GNU Queue

Load-balancing/batch-processing environment
and local rsh replacement
17 May 2000
Version 1.12.9

Werner G. Krebs

1 Introduction

GNU Queue is a UNIX process network load-balancing system that features an innovative proxy process mechanism which allows users to control their remote jobs in a nearly seamless and transparent fashion. When an interactive remote job is launched, such as say Matlab, or EMACS interfacing Allegro Lisp, a proxy process runs on the local end. (You can think of this being equivalent to a running 'telnet' or 'rsh' process, but more intelligent.) By sending signals to the local proxy - including hitting the suspend key - the process on the remote end may be controlled. Resuming the proxy process resumes the remote job. The user's environment is almost completely replicated, including not only environmental variables, but nice values, rlimits, terminal settings are all replicated on the remote end. Together with `MIT_MAGIC_COOKIE_1` (or `xhost +`) the system is X-windows transparent as well, provided the users local `DISPLAY` variable is set to the fully qualified pathname of the local machine.

One of the most appealing features of the proxy process system even with experienced users is that asynchronous job control of remote jobs by the shell is possible and intuitive. One simply runs the stub in the background under the local shell; the shell notifies the user when the remote job has a change in status by monitoring the stub daemon.

When the remote process has terminated, the proxy process returns the exit value to the shell; otherwise, the stub simulates a death by the same signal as that which terminated or suspended the remote job. In this way, control of the remote process is intuitive even to novice users, as it is just like controlling a local job from the shell. Many of my original users had to be reminded that their jobs were, in fact, running remotely.

In addition, Queue also features a more traditional distributed batch processing environment, with results returned to the user via email. In addition, traditional batch processing limitations may be placed on jobs running in either environment (stub or with the email mechanism) such as suspension of jobs if the system exceeds a certain load average, limits on CPU time, disk free requirements, limits on the times in which jobs may run, etc. (These are documented in the sample `profile` file included.)

Queue may be installed by any user on the system; root privileges are not required.

2 Installing Queue as an Ordinary User

Installing GNU Queue as an ordinary user is recommended only if you lack root (aka, superuser or Unix system administrative privileges) on your cluster.

You do not need to have system administrative privileges to install GNU Queue.

However, To allow all users in the cluster to use GNU Queue you should have your cluster's system administrator install Queue following the instructions in the chapter Install By Root. See Chapter 3 [Install By Root], page 5. However, if this is impractical, you may install Queue yourself without resorting to administrative superuser, or root, privileges by following the instructions in this chapter.

Note that, under its default configuration, GNU Queue supports only one installation per cluster, so if you install GNU Queue as an ordinary user you will be the only user able to run jobs through it. This can be overcome if another user edits GNU Queue's header files to change its network port numbers to avoid a conflict with another copy of GNU Queue running on the same cluster.

See Section 8.2 [Security for OUsers], page 17, on `-DHAVE_IDENTD` and running an RFC 931 `identd` service on cluster when installating GNU Queue as an ordinary user.

To do this, you will need write access to an NFS directory that is shared among all hosts in your cluster. In most cases, your system administrator will have set up your home directory this way.

Installing GNU Queue for one user:

1. Run `./configure` .

   When installing as an ordinary user, configure sets the makefile to install GNU Queue into the current directory. `queue` will go in `./bin`, `queued` daemon will go into `./sbin`, `/com/queue` will be the shared spool directory, the host access control list file will go into `./share` and the queued pid files will go into `./var` . If you want things to go somewhere else, run `./configure --prefix=dir`, where dir is the top-level directory where you want things to be installed.

   `./configure` takes a number of additional options that you may wish to be aware of, `./configure --help` gives a full listing of them. `--bindir` specifies where `queue` goes, `--sbindir` specifies where `queued` goes, `--sharedstatedir` where the spool directory goes, `--datadir` where the host access control file goes, and `--localstatefile` where the `queued` pid files go.

   If `./configure` fails inelegantly, make sure `lex` is installed. GNU `flex` is an implementation of lex available from the FSF, `http://www.gnu.org`.

2.

   Now run `make` to compile the programs.

3.

   If all goes well, `make install` will install the programs into directory you specified with `./configure`. Missing directories will be created.

   The name of the localhost `make install` is being run on will be added to the host access control list if it is not already there.

4.

   Try running Queue.

   Start up ./queued on the localmachine. (If you did a `make install` on the localhost the localhost should already be in the host access control list file.)

   `./queue --help` gives a list of options to Queue.

   Here are some simple examples:

   ```
   > queue -i -w -n -- hostname
   > queue -i -r -n -- hostname
   ```

   Here is a more sophisticated example. Try suspending and resuming it with Control-Z and 'fg':

   ```
   > queue -i -w -p -- emacs -nw
   ```

   If this example works on the localhost, you want want to add additional hosts to the host access control list in `share` (or `--datadir`) and start up queued on these.

   ```
   > queue -i -w -p -h hostname -- emacs -nw
   ```

   will run emacs on `hostname`. Without the `-h` argument, it will run the job on the `best` or `least-loaded` host in the ACL. See Chapter 6 [Profile], page 13, for details on how host selection is made.

You can also create additional queues for use with the `-q` and `-d` commands, as outlined for root users below. Each spooldir must have a `profile` file associated with it. See Chapter 6 [Profile], page 13, for details.

3 Installation of GNU Queue by System Administrator (Preferred)

If you want to just experiment with Queue on a single host, all you need is a local directory that is protected to be root-accessible only. For load-balancing, however, you will need an NFS directory mounted on all your hosts with 'no_root_squash' (see NFS man pages) option turned on. Unfortunately, the 'no_root_squash' option is required for load-balancing because the file system is used to communicate information about jobs to be run. The default spool directory is under the default GNU sharedstatedir, `/usr/local/com/queue`.

no_root_squash option is the GNU/Linux name. The option is named differently under different platforms, see your NFS man pages for the name of the option that prevents root mapping to nobody on client requests.

Installing GNU Queue for cluster-wide usage

1.

   Since non-administrators are generally less sophisticated than system administrators, the default `./configure` option is to install GNU Queue in the local directory for use by a single user only.

   System administrators need to specify `--enable-root` to reconfigure GNU to run with root privileges. This turns off some options, for example, privileged ports are used instead of relying on the `identd` (RFC 931) service if it is installed. See Chapter 8 [Security], page 17, for a discussion of security issues

   Run `./configure --enable-root` .

   When installing with the `--enable-root` option, configure sets the Makefile to install GNU Queue under the `/usr/local` prefix. `queue` will go in `/usr/local/bin`, `queued` daemon will go into `/usr/local/sbin`, `/usr/local/com/queue` will be the shared spool directory, the host access control list file will go into `/usr/local/share` and the queued pid files will go into `/usr/local/var` . If you want things to go somewhere else, run `./configure --enable-root --prefix=dir`, where dir is the top-level directory where you want things to be installed.

   `./configure --enable-root` takes a number of additional options that you may wish to be aware of, `./configure --help` gives a full listing of them. `--bindir` specifies where `queue` goes, `--sbindir` specifies where `queued` goes, `--sharedstatedir` where the spool directory goes, `--datadir` where the host access control file goes, and `--localstatefile` where the queued pid files go.

   If `./configure` fails inelegantly, make sure lex is installed. GNU Flex is an implementation of lex available from the FSF, http://www.gnu.org.

2.

   Now run `make` to compile the programs.

3.

   If all goes well, `make install` will install the programs into directory you specified with `./configure`. Missing directories will be created.

   The name of the localhost `make install` is being run on will be added to the host access control list if it is not already there.

4. Try running Queue.

   Start up ./queued on the localmachine. (If you did a `make install` on the localhost the localhost should already be in the host access control list file.)

   `./queue --help` gives a list of options to Queue.

   Here are some simple examples:

   ```
   > queue -i -w -n -- hostname
   > queue -i -r -n -- hostname
   ```

   Here is a more sophisticated example. Try suspending and resuming it with Control-Z and 'fg':

   ```
   > queue -i -w -p -- emacs -nw
   ```

   If this example works on the localhost, you want want to add additional hosts to the host access control list in `share` (or –datadir) and start up queued on these.

   ```
   > queue -i -w -p -h hostname -- emacs -nw
   ```

   will run emacs on `hostname`. Without the `-h` argument, it will run the job on the "best" or "least-loaded" host in the ACL. See Profile for details on how host selection is made. See Chapter 6 [Profile], page 13.

You can also create additional queues for use with the `-q` and `-d` commands, as outlined for users below. Each spooldir must have a `profile` file associated with it. See Chapter 6 [Profile], page 13, for details.

4 Setting Up Queue for Cluster-wide Usage

The GNU Queue system consists of two components, ‘queued’ which runs as a daemon on every host in the cluster, and ‘queue’ is a user program that allows users to submit jobs to the system.

The 'queue' binary contacts queued to learn the relative virtual load averages (explained in 'profile') on each host, and specifies one on which to run the job. Queued then forks off a process and works together with queue on the local end to control the remote job.

Look over the sample 'profile' file, See Chapter 6 [Profile], page 13, to learn how to customize batch queues and load balancing. 'profile' has many options. Among others, you can configure certain hosts to be submit-only hosts for all or only certain job classes by turning off job execution in these queues.

Add the name of each host in the cluster to the access control list. The default location for this is either `share/qhostsfile` or `/usr/local/share/qhostsfile` depending on how `./configure` was invoked.

Finally, if you are installing GNU Queue cluster-wide, make sure the spool directory (default is `/usr/local/com/queue`) is NFS exported root-writable on all systems in your cluster. In GNU/Linux, this is done by setting the `no_root_squash` option in `/etc/exports` (and then running `/usr/etc/exportfs` to cause the system to acknowlege the changes; if /usr/etc/exportfs is not available on your system, restart `nfsd` and the `portmapper`.)

Other operating system flavors have different names for this option. Read `nfs(4)`, `exports(4)` and other `man` pages for information on setting the `no_root_squash` equivalent on your operating system flavor.

5 Running queue

```
queue [-h hostname|-H hostname] [-i|-q] [-d spooldir]
      [-o|-p|-n] [-w|-r] -- command.options
qsh  [-l ignored] [-d spooldir] [-o|-p|-n]
     [-w|-r] hostname command command.options
```

-h hostname
--host hostname
force queue to run on hostname.

-H hostname
--robust-host hostname
Run job on hostname if it is up.

-i|-q
--immediate|--queue
Shorthand for the (`now` spooldir) and queue (`queue` spooldir).

[-d spooldir]
[--spooldir spooldir]
With `-q` option, specifies the name of the batch processing directory, e.g., `matlab`

-o|-p|-n
--half-pty|--full-pty|--no-pty
Toggle between half-pty emulation, full-pty emulation (default), and the more efficient no-pty emulation.

-w|-r
--wait|--batch
Toggle between wait (stub daemon; default) and return (mail batch) mode.

-v
--version
Version

--help List of options

The defaults for `qsh` are a slightly different: no-pty emulation is the default, and a hostname argument is required. A plus (`+`) is the wildcard hostname; specifying `+` in place of a valid hostname is the same as not using an `-h` or `-H` option with `queue`. `qsh` is envisioned as a `rsh` compatibility mode for use with software that expects a `rsh`-like syntax. This is useful with some MPI implementations; See Chapter 7 [MPI], page 15.

Start the Queue system on every system in your cluster (as you defined in queue.h) by running `queued` or `queued -D &` from the directory in which queued is installed.

The later invocation places queued in debug mode, with copious error messages and mailings, which is probably a good idea if you are having problems. Sending queued a `kill -HUP` will force it to re-read the profile files and ACL lists, which is good when you wish to shut down a queue or add hosts to the cluster. `queued` will also periodically check for modifications to these files.

If all has gone well at this stage, you may now try submitting a sample job to the system. I recommend trying something like `queue -i -w -p -- emacs -nw`. You should be able to background and foreground the remote EMACS process from the local shell just as if it were running as a local copy.

Another example command is `queue -i -w -- hostname` which should return the best host (i.e., least loaded, as controlled by options in the profile file; See Chapter 6 [Profile], page 13, to run a job on.

The options on queue need to be explained:

`-i` specifies immediate execution mode, placing the job in the `now` spool. This is the default. Alternatively, you may specify either the `-q` option, which is shorthand for the `wait` spool, or use the `-d spooldir` option to place the job under the control of the `profile` file in the `spooldir` subdirectory of the spool directory, which must previously have been created by the Queue administrator.

In any case, execution of the job will wait until it satisfies the conditions of the profile file for that particular spool directory, which may include waiting for a slot to become free. This method of batch processing is completely compatible with the stub mechanism, although it may disorient users to use it in this way as they may be unknowingly forced to wait until a slot on a remote machine becomes available.

`-w` activates the stub mechanism, which is the default. The queue stub process will terminate when the remote process terminates; you may send signals and suspend/resume the remote process by doing the same to the stub process. Standard input/output will be that of the 'queue' stub process. `-r` deactivates the stub process; standard input/output will be via email back to the users; the `queue` process will return immediately.

`-p` or `-n` specifies whether or not a virtual tty should be allocated at the remote end, or whether the system should merely use the more efficient socket mechanism. Many interactive processes, such as `EMACS` or `Matlab`, require a virtual tty to be present, so the `-p` option is required for these. Other processes, such as a simple `hostname` do not require a `tty` and so may be run without the default `-p`. Note that `queue` is intelligent and will override the `-p` option if it detects both `stdio/stdout` have been re-directed to a non-terminal; this feature is useful in facilitating system administration scripts that allow users to execute jobs. [At some point we may wish to change the default to `-p` as the system automatically detects when `-n` will suffice.] Simple, non-interactive jobs such as `hostname` do not need the less efficient pty/tty mechanism and so should be run with the `-n` option. The `-n` option is the default when `queue` is invoked in `rsh` compatibility mode with `qsh`.

The `--` with `queue` specifies 'end of queue options' and everything beyond this point is interpreted as the command, or arguments to be given to the command. Consequently, user options (i.e., when invoking queue through a script front end, may be placed here):

```
#!/bin/sh
exec queue -i -w -p -- sas $*
```

or

```
#!/bin/sh
exec queue -q -w -p -d sas -- sas $*
```

for example. This places queue in immediate mode following instructions in the `now` spool subdirectory (first example) or in batch-processing mode into the `sas` spool subdirectory, provided it has been created by the administrator. In both cases, stubs are being used, which will not terminate until the sas process terminates on the remote end.

In both cases, `pty/ttys` will be allocated, unless the user redirects both the standard input and standard output of the simple invoking scripts. Invoking queue through these scripts has the additional advantage that the process name will be that of the script, clarifying what is the process is. For example, the script might called `sas` or `sas.remote`, causing `queue` to appear this way in the user's process list.

`queue` can be used for batch processing by using the `-q -r -n` options, e.g.,

```
#!/bin/sh
exec queue -q -r -n -d sas -- sas $*
```

would run `SAS` in batch mode. `-q` and `-d sas` options force Queue to follow instructions in the `sas/profile` file under Queue's spool directory and wait for the next available job slot. `-r` activates batch-processing mode, causing Queue to exit immediately and return results (including stdout and stderr output) via email.

The final option, `-n`, is the option to disable allocation of a pty on the remote end; it is unnecessary in this case (as batch mode disables ptys anyway) but is here to demonstrate how it might be used in a `-i -w -n` or `-q -w -n` invocation.

6 Configure a Job Queue's profile File

Under `/usr/spool/queue` you may create several directories for batch jobs, each identified with the class of the batch job (e.g., `sas` or `splus`). You may then place restrictions on that class, such as maximum number of jobs running, or total CPU time, by placing a `profile` file like this one in that directory.

However, the `now` queue is mandatory; it is the directory used by the `-i` mode (immediate moe) of queue to launch jobs over the network immediately rather than as batch jobs.

Specify that this queue is turned on:

```
exec on
```

The next two lines in `profile` may be set to an email address rather than a file; the leading `/` identifies then as file logs. Files now beginning with `cf`,`of`, or `ef` are ignored by the queued:

```
mail /usr/local/com/queue/now/mail_log
supervisor /usr/local/com/queue/now/mail_log2
```

Note that `/usr/local/com/queue` is our spool directory, and `now` is the job batch directory for the special `now` queue (run via the `-i` or immediate-mode flag to the queue executable), so these files may reside in the job batch directories.

The `pfactor` command is used to control the likelihood of a job being executed on a given machine. Typically, this is done in conjunction with the `host` command, which specifies that the option on the rest of the line be honored on that host only.

In this example, `pfactor` is set to the relative MIPS of each machine, for example:

```
host fast_host pfactor 100
host slow_host pfactor  50
```

Where `fast_host` and `slow_host` are the hostnames of the respective machines.

This is useful for controlling load balancing. Each queue on each machine reports back an 'apparant load average' calculated as follows:

1-min load average/ ((max(0, vmaxexec - maxexec) + 1)*pfactor)

The machine with the lowest apparant load average for that queue is the one most likely to get the job.

Consequently, a more powerful `pfactor` proportionally reduces the load average that is reported back for this queue, indicating a more powerful system.

Vmaxexec is the "apparant maximum" number of jobs allowed to execute in this queue, or simply equal to maxexec if it was not set. The default value of these variables is large value treated by the system as infinity.

```
host fast_host vmaxexec 2
host slow_host vmaxexec 1
maxexec 3
```

The purpose of `vmaxexec` is to make the system appear fully loaded at some point before the maximum number of jobs are already running, so that the likelihood of the machine being used tapers off sharply after `vmaxexec` slots are filled.

Below `vmaxexec` jobs, the system aggressively discriminates against hosts already running jobs in this Queue.

In job queues running above `vmaxexec` jobs, hosts appear more equal to the system, and only the load average and `pfactor` is used to assign jobs. The theory here is that above `vmaxexec` jobs, the hosts are fully saturated, and the load average is a better indicator than the simple number of jobs running in a job queue of where to send the next job.

Thus, under lightly-loaded situations, the system routes jobs around hosts already running jobs in this job queue. In more heavily loaded situations, load-averages and `pfactor`s are used in determining where to run jobs.

Additional options in `profile`

`exec` on, off, or drain. Drain drains running jobs.

`minfree` disk space on specified device must be at least this free.

`maxfree` maximum number of jobs allowed to run in this queue.

`loadsched`
1 minute load average must be below this value to launch new jobs.

`loadstop` if 1 minute load average exceeds this, jobs in this queue are suspended until it drops again.

`timesched`
Jobs are only scheduled during these times

`timestop` Jobs running will be suspended outside of these times

`nice` Running jobs are at least at this nice value

`rlimitcpu`
maximum cpu time by a job in this queue

`rlimitdata`
maximum data memory size by a job

`rlimitstack`
maximum stack size

`rlimitfsize`
maximum fsize

`rlimitrss`
maximum resident portion size.

`rlimitcore`
maximum size of core dump

These options, if present, will only override the user's values (via queue) for these limits if they are lower than what the user has set (or larger in the case of `nice`).

7 Running GNU Queue with MPI and PVM.

Many MPI implementations (such as the free MPICH implementation) allow you to specify a replacment utility for rsh/remsh to propagate processes.

Just use `qsh` as the replacement. Be sure the `QHOSTSFILE` lists all hosts known to the MPI implementation, and the queued is running on them.

You have three options: place a `+` in the MPI hosts file for each job-slot you want MPI to be able to start, explicitly list Queue's hosts in the MPI host file, or use a combination of `+` wild-cards and explicitly listed hosts in MPI's host file.

The `+` is GNU Queue's wild-card character for the hostname when it is invoked using `qsh`. It simply means that Queue should decide what host the process should run on, which is the default behavior for Queue. Specifying a host instead of using the `+` with qsh is equivalent to the `-h` option with the regular queue command-line syntax.

By placing `+`s in the MPI host file, MPI will pass `+` as the name of the host for that job slot to GNU Queue, which, in turn, will decide where the job should actually run.

By running jobs through GNU Queue this way, GNU Queue becomes aware of jobs submitted by MPI, and can route non-MPI jobs around them. Normally, you would want to use a job queue (`-j` option) which has a low `vmaxexec` set and a high maxexec, so that MPI's jobs will continue to run, but GNU Queue will aggressively try to route jobs to other hosts the moment the job queue begins the fill.

GNU Queue's load scheduling algorithm is smarter than that of many MPI implementations, which frequently treat all hosts as equal and implement a round-robin algorithm for deciding which to host to run a job on. GNU Queue, on the other hand, can take load-averages, CPU power differences (via `profile` file specifiers), and other factors into account when deciding on which host to send a particular job to.

`qsh` represent a stage-1 hook for MPI. Our development team (See Section 9.2 [Getting Help], page 19, for information on joining the development team) is currently working on a stage-2 hook, in which MPI becomes aware of GNU Queue jobs as well, allowing them to work as an integrated scheduling team.

Support for PVM is currently in development as well.

8 Security Issues

8.1 Installation by System Administrator

Security is always a concern when granting root privileges to software.

I was security conscious and knowledgeable about UNIX security issues when I wrote queue. It should be paranoid in all the right places, at least provided that the spool directory is root-accessible only (standard installation) or user-accessible (installation by ordinary user) only.

Critical ports allow connections only by hosts in the access control list. Standard checks (TCP/IP wrapper-style) are made to prevent DNS spoofing and IP forwarding as much as possible. In addition, connections must be made from privileged ports (root installation version). `queue.c` and `queued.c` run with least-privileges, revoking root privileges as soon as they have verified information and acquired a privileged port.

Moreover, at the time of this writing the source code has been available for a number of months and has been used at numerous installations, including some concerned with security.

However, this does not guarantee that security holes do no exist. It is important that security-conscious users scrutinize the source code and report any potential security problems to `bug-queue@gnu.org`. By promptly reporting security issues you will be supporting free software by ensuring that the public availability of source code is a security asset.

8.2 Installation by Ordinary User

In this installation mode, GNU Queue takes many of the same precautions for these users as when it has been installed cluster-wide by a system administrator.

Unfortunately, when Queue is installed by an ordinary user, privileged ports are not available. This might make it possible for a malicious user already having a shell account on the same cluster to have `queued` or `queue` try to spoof each other.

To close this hole, Queue uses the one-way function `crypt(3)` and a cookie passed over NFS to allow `queued` and `queue` to authenticate each other. These cookies are used in the root version as well to prevent port confusion by `queued` trying to connect to a `queue` that has earlier died, although they aren't useful from a security standpoint.

When GNU Queue is compiled with the `-DHAVE_IDENTD` (and `-DNO_ROOT`), `queued` and `queue` also use the `identd` service (RFC 931) to prevent spoofing by checking the ownership of remote sockets within the cluster. For this to work proplery, `identd` must be running on all your cluster hosts, return accurate information (either the user's login name as given in the password file or his/her uid), and at least accept connections from within the cluster in a reasonable amount of time. The `./configure` script tries to set `-DHAVE_IDENTD` automatically based on whether or not your host accepts local connections to port 113, but some systems intentionally allow identd to output bogus information for privacy reasons, and `-DHAVE_IDENTD` should not be set on these; if this is the case, you may need to re-compile GNU Queue with `HAVE_IDENTD` undefined in `config.h`. Fortunately, `queue` will normally complain immediately if `-DHAVE_IDENTD` is set when it shouldn't be.

To get around the performance hit of calling the `crypt(3)`, one-way functions are not used if spoofing queued is impossible due to privilege ports (root installation) or authenticated ports (`HAVE_IDENTD` service), so running `identd` with GNU Queue or installing GNU Queue cluster-wide as root may offer a slight performance advantange. In sites which normally send user passwords over the network in cleartext it is not expected to substancial improve secure over the cookie passing mechanism, however.

These cookies are passed in plaintext, which means that a malicious user might be able to observe the NFS network traffic between the hosts and, having shell access on the cluster, might still be able to spoof `queue` or `queued`. Since most sites send UNIX account passwords over the network in cleartext as well, this is only of concern in very secure sites that do not pass passwords in cleartext over the network.

In the rare event that your site is a very secure site that does not send passwords in cleartext, and you are compiling Queue without root privileges, you should have your administrator install the `identd` (RFC 931) service and re-run `./configure` to ensure `HAVE_IDENTD` is defined in `config.h`. If your very secure site prefers to spoof `identd` for privacy reasons, your administrator may be able to restrict `identd` access with `tcp_wrapper` or install an accurate `identd` on a non-standard port which could restrict connections to within the cluster via `tcp_wrapper`. You would need to re-compile GNU Queue with this new port number set in `ident.c`. Another option is to have your system administrator install Queue cluster-wide; this uses privileged ports and therefore may operate securely with resorting to `identd`.

These concerns do not apply when Queue has been installed cluster-wide by root (`NO_ROOT` is not defined), because privileged ports are then available.

9 Sending Feedback and Getting Help

9.1 Feedback

PLEASE SEND US FEEDBACK ON QUEUE!

Whether you have a queue-tip, are queue-less about how to solve a problem, or simply have another bad queue joke, que[ue] us in at `bug-queue@gnu.org` and we'll take our que[ue] from you on how best to improve the software and documentation.

9.2 Getting Help

The application's homepage is `http://queue.sourceforge.net`.

Bug reports should be sent to the bug list `bug-queue@gnu.org`.

Users are encouraged to subscribe to and request assistance from the development list, 'queue-developers', as well.

At the time of this writing, the list was working on several fun projects, including improved MPI & PVM support, secure socket connections, AFS & Kerberos support. We're also porting and improving a nifty utility that lets you monitor and control the execution of Queue jobs throughout your cluster. The list is a great way to tap into the group's expertise and keep up with the latest developments.

So, come join the fun and keep up with the latest developments by visiting `http://lists.sourceforge.net/mailman/listinfo/queue-developers`.

It is also possible to subscribe from the application's homepage.

It is `http://queue.sourceforge.net`.

At the time of this writing, GNU Queue is being maintained by GNU Queue's primary author, Werner G. Krebs.

Appendix A GNU GENERAL PUBLIC LICENSE

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

   a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

   The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.
7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

   It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

   This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.
8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.
10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software

which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "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 THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
one line to give the program's name and an idea of what it does.
Copyright (C) 19yy  name of author

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type `show w'.  This is free software, and you are welcome
to redistribute it under certain conditions; type `show c'
for details.
```

The hypothetical commands '`show w`' and '`show c`' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than '`show w`' and '`show c`'; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright
interest in the program `Gnomovision'
(which makes passes at compilers) written
by James Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

Index

J

K

L

M

N

O

P

Q

R

S

Table of Contents