NAME

virt−rescue − Run a rescue shell on a virtual machine

SYNOPSIS

```
virt−rescue [−−options] −d domname

virt−rescue [−−options] −a disk.img [−a disk.img ...] [−i]
```

Old style:

```
virt−rescue [−−options] domname

virt−rescue [−−options] disk.img [disk.img ...]
```

WARNING

Using `virt−rescue` in write mode on live virtual machines, or concurrently with other disk editing tools, can be dangerous, potentially causing disk corruption. The virtual machine must be shut down before you use this command, and disk images must not be edited concurrently.

Use the *−−ro* (read-only) option to use `virt−rescue` safely if the disk image or virtual machine might be live. You may see strange or inconsistent results if running concurrently with other changes, but with this option you won't risk disk corruption.

DESCRIPTION

virt-rescue is like a Rescue CD, but for virtual machines, and without the need for a CD. virt-rescue gives you a rescue shell and some simple recovery tools which you can use to examine or rescue a virtual machine or disk image.

You can run virt-rescue on any virtual machine known to libvirt, or directly on disk image(s):

```
virt−rescue −d GuestName −i

virt−rescue −−ro −a /path/to/disk.img −i

virt−rescue −a /dev/sdc
```

For live VMs you *must* use the *−−ro* option.

When you run virt-rescue on a virtual machine or disk image, you are placed in an interactive bash shell where you can use many ordinary Linux commands. What you see in / (*/bin*, */lib* etc) is the rescue appliance. You must mount the virtual machine's filesystems. There is an empty directory called */sysroot* where you can mount filesystems.

To automatically mount the virtual machine's filesystems under */sysroot* use the *−i* option. This uses libguestfs inspection to find the filesystems and mount them in the right place. You can also mount filesystems individually using the *−m* option.

Another way is to list the logical volumes (with **lvs** (8)) and partitions (with **parted** (8)) and mount them by hand:

```
><rescue> lvs
LV      VG        Attr   LSize   Origin Snap%  Move Log Copy%  Convert
lv_root vg_f15x32 −wi−a−   8.83G
lv_swap vg_f15x32 −wi−a− 992.00M
><rescue> mount /dev/vg_f15x32/lv_root /sysroot
><rescue> mount /dev/vda1 /sysroot/boot
><rescue> ls /sysroot
```

Another command to list available filesystems is **virt−filesystems** (1).

To run commands in a Linux guest (for example, grub), you should chroot into the /sysroot directory first:

```
><rescue> chroot /sysroot
```

NOTES

Virt-rescue can be used on *any* disk image file or device, not just a virtual machine. For example you can use it on a blank file if you want to partition that file (although we would recommend using **guestfish**(1) instead as it is more suitable for this purpose). You can even use virt-rescue on things like USB drives, SD cards and hard disks.

You can get virt-rescue to give you scratch disk(s) to play with. This is useful for testing out Linux utilities (see *--scratch*).

Virt-rescue does not require root. You only need to run it as root if you need root to open the disk image.

This tool is just designed for quick interactive hacking on a virtual machine. For more structured access to a virtual machine disk image, you should use **guestfs**(3). To get a structured shell that you can use to make scripted changes to guests, use **guestfish**(1).

OPTIONS

--help
Display brief help.

-a FILE
--add FILE
Add `FILE` which should be a disk image from a virtual machine. If the virtual machine has multiple block devices, you must supply all of them with separate *-a* options.

The format of the disk image is auto-detected. To override this and force a particular format use the *--format=*.. option.

-a URI
--add URI
Add a remote disk. See ''ADDING REMOTE STORAGE'' in **guestfish**(1).

--append KERNELOPTS
Pass additional options to the rescue kernel.

--blocksize=512
--blocksize=4096
--blocksize
This parameter sets the sector size of the disk image. It affects all explicitly added subsequent disks after this parameter. Using*--bloc ksize* with no argument switches the disk sector size to the default value which is usually 512 bytes. See also ''guestfs_add_drive_opts'' in **guestfs**(3).

-c URI
--connect URI
If using libvirt, connect to the given *URI*. If omitted, then we connect to the default libvirt hypervisor.

If you specify guest block devices directly (*-a*), then libvirt is not used at all.

-d guest
--domain guest
Add all the disks from the named libvirt guest. Domain UUIDs can be used instead of names.

-e none
Disable the escape key.

-e KEY
Set the escape key to the given key sequence. The default is `^]`. To specify the escape key you can use:

`^x` Control key + `x` key.

`none`
-e none means there is no escape key, escapes are disabled.

See ''ESCAPE KEY'' below for further information.

−−format=raw|qcow2|..
−−format
The default for the *−a* option is to auto-detect the format of the disk image. Using this forces the disk format for *−a* options which follow on the command line. Using *−−format* with no argument switches back to auto-detection for subsequent *−a* options.

For example:

```
virt−rescue −−format=raw −a disk.img
```

forces raw format (no auto-detection) for *disk.img*.

```
virt−rescue −−format=raw −a disk.img −−format −a another.img
```

forces raw format (no auto-detection) for *disk.img* and reverts to auto-detection for *another.img*.

If you have untrusted raw-format guest disk images, you should use this option to specify the disk format. This avoids a possible security problem with malicious guests (CVE−2010−3851).

−i
−−inspector
Using **virt−inspector** (1) code, inspect the disks looking for an operating system and mount filesystems as they would be mounted on the real virtual machine.

The filesystems are mounted on */sysroot* in the rescue environment.

−−memsize MB
Change the amount of memory allocated to the rescue system. The default is set by libguestfs and is small but adequate for running system tools. The occasional program might need more memory. The parameter is specified in megabytes.

−m dev[:mountpoint[:options[:fstype]]]
−−mount dev[:mountpoint[:options[:fstype]]]
Mount the named partition or logical volume on the given mountpoint **in the guest** (this has nothing to do with mountpoints in the host).

If the mountpoint is omitted, it defaults to /. You have to mount something on /.

The filesystems are mounted under */sysroot* in the rescue environment.

The third (and rarely used) part of the mount parameter is the list of mount options used to mount the underlying filesystem. If this is not given, then the mount options are either the empty string or `ro` (the latter if the *−−ro* flag is used). By specifying the mount options, you override this default choice. Probably the only time you would use this is to enable ACLs and/or extended attributes if the filesystem can support them:

```
−m /dev/sda1:/:acl,user_xattr
```

The fourth part of the parameter is the filesystem driver to use, such as `ext3` or `ntfs`. This is rarely needed, but can be useful if multiple drivers are valid for a filesystem (eg: `ext2` and `ext3`), or if libguestfs misidentifies a filesystem.

−−network
Enable QEMU user networking in the guest. See ‘‘NETWORK’’.

−r
−−ro
Open the image read-only.

The option must always be used if the disk image or virtual machine might be running, and is generally recommended in cases where you don’t need write access to the disk.

See also ‘‘OPENING DISKS FOR READ AND WRITE’’ in **guestfish** (1).

--scratch
--scratch=N
The *--scratch* option adds a large scratch disk to the rescue appliance. *--scr atch=N* adds `N` scratch disks. The scratch disk(s) are deleted automatically when virt-rescue exits.

You can also mix *-a*, *-d* and *--scratch* options. The scratch disk(s) are added to the appliance in the order they appear on the command line.

--selinux
This option is provided for backwards compatibility and does nothing.

--smp N
Enable N ≥ 2 virtual CPUs in the rescue appliance.

--suggest
This option was used in older versions of virt-rescue to suggest what commands you could use to mount filesystems under */sysroot*. For the current version of virt-rescue, it is easier to use the *-i* option instead.

This option implies *--ro* and is safe to use even if the guest is up or if another virt-rescue is running.

-v
--verbose
Enable verbose messages for debugging.

-V
--version
Display version number and exit.

-w
--rw
This changes the *-a*, *-d* and *-m* options so that disks are added and mounts are done read-write.

See ‘‘OPENING DISKS FOR READ AND WRITE’’ in **guestfish** (1).

-x Enable tracing of libguestfs API calls.

OLD-STYLE COMMAND LINE ARGUMENTS

Previous versions of virt-rescue allowed you to write either:

```
virt-rescue disk.img [disk.img ...]
```

or

```
virt-rescue guestname
```

whereas in this version you should use *-a* or *-d* respectively to avoid the confusing case where a disk image might have the same name as a guest.

For compatibility the old style is still supported.

NETWORK

Adding the *--network* option enables QEMU user networking in the rescue appliance. There are some differences between user networking and ordinary networking:

ping does not work
Because the ICMP ECHO_REQUEST protocol generally requires root in order to send the ping packets, and because virt-rescue must be able to run as non-root, QEMU user networking is not able to emulate the **ping** (8) command. The ping command will appear to resolve addresses but will not be able to send or receive any packets. This does not mean that the network is not working.

cannot receive connections
QEMU user networking cannot receive incoming connections.

making TCP connections

The virt-rescue appliance needs to be small and so does not include many network tools. In particular there is no **telnet**(1) command. You can make TCP connections from the shell using the magical */dev/tcp/<hostname>/<port>* syntax:

```
exec 3<>/dev/tcp/redhat.com/80
echo "GET /" >&3
cat <&3
```

See **bash**(1) for more details.

ESCAPE KEY

Virt-rescue supports various keyboard escape sequences which are entered by pressing `^]` (Control key + `]` key).

You can change the escape key using the *-e* option on the command line (see above), and you can disable escapes completely using *-e none*. The rest of this section assumes the default escape key.

The following escapes can be used:

`^] ?`
`^] h`
Prints a brief help text about escape sequences.

`^] i`
Prints brief libguestfs inspection information for the guest. This only works if you used *-i* on the virt-rescue command line.

`^] q`
`^] x`
Quits virt-rescue immediately.

`^] s`
Synchronize the filesystems (sync).

`^] u`
Unmounts all the filesystems, except for the root (appliance) filesystems.

`^] z`
Suspend virt-rescue (like pressing `^Z` except that it affects virt-rescue rather than the program inside the rescue shell).

`^] ^]`
Sends the literal character `^]` (ASCII 0x1d) through to the rescue shell.

CAPTURING CORE DUMPS

If you are testing a tool inside virt-rescue and the tool (**not** virt-rescue) segfaults, it can be tricky to capture the core dump outside virt-rescue for later analysis. This section describes one way to do this.

1. Create a scratch disk for core dumps:

```
truncate -s 4G /tmp/corefiles
virt-format --partition=mbr --filesystem=ext2 -a /tmp/corefiles
virt-filesystems -a /tmp/corefiles --all --long -h
```

2. When starting virt-rescue, attach the core files disk last:

```
virt-rescue --rw [-a ...] -a /tmp/corefiles
```

NB. If you use the *--ro* option, then virt-rescue will silently not write any core files to */tmp/corefiles*.

3. Inside virt-rescue, mount the core files disk. Note replace */dev/sdb1* with the last disk index. For example if the core files disk is the last of four disks, you would use */dev/sdd1*.

```
><rescue> mkdir /tmp/mnt
><rescue> mount /dev/sdb1 /tmp/mnt
```

4. Enable core dumps in the rescue kernel:

```
><rescue> echo '/tmp/mnt/core.%p' > /proc/sys/kernel/core_pattern
><rescue> ulimit -Hc unlimited
><rescue> ulimit -Sc unlimited
```

5. Run the tool that caused the core dump. The core dump will be written to */tmp/mnt/core.PID*.

```
><rescue> ls -l /tmp/mnt
total 1628
-rw------- 1 root root 1941504 Dec  7 13:13 core.130
drwx------ 2 root root   16384 Dec  7 13:00 lost+found
```

6. Before exiting virt-rescue, unmount (or at least sync) the disks:

```
><rescue> umount /tmp/mnt
><rescue> exit
```

7. Outside virt-rescue, the core dump(s) can be removed from the disk using **guestfish**(1). For example:

```
guestfish --ro -a /tmp/corefiles -m /dev/sda1
><fs> ll /
><fs> download /core.NNN /tmp/core.NNN
```

ENVIRONMENT VARIABLES

Several environment variables affect virt-rescue. See ''ENVIRONMENT VARIABLES'' in **guestfs**(3) for the complete list.

FILES

`$XDG_CONFIG_HOME`/libguestfs/libguestfs-tools.conf
`$HOME`/.libguestfs-tools.rc
`$XDG_CONFIG_DIRS`/libguestfs/libguestfs-tools.conf
/etc/libguestfs-tools.conf

This configuration file controls the default read-only or read-write mode (*--ro* or *--rw*).

See **libguestfs-tools.conf**(5).

SEE ALSO

guestfs(3), **guestfish**(1), **virt-cat**(1), **virt-edit**(1), **virt-filesystems**(1), **libguestfs-tools.conf**(5), http://libguestfs.org/.

AUTHOR

Richard W.M. Jones http://people.redhat.com/~rjones/

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.