NAME

libguestfs−test−tool − Diagnostics for libguestfs

SYNOPSIS

```
libguestfs−test−tool [−−options]
```

DESCRIPTION

libguestfs-test-tool is a test program shipped with libguestfs to allow you to check basic libguestfs functionality is working. This is needed because libguestfs occasionally breaks for reasons beyond our control: usually because of changes in the underlying qemu or kernel packages, or the host environment.

If you suspect a problem in libguestfs, then just run:

```
libguestfs−test−tool
```

It will print lots of diagnostic messages.

If it runs to completion successfully, you will see this near the end:

```
===== TEST FINISHED OK =====
```

and the test tool will exit with code 0.

If it fails (and/or exits with non-zero error code), please paste the *complete, unedited* output of the test tool into a bug report. More information about reporting bugs can be found on the http://libguestfs.org/ website.

OPTIONS

−−help
Display short usage information and exit.

−−qemu qemu_binary
If you have downloaded another qemu binary, point this option at the full path of the binary to try it.

−−qemudir qemu_source_dir
If you have compiled qemu from source, point this option at the source directory to try it.

−t N
−−timeout N
Set the launch timeout to `N` seconds. The default is 600 seconds (10 minutes) which does not usually need to be adjusted.

−V
−−version
Display the libguestfs version number and exit.

TRYING OUT A DIFFERENT VERSION OF QEMU

If you have compiled another version of qemu from source and would like to try that, then you can use the *−−qemudir* option to point to the qemu source directory.

If you have downloaded a qemu binary from somewhere, use the *−−qemu* option to point to the binary.

Note when using these options, you can ignore the business of qemu wrapper scripts (‘‘QEMU WRAPPERS’’ in **guestfs** (3)), since libguestfs-test-tool writes a wrapper script for you if one is needed.

TRYING OUT A DIFFERENT KERNEL

You can tell supermin to try a different kernel. You do this by setting the environment variables `SUPERMIN_KERNEL`, `SUPERMIN_KERNEL_VERSION` and/or `SUPERMIN_MODULES`.

Refer to ‘‘ENVIRONMENT VARIABLES’’ in **supermin** (1) for further information.

TRYING OUT A DIFFERENT VERSION OF LIBVIRT

To find out which backend is the default in your libguestfs package, do:

```
unset LIBGUESTFS_BACKEND
guestfish get−backend
```

If you are using the libvirt backend, then you can try out a different (eg. upstream) version of libvirt by running these commands (*not* as root):

```
killall libvirtd lt-libvirtd
~/path/to/libvirt/run libguestfs-test-tool
```

The first command kills any session `libvirtd` process(es) that may be running on the machine. The second command uses libvirt's `run` script (in the top-level libvirt build directory) to set some environment variables so that the alternate version of libvirt is used to run the program.

TRYING OUT WITH / WITHOUT LIBVIRT

To find out which backend is the default in your libguestfs package, do:

```
unset LIBGUESTFS_BACKEND
guestfish get-backend
```

If you are using the libvirt backend, you can try without (ie. libguestfs directly launching qemu) by doing:

```
export LIBGUESTFS_BACKEND=direct
```

Or if you are using the default (direct) backend, then you can try libvirt:

```
export LIBGUESTFS_BACKEND=libvirt
```

or with libvirt and a specific libvirt URI:

```
export LIBGUESTFS_BACKEND=libvirt:qemu:///session
```

TRYING OUT DIFFERENT SELINUX SETTINGS

To find out which backend is the default in your libguestfs package, do:

```
unset LIBGUESTFS_BACKEND
guestfish get-backend
```

To find out if SELinux is being used, do:

```
getenforce
```

If you are using libvirt, SELinux and sVirt, then you can try to see if changing SELinux to ''permissive'' mode makes any difference. Use this command as root:

```
setenforce Permissive
```

If this makes a difference, look in the audit logs for recent failures (''AVCs''):

```
ausearch -m avc -ts recent
```

You can convert AVCs into suggested SELinux policy rules using tools like **audit2allow**(1). For more information, see the ''Security Enhanced Linux User Guide''.

To reenable SELinux and sVirt, do:

```
setenforce Enforcing
```

SELF-DIAGNOSIS

Refer to ''APPLIANCE BOOT PROCESS'' in **guestfs**(3) to understand the messages produced by libguestfs-test-tool and/or possible errors.

EXIT STATUS

libguestfs-test-tool returns *0* if the tests completed without error, or *1* if there was an error.

ENVIRONMENT VARIABLES

For the full list of environment variables which may affect libguestfs, please see the **guestfs**(3) manual page.

SEE ALSO

guestfs(3), http://libguestfs.org/, http://qemu.org/.

AUTHORS

Richard W.M. Jones (`rjones at redhat dot com`)

COPYRIGHT

LICENSE

BUGS

To get a list of bugs against libguestfs, use this link: https://bugzilla.redhat.com/buglist.cgi?component=libguestfs&product=Virtualization+Tools

To report a new bug against libguestfs, use this link: https://bugzilla.redhat.com/enter_bug.cgi?component=libguestfs&product=Virtualization+Tools

When reporting a bug, please supply:

- The version of libguestfs.
- Where you got libguestfs (eg. which Linux distro, compiled from source, etc)
- Describe the bug accurately and give a way to reproduce it.
- Run **libguestfs−test−tool** (1) and paste the **complete, unedited** output into the bug report.