially the same thing as this message, but the shell script only removes the group locally. We recommend that you leave MANUALLY defined, and when you receive mail advising you of the demise of the newsgroup, you run rmgroup by hand. This will prevent accidental or malicious removal of a good newsgroup.

9.4. cancel

This message cancels a given article. It takes one argument, the message ID of the article to cancel. It should be broadcast to the same newsgroup as the original article.

9.5. sendsys

The feeds file is mailed to the originator of the message. There are no arguments. This is used for making maps. Since your feeds file is public information, you should not remove or change this control message.

9.6. senduuname

The **uuname**(1) program is run and the output is mailed to the originator of the message. There are no arguments. This is used for making uucp maps. If you do not run UUCP or have sites in your L.sys which are a secret, you may wish to edit this. Note that only the output of uuname is mailed, not the contents of L.sys (which news does not have access to anyway). If you do make a change, you should arrange that some mail still is sent out to the originator of the message, so he will know your site received it. See the code in routine c_senduuname in control.c.

9.7. version

The local version name/number of the netnews software is mailed back to the author of the control message.

9.8. checkgroups

This control message is an attempt at semi-automatic maintenance of the list of active news groups. This control messages takes the body of the article and pipes it into LIB/checkgroups. As mentioned previously, LIB/checkgroups will update the newsgroups file, add any missing newsgroups and mail a message to **NOTIFY** about any old newsgroups that should be removed. It is expected that the person who maintains the list of active newsgroups will broadcast this control message on a regular basis.

9.9. addsub

This control message allows a downstream site to add groups to its feed. The message has two arguments; the site name and a group or comma-separated group list. The request succeeds only if the requested set of groups is within the subscription defined by the group's flexgroup information; a nonexistent flexgroups field is treated as a maximum subscription of 'all'.

9.10. delsub

This control message allows a downstream site to drop groups from its feed. The message has two arguments; the site name and a group or comma-separated group list. The sitename is checked against the incoming article headers.

9.11. Other Messages

Any unrecognized message will cause an error message to be mailed to the news administrator. Additional messages may be defined as time goes on, such as messages to automatically update directories or maps. You should be willing to go into the code (control.c) and add messages as they become standardized.

10. Administrative arcana

10.1. Preventive maintenance

There are some things you should do periodically to keep your news system running smoothly. In particular, the **history** , **errlog** and **log** files in your LIB directory will grow, and you should make sure that they are cleaned up periodically.

The LIB/expire program will remove lines from history corresponding to deleted articles, but it is a good idea to check the file every few months to make sure it is not going wild. Be sure not to completely lose your history file when you clean it up, in case another neighbor tries to send you an article you recently got. (If you only get news from one site it is safe to clean it out completely.)

The LIB/newslogs script can be invoked weekly from crontab to prune the **log** and **errlog** files. As distributed it will keep around a week of logfiles in day-sized chunks; you can season to taste.

You should also clean out old newsgroups that are no longer active. To remove a newsgroup net.foo, you should run the shell script LIB/rmgroup with net.foo as the argument. I.e.

```
/usr/lib/news/rmgroup net.foo
```

Note that clearing up UUCP constipation is another thing you'll have to do if you have flaky hardware or phone lines. If you have more than one connection, chances are that UUCP will get clogged up when one of your neighbors goes down for more than a few hours. Various spooling schemes are being worked on to help make the news/uucp system more robust, but one thing you can and should do, if you find your /usr/spool/uucp directory getting too big, is to install a subdirectory fix to UUCP. A quick and dirty version of this is available from Duke, which traps the open, creat, link, etc. system calls at the assembly language level and maps, for example, D.fooA1234 into D.foo/D.fooA1234. Since the C. and D.local directories still get big, in practice this can still create some big directories, but the directories tend to be a factor of 5 smaller, resulting in a factor of 25 improvement to speed (since a directory traversal for all files is quadratic on UNIX). Right now, UUCP is the weak link in netnews distribution, and you should certainly keep an eye on it.

10.2. Controlling spool space utilization and expire execution times

News 3.0 (beta level 7) gives you the ability to control your spool space utilization by setting message-expiration periods on a newsgroup-by-newsgroup basis. This is done using the optional fifth field of the active-groups file, which may have the following parts:

<number>
Expire articles in <number> days. The period may be shorter or longer than DFLTEXP; it overrides the default completely.

& As a suffix, means this group is volatile. Articles in it may be dropped as soon as all subscribers have seen them (on single-user machines all groups can be volatile, drastically reducing the spool space requirements).

! As a suffix, causes expire to always ignore explicit expire dates on articles posted to the group.

Note that under the new dispensation different copies of a cross-posted article may expire at different times. The history of an article is not dropped until the last copy expires. The *sendbatch*(8) utility's batch files are lists of Internet IDs, so if there are any copies of a message unexpired when it runs one will be found.

Setting the ‘‘c’’ flag on a group in the active file (by suffixing the flags field with ‘‘c’’) causes messages to be stored in compressed form. All netnews programs that read message files now open them through a routine that detects and undoes compression, so the effects of this flag are invisible except that it trades reduced spool space for increased access time. You must be running compress 3.0 or later to use this feature.

3.x versions of expire usually run much faster than formerly because the new history format includes an explicit expire date field, which means expire doesn't need to *fopen*(2) each article and read its header to find whether it should be expired. Be aware that you lose this advantage, however, when using -h, -r, -s, -p, -f or -a options.

10.3. Tips on how to keep things from bogging down

If you find that rnews processing is bringing your processor to its knees during prime time, recompile with the SPOOLNEWS option. Then your incoming news will be processed each time you run expire (actually, just before it). You can schedule separate unspooling runs (rnews -U) with crontab.

It is strongly recommended that you run with SPOOLNEWS and SPOOLPOST turned on, and start a "rnews -d" in your bootup script. In this configuration, individual invocations of rnews will place articles into a temporary spool directory. Then, the rnews daemon will pickup those articles batch them in at once. This way, an rnews run can pickup many, many articles at once, reducing the cost of its startup overhead.

NNTP or other network sites that start up many rnews instances may want to compile with SPOOLNEWS and then run rnews in -d (daemon) mode. This is the equivalent of a perpetually-repeating rnews -U; it avoids repeating the high startup overhead of a conventional rnews run.

If posting individual news articles becomes really slow, your history database files are probably getting very large. Arrange crontab to run expire -r more often.

If your machine has long alignment problems, makdatum() is going to be slow as a dog, since it is used *a lot* by the edbm routines, and it invokes memcopy() to copy four bytes two or three times.

10.4. Usage Monitoring

This distribution provides some scripts for generating some useful reports on net usage and performance. These are as follows:

arbitron	-- report group subscription ratings
delay	-- report information on net propagation delays
newsspace	-- report news group disk usage information

These tools contain documentation comments describing their use. You can help USENET readers everywhere by setting up arbitron to mail its output to the newslists moderator, see the directions.

10.5. Creating New Newsgroups

As system news administrator, you are able to create newsgroups. To create a newsgroup, first make sure this is the right thing to do. Normally a suggestion is first posted to news.groups,net.relatedgroup for a net newsgroup (net.relatedgroup should be the group which you are proposing to sub-divide. E.g. to propose creating rec.arts.tv.soaps, post the original article to rec.arts.tv and news.groups). Followups are made to news.groups ONLY.

(You can force this by putting the line:

Followup-To: news.groups

in the headers of your original posting). If it is established that there is general interest in such a group, and a name is agreed on, then someone creates it by typing the command

inews −C *newsgroup*

This will create the directory and active entry locally. It will also prompt you for a paragraph describing the group and start up an inews to post a newgroup control message announcing the group. This control message will be sent out on net.msg.ctl and other sites may have configured their systems to do something with these messages. A human readable announcement is not made - you can post this to news.group if necessary.

You must be root or the news administrator to use the −C option to inews. (That is, your uid must match 0 or the NEWSADMIN configuration symbol. It is recommended that you change NEWSADMIN to your own uid so you don't have to su to create newsgroups.)

Finally, you should choose an expiration period for the new group. This is done by setting the expiration period in the admin file as decribed above.

See the newgroups document included with this distribution for more details.