NAME

guestfs−testing − manual testing of libguestfs, you can help!

DESCRIPTION

This page has manual tests you can try on libguestfs. Everyone has a slightly different combination of platform, hardware and guests, so this testing is very valuable. Thanks for helping out!

Tests marked with a ***** (asterisk) can **destroy data** if you're not careful. The others are safe and won't modify anything.

These tests **require libguestfs ≥ 1.22**.

You can report bugs you find through this link:

https://bugzilla.redhat.com/enter_bug.cgi?component=libguestfs&product=Virtualization+Tools

or post on the mailing list (registration is **not** required, but if you're not registered then you'll have to wait for a moderator to manually approve your message):

https://www.redhat.com/mailman/listinfo/libguestfs

TESTS

Run libguestfs-test-tool

Run:

```
libguestfs−test−tool
```

This command does a very simple, non-destructive test that basic libguestfs is functioning. You don't need to run it as root.

If it *doesn't* print `===== TEST FINISHED OK =====`, report it as a bug. It is very important that you include the **complete, unedited** output of `libguestfs−test−tool` in your bug report. See the "BUGS" section at the end of this page.

Check KVM acceleration is being used.

If your host has hardware virt acceleration, then with a hot cache libguestfs should be able to start up in a few seconds. Run the following command a few times:

```
time guestfish −a /dev/null run
```

After a few runs, the time should settle down to a few seconds (under 3 seconds on fast 64 bit hardware).

If the command above does not work at all, use **libguestfs−test−tool** (1).

Check which version of libguestfs, qemu, libvirt, etc is being used.

Look at the output of `libguestfs−test−tool` and check:

- Which version of libguestfs is being used? Near the beginning of the output you'll see a line like:

  ```
  library version: 1.22.0fedora=19,release=1.fc19,libvirt
  ```

- Is libvirt being used? You can tell the difference by looking for the backend:

  ```
  guestfs_get_backend: direct
  ```

  or:

  ```
  guestfs_get_backend: libvirt
  ```

- Which version of qemu is being used? It may be printed out:

  ```
  libguestfs: qemu version 1.5
  ```

- Which kernel is being used? **supermin** (1) will try to pick the latest k ernel installed on your machine. You can see the version in the appliance output, eg:

  ```
  [    0.000000] Linux version 3.9.2−200.fc18.x86_64 [...]
  ```

Try to open a local guest image with guestfish.

You can use any guest disk image for this test. Make sure you use the `--ro` flag so that **guestfish**(1) will open the disk image read-only.

```
guestfish --ro -a /path/to/disk.img -i
```

If the command is successful, it should print out the guest operating system name and put you at the guestfish `><fs>` prompt. You can use guestfish commands like `ll/` to look inside the disk image. To exit, type `exit`.

If you get an error, try enabling debugging (add `-v` to the command line). Also make sure that **libguestfs-test-tool**(1) succeeds.

Try to open a remote guest image with guestfish.

You may also have to disable libvirt by setting this:

```
export LIBGUESTFS_BACKEND=direct
```

If you have a disk image available over HTTP/FTP, try to open it.

```
guestfish --ro -i --format=raw -a http://www.example.com/disk.img
```

For SSH you will need to make sure that ssh-agent is set up so you don't need a password to log in to the remote machine. Then a command similar to this should work:

```
guestfish --ro -i --format=raw \
  -a ssh://remote.example.com/path/to/disk.img
```

If you get an error, try enabling debugging (add `-v` to the command line). Also make sure that **libguestfs-test-tool**(1) succeeds.

Run virt-alignment-scan on all your guests.

Run **virt-alignment-scan**(1) on guests or disk images:

```
virt-alignment-scan -a /path/to/disk.img
```

or:

```
virt-alignment-scan -d Guest
```

Does the alignment report match how the guest partitions are aligned?

Run virt-cat on some files in guests.

virt-cat(1) can display files from guests. For a Linux guest, try:

```
virt-cat LinuxGuest /etc/passwd
```

A recent feature is support for Windows paths, for example:

```
virt-cat WindowsGuest 'c:\windows\win.ini'
```

An even better test is if you have a Windows guest with multiple drives. Do`D:`, `E:` etc paths w ork correctly?

*** Copy some files into a shut off guest.**

virt-copy-in(1) can recursively copy files and directories into a guest or disk image.

```
virt-copy-in -d Guest /etc /tmp
```

This should copy local directory */etc* to */tmp/etc* in the guest (recursively). If you boot the guest, can you see all of the copied files and directories?

Shut the guest down and try copying multiple files and directories:

```
virt-copy-in -d Guest /home /etc/issue /tmp
```

Copy some files out of a guest.

virt-copy-out(1) can recursively copy files and directories out of a guest or disk image.

```
virt-copy-out -d Guest /home .
```

Note the final space and period in the command is not a typo.

This should copy */home* from the guest into the current directory.

Run virt-df.

virt-df (1) lists disk space. Run:

```
virt-df
```

You can try comparing this to the results from **df** (1) inside the guest, but there are some provisos:

- The guest must be idle.
- The guest disks must be synched using **sync** (1).
- Any action such as booting the guest will write log files causing the numbers to change.

We don't guarantee that the numbers will be identical even under these circumstances. They should be similar. It would indicate a bug if you saw greatly differing numbers.

Try importing virt-df CSV output into a spreadsheet or database.

Run:

```
virt-df --csv > /tmp/report.csv
```

Now try to load this into your favorite spreadsheet or database. Are the results reproduced faithfully in the spreadsheet/database?

http://www.postgresql.org/docs/8.1/static/sql-copy.html
http://dev.mysql.com/doc/refman/5.1/en/load-data.html

*** Edit a file in a shut off guest.**

virt-edit (1) can edit files in guests. Try this command on a RHEL or Fedora guest:

```
virt-edit LinuxGuest /etc/sysconfig/network
```

On other Linux guests try editing other files such as:

```
virt-edit LinuxGuest /etc/motd
```

Are the changes seen inside the guest when it is booted?

Display the filesystems / partitions / LVs in a guest.

virt-filesystems (1) can be used to display filesystems in a guest. Try this command on any disk image or guest:

```
virt-filesystems -a /path/to/disk.img --all --long -h
```

or:

```
virt-filesystems -d Guest --all --long -h
```

Do the results match what is seen in the guest?

Run virt-inspector on all your guests.

Use **virt-inspector** (1) to get a report on all of your guests or disk images:

```
virt-inspector -a /path/to/disk.img | less
```

or:

```
virt-inspector -d Guest | less
```

Do the results match what is actually in the guest?

If you have an unusual guest (a rare Linux distro, a very new version of Windows), does virt-inspector recognize it? If not, then it's probably a bug.

Try the auditing features of virt-ls on all your guests.

List all setuid or setgid programs in a Linux virtual machine:

```
virt-ls -lR -d Guest / | grep '^- [42]'
```

List all public-writable directories in a Linux virtual machine:

```
virt-ls -lR -d Guest / | grep '^d ...7'
```

List all Unix domain sockets in a Linux virtual machine:

```
virt-ls -lR -d Guest / | grep '^s'
```

List all regular files with filenames ending in '.png':

```
virt-ls -lR -d Guest / | grep -i '^-.*\.png$'
```

Display files larger than 10MB in home directories:

```
virt-ls -lR -d Guest /home | awk '$3 > 10*1024*1024'
```

Find everything modified in the last 7 days:

```
virt-ls -lR -d Guest --time-days / | awk '$6 <= 7'
```

Find regular files modified in the last 24 hours:

```
virt-ls -lR -d Guest --time-days / | grep '^-' | awk '$6 < 1'
```

Do the results match what is in the guest?

Create a disk image from a tarball.

Use **virt-make-fs** (1) to create a disk image from any tarball that you happen to have:

```
virt-make-fs --partition=mbr --type=vfat /any/tarball.tar.gz output.img
```

Add 'output.img' as a raw disk to an existing guest. Check the guest can see the files. This test is particularly useful if you try it with a Windows guest.

Try other partitioning schemes, eg. *--partition=gpt*.

Try other filesystem formats, eg. *--type=ntfs*, *--type=ext2*.

*** Run virt-rescue on a shut off disk image or guest.**

Use **virt-rescue** (1) to examine, rescue or repair a **shut off** guest or disk image:

```
virt-rescue -a /path/to/disk.img
```

or:

```
virt-rescue -d Guest
```

Can you use ordinary shell commands to examine the guest?

*** Resize your guests.**

Use **virt-resize** (1) to give a guest some more disk space. For example, if you have a disk image that is smaller than 30G, increase it to 30G by doing:

```
truncate -s 30G newdisk.img
virt-filesystems -a /path/to/olddisk.img --all --long -h
virt-resize /path/to/olddisk.img newdisk.img --expand /dev/sda1
qemu-kvm -m 1024 -hda newdisk.img
```

Does the guest still boot? Try expanding other partitions.

*** Sparsify a guest disk.**

Using **virt-sparsify** (1), make a disk image more sparse:

```
virt-sparsify /path/to/olddisk.img newdisk.img
```

Is *newdisk.img* still bootable after sparsifying? Is the resulting disk image smaller (use `du` to check)?

Build and boot a guest

Using **virt-builder** (1), choose a guest from the list:

```
virt-builder -l
```

build it:

```
virt-builder -o disk.img [os-version from list above]
```

and boot it:

```
qemu-kvm -cpu host -m 2048 -drive file=disk.img,format=raw
```

Does it boot?

*** "Sysprep" a shut off Linux guest.**

Note that this really will mess up an existing guest, so it's better to clone the guest before trying this.

```
virt-sysprep --hostname newhost.example.com -a /path/to/disk.img
```

Was the sysprep successful? After booting, what changes were made and were they successful?

Dump the Windows Registry from your Windows guests.

Use **virt-win-reg** (1) to dump out the Windows Registry from any Windows guests that you have.

```
virt-win-reg --unsafe-printable-strings WindowsGuest 'HKLM\Software' |
  less

virt-win-reg --unsafe-printable-strings WindowsGuest 'HKLM\System' |
  less
```

Does the output match running `regedit` inside the guest?

A recent feature is the ability to dump user registries, so try this, replacing *username* with the name of a local user in the guest:

```
virt-win-reg --unsafe-printable-strings WindowsGuest 'HKEY_USERS\username' |
  less
```

SEE ALSO

guestfs (3), **guestfish** (1), **guestfs-examples** (3), http://libguestfs.org/.

AUTHORS

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

COPYRIGHT

Copyright (C) 2011-2012 Red Hat Inc.

LICENSE

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

This library 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 Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

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.