Installing ‘‘C News’’ Network News Software

Geoff Collyer

Department of Statistics
University of Toronto

ABSTRACT

This document describes the flow of network news within and between machines, what each component of C news does, and how to install C news.

Introduction

Network news (or *netnews* for short) consists of a collection of messages formatted similarly to ARPAnet mail (see ARPA Internet RFC 1036 for details), widely spread. The logical network, imposed on top of various real networks, formed by the set of all interconnected sites exchanging network news is called ‘‘Usenet’’ and was formed in 1979, radiating out from Duke University. Netnews is propagated between cooperating machines by a flooding algorithm, with some loop prevention heuristics: each machine sends its neighbours news that the neighbours have (probably) not yet seen.

Flow of netnews between machines may be achieved by use of any network which can transmit an arbitrary stream of (at least 7-bit) ASCII code, unmodified. If a network cannot transmit ASCII intact (e.g. BITNET), it is possible to encode netnews before transmission across the network and decode it upon reception from the network. Since one cannot be certain that one’s network neighbours will be up and reachable at all times, outgoing netnews must be queued, at least in the case of network trouble. To date, at least these networks, protocols and media have been used to transmit netnews correctly: UUCP, RS232, NNTP, Ethernet®, the ARPA Internet, Datakit®, ACSnet, magnetic tape, SMTP, and BITNET, though at least the last two require some form of encapsulation to avoid corruption of articles; SMTP because some common implementations get the newline-CRLF mappings wrong, thus throwing off byte counts in batches, and BITNET because of its Procrustean chopping, expanding, mapping, bashing and smashing of all data sent through it (sending lines of 80 or fewer characters of letters of either case and digits and plus and minus appears to be safe).

Netnews arrives via some network, which causes a command to be executed upon arrival (e.g. *rnews*). *rnews* typically spools the inbound netnews for later processing. Eventually (often within the hour or at the end of the business day), the input queue is run and the netnews is stored locally, typically under */usr/spool/news*, and queued for transmission to netnews neighbours. Once stored on disk, netnews may be read by any of a collection of unprivileged news readers, including *cat*(1). *Expire* is run typically each night to remove old netnews from disk.

C News was originally written to provide a much faster and smaller, more robust, reliable and correct implementation of netnews software than B 2.11 news. We believe that C News is also faster, smaller and more robust than B 3.0 news. B 3.0 news has many more features; take that as you will.

C News Components.

Rnews invokes the input subsystem, which spools incoming netnews in its original form, as received, typically in compressed batches. Periodically, the input queue should be run by invoking *newsrun*, thus uncompressing any compressed input and feeding it to *relaynews*.

Relaynews files incoming netnews as articles on disk and initiates spooled transmission to other machines, often by simply writing the names of the disk files containing the articles on the ends of ‘batch’ files of file names. Quite a bit of policy from RFC 1036 is embedded in *relaynews*. *inews* is a complex front-end for *relaynews* which implements much of the per-site policy on news posting. *inews* is a fairly-slow shell script which is adequately fast for most sites, but it will need to be replaced by something more streamlined for sites which gateway mailing lists into netnews, for example.

The output *batcher* reads lists of file names and generates news batches (see RFC 1036 or *news*(5) for the format) of prescribed sizes and queues the batches for transmission to other sites. The batcher is run asynchronously with *relaynews*, typically once an hour outside of business hours.

Expire is generally run once per night to remove from disk news articles older than a few days. *Expire* can use different expiry criterion for different newsgroups and can archive articles instead of removing them. *Expire* also runs asynchronously with *relaynews*.

There are many news readers. C News comes with a limited news reader (*readnews* by Michael Rourke) sufficient to replace long-winded */etc/motd*s but you will want a heavy-duty news reader if you plan to read more than local news groups. We recommend *rn* by Larry Wall, available from your netnews neighbours or your nearby **comp.sources.unix** archive site. There are others: *vn* and *vnews* are two.

Preparation for Installation

Netnews consumes a lot of disk space and often a lot of transmission time. Here are some relevant measurements regarding a full netnews feed as of the time of writing (January 1989), taken from *news.lists*: a day’s netnews is about 3Mb and 1,400 articles in 450 newsgroups. Years ago, sites often kept 14 days of netnews on disk, but now many sites keep news for 3 to 5 days, thus allowing for the occasional long weekend. Thus a full news feed expired after 4 days will consume about 12Mb. Some people feel that news volume is doubling roughly every 16 months. If this is true, we can expect volume to increase by a factor of 10 in about 4 years to 30Mb per day or 115Mb for 4 days. It is thus wise planning to set aside a lot of disk space for netnews. There are two ways to cope with ever-increasing volumes of netnews: refuse to accept more newsgroups, or expire news after shorter intervals on disk. A current full feed takes just over 7 hours to transmit at 1200 baud, so for dial-up connections one clearly wants the fastest modems one can afford.

Clearly, transmitting a full news feed is a non-trivial commitment of resources, so you may have some difficulty in finding a site willing to supply one. Such a site may in turn expect you to feed yet other sites. You will need to agree with your prospective netnews neighbour(s) upon transfer media, protocols and networks.

Before proceeding to install C News, you should read this document through to the end, probably read the companion document *The Interface Between C News And The Outside World*, and possibly read selected items in the *C News Implementor’s Notebook* and the manual pages.

You will need to assign a user id and group id to netnews (often new ids called ‘‘news’’); initialise these directories: NEWSCTL (typically **/usr/lib/news**), NEWSBIN (**/usr/lib/newsbin**), and NEWSARTS (**/usr/spool/news**); and install each subsystem. NEWSCTL and NEWSARTS are logically one subtree, defining a news data base, but are split for backward compatibility with older news software, particularly old news readers. NEWSBIN contains programs and shell scripts which might be common amongst machines sharing a common architecture (e.g. Sun 3’s); NEWSCTL/**bin** may override these. The goal is to install the subsystems, integrate them into a working news system, and configure the news system to communicate with other news systems.

There are a few key files that must exist before any serious attempt may be made to operate the news software. NEWSCTL/**active** is the list of newsgroups that this site knows (is willing to accept or individually reject), and must be owned by the NEWS userid (the userid that owns NEWSBIN/**relay/relaynews**, typically *news*). You will probably want to get your initial **active** file from your upstream feed and edit it to suit the set of groups you wish to receive. Be sure to make the second field more than five digits wide, by adding leading zeroes (ten digits is a conservative width). NEWSCTL/**sys** defines the newsgroups that this site is willing to accept and describes how they are to be forwarded to its neighbours. NEWSCTL/**server** contains the hostname of your file server, if you have multiple machines sharing news via a network file system. NEWSCTL/**whoami** similarly contains the name by which a cluster of machines sharing news is to be known for purposes of news. NEWSCTL/**mailname** is optional and contains the full (possibly dotted) name by which your cluster is known for purposes of mail. NEWSCTL/**organisation** (or

NEWSCTL/**organization** if you insist) contains the name of your organisation, which will be copied into the **Organization:** header of articles posted locally, by default. NEWSCTL/**mailpaths** defines mail routes to machines which contain aliases for postings to moderated newsgroups. NEWSCTL/**log**, NEWSCTL/**errlog**, NEWSCTL/**history**, NEWSCTL/**history.dir**, and NEWSCTL/**history.pag** must exist and be owned by the NEWS userid. Tentative versions of all these files are built by the installation procedures, but it is quite likely that you'll have to edit some of them.

C News Installation

Proceed to the **conf** directory of the distribution. There is a major shell file here, named **build**, that will interrogate you at length and construct shell files to do the real work. You may need to do ‘‘chmod +x build’’ before running it, to make it executable.

You will probably need your system's manuals on hand to answer **build**'s questions. Another terminal (or another window, on a bitmap display) would also be useful. You'd best be prepared to take notes, also, as **build** will occasionally suggest that something be checked when you're done.

Build itself does not alter any files or perform any installation chores: all it does is create shell files in the **conf** directory. If you already know something about news software, or are merely suspicious that **build** hasn't done everything right, you should probably read the shell files before running them. There are four of them: **doit.root**, **doit.bin**, **doit.news**, and **again.root**.

Doit.root sets up the major directories for news, and sets their ownerships correctly. It typically will have to be run as *root* to have adequate permissions for all of this. It is brief.

Doit.bin does most of the work of building and installing the programs. It should be run as the user who owns the distribution directories and will own the executable programs under NEWSBIN.

Doit.news does some other small chores and installs control files. If any of the control files already exist, it will complain and refuse to overwrite them, as a safety precaution. It should be run as the owner of the news files. Since many of the files it is installing are built by **doit.bin**, that should be run first.

Finally, **again.root** tends to ownership and permission changes on a few programs that need to run set-userid. It requires the ability to change ownerships and to set permissions on the files afterwards, which usually means running it as *root*. It too is brief.

There are undoubtedly strange systems out there that **build** and friends are not smart enough to cope with. In such cases it will be necessary to edit the shell files before running them, or to use them as guides and do the work by hand. In particular, systems that require strange options in **Makefile**s will need to have those inserted by hand.

If you want to test pieces of C News without installing them, some (not all) of the subsystems have a ‘‘make r’’ feature that runs a regression test. Note: almost all of these require that NEWSCTL/**bin/config**, or its local equivalent, be in place already so that shell files can find out what PATH (etc.) they should be using.

Note that it is easy to build a test news system which is completely independent of other existing news systems on the same machine, or to build one which shares its NEWSBIN with another C news system, or shares input of articles (e.g. running an old B news system and a new C news system off the same stream of input until you are confident that your new C news system is operating to your satisfaction). See *subst*(1) for the mechanism which permits quickly changing all the places that know the location of NEWSCTL/**bin/config.** After that, edit this news system's NEWSCTL/**bin/config** and things should be set up for separate existence.

First News

When you arrange to get a news feed from your neighbor, you should also ask him to send you the current set of articles in the newsgroup **news.announce.newusers**. Several of these are very important reading for people who are new to the net.

Unusual Systems

We believe that C News runs fine on 16-bit machines, but it hasn't been tested very thoroughly on them lately. It will not perform quite as well with the more limited space.

Machines with very old compilers can be a headache. There are some hooks in **h/news.h** for doing without ‘‘void’’ and ‘‘unsigned long’’, two particular problem areas, but they have to be arranged manually; **build** does not know about them.

Some very old systems cannot do *setuid(geteuid())*, which makes it impossible for a daemon to make directories and get the ownership right. We provide a small program, **setnewsids**, to run setuid-root. **Relaynews** knows about it and invokes it if *setuid(geteuid())* fails; it then sets permissions correctly and re-invokes **relaynews**. The code is short enough to be read and understood in full, so that the suspicious system administrator can be sure that this setuid-root program is not going to do something awful.