Introduction to INN

James Brister `<brister@vix.com>`
Paul Vixie `<paul@vix.com>`

Tutorial Overview

- What is INN?
- Hardware requirements
- Fetching building and installing INN.
- Transport servers vs. reader servers.
- Article transfer vs. article replication
- Innxmit, nntplink and innfeed.
- Expire multiple spindles.
- Reports and monitoring.
- Troubleshooting.

What is INN?

- A transport system – no readers included.
- An implementation of the NNTP RFC (977).
- Can be used for UUCP sites, but targeted at IP-connected hosts.
- Runs on almost all modern UNIX™ systems.

Hardware: RAM

- The Golden Rule: *More is better*.
- Get as much RAM as you can so swapping is limited. A running `innd` of 60MB is not unreasonable for a busy system (lots of incoming feeds).

Hardware: Swap Space

- Golden Rule (revisited): *More is better*.
- On many versions of UNIX™, when `innd` has to `fork` there must be at least as much free swap as `innd`'s current size (even when using `vfork`).
- `innd` `forks` off each version of `nnrpd`, so on a reader machine this is very important.

Hardware: Disk Space/Speed

- The Golden Rule (revisited): *More/Faster is vital*.
- A full news feed in mid 1996 is around 3GB per day.
- Inn's article storage scheme (one file per article) is hard on your disk arm actuators.
- Available disk space and number of spindles is most important.
- Low seek times are very important
- CPU speed is probably least important for now, but with future capabilities like SPAM™ filtering, it is becoming more and more important.

Getting INN

- Last official release of INN:

```
ftp://ftp.uu.net/networking/...
   news/nntp/inn/inn-1.4sec.tar.Z
```

- Home of the “unoff” versions:

```
http://www.math.psu.edu/...
    barr/INN.html
```

- First beta test version of INN 1.5

```
http://www.vix.com/isc

ftp://ftp.vix.com/pub/inn/...
   inn-1.5b1.tar.gz
```

Configuring INN

- One compile-time config file:

  ```
  config/config.data
  ```

  ...which you must create and edit before building. This file specifies the paths of all the other files.

- ...and many run-time config files:

  ```
  site/expire.ctl
  site/hosts.nntp
  site/inn.conf
  site/newsfeeds
  site/nnrp.access
  site/passwd.nntp
  site/control.ctl
  ```

  ...which you must edit after building.

Configuring: `config/config.data`

- Specifies various platform dependencies and site preferences.
- Editing this correctly is the most time consuming and error prone part of the process.
- Section 9 of the INN FAQ has suggested values for various platforms.

Configuring: `site/expire.ctl`

- Defines the expiry policy for the groups you carry.
- Generally we throw `alt.*` and other junk groups out after 3 days, and save the more useful hierarchies for as long as we have disk space.
- Your users will tell you which groups they want to be able to read even after a three day weekend.
- This is what you experiment with if disk space starts getting low.
- Consider tuning when free space after expire is less than %150 of a day's feed

Configuring: `site/hosts.nntp`

- Specifies which hosts can connect to you to feed you articles.
- You have to specify each host (no wildcards).
- It's sometimes wise to put IP addresses here rather than names for when DNS is broken (or just not running) when `innd` is started.
- Beware that a neighbor's address can change even if its name does not.
- Any hosts not listed in this file will be given to `nnrpd` to process if they connect.

Configuring: `site/hosts.nntp`

```
## hosts.nntp format
##
##      <host>:
##      <host>:<password>
##      <host>:<password>:<groups>
##
##  <host> can be a name or IP
##  address; no wildcards.  Any
##  hosts not listed here are
##  handed off to nnrpd.
##
news.foo.com:
news.bar.com:their-password
news.uu.net::!foo.*
```

Configuring: `site/inn.conf`

- Specifies various run-time parameters for INN.

```
organization:    Vixie Enterprises
server:          nnrp.vix.com
domain:          home.vix.com
fromhost:        vix.com
pathhost:        vixie
moderatormailer: %s@uunet.uu.net
```

- `pathhost` is what gets put in the Path header. Use your host's FQDN unless you're registered with the UUCP names project.

Configuring: `site/newsfeeds`

- Specifies what you do with articles you receive (i.e. who you feed them to).

- *Does not* specify which newsgroups you'll take articles for (the active file does that).

- The ME entry's newsgroup list gives the default subscription list for all neighbors, but this defaulting feature is usually not used.

- ME's distribution pattern limits incoming articles. You should probably put `!local` there to prevent incoming leaks.

Details: `site/newsfeeds`

- *site*[/*exclude*,*exclude*...]\
  :*pattern*,*pattern*...[/*distrib*,*distrib*...]\
  :*flag*,*flag*...:*param*
- *site* is the name `innd` will use to refer to this peer. Is usually the same as what that the peer puts into the `Path` header of articles.
- If *site* or any of the *exclude* names are in the `Path` header, of a just-arrived article, then the article will not be offered to this peer.

Examples: `site/newsfeeds`

- *site*[/*exclude*,*exclude*...]\
  :*pattern*,*pattern*...[/*distrib*,*distrib*...]\
  :*flag*,*flag*...:*param*

- *pattern* is a *wildmat*-style pattern to match against group names.

- Any *pattern* list from the `ME` line is prepended to this list.

- To subscribe to all of `comp` but none of the sources except for `comp.sources.unix`:

```
comp.*,!comp.sources.*,\
comp.sources.unix
```

- The *distrib* section can be used to limit what distribution a peer gets (or doesn't get).

Examples: `site/newsfeeds`

- *site*[/*exclude*,*exclude*...]\
  :*pattern*,*pattern*...[/*distrib*,*distrib*...]\
  :*flag*,*flag*...:*param*
- The *flag* entries let you limit when a site gets an article (by size, for example), and describe the mechanism for feeding the peer.
- The two most important flags are `T` (the type of feed) and `W` (what data to give to the feed–such as pathname or size of article).
- The documentation for feeder programs such as `innfeed` will tell you what to use here.
- The *param* field depends on the type of feed. For example it could be the pathname of the file to record article pathnames in for `innxmit`.

Examples: `site/newsfeeds`

- ```
  ME:!junk,!control/\
       !local,!fj,!de,!aus::
  ```
- ```
  feedstats!:*:Tc,Wtbs\
       :/var/news/bin/statChan -t 60
  ```
- ```
  decwrl/news1.digital.com\
       :*,!local.*,!control,!junk/!local:\
        Tf,Wnm,H2:
  ```
- ```
  aladdin:!*,\
       comp.fonts,comp.lang.postscript,\
       comp.sources.postscript,\
       gnu.announce,gnu.utils.bug\
       /!local:Tf,Wfb:
  ```

New Article Processing Sequence

- Look for group(s) in active file. If none there then drop
- Validate against distributions in `ME` entry of `newsfeeds` file.
- Go through `newsfeeds` file and try every entry.
  - Check entry's name against `Path` header components.
  - Check entry's pattern against the `Newsgroup` header
  - Check entry's distribution against the `Distribution` header
  - Check entry's FLAGS (like "H2")

Configuring: `site/nnrp.access`

- Tells `nnrpd`, the reader daemon, who can do what.
- Format:
  `hosts:perms:username:password:patterns`
- Only looked at by nnrpd once it's started by `innd`. `innd` doesn't look at the file.
- If username and password specified then client must send them to Post or Read
- Lets you limit the newsgroups a set of hosts can see.
- Example:

```
*.home.vix.com:Read Post:::*
localhost:Read Post:::*
193.71.66.49:Read:brister:foobar:local.*
*.de:Read Post:::*,!alt.sex.*,!alt.sex
```

Configuring: `site/passwd.nntp`

- Defines the passwd used by various local programs to give to remotes when connecting to send articles.
- Example:

```
news.foo.com:rsalz:martha
```

- Each NNTP neighbor will tell you if you need to add an entry here.

Configuring: `site/passwd.nntp`

- Used by `innxmit`, `innxbatch`, `nntplink`, `nntpget`, `inews`, `rnews` and `nnrpd`. Not yet used by `innfeed`.
- For feeders, the password here must match the password in the `hosts.nntp` on the remote machine.
- For readers, the username and password must match the username and password in the nnrpd.access file on the remote machine.

Configuring: `site/control.ctl`

- Specifies how control messages (other than `cancel`) are handled.
- Format:

```
<message>:<from>:<newsgroups>:<action>
```

- Sample version is usually sufficient

Configuring: `site/control.ctl`

-

```
<message>:<from>:<newsgroups>:<action>
```

- `message` is the type of control message: `newgroup`, `rmgroup`, `checkgroups` etc.
- `from` is a pattern that must match the `From` line.
- `newsgroups` is a pattern that must match the groups being newgroup’d or rmgroup’d (ignored for other messages).
- `action` is what to do: execute it, log it, mail it, ignore it.

Building INN: Documentation

- Build and read (maybe more than once) the Installation document:

```
make Install.ms
nroff -ms Install.ms > Install.txt
```

- Read the FAQ:

```
ftp://ftp.xlink.net/pub/news/docs/
```

Building INN: Software

- Create config.data, edit it and then build with it:

```
cp config/config.dist \
   config/config.data
vi config/config.data

make
```

- Save the `config.data` for the next time you build.

Installing INN

- The various config files in the `site/` directory will be used during the installation procedure. Edit them in the site directory first.

```
vi site/expire.ctl
vi site/hosts.nntp
vi site/inn.conf
vi site/newsfeeds
vi site/nnrp.access
vi site/passwd.nntp

make install
```

- Use `make update` to only install newly built binaries.
- `make update` and `make install` must be run as `root` as `inndstart` is installed setuid root.

Database Initialization

- The `active` file defines what groups you'll accept.
- The format of an `active` file entry is:

  groupname high# low# flags

- For example:

  ```
  alt.flame 0000003598 0000003185 y
  ```

- The most up to date `active` file can be found in:

  ```
  ftp://ftp.uu.net/networking/
      news/config/active.gz
  ```

- You will also need an up to date `newsgroups` file (which describes the newsgroups), from:

  ```
  ftp://ftp.uu.net/networking/
      news/config/newsgroups.gz
  ```

Managing Transport Servers

- These are fairly easy to manage – no users.
- Generally fewer system resources required (swap, RAM, etc).
- Usually a corporate firewall will include one or more transport-only servers.
- Expiration limits are very short since sending stale news onward to other transport servers is almost never the right thing.
- No need to have `innd` create extra links for cross posts.

Managing Reader Servers

- Hard to manage – users are very excitable about netnews.
- Reader machines might also archive news.
- Reader machines are where news/mail gateways are often found.
- 100 `nnrpd` processes can consume all available resources.
- Especially with readers like Netscape™ Navigator™.
- May have to support "sucking" feeders.

About Article Transfer

- Normal way of feeding.
- Possibly multiple remotes send articles via standard protocol.
- Local INN determines article numbers.
- Article IDs $\neq$ article *numbers*.

About Article Replication

- Slave vs. Master.
- Used to make one system a mirror image of another.
- Master feeds articles in such a way that the article *numbers* remain consistent across the two machines.
- News readers can therefore reuse a `.newsrc` file even while switching among several replicated servers.

More About Article Replication

- Articles posted by readers of the slave machine are actually handled by the master.
- Currently uses the private `XREPLIC` extension to NNTP.
- Will be moving to standard `IHAVE` or streaming `TAKETHIS` with `Xref` header parsing.
- Cnews machines can slave off of INN masters.

About `ctlinnd`

- The program that controls the `innd` server.
- Is the program first used to check the state of the server:

  ```
  ctlinnd mode
  ```

- Used to unthrottle the server after it throttled itself due to low disk space (for example):

  ```
  ctlinnd go ''
  ```

- Is used by all external programs that need to control article flow or file maintenance.
- Run

  ```
  ctlinnd -h
  ```

  to see the many possible commands

.

About `innxmit`

- Comes with INN.
- Works on batch file created by `innd`.
- Handles streaming and non-streaming peers.
- Negotiates the streaming extension automatically on each connection.
- Uses `ctlinnd` to have innd flush batch files before processing them.

More About `innxmit`

- Feeds only one remote.
- Good if you want to tightly control when articles get sent (it is run via `cron`).
- There is dead time between the finish of the last run and the start of the next.

About `nntplink`

- `ftp://ftp.math.ohio-state.edu/pub /nntplink/nntplink.tar.gz`
- Runs in several modes: under `innd`, batch file (like `innxmit`), a log file, or a *funnel* file.
- Keeps connection open all the time (modulo idle time).
- `nntplink` has been around a long time and has a lot of users.

About `innfeed`

- `ftp.vix.com:/pub/inn/innfeed`.
- Will be part of INN as of Version 1.5.
- Runs all the time underneath `innd` (and/or will process batch files).
- Handles streaming and non-streaming peers.
- Can result in subsecond article propagation to hundreds of hosts.
- Most efficient with fast peers due to article still being in disk buffer cache.

More About `innfeed`

- Feeds more than one remote simultaneously.
- Can feed each remote over more than one simultaneous connection.
- Keeps connections open for long periods of time (modulo idle time).
- Too easy to hammer a neighbor with parallel connections–limit the number of parallel connections and number of articles sent in streaming mode.
- `innfeed` is only in beta test at the moment.
- Even so, several large and busy news relays are using it.
- Does not (yet) do nntp passwords.

Cross Posting: Hard vs. Symbolic Links

- In a cross posted article the extra article references are links to the original article.
- For newsgroups located on the same filesystem, the links are *hard*.
- On different filesystems, the links are *symbolic*.
- This means that a system that doesn't support symbolic links can only have one filesystem for article storage.

About Using Multiple Spool Disks

- 6 files are involved when receiving any new article:
  - Article file
  - History files (2 – text and db file)
  - Active file
  - Overview file (summary of headers)
  - Log file

- If you can afford enough disks, then put history files, overview files and articles files on their own spindles.

More About Using Multiple Spool Disks

- Mount or symlink another filesystem at the appropriate point, e.g. `/var/spool/news/alt`.
- In `news.daily`, change:

```
EXPIREFLAGS="-v1"
```

...to:

```
EXPIREFLAGS="-v1 -l"
```

- If `/var/spool/news/alt` is a symlink, then in `expirerm`, change:

```
RMPROC="fastrm -e -u -s $SPOOL"
```

...to:

```
RMPROC="fastrm -e -s $SPOOL"
```

More About Using Multiple Spool Disks

- Edit `innwatch.ctl` so that it checks all the spool disks, not just “**.**”. See the lines containing `No space (spool).`
- Edit `innstat` (the line with the “df”) so that all spool disks are included.
- The need for these modifications will be removed in a future release of INN.

About Expiration

- The `-l` (*ell*) option tells expire to assume that cross-postings are implemented with symbolic links.
- The original article is not removed until all the cross postings are removed.
- The file system's hard links are much more efficient for this kind of thing, but they can only span a single file system.

Monitoring INN

- Use `innwatch`.
- Look at your system log files.

  `news.notice`, `news.err`,

  `news.crit`, `news`
- Read the mail from `news.daily`.
- `tail -f` is your friend.

More About Monitoring INN

- To verify that news is coming in, tail the `news` log file.
- To check which groups new articles are going into, tail the `history` file.
- To check:

  incoming connections (reader or feeder)
  feeder statistics
  reader statistics

  ...tail the log file `news.notice`.
- Check for very high reader usage (for sucking feeds) in the `news.daily` report.

The `news.daily` Report

- Shows current `innd` status.
- Shows disk usage.
- Shows batch and log file sizes.
- Shows current connections to `innd`.
- Shows expiration statistics.
- Summarizes bad article information (who and what).
- Shows incoming reader and feeder statistics.
- Gets all of the above details from `news.notice` log file
- History file processing by expire should take less than 10 minutes (or so). Removing the expired articles is the time consuming part.

Example `news.daily` Report

- ```
  Server status:
  Server running
  Allowing remote connections
  Parameters c 14 i 0 (4) l 1000000...
        o 51 t 300 normal specified
  Not reserved
  Readers separate enabled

  Disk usage:
  /dev/sd0e  566254  404466 133474 75% /var
  /dev/sd0h 1007730  784964 172378 82% /gw0
  /dev/sd1h 1916239 1700757 119670 93% /gw1
  ```

- `expire` reserves the server. To manually unreserve the server if expire fails:

  ```
  ctlinnd reserve ''
  ```
  ```

Monitoring: `innwatch`

- `innwatch` will monitor disk space, load average and anything else you put in its control file.
- It will perform any action on `innd` you choose based on rules in the control file.
- To use `innwatch`, change the line in `rc.news` to read

```
DOINNWATCH=true
```

Troubleshooting Miscellany

- run `inncheck` (especially after initial install).
- Reread the Install document.
- *Read the FAQ!*

  `ftp://ftp.xlink.net/pub/news/docs/`
- Telnet to port 119 and issue NNTP commands by hand.
- If remote is paused, then it will hang when fed an article with `IHAVE` or `TAKETHIS`.
- Make sure connection is to correct program (innd vs. nnrpd). Check the banner message.

NNTP Protocol – Transport Verbs

- `IHAVE` *message-id*
- `CHECK` *message-id*
- `TAKETHIS` *message-id*
- `XBATCH` *byte-count*

NNTP Protocol – Reader Verbs (1)

- `NEWGROUPS` YYMMDD HHMMSS
- `GROUP` *group*
- `LIST OVERVIEW.FMT`
- `XOVER` *nnn* | *nnn-* | *nnn-xxx*

NNTP Protocol – Reader Verbs (2)

- `MODE READER`
- `LIST`
- `LIST ACTIVE` [*pattern*]
- `LIST DISTRIBUTIONS`
- `LIST NEWSGROUPS`

NNTP Protocol – Reader Verbs (3)

- `NEWNEWS` *pattern* YYMMDD HHMMSS
- `ARTICLE` *message-id* | *nnn*
- `HEAD` *message-id* | *nnn*
- `BODY` *message-id* | *nnn*
- `POST`

NNTP Protocol – Obscurities (1)

- `STAT` *nnn* | *message-id*
- `LAST`
- `NEXT`
- `QUIT`

NNTP Protocol – Obscurities (2)

- `SLAVE`
- `AUTHINFO USER` *username*
- `AUTHINFO PASS` *password*
- `AUTHINFO GENERIC` *authenticator-arguments*

NNTP Protocol – Obscurities (3)

- `XPATH` *message-id*
- `XREPLIC` *group/nn,group/nn*
- `XHDR` *header* [*range* | *message-id*]
- `DATE`

NNTP Protocol – Obscurities (4)

- `LIST ACTIVE.TIMES`
- `LIST DISTRIB.PATS`
- `LIST SCHEMA`
- `LIST OVERVIEW.FMT`
- `LIST SUBSCRIPTIONS`

NNTP Protocol – Obscurities (5)

- `LISTGROUP` [*group*]
- `XGTITLE` [*pattern*]
- `XPAT` *header range* | *message-id pattern* [...]