WARNING

BETA DOCUMENTATION.
PLEASE DESTROY AFTER APRIL 1, 1996.

Garbage In, Garbage Out
Internet/Fidonet Gateway for the DOS and OS/2 Platforms

Jason Fesler

WARNING

BETA DOCUMENTATION.
PLEASE DESTROY AFTER APRIL 1, 1996.

A note from the author..
These docs are only a rough draft. No spell checking or grammar checking have yet been done. Please, anything that seems missing, let me know. If you can contribute, I would REALLY appreciate the help! If you want to replace anything that I have written so far, feel free to help also :-)

CAVEAT EMPTOR!

NO WARRANTEE

THE AUTHOR PROVIDE ABSOLUTELY NO WARRANTY. EXCEPT WHEN OTHERWISE STATED IN WRITING, THE AUTHOR AND/OR OTHER PARTIES PROVIDE GIGO "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF GIGO, AND THE ACCURACY OF ITS ASSOCIATED DOCUMENTATION, IS WITH YOU. SHOULD GIGO OR ITS ASSOCIATED DOCUMENTATION PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IN NO EVENT WILL THE AUTHOR BE RESPONSIBLE IN ANY WAY FOR THE BEHAVIOR OF MODIFIED VERSIONS OF GIGO. IN NO EVENT WILL THE AUTHOR AND/OR ANY OTHER PARTY WHO MAY MODIFY AND REDISTRIBUTE GIGO AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY LOST PROFITS, LOST MONIES, OR OTHER SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS) GIGO, EVEN IF THE AUTHOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY.

Contents

Introduction

GIGO is an internet <-> fidonet technology gateway program. GIGO does NOT read your message bases, nor will it touch them. GIGO only understands .PKT files that your echomail and netmail processors create. It will convert these packets into format ready for sending and processing by your UUCP host. For Fidonet bound mail, GIGO will take the mail bundles received from your host system and process them into .PKT files for your mail tosser to distribute.

For those of you running OS/2, with a dedicated connection to the internet, an optional add-on for GIGO allows for using NNTP and SMTP instead of using UUCP. If you are not able to run your site full-time, however, I would strongly urge sticking with UUCP.

Things that GIGO will do:

- Convert between internet email and fidonet netmail
- Convert between internet newsgroups and fidonet echomail
- Catch internet mailing lists to convert them into echoes
- Handle email addresses for you, and the fido sites that you wish to feed (using sub-domains, bangpaths, or percent hacks)
- Catch specified email or newsgroups and save as raw ASCII files to a directory of your choice (i.e., the *.binaries.* groups)
- Host your own mailing lists
- Automate having users subscribe and unsubscribe from mailing lists you host
- Run user-defined external function requests
- Perform nodelist lookup, to verify deliveries.

Things that you can NOT do:

- GIGO does not hub. It counts on one UUCP or TCP/IP host. For fidonet distribution, it counts on your fidonet software to do the hubbing.
- GIGO does not touch message bases. It counts on your fidonet software to interface directly to the message bases.
- GIGO does not handle multiple FTN domains well; GIGO is only 4D aware, not 5D. It also does not do what is known as “AKA matching” on the fidonet side.
- The German “MSGID.DOC” standard. It’s something I’d like to add, but lack the time to do it. The politics don’t help, either. (IF YOU _NEED_ MSGID.DOC CONFORMANCE.. USE A DIFFERENT PACKAGE.)
- Expect legible readable documentation, unless I can get some help on them (grin). I’m terrible at writing documentation.

Requirements

- an IBM compatible computer (386 or higher)
- MS-DOS or OS/2 compatible operating system
- DOS: 2 megs _available_ DPMI or EMS memory. (DV users _must_ give that window 2048k EMS).
- OS/2 Memory. :-)
- Lots of disk space; GIGO itself, 2-3 megs. The biggest factor, is how much mail you will be moving through your gateway.
- One database option of GIGO, can easily generate a 20 meg database. If you want to use this option, be sure to account for that database!

Others:

- Waffle's uucico or Fxuucico communications program to send/receive your mail bundles from your host
- Alternately, under OS/2 and a dedicated TCP/IP connection, the optional add-on for GIGO (the GIGOT package) which includes SMTP and NNTP send & receive abilities, for "live connections".
- any mail processor that can handle .PKT files for exporting and importing from/to your message base. (i.e., Gecho, Squish, FastEcho, Fmail, etc.).

Contacting the Author

The author may be reached at any of the following addresses:

Fidonet: Jason Fesler, at 1:203/7707
Internet: jfesler@gigo.com *(Preferred)*
Surface mail:
Jason Fesler
2405 Carta Court #4
Sacramento CA 95825 USA

At this time, voice calls are not an option unless otherwise prearranged via email, as I am working two jobs. I'm simply _not home_ to take calls :-).

GIGO Registration Information

GIGO is not free - a lot of time and money went in making this product. Writing GIGO directly costs me us$100-200 in phone bills each month, as well as many hours of my time. GIGO has several hundred K worth of source code lines going into this project now. Some of that source code is even included for further enhancements and custom modifications by you.

DO NOT register GIGO until you are happy that it is running. The worst it will do is remind you to register every eight thousand messages that you gate. If you are just starting to set it up, chances are you are not going to move that much mail; those reminders won't be that often.

If, however, you do have a stable gateway set up with GIGO, please, register it! I'm only asking for us$20. And, I'm even willing to barter, send me a message if you can offer me alternative goods or services. I'm mostly interested in hardware; alternately, os/2 and unix based items that either make my hobbies (os/2) or my jobs (unix) easier.

There are several levels of registration; you decide which one you qualify under. I only expect people to use the Commercial Registration in environments where the company deems it as neccessary; if you are running the gateway for personal reasons, at a commercial site, you may choose the registration level deemed appropriate to you.

Normal Registration: us$20

This level of registration will give you email support, as well as netmail. More than 2-3 replies via crash netmail will result in replies being left on hold instead of crashed, as Imy phone bills are too high to carry on extended conversations with people. If you're willing to pay for the long distance (voice or data), I'll give happy to give you support. My voice number is available via directory assistance and on request.

Extended Registration: us$20+ [negotiable]

Same as above, except that I'm willing to call you or carry on extended crashmail conversations, etc. The [negotiable] part is to cover my phone costs. Ie, if the [netotiable] part is $20, almost 3 hours of phone time will be paid for. It's easier if you just pay me the us$20, and call me directly, or let me call you collect. This method is just an option for me to crashmail back answers to you sooner instead of you having to poll me, etc. In reality, this option is the same as Normal Registration, except it gives (me/us) the flexibility to return mail immediately, or call voice, etc, without having to worry about running up my phone bill. This method of registration is fairly popular.

Commercial Registration: us$99

For this amount, I'll mail you:

- Certificate of registration (immediately)
- Serialized floppy disk (see below!)

- Printed manual and sample configs (see below!)
- Telephone support
- On-site setup and configuration options negotiable
- Updates are free via email to site administrator

Please inquire via email to see if packaging etc are ready; at this time, it is not available. Anyone who does register at this time, will be shipped materials when they are made availabe; the key, however, will be shipped immediately via crashmail, email, or whatever method is desired. (Sorry, no smoke signals!)

Feature Requests

If the feature is only going to be useful for your site, it will depend on the size of the hack needed to perform your request. Call me or email me to discuss details. Generally, I'm cheap. =O

If the feature is deemed as useful to everyone, and I have time to put it on my agenda, the feature will be free. I have so far been willing to incorporate most of the ideas passed my way. Only one person has had to bribe me to put in key features for his site so far. ;-)

By the same token, depending on the nature of the request, I may not be able to help out at all. Some requests that I have gotten, are simply _way_ beyond the scope of anything simple.

The future..

My plans for future versions of GIGO include:

A 1.00 release that I can send out to all of the commercial sites :-)

Better documentation

Better MIME handling for FILESERV and file attaches

Help someone else do SMTP and NNTP for Windows (I myself, will not be doing that development.. but I'm willing to assist whoever wants to do the job..)

MSGID.DOC standard integration (this will take a large amount of time and effort, as well as trying to figure out a way to prevent many of the problems that MSGID.DOC are creating.)

Possibly: a unix version, should any decent fidonet mailers become available on the unix platform that do EMSI, etc (I somehow think, that I would have to write a bloody mail tosser before that day happens, though..)

GIGO Registration Form

If registering by credit card, please only send this information via a secure chanel (PGP, or direct crash fidonet netmail). Routed netmail and email are insecure! Use 1:203/7707 or jfesler@gigo.com when sending your registation information.

Card holder's name:
Card holder's voice number:
Card number:
Card expiration date:
Authorization for Record Place BBS to charge _____ for the purpose of registration the GIGO internet gateway software package.

us$20 [ ] Shareware registration; key via email; netmail queries are left on hold unless an email address is given.

us$20+___[ ] Shareware+ registration; extra amount is up to you. This amount is used to be able to send crashmail netmail messages back to you, in the interest of not requiring you to poll 24 hours later when netmail is left on hold at my site.

us$99[ ] Commercial; note that materials and such are not yet ready, and will not be ready until version 1.0 (and mainly, REAL DOCS!) are completed. At that time, the full package will be sent out to everyone who has registered at this level. Registration certificates on the official color GIGO letterhead are sent out immediately, however. Email me for any queries or concerns.

To register, send your check or money order (in US funds) to:

Jason Fesler
2405 Carta Court #4
Sacramento CA 95825 USA

Bank checks ****MUST**** be drawn on US funds, on a US bank. Period. It costs me ~$25 to collect on a check that is drawn from outside the country, and takes a few weeks. If you can not get a check drawn from a US bank, I encourage you to use the credit card method instead; the added advantage being that they will take care of the currency exchange for you.

Average turn-around will be 12-96 hours, depending on timing. Credit cards are not processed over weekends, holidays, etc, as this is a business that will be processing the credit transactions on my behalf.

IMPORTANT!

You can send me your credit card one of the following ways ONLY. Please do not use alternate methods, as it is not worth the risk of someone getting your card number!

- Direct/Crash Fidonet Netmail (1:203/7707, 1-916-483-8486)
- PGP encryped email (be sure to include your public key!) Do not use PGP'd routed netmail; that's against fido policy. To get my key, email or finger "pgp@gigo.com".
- Voice communication via telephone (number avail on request)

These tips are for _your_ protection.. You don't want other people to have your credit card number!

Some questions to ask yourself..

Or, so you want to set up an Internet/Fidonet gateway..

Who are you? Are you going to have an internet address based on your host's (or provider's) name? Or, are you going to have your own domain (or internet address, in a way) created for your gateway? You will need to ask your provider on this; different providers have different policies.

Are you going to be feeding other BBS's in your area, either individually or publicly? If so, you will need to make sure that your provider knows this. If you don't, then there is a strong likelihood that mail going to those BBSs will be bounced in the internet world, since those BBSs will not be properly represented. If you plan on feeding a LOT of BBS's, it might be easier to ask your host to send _everything_ based on your name to you, instead of just users at your own system. I.e., if you were foobar.com, **ask your host to both foobar.com as well as *.foobar.com** to your system, and let your gateway figure out who to forward mail for and what to return back to the sender as undeliverable.

Another issue on whether you will be a public gateway or not is the FidoNet address that you have assigned to it. If it is a private gateway, you will likely choose to give it either a fake node address, or an address that will be a "point" off of your own BBS. If you are going to be a public gateway, it is probably better that you try to get a real FidoNet style address for your gateway assigned to you by your net coordinator. Many net coordinators are willing to do this, for a gateway that is set up for public usage.

Support Site, newsgroups and the mailing list

Assistance is available via a variety of means, some of which are already available to you:

- The UFGATE echo; an echo where gateway people all hang out. Lots of people using various different gateway packages are all there.
- The GIGO echo; it’s available by private arrangement from the author.
- The GIGO mailing list - send email to listserv@gigo.com, message body “Help” and “index” on seperate lines. More information will be sent to you automaticly.
- World Wide Web: visit http://www.gigo.com (T1 speed)
- FTP: visit by ftp, at ftp.gigo.com (165.90.138.208 as of 1/96) (Also, on the T1)
- BBS - either by standard modem to +1-916-483-8486, or via internet telnet or vmodem, at gigo.com. Note that my home system is only connected via a 28k modem to the internet; you may experience lag if telneting in. However, it _is_ cheaper than paying long distance.
- Unfortunately, the newsgroup alt.bbs.gigo-gateway is no longer used, due to usenet spammers poluting the alt.* hiearchies. Please, stay with us! but via something other than the newsgroup.

How to get mail from your mail processor, to GIGO:

If you are still reading, that means I have not scared you off yet (grin). As mentioned, this produce _does require_ that you have a fairly decent working knowledge of how your fidonet mail system works. There is no way to cover all fidonet systems available; there are simply too many of them out there. A brief overview of how to work with FrontDoor style systems will be given, as well as the more preferred Binkley style system.

GIGO itself, will _only_ read and write *.PKT files. These are the most generic form of the mail packet format used by Fidonet worldwide. Nobody can communicate without handling this format. You may not see it though - often, these packets are compressed before transferring from one node to another. Since GIGO works _only_ with these files, it does not directly support any one type of mailer system.

Some mailers are easier to interface than others into GIGO. Choice of mail processing can also affect how easy or hard things will be for you.

Front Door, Intermail, MainDoor:

FrontDoor is an excellent mailer written by Joaquim Homrighausen (JoHo, for short). I own a copy (indeed, JoHo and I swapped a copy of GIGO for FrontDoor, we each like each other's work that much :-).

Intermail is a very similar product by Peter Stewart (both are based off of identical code; long story; short of it is, that JoHo and Peter were once partners). Lastly, "MainDoor" is a clone mailer written in Spain, with both DOS and OS/2 versions.

For the purposes of this document, I will only be referring to FD (FrontDoor). All three packages are mostly identical to each other in terms of how to interface to them.

FrontDoor's method of knowing who to call next, and what to deliver, is made by scanning a *.MSG netmail directory. For most people, this is "C:\FD\NETMAIL". It's highly suitable for simpler sites, as it's easy for the end user to tell whether or not their message has gone out. However, the rub is, that it's actually _harder_ to use when interfacing to other programs. (Did I mention, that you need a complete knowledge of your fido system? You do, especially if you're running FrontDoor and it's look-alikes!)

Since GIGO does not read *.MSG netmail messages generated by a FD-type ("arcmail attach") system, you need to get the mail packets to GIGO in one of two fashions.

Method 1: Some echomail processors can pack outgoing netmail up for delivery along with the echomail. They can also generate *.PKT (uncompressed packets) instead of doing the normal compression routines. This method is not common, but if it is available (and does the job right), it will be the fastest way to do things.

Method 2: For everyone else, grab a copy of Gerard van Essen's great

"NETMGR" program. It has the ability to scan a netmail directory, and move any mail destined for a particular system into a new directory, automatically converting *.MSG messages into *.PKT at the same time. (I gave Gerard the code to do this :-). In addition, NETMGR has *lots* of other great neat options. Grab it even if you don't need it! It's worth it's weight in gold.

The specific options to look into NETMGR for, will be the "ACTION MOVEMAIL" command. You will want to scan the netmail directory, moving any netmail and their file attaches to a new directory (that GIGO will look into).

After moving these files, if they are compressed, you will need to decompress them. For most people, the files will always have the same base part of the name; ie, 0010125E.*. You can simply use your unarchiver in a batch file to decompress any files by whatever pattern is needed in YOUR case. Once again, check to see if you can simply get your mail processor to make uncompressed packets - it's both faster and easier this way.

Binkley, Xenia, Portal of Power

This actually covers any mailer that uses a Binkley style outbound. Binkley style systems do not scan *.MSG messages at all when determining where they need to call. They instead require that the mail processor already have converted such messages into *.PKT (or compressed arcmail) form, and placed into the outbound directory with a special naming method. The names of those files, contain the fidonet addresses, in a special encoded format (hexadecimal), which tells the mailer where it needs to call.

The great thing about this method, is that it's highly predictable. And, if you happened to get one of these mailers running, chances are you are halfway familiar with fidonet technology :-) as those mailers do not make it easy to get things going without that knowledge.

Method 1: If you are running Squish, you are in great luck. Add to your ROUTE.CFG file, near the top:

```
send hold noarc 1:2/3   ;Instead of 1:2/3, use whatever address
                        ;your gateway is going to become.
```

You will need to know what directory files are going to be stored in by Squish. Normally, this will be in \outbound unless you are running the gateway as a point. If you are going to run it as a point, you will have one or more *.PNT directories in your outbound; you'll find that mail for your gateway will be stored in one of them. You'll need to figure out which one. If you know hexadecimal numbers at all, you can just convert the net/node to \outbound\xxxxyyyy.pnt\ to figure it out; alternately, you can peek at the files.

When Squish writes it's "hold noarc" files, it will create a file ending in .HUT. GIGO can read .OUT, .CUT, .HUT, .DUT, and .PKT files without/out any problems.

(In reality, they are all the same, with different extensions to tell the mailer whether to send them crash, hold, direct, etc).

METHOD 2: For everything but Squish. :-)

You may wish to get the "netflo" program from ftp.gigo.com. It has provisions for moving all of the mail from a binkley style outbound for a given address, into a new directory. *See "Method 2" of the FrontDoor instructions* for information on decompressing the files that get moved afterwards.

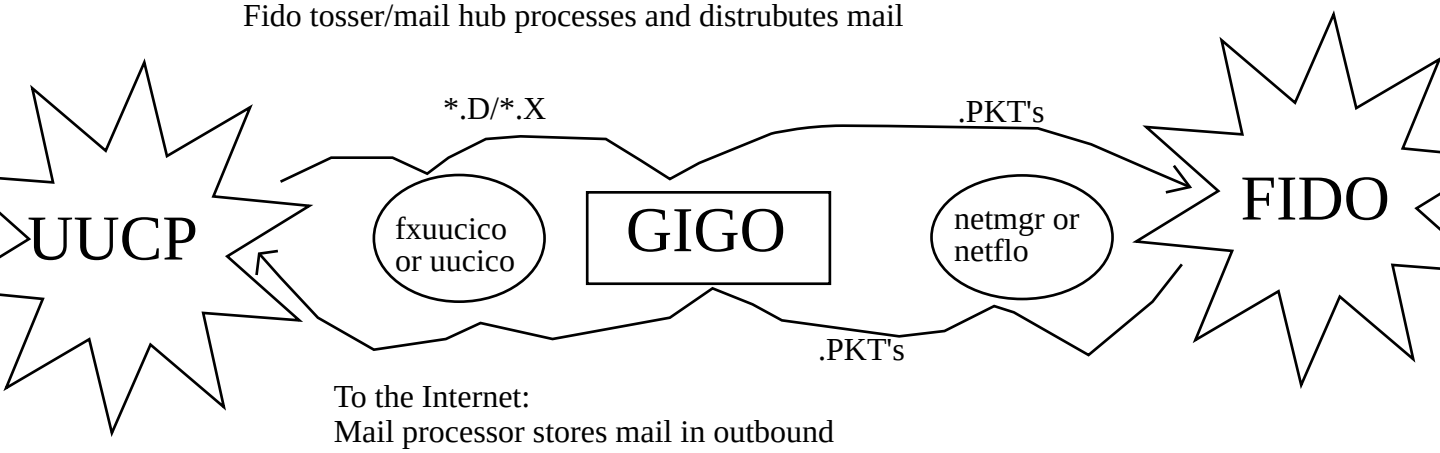
BOTTOM LINE:

No matter what mailer or echomail processor you are using, you will need to give GIGO raw *.PKT files. It doesn't matter what name they are, so long as they are decompressed. GIGO will ignore any packet that have the wrong fidonet address in the headers; it's safe to point GIGO at any directory without fear that the wrong files will be picked up. (Unless, of course, GIGO is looking for the wrong address because of misconfiguration :-).

GIGO will NOT read compressed files; it will not read *.MSG files; it will not directly read any message base format. It's merely a translator, and not a message base agent at all.

If you have problems, consider joining the GIGO mailing list and asking for assistance. Be sure to indicate what mailer and mail processor you are trying to work with; if anyone has similiar experience, they will be able to help you.

To Fidonet:
UUCP feed gives mail and news to fxuucico when you poll
GIGO translates *.D/*.X files from fxuucico to fidonet .PKT's
Fido tosser/mail hub processes and distrubutes mail



To the Internet:
Mail processor stores mail in outbound
NETMGR (for FrontDoor,etc) or NETFLO (for Binkley, etc) move mail
from outbound to GIGO's inbound directory
Make sure it's uncompressed!
GIGO translates *.PKT files to *.DAT,*.XQT, and *.CMD files for fxuucico
fxuucico polls UUCP site, and delivered outgoing files

RUNNING GIGO

Running GIGO is as simple as:

```
GATEWAY.EXE  [configfile]  [options]
```

If no config file is specified, then "gateway.cfg" will be used. You may also specify multiple config files; GIGO will run once for every config file specified (without having the overhead of reloading once for each config you may have).

Options are specified below.

DOS USERS:

GIGO requires the use of a 32 bit DOS extender. To run GIGO, you will need to ensure that GIGO has access to 2 megs of EMS or DPMI memory, and that GIGO is running on at least a 386. Typing "mem" at the DOS prompt should show that you have at least 2000000 bytes of EMS free. If you do not, GIGO may fail to load.

Under DesqView, special attention must be made. The settings for the window you plan on running GIGO in, requires custom settings. Under "Advanced Settings", you will need to fill in the box labeled "EMS", with 2048. Otherwise, GIGO will not be allowed to have the memory it needs.

For your information: GIGO is using the CauseWay DOS extender for Watcom C/C++ and Asm. It seems to be a very stable (and fast, and small, and lower memory requirements, and on-and-on) than the popular DOS4GW DOS extender. Older versions of GIGO used DOS4GW, and a few people noted compatibility problems under some network situations. If you were one of these people, you might try things again with any 1996 release of GIGO, and let me know the results.

OS/2 USERS:

GIGO runs in a VIO session (ie, a text window or full screen). Some people have reported problems with running GIGO under Warp Connect, when running full screen; if you experience this problem, try running GIGO in a window instead. One other suggestion, for full-screen, is to use "start gateway.exe", except that it will start a new copy running in the background. There are no special memory concerns (other than the normal OS/2 memory concerns) when running GIGO.

A few people have also mentioned that GIGO eats up most of their computer's CPU when running. It does. It's busy (grin). A freeware program called "sp" (latest version I saw, was in sp103.zip on ftp://hobbes.nmsu.edu), allows for running programs with different levels of priority. I run my entire mail processing (including the gateway) as "i0", which says only let the mail processor run when the rest of the computer has nothing better to do. The mail processing has only a slight slowdown, yet the computer is still highly responsive - even if there is a high CPU load. Any utility that provides this functionality will do; sp103.zip is merely the one that I personally use.

Command Line Options

GIGO only supports a few command line options. Most options, are instead handling in the config files directly. Only options that affect GIGO *before* GIGO has had a chance to load it's config files, are specified on the command line.

Those options are:

/MAIL	Only process "rmail" bundles in your spool directory
/NEWS	Only process "rnews" and all BAG files (both email and news)
/FIDO	Only process .PKT files
/NOLOGO	Do not show opening GIGO logo. Registering GIGO, will make the closing GIGO logo disappear as well. Ctrl-Break will abort the closing logo, should you be running GIGO manually.
/NOVIDEO	Specifies that no video writes are to be performed. Video output is produced, but only while loading the configs and shutting down. During actual processing, the video is turned off in the interest of speed. (This can be specified in the config file as NOVIDEO).
/NOLOCK	Do not use a lock file to make sure that no other copies of GIGO are running (I do NOT suggest this!) If you are running in an environment where GIGO does not run, but this option fixes things, ***please*** let me know the details.

If you are running the DOS version..

/BIOS	Use BIOS video writes (slow!)
/COLOR	Use color, even if mono is detected
/MONO	Use mono, even if color is detected

CONFIGURING GIGO

The configuring of GIGO is done through the use of plain ASCII text files. Some argue that a menu driven setup is the best way to go; others, prefer ASCII, as making large changes or converting between mail systems is faster and easier. Alas, there is not much of a middle ground. Since I prefer to work on GIGO's functionality the most, I chose to use the plain ASCII text file method. It's major advantages:

- Fast to manipulate (using your favorite text editor, and all it's capabilities)
- Making bulk changes are easier using search-and-replace with your editor
- Config files are both forward and backward compatible - no binary structures that would need conversion utilies to manipulate between versions
- Information about every configurable item in GIGO, is in the config file itself. You do not have to refer back to these documents in most cases, since the config file itself is documented. These config files have served as documentation for the first 2 years of GIGO's life.

Configuring GIGO is not an instant thing - you can't just hit a few keys, move the cursor, toggle a few buttons, and go. There is too much involved in a gateway system, for it to be that simple. Please, take the time to read through each of the config files. Take them one by one.

The configs are broken down to major levels of functionality. Should it be easier for you , you can break them down even more (and use the "INCLUDE" keyword to include the extra parts).

Don't let the size of those configuration files daunt you; they are only that big, because of the amount of documentation that is built into them. Stripping out all of the comments, would leave almost nothing left over.

CONFIG FILES

Included in the archive are configurations which you can start modifying to suit your purpose. The succeeding sections will through each config files and go over the keywords used in them. This is provided to give you a more detailed explanation of each function. The sample config files have enough comments on it to get you by without having to wade through this.

GATEWAY.CFG

This is the main control file for GIGO. For all the configuration files, any line beginning with a semi-colon are comments and will not be processed by GIGO. Lines starting with "DOS" or "OS2" will only be used when running the appropriate version of GIGO. Ie,

```
DOS INCLUDE DOS.CFG ; only includes DOS.CFG if running under DOS
DOS SWAP   ;  Only enable swap, if running under DOS
OS2 INCLUDE OS2.CFG ; only includes OS2.CFG if running under OS/2
```

SERIALNUMBER xxx
REGISTRATION xxxxxxxxxxxxx

Once you have registered GIGO, fill in the SERIALNUMBER and REGISTRATION fields. These values are keyed to your MYSITE entry, so ensure you report your site correctly when you register.

TZ xxxx

TZ = Your timezone. In most cases it will be EST5, CST6, MST7, PST8. If you remove or leave out this field, the default is EST5 The numeric portion should be the number of hours away from UTC you are; positive numbers are west of the Greenwich or prime meridian, and negative numbers are to the right of the meridian.

TEST

Do not delete files when done. This is useful for checking the results when you run gigo after making changes in your configuration files. Very good if you can't bug your host to send you test msgs everytime you make changes. (This applies only to fido *.PKT and to UUCP bundles. The code requires that .BAG files do not honor TEST mode.. sorry!)

MAX_FIDONET 64000
MAX_USENET 1000000

Define the maximum buffer sizes to allocate for usenet and fidonet style messages; these can be made larger than default. For people low on memory, they can be made somewhat smaller.

LOG gigo.log

Where to place the log file. There is no default. Comment or remove this line to skip making a log file. Drives and paths may be specified else they default to the directory where your run GIGO in.

LOGKEPT gigokeep.log

GIGO is able to keep track of how each piece of news is processed. LOGKEPT will log all newsgroups that were kept and what echo names they were given.

LOGDUMPED gigodump.log

LOGDUMPED logs which newsgroup areas you received but were dumped, i.e., because the name of the area could not be found in your MAPPING.CFG file, they were not added to your BBS.

LOGBOUNCED gigobncd.log

Logs of message headers that were returned to sender. This also includes messages processed by GIGO for users coming from the same domain.

LOGFILESERV gigoftp.log

Logs of FILESERV requests processed by GIGO. See FILESERV commands below.

LOGTRAFFIC gigotraf.log

Log of all inbound and outbound traffic stats. This is a straight ascii file and you will need a separate utility to make more sense out of it.

The format of the log file is:

<type> <size> <from/to> <to/from> <address> <who>

TYPEs may be any of the following:

ei	incoming e-mail
eo	outgoing e-mail
ni	incoming newsgroups/mailing list to echomail
no	outgoing newsgroups
li	incoming mailing list messages to echomail
xo	mailing list “exploded” out to list subscribers

SIZE is in the format of xxxx/x where xxxx is the size message and /x the number of messages it was split (to/from?)

FROM is the sender’s address or where the message originated

TO is where the message is addressed as

ADDRESS is the converted FidoNet address

WHO is the converted full name of the sender or receiver

If you do not need all these logs, turn them off. These log files all cause GIGO to run slower. gigo.com runs without logs at all :-).

The following INCLUDE statements will read other configuration files. Putting them into separate files allows grouping the different process.

INCLUDE headers.cfg

Controls what headers are kept and allowed in the FTSC messages. See HEADERS.CFG below.

INCLUDE mapping.cfg

Controls which newsgroups and/or mailing lists are forwarded to your Fidonet node for converting to echomail and back. See MAPPING.CFG below.

INCLUDE users.cfg

Defines how users and sites are resolved as mail is processed by GIGO. See USERS.CFG below

INCLUDE maillist.cfg

Used for running your own mailing list. If you will not be hosting a mailing list, you don't need this. See MAILLIST.CFG below.

MESSAGES + SIGNATURE FILES

GIGO has a variety of messages that it can give it's users, usually associated with user errors or permission errors. These messages are all sysop-definable; you can rewrite it to other languages, or perhaps change the wording to better suit your users, or mention certain pecularities about your specific system.

MESSAGES GATEWAY.MSG

The file that GIGO's messages and the signatures are stored. It defaults to GATEWAY.MSG unless you specify otherwise here.

Note that this file MUST be here for GIGO to run properly; if the file is missing, or any of the messages are missing, the messages sent to the users will be practically meaningless. Signatures aren't required, but the messages themselves are.

SIGNATURES

GIGO can automaticly add a signature at the bottom of messages, on the behalf of the BBS or the user. You can define these signatures yourself; you can define their behavior with wildcards, etc. The main drawback, is that this file has to be searched once for every outgoing message that you send out. signatures are only attemped _IF_ this keyword is used. The format for the messages and for the signatures is in the sample GATEWAY.MSG file provided in the archive.

GATEWAY ADDRESSING

GATEWAY <address> <path> <extension>

The Gateway address is the address for GIGO to use when sending and receiving mail from & to Fidonet. This may be a real address, listed in the Nodelist, a fake address, or a point address.

Under _NO_circumstances_ should the gateway's address be the same as your address. If you can not get a nodelist entry for it, then set it up as either a fake node, or as a point off of your system. If you do not follow this precaution, you will find that the gateway will be attempting to process and bounce mail that was meant for you, instead of the gateway.

Path is where your mail packer puts its outbound files in. Squish normally defaults to C:\SQUISH\OUTBOUND\

Extension refers to the file extension for GIGO to search through in the path. GIGO looks for _uncompressed_ FTSC files. This may be: PKT, OUT, HUT, CUT, or ?UT. If you are not sure, use * instead.

Note that GIGO does NOT read *.MSG messages at all - - if you are using FrontDoor, you will need to use "mailscan", "netmgr", or the mail packing features of your mail procsesor.

Valid suggestions:

```
GATEWAY 1:203/2 C:\OUTBOUND\ ?UT
GATEWAY 1:203/2 C:\GECHO\PKTS\ PKT
GATEWAY 1:203/2 C:\INBOUND\ *
```

Note, you may only specify this once.

SENDTO <address> <path>

SENDTO is the address where packets are sent to by GIGO. The address shows what address the .PKT file should be addressed to. This is usually your BBS's main FidoNet address. The directory should be your protected inbound directory.

SEEN-BY <address>

Seen-by adds whatever 2-dimensional address(es) to the seen-by's. Generally, this is going to be the net/node of the gateway. Do NOT (*repeat*) do NOT place zone or point information here, as it breaks some mail processors!

PATH <address>

PATH adds whatever 2-dimensional address(es) to the path line. Generally, this is going to be the net/node of the gateway. Do NOT (*repeat*) do NOT place zone or point information here, as it breaks some mail processors! For many mail processors, this line _may_ be commented out. [This command added for compatability with WildCat's mail processor, which _demands_ path information]

The succeeding lines are your mail addresses and your host. Examples below will be taken from the internet address:

```
user.name@swizzle.sacbbx.com
```

MYSITE <sitename>

This is your site uucp name. It should be a single word name. eg. MYSITE swizzle

DOMAIN <domain>

This is your site's address, minus the site UUCP name above. eg. DOMAIN sacbbx.com

MYFEED <host machine name>

This is your host's UUCP machine name. This is used for spooling. This should be the same as one of the subdirectories in your C:\GIGO\SPOOL\ directory

MYPATH <site.domain>

If you want to override the site name used in the Path: line, you may specify it here. Leave alone unless you need it. It is provided for cases where either the host expects something non-standard, or where GIGO's guess turns out to be wrong.

MYFROM <site.domain>

Only use this to modify your "From " line (ie, From user xxxxdatexxxx remote from site to From myfrom!user xxxxdatexxx remote from site Added for compatability w/centron.com's host. If you don't need it, leave it out.

POSTMASTER_HACK

Use this to force the "From " line to be sitename!postmaster .. (also for centron.com) (Can be used at same time as MYFROM). If you don't need it, leave it out.

CRLF

Uncomment this line if your feed is sending mail or news bundles with carriage returns at the end of the line, instead of just line feeds. Enabling this option _will_ slow down processing some, as it will have to translate cr/lf lines to just lf lines. (Most sites won't need this option.) (It's prefered that they turn OFF the CR's)

SPOOL <path>

Where your Waffle style UUCICO spool dir is. This must be the same as UUCICO's spool! eg. SPOOL C:\GIGO\SPOOL\

BAGS <path>

If you get .BAG files from:

Planet Connect
Pagesat
The GIGO TCP/IP add-ons

you'll want to tell GIGO what directory to look for the bag files in. Note that you will still need to set up the SPOOL, MYSITE, DOMAIN, and MYFEED options. eg. BAGS C:\BAGS\

Packet Generation

ORIGIN=ORGANIZATION Usenet:

In messages from usenet to Fidonet, ORIGIN=ORGANIZATION will place the "Organization" field from the usenet header into the Fidonet origin line. You may change the string following ORIGIN=ORGANIZATION to meet your gateway's needs. Example:

```
Organization: University of the World
```

- will become -

```
* Origin: Usenet: University of the World (1:125/110.99)
```

ORGANIZATION=ORIGIN Fidonet:

Along the same lines, ORGANIZATION=ORIGIN will place the Fidonet origin line into the "Organization" field to messages bound for usenet. Example:

```
* Origin: Swizzle Stick BBS (1:125/110)
```

- will become -

```
Organization: Fidonet: Swizzle Stick BBS
```

ORIGIN Swizzle Stick <-> Internet Gateway ORGANIZATION Swizzle Stick

These two keywords are used instead, if either ORGANIZATION=ORIGIN or ORIGIN=ORGANIZATION are excluded, _or_ simply left unspecified. They will be used as in the respective place holders in the outbound mail.

ORIGINADDRESS 1:2/3.4

ORIGINADDRESS allows you to specify a fidonet address other than the gateway's address, for echomail that is produced by the gateway. Ie, ORIGINADDRESS 1:2/3.4 will make * Origin: xxxxx (1:2/3.4) Most people should leave this commented out.

> I've taken the translated stuff out for the moment.

TRANSLATE

8BIT

The TRANSLATE keyword defines the translation table from fidonet->internet, as 8 bit characters are a naughty thing when posting world-wide. This table defines how GIGO will translate the characters. It should be 128 characters long, and represents the 8 bit characters only. Alternately, you can disable TRANSLATE, and uncomment 8BIT, and GIGO will perform no translation at all. Beware, you will get flamed from all parts of the world if you do this without a real good reason for it.

Note that the TRANSLATE table may not have 2 consecutive spaces, due to the way that GIGO converts these config files to save memory. It is a known bug, that I will be seeking a workaround for in a future version (ie, perhaps an external file for inbound and outbound filtering).

COMPRESSION

Newsgroups are usually sent batched together and compressed. These files usually say "#! cunbatch" at the top, and it tells the receiving end how to process them.

The following two parameters are the command lines needed for the standard unix-world COMPRESS program. This program is being included in the GIGO extras.

COMPRESS COMPRESS.EXE -m

Allow GIGO to handle compression of rnews files. COMPRESS.EXE may be replaced with GZIP.EXE if and only if your UUCP feed allows it on the host system. If commented out, GIGO will prepare mail to be sent uncompressed. This file must be in your GIGO directory or in the path.

DECOMPRESS COMPRESS.EXE -d

Allow GIGO to handle decompression of cunbatched files. You may use GZIP.EXE -d instead to allow decompression of both compressed and gzip'd files (gzip autodetects the format being used). This file must be in your GIGO directory or in the path.

FASTCOMPRESS

FASTCOMPRESS tells GIGO that we are using a specially modified version of the public domain COMPRESS program. This modified version will take care of adding or removing the "#! cunbatch" line on newsgroups for us, saving the gateway a lot of time with messy tempory files.

If you are NOT using GIGO's compress program, which has been tailored for use with the gateway, you will need to run the following instead.

```
COMPRESS compress.exe
DECOMPRESS compress.exe -d
;;;FASTCOMPRESS should not be used.
```

Special Features by Request

The following keywords are made by special requests. If you don't understand these options, don't use it. If you don't need them, leave them alone.

ROUTESPOOL <re-address to> <for>

At the request of garux@mdtn_bbs .. a specialty option to allow outgoing _email_ only to be routed to different spools, based on where it is headed.

Most sites don't need this, and the inclusion of even one of these statements will slow down _all_ outbound email, as it turns on the extra checking needed to figure out where to send things. The routines had to make use of scanning the config memory every pass, and make use of regex routines (which are always slower processing).

Examples:

```
ROUTESPOOL host1  host1
```

Sends all mail to host1 directly to host1 spool

```
ROUTESPOOL frackit frackit.uucp
```

Re-route all mail for frackit.uucp to frackit

```
ROUTESPOOL frackit wfs.uucp
```
Put all user@wfs.uucp mail in frackit's spool
```
ROUTESPOOL host  *.uucp
```
Sends all user@....uucp messages to "host"

BACKWARDSFEED

Another option, that's for kludging value only, is to generate outgoing spool file names with your host's name in it, instead of your name in it. Don't use this option unless you know WHY you need it (it will be used by maybe 1% of the people around..) Anyone using this option when they are not supposed to has about a 99% chance of losing their mail.

BURT_JUDA_HACK

The BURT_JUDA_HACK option is meaningless for most people. This option creates email headers with "From user@site.domain date" instead of "From site!user date+time remote from site". *If this option helps you, great. Don't do it unless you _know_ you need *it though. Burt needed it for his special smtp scripts.

XQT_FROM_UUCP

The XQT_FROM_UUCP option is to force the XQT files to pretend to come from a user named "uucp" instead of from the original user. Only use this if your host is having problems with your email packets; most hosts can handle actual user names in the xqt packet.

ALTMUNGING

If your host is sending you names that GIGO is complaining about (ie, MUNGE_ERROR(....) or not finding any of your files). it could be that you are getting files that are not using the standard naming conventions that most hosts are using to send files to their feeds. Uncomment the ALTMUNGING option, and GIGO will attempt an alternative method to locate the files in your spool directory. Most people will not need this option.

FSC-35

Always do FSC-35. If your BBS or editor supports the FSC-35 specification, the FSC-35 command should be enabled. This causes GIGO to add the ^aREPLY-TO and ^aREPLYADDR kludges to messages. If you are not sure if your BBS supports FSC-35, leave the command in, as it will not cause a problem for most BBS or editor software.

PARTIAL-35

Only use FSC-35 when the fidonet headers are not capable of returning a reply correctly. Disable FSC-35 if you're using this.

OLDTOSSER

If you are running GIGO as a point, and your mail tosser is telling you that GIGO mail bundles are from Net 65535 or Net -1, enable this keyword. (GIGO by default makes FSC-0048 compliant packets, noting that NET values should be set to 65535 when dealing with points. Wierd stuff, I know..)

RETURNADDRESS_EMAIL <optionalname>
RETURNADDRESS_NEWS <optionalname>

This tells GIGO to use the person's email address (name@site) in the From: field of a fidonet message if it is small enough to fit in. If the address is too long, the person's name is used instead. Commenting this verb will only import names in the From portion rather than email addresses.

By default, a person's real name is placed in the fidonet header.

<optionalname> may be something like "UUCP".

SPLIT 15000

Split up large messages SQUISH, QMAIL, TOSSCAN, GECHO, FMAIL, FASTECHO, WILDMAIL, RENEMAIL, TELEMAIL, and 95% of the rest of the tossers out there require a relatively small individual message size to work smoothly. Nearly every tosser since then has taken this example and imposed a relativley small limit on fidonet messages. if you change this from 15000, be sure you know everyone receiving from you will be able to handle the incoming large mail you send. Technote: Splits are made using FSC-0047; "SPLIT" and "UNSPLIT" will work on splitting/unsplitting *.MSG messages with this format. Also, PKTSORT by Rolf Wilms (sp? please correct me!) has the capability of re-merging these messages while still in .PKT format.

MAXPKT 150000

Maxpkt provides a (approx) cuttoff point per .PKT do you REALLY want to toss a 3.5 meg single packet? didn't think so, one small slipup and you might lose a whole day's mail. commenting this verb will generate one continuous mail packet, that may become questionable in the event of an emergency abort. Unlike any other currently available gateway package GIGO splices USENET batches into a seamless stream of fidonet packets, without pausing between each batch. This is necessarry to keep things within a reasonable managability level.

LINEBREAK 80

LINEBREAK - specifies the longest line length allowed to to be sent to the usenet site. Any messages that have lines that are too long will be wrapped. This option is usually needed, as many fidonet editors do not wrap the text before sending it out (it may look wrapped, but it's no in the .pkt). Suggested value: 80. Things *do* run faster with this option turned off.

MAXNEWSGROUPS 20

MAXNEWSGROUPS xx tells GIGO to allow up to xx copies of the same message to be cross-posted. If the message is cross-posted to more newsgroups than the number you specify, GIGO will publish only the first copies. Default=20

TRANSLATE_CONTROL

DELETE_CONTROL

Control messages are regularly sent out in the newsgroups; these messages control automatic wide-area message deletion, area creation, etc. GIGO has the ability to intercept the delete message; when used in combination with the NIKE option, GIGO can completely automate the wide-area message deletion when moderators and users wish to unpost their messages. Alternate, GIGO can strip Control: messages entirely, since for all other practical purposes, they make no sense in a Fidonet environment.

SAFE_THREADS

Message threading! If you can afford the time and drive space, this is a nice option. It can:

- Allow consistant MSGID/REPLY linking (if your tosser supports it)
- Insure that outgoing newsgroups have proper“Newsgroups:” lines
- Insure that if “Followup-To:” is specified, that it’s honored
- Insure that outgoing newsgroups have proper “References:” lines

However, this does come at a price. The database takes approximately **80 characters per message**. For gigo.com, a 20 meg database holds 2 weeks worth of messages. This also slows down processing of outgoing messages (roughly, an extra 1 second per outgoing fido->internet message).

If you use this option, be sure to look at the “CLEANID.EXE” program and it’s documentation. They will help keep your database at a managable size.

KEEP_THREADS ; Also known as “NIKE”

This is the least-prefered way of preserving MSGID/REPLY linking and “References:” lines. It’s not quite as accurate as SAFE_THREADS can be; in addition, some argue that the MSGID’s created by KEEP_THREADS go against the FTS-4 specifications. Chances are, if you know somebody who constantly says “Straw man”, you don’t want to use this option.

BLOCKADDRESS <search pattern>

The mailer daemon can allow you to prevent fidonet users from sending email to various addresses. This is useful, for instance, if you have a measured-rate UUCP service (or long distance charges), and you don’t feel like letting users request files on _your_ call. BLOCKADDRESS allows you to specify a pattern for an address; if the address that the user is emailing fits any of these patterns, the message will be returned to him as undeliverable.

<search patterns> maybe specific works, eg. listserv or containing wildcards ie. *server* will blockout all messages to mail-server, file-server etc.

Note that this will block even YOUR attempts. It doesn't discriminate on who it blocks. Also, be careful of using the wildcards. Blocking out *serv* includes all mail going to compuSERVe accounts as well.

NODELIST <path>

GIGO has the ability to check a nodelist index. A seperate nodelist compiler is included with GIGO, and comes with it's own docs.

```
<path> points to where NODELIST.GIG will be found. If you specify the di-
rectory where NODELIST.GIG can be found, GIGO will look up this nodelist
index any time an email address comes in in the form of
user@f###.n###.z###.site.domain
```

or in the form of

```
user@##-##-##-##.site.domain
```

If GIGO can not verify the existance of that nodelist entry, GIGO will send the message back to the internet user.

This option is here mainly because MsgTrack is not gateway aware, and will bounce messages back to the gateway. The gateway would in turn send a message back to MsgTrack saying that it wrote the message wrong. What a mess! If possible, please, use the NODELIST option in GIGO, to prevent such a mess happening for you..

FILESERV FUNCTIONS

GIGO can act as a server for FILESERV requests. This allows people on the internet to make file requests on your system or for people on FidoNet to send files to the Internet. If you don't want to provide such functions, comment them out.

FILESERV-LIST <file>

This is a list of your file requestable directories, one on each line. You can point this to your bink style okfiles catalog if you wish; magic names will be honored. GIGO will understand both FrontDoor style and Bink Style request lists. You can list magic names and directory names with either method.

Sample: FILESERV-LIST C:\BINKLEY\OKFILE.TXT

In OKFILE.TXT:

The format of this file is as follows:

```
MAGICNAME   C:\path\realname.ext
@MAGICNAME  C:\path\realname.ext
@GIGO C:\pub\ufgate\gigo1129.zip
C:\PUB\DIRECTORY\
C:\PUB\DIRECTORY\*.*
C:\PUB\UFGATE\*.*
```

FILESERV-MAXFILES <number>

The maximum number of files that may be requested.

FILESERV-MAXBYTES 10000000

The maximum binary size per request.

FILESERV-DEFAULT 75000

Default segment size for binaries.

FILESERV-FIDONET

This keyword allows fidonet style people to send messages to fileserv. This is NOT ALLOWED IN FIDONET! IT'S CONSIDERED FILE ROUTING!

FILESERV-ATTACH

This keyword allows file attachments to the internet. File attach statements addressed to the gateway will be processed, uuencoded, and sent to the specified internet address. If you use this, you'll want to make sure that your inbound directory is listed as a requestable directory.

FASTRETURN fidonet.org

FASTRETURN myspecialnetwork.org

The following command allows for "fast return" of mail that is addressed to the internet, but is actually supposed to be processed by your system. In these cases, the gateway will generate _incomming_ bundles instead of outgoing bundles, making the messages never even leave your system. Best use for this is "fidonet.org" addresses, since we are not allowed to send from fidonet, to fidonet, via the internet backbone (as part of our charter to get .fidonet.org).

Note, to force compliance with this charter, "fidonet.org" is a forced issue with GIGO. If you do not want GIGO to FASTRETURN anything for fidonet.org, contact the author. The key will only be granted to disable this function to very special circumstances that do not violate fidonet.org's charter. An example of this exception would be if you are the gateway for a different (and registered) organization, ie rbbsnet.org, where it _is_ desirable to be able to email someone@*.fidonet.org ..

IGNORE_CHARTER xxxx

If you do NOT want fidonet.org put into the FASTRETURN statements automaticly, you must get a special key, simply email me with a good reason why. Mostly, this is for people gating networks other than fidonet. The key itself is free, so long as the circumstances do not violate the fidonet.org charter.

USERS.CFG

The USERS.CFG file defines usernames and sitenames. It controls how addresses are handled.

KEEPUNKNOWNUSERS

Keeps messages to users on the LOCAL system that are not explicity defined.

KEEPUNKNOWNSITES

Keeps messages to other systems that are not yet defined, so that you can manually forward/bounce/etc via netmail.

KEEPUNKNOWN

Same as defining both KEEPUNKNOWNUSERS and KEEPUNKNOWNSITES

MAPSTYLE

Mapping style for default outbound email. Gigo supports practically any style of generating addresses now, as it is completely definable by you, the admin.

If the message is being launched from the gateway itself, then no "othersite" should be evident in the outbound headers. See "SITE local" command below.

The following are variable translations for use with the mapstyle keyword. The variables are taken from the GATEWAY.CFG file.

*user translates to the current user.name
*site translates to the current SITE parameter
*mysite translates to the current MYSITE variable
*domain translates to the current DOMAIN variable
*myfeed translates to the current MYFEED variable
*fnz translates to .f##.n##.z## (ie, *user@*fnz.fidonet.org)
*ftn translates to ##-##-##-## (may have fewer #'s; this variable is an alternative to *fnz, using fewer periods

and (usually) a shorter string. The number of ##'s will vary from 1 to 4 (ie, 203-7707 is enough to say net 203, node 7707, in my zone).

You MUST define **MAPSTYLELOCAL** and **MAPSTYLESITE**; **MAPSTYLEUNDEF** is optional, and defaults to being the same as your MAPSTYLESITE template. [This used to instead default to a .fidonet.org template, but the default .fidonet.org gateway will be going down soon.]

You may use any of the above variables; most importantly, *user and *site. Anything else can be filled in by you with specific text, and may use the above variables if you wish.

MAPSTYLELOCAL *user@*mysite.*domain

Defines how the address of messages from the local site are formatted. This is used with the SITE LOCAL statements below.

MAPSTYLESITE *user@*site.*mysite.*domain

Defines the format of the address of messages coming from nodes other than the local node. This is used with other SITE statements.

MAPSTYLEUNDEF *user@*ftn.*mysite.*domain

Use this format when no SITE statement or bounced site is defined.

USER <internet aka name> <fido address> <real name>

This is the most basic of defining users for your system. Each statement tells GIGO how to convert each individual name it encounters.

Samples used here will assume a site name of _swizzle_ and a domain of _com_. Both defined by GATEWAY.CFG keywords as MYSITE and DOMAIN respectively.

```
USER richard.bollar   1:125/110   Richard Bollar
USER bollar     1:125/110   Richard Bollar
USER root 1:125/110   Richard Bollar
USER postmaster 1:125/110   Richard Bollar
USER sysop      1:125/110   Richard Bollar
USER d.j.hannah 1:125/110   D.J. Hannah
```

Messages coming from the Internet addressed to either:

richard.bollar@swizzle.com
bollar@swizzle.com
root@swizzle.com
postmater@swizzle.com
sysop@swizzle.com

will be converted to Fido format with the message addressed To: Richard Bollar with a netmail address of 1:125/110. Messges coming from the Internet address to d.j.hannah will be sent to D.J. Hannah at 1:125/110

When Richard Bollar of 1:125/110 sends messages to the Internet, GIGO will convert it to:richard.bollar@swizzle.com.

You may have as many internet aka names as you want for the same name but the first statement will always be the default.

```
USER jdoe@somebbs.swizzle.com 1:125/1234.0  John Doe
```

Above is a sample for a user on some other BBS. This is used if you're gating mail for other BBS users. The entire address has to be specified even if a SITE statement has been defined for their BBS. USER statements takes priority over SITE statements.

There is one internal option for the <Internet name> parameter: **BOUNCE** Should someone starts abusing the gateway, you can block them out. When the gateway tries to create this user's email address, it will see that their account says "bounce", and will send them back a nasty-gram(tm). The message sent back, can be edited in your GATEWAY.MSG file, with a regular text editor.

```
USER bounce 1:2/3.4     User Name
USER bounce 1:203/7707.1  Twit
```

USER <internet aka name> <function> <file>

GIGO has a built-in mail-daemon that can perform various functions. Functions supported are:

INFO	sends the text file as defined in the <file> parameter.
BOUNCE	bounce with a generic message or <file> if defined.
FORWARD	re-route mail to a different address.
SAVETEXT	saves the message to <d:\path*.in> file
LOGTEXT	appends the message to <file>
FUNCTION	runs external programs and mails result back

```
USER info@swizzle.com  INFO  infofile.txt
```

Whenever someone sends email to info@swizzle.com, GIGO will reply by sending INFOFILE.TXT

```
USER dead2@swizzle.com BOUNCE
```

Whenever someone sends message to dead2@swizzle.com, the message will be bounced back with a generic message

```
USER dead@swizzle.com  BOUNCE deaduser.txt
```

Whenever someone sends a message to dead@swizzle.com, GIGO will bounce the message back with the “deaduser.txt” file

```
USER jfesler FORWARD jfesler@gigo.com
```

Should someone send mail addressed to jfesler at your site, the message is forwarded to jfesler@gigo.com instead.

```
USER incoming SAVETEXT C:\incoming\
```

This captures messages addressed to incoming@swizzle.com to the c:\incoming directory. Each message received is named successively with the extension of .in

```
USER  logme LOGTEXT  incoming.log
```

This captures messages addressed to logme@swizzle.com and appends it to the file INCOMING.LOG.

USER .. FUNCTION requests

If you are willing to make your own hacks, interfacing with GIGO is easy. GIGO supports function requests triggered upon email to a given user. GIGO saves the entire message as FUNCTION.REQ, and runs your program or batch file. Upon return, if a file called FUNCTION.REP exists, it will be sent back to the party to sent you the email.

Samples:

```
USER  dir   FUNCTION dir c:\*.* /s >function.rep
USER  listserv    FUNCTION listserv.exe
USER  gigo-request FUNCTION listserv.exe s gigo
USER  kaboing     FUNCTION copy function.req function.rep
```

MAPUF <internet aka name> <real name>

The mail-daemon can even attempt to catch fidonet-style names and give them email addresses. For example, if someone sends a message to “Sysop” at your gateway’s address, you can have it sent to your main account instead.

Please note that this only affects Fidonet -> Usenet mail. Also, note that the internet aka name _must_ be the _full_ internet account name, even if it is going to your own gateway.

```
MAPUF  root@swizzle.com Sysop
```

Mail sent to “sysop” is forward to root@swizzle.com

```
MAPUF  jfesler@gigo.com  Jason Fesler
MAPUF  jfesler@gigo.com  Bugs
MAPUF  jfesler@gigo.com  Author
```

Mail sent to “Jason Fesler”, “Bugs” and “Author” are all forwarded to jfesler@gigo.com

HEADERS.CFG

The HEADERS.CFG file controls how the internet message header lines are treated when they are converted to FTSC messages.

The syntax is: Keyword_<headerline>

Keywords can be:

```
Keep_headername   to keep it in the message
Kill_headername   to delete a line totally
Hide_headername   to keep it, but behind a hidden kludgeline
```

Sample header lines to keep, kill, or hide include:

Apparently-to: The internet address a netmail was originally addressed to (this is generated by GIGO internally, and is useful in determining why and where things are)

From: Who the message is from. This is the same as the From line at the message headers (unless you have specified to use RETURNADDRESS_xxxx UUCP).

Organization: Organization (if any) of the person writing the message.

Path: Path from where the message passed through.

Subject: Full subject line

The following keywords, can set the default behavior for all headers. It's usually easiest, to set the defaults, and only override those headers that you want to keep or kill specifically.

MULTIPLEHEADERS

States that you want to have the message headers from the usenet message placed in _all_ fidonet messages. Comment this line out to have GIGO only place the usenet headers on the first message of a split message.

RMAIL_KEEP_ALL

RMAIL_HIDE_ALL

This command specifies that GIGO should keep ALL header lines from an email message. You may instead use a HIDE keyword, which will default all headers to be behind kludge lines. Note that any KEEP, HIDE, or KILL statements will override the default, no matter what.

RNEWS_KEEP_ALL

RNEWS_HIDE_ALL

Same as above, except applied to newsgroup headers be hidden behind a kludge line.

"ALLOW"ING USERS TO MAKE THEIR OWN HEADERS

One of GIGO's features, is the ability to let the user create any header they wish to send out to the internet. Any user can create any header starting with "X" (ie, "X-Stupid-Header: Foobar") at the top of the message. For any header not starting with "X", you must expressly "ALLOW" it in this config file. Only headers that you ALLOW will be permitted from the user.

```
ALLOW Newsgroups:
ALLOW Subject:
ALLOW Reply-To:
```

The above samples will allow your users to arbitrarilly write any Newsgroups: , Subject: , or Reply-To: line they want. No checking is done for correctness!

SAMPLE HEADERS.CFG

```
MULTIPLE_HEADERS ; I want all of the headers on each msg
RMAIL_KEEP_ALL   ; default: show all email headers
RNEWS_KEEP_ALL   ; default: show all news headers
KILL Path:       ; this is useless to me, and often WAY big!
Hide Date:       ; We already know the date; this will hide it.
Hide Subject:    ; Same for subject!
Hide Received:   ; Most users don't care about Received:lines
ALLOW Reply-To:  ; I allow people to set the Reply-To: line
```

THE MAPPING.CFG FILE

Getting newsgroups is a simple prodedure for most people. If you want to get newsgroups, and convert them into fidonet echos, you will need to be sure to tell your provider what groups you want to get.

Many providers hav an automated email system for turning various newsgroups on and off. None of them have standardized; if the day comes that they do, I'll consider making a translation between fidonet areafix requests to whatever standard becomes available. However, after so many years, it still hasn't happened...

Once your feed has turned on the newgroups you want, you will need to edit the mapping.cfg file to tell GIGO what to do with those newsgroups. The default, is to dump them.

There are a few different ways of telling GIGO how to process the newsgroups. One method involves wildcards; the other method is by telling GIGO explicitly (newsgroup by newsgroup) what to do. A few people, do a mixture of both.

The easiest to understand method, is to explicitly tell GIGO how to handle each newsgroup, one by one. The "=" (equal) token is used for this purpose. Note: newsgroup names are case sensitive!

EXPLICIT MAPPING, USING "="

UNMODERATED NEWSGROUPS

The first form, is to state the newsgroup name and what it should translate to.

```
= newsgroup.name ECHONAME
= comp.mail.headers COMP.MAIL.HEADERS
= news.admin.net-abuse.announce NETABUSE-ANNOUNCE
```

These examples will always translate using the first applicable form that it can. When processing newsgroups to echomail, the first matching newsgroup is used. When processing fidonet to usenet, the first rule where the echoname matches, is used.

MODERATED NEWSGROUPS

In Usenet, the concept of a moderated newsgroup exists, that differs from what Fidonet considers a moderated echo. In Usenet, if a post is made in a moderated newsgroup, the news system is supposed to email the moderator, the posting that is being made. If it is not emailed, then the internet host will (silently) disregard the article completely. GIGO does support moderated newsgroups; however, you do have to tell GIGO where to send the post. :-)

The following example will take messages from "moderated.newsgroup", and turn it into ECHONAME. Any posts by you or your users, will be automaticly emailed to "moderator@some.address".

```
= moderated.newsgroup ECHONAME  moderator@some.address
```

A more real-world example would be (all on one line...!):

```
= comp.os.os2.announce OS2-ANNOUNCE comp.os.os2.announce@uunet.uu.net
```

You will want to ask your news host for a current list of moderators, if you plan on posting to any moderated newsgroups.

Making one-way translations with ">" and "<"

If you have need to make a read-only newsgroup, simply use ">" instead of "=". Likewise, to make a post-only fidonet to newsgroup translation, use "<". An example read-only newsgroup would be:

> comp.os.os2.announce LOCAL-OS2-ECHO

This will allow you to have comp.os.os2.announce newsgroup messages posted to your local OS2 echo, and allow the fidonet members to actually hold discussion in the echo. If any messages make it to the gateway, the gateway will discard the messages since it's only a one-way gateway.

Wildcard mapping, with "+" and "-"

Unmoderated newsgroups

One of GIGO's greatest abilities (at least, to some people) is the ability to use wildcards to include and exclude various newsgroups. The operators "+" (to include) and "-" (to exclude) are used to tell GIGO what you are interested in getting.

One concern of using wildcards: They can be slower than explicitly stating what newsgroups you want. However, for some, it may be signficantly faster. How do you know which method to use?

If you are only getting a few newsgroups, it's probably best to just explicity state the newsgroups you want (see the "=" operator). If you are getting a lot of scattered newsgroups, that wildcards do not work well, then explicity state which newsgroups you want as well in that case.

However... if you have large groups of newsgroups, that can be easily stated with a wildcard (ie, if you want _all_ of the comp.os.os2 newsgroups), do so. The fewer lines in mapping.cfg, the better, in terms of speed. The break-even point seems to be around 5 newsgroups (using "=") take about as much time to process the config file, as 1 line using wildcards.

If you can cut a 100 line newsgroup list, down to 5-10 wildcard entries, do so! It will be more than well worth it in terms of time to process it all.

When you use "+" and "-" operators (to use wildcards, etc) GIGO has to read the entire set of mappings - this is the key difference between "+" and "-", versus "=". When using "=", GIGO stops at the first one it finds that matches. Please keep this in mind!

When starting out, GIGO will assume that it is not going to keep the newsgroup at all. It will then read all of the "+" and "-" statements in order, deciding to keep and not keep the newsgroup in question. When it reaches the end of the list, it's decision becomes final. Let's look at a small example:

```
+ comp.os.os2.*
+ comp.os.*windows*
- *.advocacy
```

This example would keep all comp.os.os2 newsgroups, as well as any comp.os.(any of the windows related) groups. However, it would discard the advocacy related newsgroups, as they are always full of flames and no actual content. This cut a list of 60+ newsgroups, down to 3 lines - a definate win in terms of processing time for GIGO.

Moderated newsgroups

You can not use wildcards with moderated newsgroups. You will need to explicity list those newsgroups wiht their moderators. See "=", above, under **moderated newsgroups**.

Converting mailing lists to echos with "@"

One of GIGO's earliest features was the ability to translate messages from an outside internet email mailing list, into a fidonet echo. This allowed for several things:

- Got all those messaages out of your netmail folder
- Allowed several people to read the mailing list, without each person subscribing (and using up bandwidth)
- On some lists, allowed users to post in the echo, and have the post get sent back to the mailing list automaticly

However, a few things quickly became apparent, and unfortuantely, are "just the way things are":

- Some mailing lists are not open mailing lists; they only allow people to post if they are actually subscribed to the list. This means that your users can't post in the echo and except to be broadcast through the list.
- Most mailing lists, send you back a copy of your post. This is because they are sending the message to everyone on the list, and you're on that list.

GIGO can't help on either of those problems. If you are unable to have users post to the mailing list, because it is a closed list, the best you can hope to do is ask the mailing list owner to reconsider. In many cases, they won't (and for good reasons). The best you can do, is have each user that is interested, subscribe. (sigh).

However, for lists that are either read-only, or where the list is an open list (and anyone can post), GIGO will be very helpful in turning your mailing list subscription into an echo, and letting your users post.

For completeness, both the old and new methods will be shown. Versions older than 1996 can only use the old method; any version from 1996 (and after) will be able to use both methods. The second method is a little bit more flexibile, and definately more usable.

The Old Method

You will need to change your name in your message editor to some name that will

help identify the email. If you were subscribing, to the FOOBAR mailing list, you could change your name to "foobar-list". Send in the subscribe message, and get yourself connected. Change your editor's name back to normal :-).

In mapping.cfg, you will need to tell GIGO what fake name you used. GIGO will convert any mail coming to that fake name into an echo of your choosing, and take any new posts in that echo and send them to the address of your choosing (usually, the post address for the mailing list you just subscribed to). Ie,

@ list-foobar@mysite FOOBAR-LIST foobar-l@theirsite

This example will take any messages sent to your "list-foobar@mysite" address, and turn it into the FOOBAR-LIST echo. Any posts in that echo, will automaticly be emailed to "foobar-l@theirsite".

The New Method

The newer method, will allow you to request the mailing list with your regular email address. GIGO will use a combination of the recipient's address, as well as the sender's address, to determine if it's supposed to be translated as a mailing list or not. You need to tell GIGO what address made the subscription; what echo name to give it; what address to send posts to; and (additionally) what address will be sending the messages to the recipient. For this conversion, the "Errors-To:" address, the "Sender:" address, or the "From:" address can be specified.

A real world example of the new method is illustrated below; you can use it to subscribe to the GIGO mailing list, and have it converted automaticly into an echo on your BBS.

"listserv@gigo.com", message body "subscribe gigo". You will get back a confirmation message; confirm it, to finish the subscription process. Once you have done this, put into your mapping.cfg file:

@ youraddress@yoursite GIGO-LIST gigo-l@gigo.com gigo-owner@gigo.com

You will now have the GIGO mailing list coming in, and automaticly translated to the GIGO-LIST echo. Any posts in that echo (by you or your users) will automaticly come back to gigo.com to be posted in the mailing list.

Read-only mailing lists

If you need to make a read-only mailing list, you can specify "null@yoursite" as the posting address, using either the "new" or the "old" methods.

Saving newsgroup messages as files with "!"

Any mapping.cfg entries that start with "!", are extended-syntax rules. The "!" always says that the command will be following separately. The reason for this, is to keep the number of symbols used in mapping.cfg to a minimum (for speed).

Syntax is in the form "! newsgroupname METHOD parameter" and "! newsgroupname METHOD parameter ECHONAME". When the second method syntax is used, the incoming messages are also sent to the echo specified. Note that this does not automaticly catch posts to that echo - these functions are highly special

purpose, and no attempt is being made to make them as generic as the rest of GIGO.

Methods include:

SAVETEXT Saves the entire message as one file, in the dirname specified as the parameter

LOGTEXT Appends the entire message to the filename specified in the parameter

A few example uses of these extended commands follow.

! news.answers SAVETEXT c:\inbound\rtfm NEWS.ANSWERS

(All FAQ's posted to news.answers are saved as raw ascii files in c:\inbound\rtfm, as well as sent out as echomail)

! alt.binaries.dirtystuff SAVETEXT c:\batch\

(You'd want to decode the files in c:\batch\ after running GIGO).

Threading Options

One of the biggest complaints of Fidonet/Internet gateways, as that they loose critical information when replies are made in fidonet, and those replies have to be translated back to Usenet. The biggest complaint, is the lack of "References:" headers; other complaints are related to the fact that most gateways do not preserve or honor the "Newsgroups:" and "Followups-To:" line.

GIGO has two options to address those issues - **SAFE_THREADS** and **KEEP_THREADS**. Of the two, I would prefer to see **SAFE_THREADS** used; it is a far cleaner and safer option compared to the (older) KEEP_THREADS option. Both, are explained further below.

SAFE_THREADS

This option tells GIGO, to start building a database of all incoming articles. The information required for a proper "References:" line is stored, as well as any information for "Followups-To:" and "Newsgroups". An average of 80 bytes is used up, per incoming message, to hold this information.

MSGID and REPLY kludge lines are generated, based on the CRC's of Message-ID and References headers. This way, fidonet based message linkers can perform message threading on the BBS message base.

When a message goes back out, this database is checked. If the fido message as a ^aREPLY kludge line that matches the database, the database is consulted for information on how to build the References: and Newsgroups: lines. If the original message said "Followup-To: poster", then GIGO will automaticly make the reply be sent to the original poster via email. In any case, full threading information is preserved.

This method is definately a bit slower; incoming traffic slows GIGO down a small bit (not too bad). Outgoing traffic, will be the noticable delay, as the database index needs to be scanned once for every outgoing message. At gigo.com, that means an additional 1 second per outgoing message.

Should you decide to use this option, you'll want to occasionally run **cleanid.exe**. This program will check the size of your database, and purge old information once it reaches whatever size you specify. It takes two parameters: the size to trim the database to (in terms of megs), and the maximum size it can reach before trimming. At gigo.com, I use:

```
cleanid.exe 15 25
```

which guarantees that the database will always be between 15 and 25 megs (between 12-20 days of database history), and it will actually do the purge around once every 3-5 days, depending on how busy the gateway is.

KEEP_THREADS

This is the nastier way to do things. Message threading and Reference: information is stored directly in the fidonet MSGID and REPLY lines. Some would argue that

they break the standards; I feel that the standards require GIGO to be able to work in this situation. In any case, some mail processors may not take kindly to the kludge lines produced.

The good part, is no database is used. The bad part, is the compatibilty issue, and the fact that only the _last_ reference item is preserved (not all of them), and the Newsgroups: line will not be preserved or honored either.

I'd use this only if time/space is a serious problem to use SAFE_THREADS, and if you're already familiar with what this option does (it used to be called the NIKE option.. as in "just do it").

External Function Requests

What they are, and how to use them

One of the key features of GIGO, that keeps it always ready for expansion, is the external function interface. GIGO can, when it receives email for a special address, save that message as a raw ASCII file, and run the program of your choice. If your program wishes to mail a response, all it has to do is write to the reply file, and let GIGO take care of addressing it back to the user.

Included with the GIGO distribution, is the "LISTSERV" program; it's source code (written in C) is also include, should you wish to build upon it, or just peek at what I do with it.

Should you wish to build your own external function, you'll want to first define the USER address that will be used for your special function. For our demonstration, we will make a simple auto-reply robot.

```
USER  test@yoursite  FUNCTION  reply.bat
```

The above line, tells GIGO that any time it gets mail to "test@yoursite", it is to run the program called "reply.bat". When it does so, GIGO will create a file called **FUNCTION.REQ**, which your program can read if it wants to.

This function is going to be very simple; it's not going to see who it is, it's just going to reply. Create the batch file called "reply.bat", with this in it:

```
echo We heard you here!   >function.rep
type function.req             >>function.rep
exit
```

All that our function does, is reply "We heard you here!", and quote back the entire message back (headers and all!). All we have to do, is create our reply in a file called **FUNCTION.REP**. When GIGO returns back from that batch file, it will see the .REP file, and automaticly send it back to the user. Done. Easy, huh?

More advanced uses, may need to actually see who the user is that is requesting the service. GIGO makes that simple, as well - the critical information is always written in the first two lines of the FUNCTION.REQ file. The first line will say "Apparently-To: xxxx" (where xxxx is the address that the request was sent to), and then next line will say "Apparently-From: xxxx" (where xxxx is the sender's email address).

NODELIST LOOKUP

GIGO has the ability to verify recipients against a nodelist index, making sure that they are deliverable (or bouncing the message back if they are not). This works only on numeric address; it is assumed that named addresses will always work correctly. Ie, "john.doe@203-7707.gigo.com" would be checked, but "john.doe@bbs.gigo.com" would not be.

If nodelist lookup is enabled, and the nodelist indexes are found, GIGO will actively check it on all incoming mail. If it does bounce a message, the bounce message will contain text found in **GATEWAY.MSG** if you wish to customize it.

Note that GIGO uses it's own proprietory nodelist index. The index was designed to be super-fast and super-tiny, and contains only enough information to know whether a given fidonet address is valid or not. My tests created approximately 100k index files (for the entire nodelist!), and took 8 seconds to build.

To build the index, use the "compile" program; ie,

```
COMPILE c:\nodelist\  NODELIST
```

If you have more than one nodelist to index (ie, if you're part of two networks, say FIDONET and EGGNET) you'll need to run it once for every nodelist. On all but the first, use "/A" as well - that puts the compiler into "append" mode.

```
COMPILE c:\nodelist\  NODELIST
COMPILE c:\nodelist\  EGGNET  /A
```

Once the indexes are built, you can just add "NODELIST .\" to gateway.cfg (you may need to change the directory name, if the nodelist indexes are built somewhere else).

Don't forget to have the compiler run every time you do your other nodelist compiling! Keeping it up to date will be important, should you decide to use this feature.

NLOOKUP

A small utility called NLOOKUP.EXE is provided, to verify that the nodelpist is compiled correctly. It's lookup is identical to the one internal to GIGO. If you suspect that the nodelist feature is misbehaving, try using this utility to help isolate the problem.

```
NLOOKUP 1:203/7707
```

will cause it to look up that address, and report whether or not it found it in the index.

TCP/IP AND OS/2

OS/2

Starting with the August 1995 release, GIGO has supported TCP/IP connections for people running either OS/2 Warp (with the Bonus Pack), or for people running OS/2 version 2.1 along with the expensive TCP/IP add-on package from IBM.

OS/2 Warp with the Bonus Pack is fine for anybody who intends to use only a modem connection to the internet. GIGO.COM is hooked up like that. If you have instead an ethernet connection to the internet, then you will need the OS/2 Warp Connect package.

Please, before working on adding SMTP and NNTP capabilities to GIGO, ensure that your OS/2 internet connection is working properly. You should be able to web browse, telnet, etc properly before attempting to play with SMTP and NNTP.

WINDOWSETC

TCP/IP under Windows (3.1, 95, NT, or anything else) is not supported at this time. It is likely that somebody will build WinSock programs for NNTP and SMTP; it is likely that they will be released as shareware. In any case, I won't be able to provide those programs myself.

If you are a WinSock developer, I'll be more than happy to colaborate with you in building those programs. I'll over the technical information, and even my source code if it helps. SMTP and NNTP are very simple protocals; they are only ASCII based.

THINGS YOU MUST DO

Other things you **must** do to make GIGO work with SMTP and NNTP:

• I strongly urge you only go this route if you are going to have a dedciated internet connection, ie one that is hooked up 24 hours a day. If it is not, I suggest UUCP instead. If you must do part-time TCP/IP, then ask for help in the GIGO mailing list - it is not something I support, but others *are* doing it with some difficulty.

• You must get the "emxrt.zip" package. This is the runtime library DLL's needed to run EMX-compiled programs. All of the TCP/IP tools built for GIGO require this package (as well as nearly every other freeware or tcp/ip related package for OS/2). It is not included automaticly because many people already have it. A copy can be found at ftp.gigo.com, as well as at hobbes.nmsu.edu. Make sure it is version 0.9a,rev6 or later if you get it from any other source.

• You must tell GIGO that you are using the TCP/IP add-ons, by adding "TCPIP" to your config file. This alters the way outgoing messages are created slightly (it eliminates *.CMD files, as well as omits the "From " line that uucp needs). Ie,

```
TCPIP
```

• You must tell GIGO where to expect .BAG files. This is done with the BAGS directive. All BAG files need to go to this directory, whether they are for email or for news. Ie,

```
BAGS C:\BAGS\
```

• You must have a static IP address; ie, every time you get on the internet, your machine's IP address will be the same.

• You must have DNS set up properly. You or your ISP will need to make sure:

forward DNS works: ie, your machine name -> IP address

reverse DNS works; ie, your IP address -> your machine name

MX records for your machine are set up. These are needed if you plan on receiving email.

MX records with wildcards are set up, if you plan on feeding other local BBS's. (ie, in additional to gigo.com, I have *.gigo.com MX records pointing to my machine).

Your ISP will know what this means. If you want to learn what all this means, I suggest any of several **books** on the subject matter by O'Reilly and Associates. Look for them in the library; most of the chapters won't be terribly useful for you unless you are also getting a unix box. Hopefully, your ISP has already *read* these books..

NNTPD - INCOMING NEWSGROUPS

This program is only meant for people who are getting an NNTP feed from their provider. NNTPD does not "fetch" the newsgroups; instead, it sits and waits for your news feed to deliver the news to you. If you instead need to fetch the newsgroups from your provider, instead of having them delivered to you, look at the section on "Using UQWK and SOUPER".

Using nntpd.exe is simple. The following script will run nntpd.exe in a loop (in case of critical errors, the loop ensures that nntpd reloads automaticly). At the beginning of the script, we need to set NNTPSERVER to the name of the server that will be sending us news. In my case, news.calweb.com delivers news to me.

WARNING: This must be the "canonical" name - and not an alias of the machine. The reason for this, is that GIGO will only know the IP address of the machine, and will have to look up it's name. Reverse DNS lookups only return the CNAME (canonical name) of the IP address in question.

NEWS.CMD:

```
set nntpserver=news.calweb.com
:loop
 nntpd -p c:\bags\
 rem  in case of errors...
 goto loop
```

After running this script, nntpd will load and await a connection from the news server. The "-p c:\bags\" tells it to write the incoming news files into c:\bags\ instead of it's current directory. You'll need to make sure that you tell GIGO where to find the **BAGS** directory.

USING UQWK OR SOUPER TO GET NEWSGROUPS

Some people may not have (or want) the luxary of having news delivered to them,

for a variety of reasons. Some news servers simply don't offer the service. Some gateways, are not connect 24 hours a day (I still suggest UUCP if you're not hooked up full time..).

You **can** use a program such as UQWK or SOUPER to fetch the news from the news server. The benefits include being able to edit the "lastread" entries of those programs, as well as easily adding or removing the list of newsgroups you want to get. The downside is, is that it's resource and bandwidth hungry, as it has to ask once for every single newsgroup for the news server to get any new messages. There's a fair amount of time overhead involved, no matter how much or how little you are receiving for that batch - it simply requires "x" amount of time per newsgroup minimum, period.

Whether you run UQWK or SOUPER, you will want to have them configured to download "SOUP" format news files. The will named something similiar to "00000001.MSG", "00000002.MSG", etc. These are not fidonet .MSG files; they are (essentially) .BAG files. Merely rename them, and move them to your BAGS directory for GIGO; GIGO will then be able to toss them.

If you decide to try other methods of getting newsgroup files, the only thing to keep in mind is that they must be standard "#! rnews" bundles, raw ascii (or compressed with "compress", and start with "#! cunbatch"). I've heard of "slirp" being a viable alternative, but I can not tell you anything about it.

Information on how I got "souper" to work with GIGO is in the GIGTxxxx archive. It has sample files to work with, and is likely to be one of the more volatile documents. I'm hoping that others can give me a cleaner looking alternative for it, as well as perhaps something for uqwk and slirp.

SMTPD - Incoming email

This is the mail receiver program. It can be ran one of two ways: either standalone (like the NNTPD program), and only accepting one connection at a time; or off of inetd (NOT the IBM released one - see the section on "Via INETD", below). The inetd method will allow multiple connections at once; however, there is no protection against having too many of them running, so beware!

Standalone:

Usage is rather similiar to the NNTPD program. Make the following script:

MAIL.CMD:

```
:top
 smtpd -p c:\bags\
 rem just in case of error..
  goto top
```

Via INETD.CNF:

Note: Do not use IBM's INETD. If you are using it, and you can not switch, then use the standalone version of smtpd instead. The IBM version is unable to pass unix socket numbers to it's spawned children, and rely instead on only using STDIN/

STDOUT.

For our purposes, use the unix port of inetd. I have a copy of it at ftp.gigo.com:/pub/misc/inetd10.zip ; it's also at hobbes.nmsu.edu:/os2/unix/inetd10.zip.

Here is the line I use at gigo.com:

```
smtp stream tcp nowait root smtpd -s %s -p c:\bags\
```

The "-s %s" tells smtpd what socket number SMTPD should pick up. Don't forget it - without it, smtpd will run in standalone mode! The "-p c:\bags\" tells GIGO to place all incoming mail bags into c:\bags\ instead of the current directory.

GIGOSLIP - OUTGOING EMAIL AND NEWS

GIGOSLIP, GIGOHELP, GIGOMAIL, AND INEWS

These are the programs that will get your outgoing traffic out. GIGOSLIP will scan your UUCP spool directory, and determining what packages are news, and which ones are email. It will then call either GIGOMAIL or INEWS (via GIGOHELP).

Everything is configured through environment variables. A future version, should be able to use a config file.

A sample batch file follows:

PBATCH.CMD

```
set mailserver=mail.calweb.com
set nntpserver=news.calweb.com
set bags=c:\bags\
set tz=PST8
cd \gateway\spool\myfeed
gigoslip
```

These environment variables can be set via your CONFIG.SYS if you wish (don't forget to reboot!). The "cd" command switches to the directory where you have outgoing mail or news; it is always a combination of your SPOOL and your MYFEED variables from gateway.cfg.

This program **is** safe to run multiple copies of; any files that are presently being transfered are locked until they are done being sent. However, you'll want to make sure that you don't have too many copies running, or else your CPU or bandwidth limits may be reached. On my 28k modem, I only allow up to 2 copies to run simultaneously. One idea is to make a "cron" type even that starts a new copy every 15 minutes. Another idea, is to make the batch file loop, with a few minute delay between loops, and perhaps have two copies that batch file running.

Hosting Mailing Lists

Hosting a Mailing List

GIGO has internal support for hosting mailing lists. These mailing lists can also optionally be gated to echos and newsgroups at the same time, potentially performing up to a three-way gating of the various mediums.

Before you commit yourself to hosting a mailing list, be aware of the traffic and headache that you will be starting. Mailing list traffic can be a very heavy toll on a gateway, especially those that are running under UUCP. You will need to send out one message per recipient, per message received. That works out to a thousand messages in one day if there are only ten posts and a hundred recipients.

Most UUCP hosts can take multiple recipients on each message, cutting some of your transfer time down. I've found that most sites can fit a hundred characters worth of addresses per outgoing message; realisticly, that's only two or three addresses. If your site can take larger lines (hopefully, 256, 1024, or 2048) take advantage of it! (See **MAXRMAIL**). It will cut your transfer times down immensely. Using SMTP at gigo.com, I can use 2048 (GIGO's maximum) to cut my outgoing traffic down immensely.

Another thing to be wary of: People will close their internet accounts or move, and not unsubscribe. Others, will post "subscribe" messages to the mailing list. Others, will refuse to use any automated means to add and drop themselves from the list. Others, are a simple pain in the exhaust pipe.

If you still want to host a mailing list, read on..

GIGO's handling of mailing lists, is actually quite simplistic. It merely captures messages from various sources, and saves those messages as files in a directory. Those messages can be captured from a variety of sources: newsgroups, email, echomail, and any custom methods that you might add on later. Later, it scans all of the mailing list directories, looking for new files to send out. If any are found, every recipient of the mailing list will get a copy. The newsgroup and/or echo may also get a copy, via a special email->newsgroup and email->echomail gateway that GIGO provides.

When capturing messages, GIGO will save the messages as *.MLQ (mailing list queue) files in the designated directory. As GIGO sends each of them out, the files are renamed to *.MLS (mailing list sent) files. Lastly, if the messages are determined to be bad (as per the badmail.ml file), the messages are instead named *.MLB (mailing list bad). If you see any *.MLB files, and you wish for them to actually be distributed, you merely only have to rename the file to *.MLQ; GIGO will pick them up the next time GIGO runs.

You will need the following definitions set up.. Place this into your **GATEWAY.CFG** file:

INCLUDE MAILLIST.CFG

This tells GIGO that you want to import the configuration options contained in MAILLIST.CFG. By default, this option is not enabled, as most people do not need it.

In your **MAILLIST.CFG** file, you'll want to add..

SITE pnews news

SITE pecho echo

These two lines define the email->newsgroup and the email->echomail gateways. These gateways **must** be defined if you are planning on having your mailing lists translated into newsgroups or into echomail conferences. *You may consider using names other than "pnews" and "pecho" for security reasons* (anybody can use these gateways, so keeping their names a secret is desirable).

MAXRMAIL 200

The MAXRMAIL command specifies the maximum length of the "rmail" lines put into the UUCP packets. 200 is generally safe; however, if many people complain that they are not getting a copy, lowering this may be needed. Notably, people using "waffle" (the DOS version) for their UUCP feeds, will want to drop this to 100.

You may want to ask your provider what the maximum size of that line can be; if you can raise this value to 1024 or 2048, you can cut down a major portion of your online time transfering email. This is due to being able to fit more people on each single message from the mailing liset.

OS/2 SMTP users: You can safely set this to 2048, as long as your host's SMTP accepts that many recipients. Sendmail on the host side definately handles it fine. I run the gigo.com mailing lists that way; it cuts down traffic immensely for me.

ML_COPY

By default, GIGO will send mailing list messages to only those people who need to see them. The person posting the message, does not need a copy (they've already seen it). Some mailing list administrators prefer, however, that the users who post do get a copy back. Enable the ML_COPY option, and GIGO will start using this new behavior.

The rest of the configuration is used to cover the individual mailing lists themselves. Each mailing list needs to have at least one "CAPTURE" statement (to capture postings sent as email, echomail, or as newsgroup messages), as well as one "EXPLODE" statement (to send newly captured messages back to me internet).

You can have several CAPTURE statements that all capture to one common directory; this is how you set up multiple posting addresses for the same one list. An example: gigo's mailing list has both "gigo-l@gigo.com" as well as "gigo@gigo.com", both common posting addresses for the list. They both point to the same directory, so

they effectively become the same list.

CAPTURE dirname\ inputaddress owner newsgroup ECHONAME badmail.ml Description

CAPTURE statements require the following parameters:

dirname All captured messages have to be stored in a directory. You can not have several mailing lists share one directory; however, you can have several CAPTURE statements for one single mailing list, going to the one directory.

inputaddress This is the address to capture. It can be a simple name (ie, gigo-l), or a complete address (ie, "gigo-l@gigo.com"). Any messages to this address that are received by your gateway, will be stored in the directory you specified.

owner This is the "owner" of the mailing list. Errors (such as mail bounces, etc) will go to this address. You do not want this address to be that of the mailing list itself, or else the bounce messages would loop into the mailing list. Normal convention is "listname-owner@sitename". **NOTE**: If you do use this format, make sure you define a USER statement with that address!

newsgroup This is a newsgroup name to monitor and capture. Any posts to this newsgroup will be saved for the mailing list. If you do not wish to capture any newsgroups, use "-" instead. You must either specify a newsgroup name or "-".

ECHONAME This is an echomail area name to monitor and capture. Any posts to this echo will be saved for the mailing list. If you do not wish to capture any echos for this mailing list, use "-" instead. You must specify an echomail area, or "-".

badmail.ml This is a "badmail.ml" type file, that describes which messages you want to automaticly filter from the mailing list. This filter can handle things like dirty words, words commonly used to slander with, email addresses of problem users, etc. You can also use it to filter out messages that say "unsubscribe", to prevent such lists that went to the wrong address from being rebroadcast to everyone on your list.

Description This is the description of the mailing list. It will be used when generating the mailing list messages.

EXPLODE dirname\ listfile.ml owner postnewsaddress postechoaddress

EXPLODE statements are what actually send the mailing list messages out to the world. You can only have one EXPLODE statement per directory. The parameters are as follows:

dirname\ EXPLODE will look for new messages in this directory to send out.

listfile.ml This is an ascii file of recipients who are supposed to receive messages for this list. There should be exactly one person per line in that file. (LISTSERV.EXE can automaticly maintain that file for you if you wish).

owner This is the same owner address as what you used in the CAPTURE statement. It is far faster to configure it twice, than to read each message to get the information. (It is needed twice; once for the message that is created, and once for the UUCP packets that are generated).

postnewsaddress If you wish to have messages from this mailing list posted to a newsgroup, you will need to specify the posting address for the newsgroup message. For this, you will be taking advantage of GIGO's email->newsgroup gateway, that was previously defined as "SITE pnews news". An example email address would be "newgroup.name@pnews.yoursite". (Of course, if you are using a name other than pnews, you'have to make that adjustment; as well, yoursite should be your real site name.). If you do not wish to post to a newsgroup, use "-". You must specify either an address or "-".

postechoaddress If you wish to have messages from this mailing list posted to an echomail conference , you will need to specify the posting address for the echomail message. For this, you will be taking advantage of GIGO's email->echomail gateway, that was previously defined as "SITE pecho echo". An example email address would be "ECHONAME@pecho.yoursite". (Of course, if you are using a name other than pecho, you'have to make that adjustment; as well, yoursite should be your real site name.). If you do not wish to post to an echo, use "-". You must specify either an address or "-".

Confused yet? A few examples are listed below. They will also be found in the sample **maillist.cfg** file.

```
;
; SITE  pnews  news ; Any mail sent to  news.group@pnews.yoursite.domain gets
  posted
; SITE  pecho  echo ; Any mail sent to  echoname@pecho.yoursite.domain gets
  posted

; This is a 3-way gate between foobar@wmeonlin.sacbbx.com, alt.foobar, and the
  FOOBAR echo
CAPTURE c:\ml\foobar\ foobar    foobar-owner alt.foobar FOOBAR badmail.ml The
  Discussion on the meaing of Foobar vs Fubar
EXPLODE c:\ml\foobar\ foobar.ml foobar-owner alt.foobar@pnews.wmeonlin.sacbbx.com
  FOOBAR@pecho.wmeonlin.sacbbx.com

;This is just a mailing list between quasi@wmeonlin.sacbbx.com and the QUASI echo
CAPTURE c:\ml\quasi\ quasi     quasi-owner -  QUASI  badmail.ml The Disc
EXPLODE c:\ml\quasi\ quase.ml  quasi-owner -
  QUASI@pecho.wmeonlin.sacbbx.com

;This is a between a newsgroup called alt.caffeine and a mailing list - but no
  echomail copy at all.
CAPTURE c:\ml\caffeine\ caffeine    caffeine-owner alt.caffeine -  badmail.ml
  Caffeine Addicts Anonymous
EXPLODE c:\ml\caffeine\ caffeine.ml caffeine-owner
```

```
alt.caffeine@pnews.wmeonlin.sacbbx.com -
```

USING LISTSERV TO AUTOMATE MAILING LISTS SUBSCRIPTIONS

Included with the GIGO distribution is an external utility, called LISTSERV.EXE. It has the ability to read incoming requests; decide if the person is trying to subscribe, unsubscribe, or get more help; and alter your *.ML files for your mailing lists automaticly. It also has an additional optional feature of requiring people to confirm mailing list subscriptions, to help prevent people with bad email addresses from getting onto the list in the first place.

Make sure that LISTSERV.EXE (and it's associated documents) are in your GIGO directory. Any *.ML files that you wish to automaticly maintain, must also be in your GIGO directory. (Any *.ML files that are not in the GIGO directory, will require that you manually maintain the mailing list file - a good option for private mailing lists).

You will want to edit the following files:

LISTSERV.TXT This message is sent at the beginning of every response.

LISTSERV.IDX This is a message sent back in response to the "INDEX" command; it should contain a list of your mailing lists and their descriptions.

LISTNAME.ML For each mailing list, you may wish to add yourself in as a recipient, at least at first. This helps start the mailing list, as well as help you confirm that things are working.

LISTNAME.FAQ This file is sent whenever somebody subscribes to a mailing list. It can be customized to each particular list. Common information would be where to send mailing list posts, official language, how to remove themselves, etc.

Next, you'll want to add a USER statement into your USERS.CFG file:

```
USER listserv FUNCTION listserv.exe
```

Any messages sent to listserv@yoursite, will be ran through GIGO's external function request processor. LISTSERV.EXE will read the message, parse out any instructions, and write a reply for GIGO to send back to the user.

MESSAGE DIGESTS

This section has been written by Stephen Griffin, for which I am grateful.

```
Gigo Digest Processor Documentation v. 950801
This software is for Digest versions of your mailing-lists. A Di-
gest is a series of individual mailing-list messages packaged up
into one message (so you may for instance get 10 mailing-list mes-
sages sent as one email.)
1) System Requirements: A working copy of GIGO. A list-server (see
the one that came with GIGO) A working knowledge of mailing-lists
and how GIGO utilizes them in particular)
2) Installation:
```

For each mailing-list you wish to have a Digest for, you must place a copy of digest.cfg in the directory where the .MLQ files for that list are stored.
Sample digest.cfg:

```
GIGO
From gigo-owner %s remote from gigo.com
Received: gigo.com (digest proc); %s
Date: %s
From: gigo-owner@gigo.com
Reply-To: gigo-l@gigo.com
To: gigo-l@gigo.com
Administrative note:
This is the GIGO support mailing list, in digest format.  To unsubscribe,
send a message to "listserv@gigo.com", message body "unsub digest".
Please remember that when you post, people all over the world will be read-
ing. The official language is English to help make sure as many people can
participate as possible.
```

Explanation of digest.cfg line-by-line:
Line 1: Name of the associated mailing-list (ie if you have widg-d.ml for the digest and widget.ml for the mailing-list, the first line would be WIDGET.
Line 2-7: Must exist in the specified order (%s's mandatory), but you should change the value's to what they need to be such as "Received: widget.widget.com (digest proc); %s" if you were widget.widget.com.
Line 8+: This will be prepended to all digest messages, so you may want to state what digest this is, how people can unsubscribe etc..
Sample maillist.cfg

```
CAPTURE gigo\ gigo-l gigo-owner@gigo.com - GIGO badmail.ml GIGO Support
CAPTURE gigo\ digest gigo-owner@gigo.com - GIGO badmail.ml GIGO Support
EXPLODE gigo\ gigo.ml   gigo-owner@gigo.com - GIGO@pecho.gigo.com EXPLODE
digest\ digest.ml gigo-owner@gigo.com - -
```

The CAPTURE line for the digest should point to the same directory as the mailing list (this is so that if someone replies to the digest message the people on the mailing-list will see it). The EXPLODE line however should point to a directory specific to the digest itself. The digest should also have it's own .ml and .faq.
3) Subscribing to a digest:
A person subscribes to a digest in the same method as they would a normal mailing-list (as the digest uses the standard .ml file produced by the listserver software) except they would subscribe to widg-d instead of widg (assuming you had widg.ml for the Widget Mailing list and widg-d.ml for the Widget Mailing list (Digest).
4) Operation: However often you wish to have a Digest produced (once a day is recommended) you should have an event which calls DIGESTP.EXE (or DIGEST.EXE for DOS people) from inside the associated mailing-list directory for a digest. This will create a bunch of *.DLQ files (Digest Queue files.) These should then be moved to the directory pointed to by the EXPLODE statement for the digest

(but they should be renamed to *.MLQ (so GATEWAY.EXE can process them.)) This should be repeated for each digest you have.
NOTE: Make sure you set the MAILSERVER environment variable before calling DIGEST.EXE if you are operation TCP/IP instead of UUCP (UUCP people should not set this variable.)
Sample .cmd/.bat
———————

```
rem process digest wsomr-l
cdd n:\gateway\wsomr-l
rem set MAILSERVER so that digestp realizes we are TCP/IP set
mailserver=mail.calweb.com
n:\gateway\digestp
move *.dlq n:\gateway\wsomr-d\*.mlq

rem process digest changi
cdd e:\gateway\changi
rem set MAILSERVER so that digestp realizes we are TCP/IP set
mailserver=mail.calweb.com
n:\gateway\digestp
move *.dlq e:\gateway\changi-d\*.mlq
```

———————
That’s it, if you need furthur assistance please contact: jfesler@gigo.com
This Documentation was written by Stephen Griffin (root@bdragon.jjm.com) and Jason takes no responsibility for typing errors, grammatical foul-ups and other minor mistakes.
(c) Copyright 1995 Jason Fesler
Stephen
— GoldED 2.41 —Stephen Griffin -Sysop/Sysadmin bdragon.jjm.com f401.n333.z1.fidonet.org Stephen Griffin 1:333/401

File Formats and Structures

UUCP files

UUCP files are documented thorougly in Ian Taylor's UUCP Internals FAQ. If you are planning on creating or modifying files in the UUCP spool directories, get the current version of the FAQ. It has lots of great information and background.

Email files

Incoming email messages are split into two parts: the .X file and the .D file. The .X file will tell GIGO where the message is supposed to go. It doesn't matter what the headers say; it is the .X file that has the authority on the manner. One can not trust the headers at all. Case in point: Mailing lists often have the mailing list address, for the "To:" field, but not the subscriber's address. The .D files, are merely raw ascii files with all of the headers and message body as one document. Further information is in RFC-822.

News files

Newsgroup files are standard RFC-1036 news messages. For batching, they may use the "#! rnews" standard: The first line of each message, must say: "`#! rnews 1234`" where 1234 is the size of the following message, in bytes. The whole string is followed by a new line. **NOTE**: Carriage returns are not used at all; unix standards use only a linefeed for the end-of-line.

Any newsgroup files that lack the "`#! rnews`" line, will be considered to be single messages, and will assume the entire file is the content of the message.

.BAG files

.BAG files can hold either news or email. For news bags, they are just the standard news files, where each message is preceded by "`#! rnews size`". If that line is missing, it will be assumed that the entire file is a single large newsgroup message.

For email, instead each line is preceded by "`#! rmail size recipient1 recipient2 recipient3`" etc. The size, is the size of the email message, including headers etc - just like a news file would be. The only major difference, is that a list of recipients is also on that line. This list should be the same list that the delivery envelope knows about - ie, if it were UUCP, the contents of the .X file, or if it was SMTP, the contents of the SMTP exchange itself.

FAQ'S - ANSWERS TO COMMONLY ASKED QUESTIONS

Q: Why does my log say I am dumping all this stuff I am not signed up for?

Q: Why am I getting all these areas that I am not subscribed to?

A: When newsgroups are crossposted, only a single copy is actually sent to you; in that message, there is a line that says what newsgroup(s) that single message needs to be placed into. This line _may_ (and probably _will_) contain one or more groups that you are not subscribed to. However, the "Newsgroups" line is not altered.

When GIGO gets messages with crosspostings to newsgroups that you do not want, GIGO will automaticly keep copies of the appropriate areas, and dump those that you do not want. If you enable logging of dumped messages, it is these message areas that are being dumped, and _is_ perfectly normal.

Personally, I think that logging dumped messages is useless, but I had requests for it, so I programmed it in. I have that option disabled on my system.

jason

Q: How do I tell GIGO which headers I want from the messages?

Q: What are KEEP, HIDE, KILL, and RMAIL_KEEP_ALL

A: GIGO uses the KEEP, HIDE, and KILL keywords to determine which headers to import from the original message. By default, no headers are kept. If you specify RMAIL_KEEP_ALL, then the default for _email_ is to _KEEP_ all headers. Be sure to end the header with ":". Most header lines have the ":", so GIGO is picky on this one. To specify a header that does not use the colon, simply omit it. (Clarification: "From " is different from "From:"; both are present in email messages.)

To KEEP certain headers:

```
Keep Apparently-to:  ; Shows who it's going to
Keep Archive-Name; Used in news.answers and elsewhere
Keep Date; What date the message was written (orig text)
Keep From; Who it's from (highly suggested)
```

To HIDE (as in, keep them, but hidden in the kludge lines)

```
Hide Newsgroups; What newsgroups the message was posted to
Hide Message-ID; Original MSGID from the usenet message
Hide Organization; Organization (if any) of the person writing the msg
Hide Subject; Full subject line
Hide To:
Hide Received:
Hide Reply-To:
Hide Sender:
Hide In-Reply-To:
Hide Mime-Version:
Hide Content-Type:
```

```
Hide Errors-To:
Hide Precedence:
Hide X-To:
Hide X-Sender:
Hide X-Ftn-To:
Hide X-Ftn-From:
Hide X-Resent-By:
Hide X-Admin-Addr:
Hide X-List-Software:
Hide X-Listname:
```

To KILL (as in, useless info, we don't want it)

```
Kill Path; This field is a USELES field for most gateways
```

Q: How do I tell GIGO which headers I want to allow my users to create?

Q: What is this ALLOW keyword for?

A: For this, the ALLOW keyword is used. By default, users are allowed only the "To:" header, so as to tell the gateway where they want the message to go to. In addition, any header starting with "X-" is allowed, since those are generally used with various email services. You may have need to allow users to create and/or override certain headers, a practical example would be the "Reply-To:" header.

Assuming that you add "ALLOW REPLY-TO:" to your config file, your users will be able to write a message such as:

```
+————————————
|From:  User Name,  1:2/3
|To  :  UUCP, 1:203/2
|Subj:  Whatever subject
+————————————
|To: null@wmeonlin.sacbbx.com
|Reply-To: User.Name@othersite.org
|
| Message body goes here
```

Q: What if the header is too big for one line?

A: Simple, use ".." on the second line. Ie,

```
+————————————
|From:  User Name,  1:2/3
|To  :  UUCP, 1:203/2
|Subj:  Whatever subject
+————————————
|To: null@wmeonlin.sacbbx.com
|Organization:  This is going to be a really long Organization
|.. line that will probably not fit into 80 characters or less.
|
| Message body goes here
```

Note that there is a space after the ".." ; GIGO will not put anything in by itself - if it needs a space, or a comma, put it in!

Q: How do I get the latest version of GIGO?

[boilerplate message]

Places to check for the new GIGO release will be:
ftp: ftp.gigo.com, in /pub/gigo
www: http://gigo.com
Ways to *get* GIGO:
ftp: ftp.gigo.com, in /pub/gigo
www: http://gigo.com
fido: File request “GIGO” (dos) or “GIGOP” (os/2) from 1:203/7707 or from 1:203/8055, or through IP address 165.90.138.204 (please, use FTP instead, I have limited bandwidth to the net)
email: Email fileserv@gigo.com, message body “get gigo” or “get gigop”.
Mailing lists:
Send email to listserv@gigo.com, message body “subscribe xxx” to get on any of these mailing lists.
GIGO GIGO Support (Internet/fidonet gateway software)
DIGEST GIGO Support (Digest message format; one _big_message per day, with all of themessages from “GIGO” in it)

GIGOALPH GIGO OS/2 alpha version binaries (distribution)
GIGOBETA GIGO DOS beta version binaries (distribution)
NOTIFY GIGO new version notifications (no binaries or discussion)
UFGATE The UFGATE conference from Fidonet (internet/fidonet gating)
WSOMR-L Support list for WSOMR offline reader (BW,QWK,SOUP,RFC)
WSOMR-D Daily digest of WSOMR-L
CHANGI Support list for Changi OS/2 NNTP server
CHANGI-D Daily digest of CHANGI

Q: How do I host a mailing list?
Q: Sample of hosting a mailing list

A: [This _only_ applies if you are trying to HOST the mailing list; if all you want to do is _translate_ an existing mailing list from another site into echomail and vice versa, see the section on MAPPING.CFG.]

Hosting a mailing list is easy with GIGO. GIGO contains two verbs, called CAPTURE (which intercepts messages to certain email addresses, newsgroups, and/or echos), and EXPLODE (which sends the captured messages to the various recipients).

In addition, you may find LISTSERV to be a valuable tool; that is discussed seperately. It’s main function is to automate
subscribe and unsubscribe requests from various users.

The best way to explain how this works, is with a real example of hosting such a mailing list. This example actually spans several Q/A sections.

Problem:

You wish to host a mailing list (called “fubar-l”) at your site. You wish to also have this translated to an echomail conference called “FUBAR”, as well as to the newsgroup “alt.fubar”.

Q: How do I create a .ML file?

Q: What is a .ML file?

A .ML file is simply an ASCII file, with a list of recipients. Each line should have exactly one address on it, and should not include the real name or any other extraneous information. Create a file called “fubar.ml” in your GIGO directory. You will want to place one email address per line in this file. Ie,

```
+————————
|null@wmeonlin.sacbbx.com
|somebody@somewhere.else.org
|root@yoursite
|
```

It is probably a good idea to include your regular email address on this list, at least at first, to help confirm that everything is working as planned.

Q: How do I set up addresses for posting news and echo-mail by email?

Skip these instructions if you have already done this for another mailing list; they are only needed once for your gateway.

Next, you are going to add a couple of SITE statements in your gateway.cfg file. You may instead wish to INCLUDE MAILLIST.CFG, and keep your mailing list configuration in this config file instead.

A sample one should be included in the GIGO archive. Add:

```
SITE pnews new; creates a SITE to post to newsgroups with
SITE pecho ech; creates a SITE to post to echomail with
```

These two SITE statements allow posts to go to places such as ECHONAME%pecho@yoursite.domain. You may change “pnews” and “pecho” to something else of your choice for security reasons :-) ; these are just the examples that I am using. It is suggested that you do use a different set of names, so that you don’t get some twit hacking via your gateway.

Q: What is this “badmail.ml” file?

Q: What are .MLB files?

“BADMAIL.ML” is a file a bit unlinke the other .ml files; this file is actually a set of rules that tell GIGO what mail should _not_ be re-broadcast in the mailing list. You would want to tell GIGO that messages from mailer-daemons, etc are not to be resent, as these are normally bounce messages that did not go to the bounce address.

Each line in this file should be the token name to look for (ie, “From:” or “Subject:”), followed by a space, and then the text that defines it as a bad message. An example file would be:

```
+————————————
```

|From: daemon
|From: Mailer-Daemon
|From: daemon
|From: DAEMON
|From: !uucp
|From: uucp@
|From: Post Office
|Subject: Returned mail: Host
|

Any message that has been determined to be bad by the rules defined, will be saved as a .MLB (Mailing List - Bad) file instead of a .MLQ file. You can either delete these messages, or (preferably) take a look at them first, and possibly rename it to a .MLQ file and let it actually be sent out.

Q: How does one CAPTURE messages for a mailing list that I am hosting?

Now, you must tell GIGO what messages to CAPTURE, and where to place them. You will also want to tell GIGO where bounced messages from the mailing list should be sent (undoubtedly, someone's address who subscribes will not be valid).

Assuming:

you want to use the "fubar\" subdirectory
you want error messages to be sent to "fubar-owner@yoursite.etc"
you want _email_ to fubar-l captured
you want _news_ in "alt.fubar" captured
you want _echomail_ from "FUBAR" captured
you have a defination of "bad mail" in badmail.ml
you want to name the mailing list "The Fubar Debate"
you will want to create the following CAPTURE line:

```
CAPTURE fubar\ fubar-l fubar-owner alt.fubar FUBAR badmail.ml The Fubar Debate
```

Q: What is a .MLQ file?

Once this is defined, any messages sent in any of the three mediums will be captured, and placed into the fubar\ directory as a *.MLQ file (MLQ stands for Mailing List Queue). It is a plain ASCII text file, with the first line of that text file being a Ctrl-A followed by "ECHO", "NEWS", or the email address of the person that posted the message. This information is used by EXPLODE so that the message is not sent back to the same place the message originally came from.

Note that you may have multiple CAPTURE lines, even pointing to the same directory (ie, if you have multiple incoming email addresses). Only the first applicable CAPTURE statement for a given message will be used.

Q: How do I send out the .MLQ files that were CAPTURED

Now that you have these captured messages, you are going to need to send them out to the various recipients.Assuming that you created “SITE pnews news” and “SITE pecho echo”,and assuming that we are following up on the previous example for the FUBAR mailing list:

```
EXPLODE fubar\ fubar.ml fubar-owner alt.fubar@pnews.mysite.org
FUBAR@pecho.mysite.org
[Quoted from MAILLIST.CFG]
 ;  Whenever GIGO runs, if there are any EXPLODE statements in the config
 ;  file, GIGO will look at the specified files for *.MLQ files.  If
 ;  it finds any, it will process those files, and rename them to
 ;  *.MLS files (Mailing List Sent).  You may either do further processing
 ;  on them, or simply delete them - that is up to you.  The author
 ;  saves them in an archive for later use.
 ;
 ;  When GIGO finds a MLQ file to be sent out, it will take a look at the
 ;  first line of the file.  If it starts with Ctrl-A (“ ”), it will
 ;  assume that that first line contains the phrase “NEWS”, “ECHO”,
 ;  or the email address of the person that wrote the message (if the
 ;  message was sent as email instead of via the newsgroup or echo).
 ;
 ;  GIGO will then take a look at the list of recipients, and start gener-
ating
 ;  the appropriate information in the spool directory for sending messages
 ;  out to all of the recipients.  GIGO will bundle roughly 10 addresses at
a
 ;  time to a single message (if you have 100 recipients, you are likely
going
 ;  to send out 10 messages with 10 people on each one).  The exact number
 ;  will vary ; GIGO will never place more than 250 characters on the
“rmail”
 ;  line in the XQT files.
 ;
 ;  GIGO will also want to send the message to an email->newsgroup address,
 ;  and an email->echomail address.  GIGO can provide these address.  These
 ;  are only neccessary if you wish have messages from the mailing list
 ;  sent to a newsgroup, or sent to an echo, or both.  This provides a
 ;  3-way gate between all 3 medias.
```

Q: What is the difference between the various mapping commands?

Q: What are the differences between +, -, =, <, >, @, and ! ?

Seriously.. see the large section on "MAPPING".

This version of the documentation has expanded quite a bit on how the mapping function work.

Q: How can I translate a mailing list into echomail format?

A: See the section on MAPPING.CFG, and look for the "@" operator.

Q: How come the mailing list keeps rejecting messages from my users?

A: Some mailing lists are ran in a “secure” mode. What this means, is that only the users actually registered in the mailing list are allowed to post to the mailing list. When you tell GIGO to translate a mailing list, you are subscribing with a name (such as list-gigo). When you or a user posts, however,that name is not what is sent to the mailing list - the user’s real address is sent. When the mailing list is being ran in a secure fashion, it automaticly rejects the message, due to that user not being a participant in the mailing list. The only way around this, is to contact the administrator of that site. However, most running a secure mailing list are not willing to compromisethe security of that mailing list - your only option in this case is to either make the mailing list read-only, or to let individual people to subscribe, and let them receive the mailing list as [net/e]mail.

Q: How do I use BIOS video writes?

A: Use the command line switch, “/BIOS”.

Q: How do I use direct video writes?

A: That is the default.

Q: How do I turn off the video display?

A: Specify NOVIDEO in your configuration file. Once GIGO is past the configuration stage, and is fully loaded, GIGO will stop generating video output, giving you an approximate 20% increase in speed.

Q: How do I stop GIGO from deleting files, when I am testing?

Q: Why am I getting things in multiple when I use the TEST keyword?

A: Specify the TEST keyword. When you use the TEST keyword, some actions will still be ran more than once, and you may see multiple instances of the same message.

Q: GIGO can find the .X files, but can not find the .D files. What now?

A: Enable the “ALTMUNGING” keyword, and see if that works. Some sites are feeding you non-standard unix filenames; they should be handled anyways. ALTMUNGING will only be needed by an estimated 7-8 sites in the world.

Q: The filenames contain my host’s name instead of mine.

A: This is _normal_. However, if you want to switch it to have your site’s name instead of your host’s, for whatever reason,specify “BACKWARDSFEED”. Don’t do this unless you really need it.

Q: My “From “ lines need FQDN names, not bang paths..

A: Look at the BURT_JUDA_HACK. This is only required if you are directly piping the files into a sendmail session on unix; most UUCP hosts handle this stuff properly automaticly.

Q: My host requires the .XQT file to say it's from "uucp"..

A: Enable XQT_FROM_UUCP in your configuration. Normally, the address that errors should be sent to would be the original person who wrote the message. For some people, however, their host will only accept "uucp" for the name that errors should goto. [Okay, not some people, but one specific person..:-)]

Q: People say that my entire message is on a single line. What I do?

A: LINESPLIT 80 tells GIGO to take all fidonet messages, and make sure that no line is longer than 80 characters, so that peopleon the internet can read your messages.

Q: Arg, so many crossposts! How do I limit the number of copies GIGO makes?

A: Use MAXNEWSGROUPS 5 if you want to limit GIGO to only 5 copiesof the same message, one copy per newsgroup that you are keeping.

Default is to allow for up to 20 crossposts.

Q: How do I get rid of "cmsg cancel" messages?

A: Use CMSG_CANCEL - it will look for any message with a "Control:" header in it (mostly used for cancellation messages), and junk the message.

Q: Why is my tosser saying GIGO's pkts are from Net 65535 or Net -1 ?

Q: What is the OLDTOSSER option for?

A: This only happens when GIGO is running as a point off of your mainaddress; this behavior is not apparent when GIGO has a dedicated node address. Some fidonet software packages are not entirely FSC-0048 aware. GIGO generates FSC-0048 mail packets, which specify that when sending to a point, that the NET field be specifed as "65535" (or "-1"). To accomodate for the older programs that get a tad bit confused, add the config option "OLDTOSSER". It will generate "Almost-FSC-0048"mail packets instead, which will be properly understood by those tossers.

Q: What is FSC-0035?

Q: What are the ^aREPLYTO and ^aREPLYADDR kludge lines?

A: An option in GIGO is called FSC-35. FSC-35 states that messages should contain kludge lines called REPLYTO and REPLYADDR at the top, which help certain message editors reliable replyto the authors of various messages. These kludge lines contain the address of the internet gateway, as well as the complete internet address of the person that should receive the netmail reply.

Q: What is PARTIAL-35?

A: Some people are annoyed with all of these extra kludge lines in the messages. PARTIAL-35 is a compromise; PARTIAL-35 will only introduce the FSC-35 kludge lines, _if_ and _only if_ the netmail“From:” field is not enough to get a guaranteed reply back to the proper place. For example, if the message was from “jfesler@netcom.com”,the fidonet header is big enough to hold the email address. However, “Joe.Schmoe@Barbeque-BBS.wmeonlin.sacbbx.com” is too long -FSC-35 kludge lines would be included, so that you can accurately reply to him.

Q: How do I change the size of the .PKT files GIGO creates?

A: Use the MAXPKT command. It specifise the approximate .PKT filesize that you wish individual .PKT files to be. MAXPKT 150000 seems to be a decent number, your mileage may vary.

Q: How do I stop large messages from coming in?

Q: How do I split up such large messages?

A: Use the SPLIT keyword. Suggested: SPLIT 15000 ; which leavesenough room for other mail processors to keep the total messageunder 16k. Many mail processors (including those that the backboneruns on) break after 16k; your mileage may and will vary. As to stopping them from coming in: Sorry, by the time GIGO gets a hold of the message, you've already downloaded it!

Q: How do I get people’s addresses to show up in the fidonet header?

Q: Why are people’s names in the fidonet header instead of their address?

A: You can change what information GIGO places into the fidonet header pretty easily; it is definable for both email and for newsgroups. Look at the section on "RETURNADDRESS" in the documentation for a full description.

Q: How do I get threading information from the usenet messages?

Q: How do I generate References: lines in my echomail replies?

A: See SAFE_THREADS and KEEP_THREADS.

Q: People are complaining that my MSGID fields are causing problems..

A: Turn off the KEEP_THREADS option. This option has proven safe formy area, but you may have someone that is extremely [expletive adj] on adhering the technicallity instead of the spirit of the specifications. Look instead into SAFE_THREADS.

Q: Why can’t GIGO see my netmail?

Q: Why can’t GIGO see my echomail?

A: GIGO only understands *.PKT files. If you can not convince your mail processor to generate *.PKT files, you may find that the “NETMGR” program is helpful. This is particularly true of people running FrontDoor ;-). NETMGR will scan your netmail directory, and convert any messages into *.PKT’s. See the section on "How to get mail from your mail processor, to GIGO"

Q: What is the structure of .REQ files for fileserv request?

A: The structure of the .REQ file is very similiar to a Binkleyflo file. The first line _must_ be the email address that the resulting messages should be sent to. The remaining lines can be comments that are sent to the user (lines starting with ";" are considered comments). Or, they may be file names to be sent. If the path is not specified, the FILESERV-OKFILES listing will be checked for the file. To delete the file afterwords, start the line with a # or ^ symbol.

Example .REQ file:

```
1: joe.schmuck@somewhere.org
2: ; This file has been created by the sysop
3: ; any questions, email joe@mysite.bbs.org ..
4: GIGO
5: e:\files\gateways\gigo0829.zip
6: #f:\bbs\outbound\00010414.mo1
7: ;
8: ; That’s all folks!
```

Line 1: email address Line 2,3,7,8: Sent to user in a message from fileserv. Line 4: Looks for a file or magic name called GIGO in your fileserv-okfiles listing. If found, sends it. Line 5: Absolute path name. Sends the designated file. Line 6: Absolute path name; mail bundle from fidonet. Deletes when done.

Q: How can I speed GIGO up?

A: Things you can do to speed GIGO up:

•If you load GIGO several times an hour, usage of INCLUDE statements is discouraged, as it takes longer to locate and read in all of your INCLUDE’d segments. Placing everything into one config file will help speed things up for you on the startup-time needed.

•Disable on the logging that you don’t need. If you have problems, reenable the logging that is appropriate for the job at hand. Logging _really_ slows things down. Don’t forget that you have multiple log files to play with..

•Cut down on the KEEP, KILL, and HIDE statements -only use what you need. They are all scanned once for every single fidonet message created. Using RMAIL_KEEP_ALL (and others like it) can speed things up significantly, by reducing the overall number of keep/kill/hide statements.

•If you don’t need the FIDOECHOS functionality, comment it out. Saves memory, and saves time converting newsgroups.

•Are you getting everthing from several heiarchies? Use wildcards instead! See the section on MAPPING.CFG for more information.

•Even thoug “+ newsgroup.name” and “= newsgroup.name”are (roughly) the same, use the “=newsgroup.name” method instead when possible. The “+” syntax tells GIGO to keep checking, for other mappings that might be more suitable. The “=” syntax says,‘hey, we’ve got the one we want, stop checking!’.

•If you are using PARTIAL-35 (Partial FSC-35), and don’t really need it, you can turn it off for a bit of a speed increase due to GIGO not having to actually compare the fido header and the FSC-35 extentions. (If you actually find this feature useful, however, it’s well worth the speed hit..)

•Turn off the video. NOVIDEO in the config, or /NOVIDEO on the commandline, will tell GIGO to turn off video output. The normal startup screens will still be shown, but once the gateway actually starts processing mail, the video gets turned off. My tests have shown this to speed things up by about 25%. Do this _after_ you’ve got the gateway up and running. Between this, and disabling your logging, you can really speed things up.

•Don’t use virtual memory unless you really need it!

Index

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X