ORACLE

Setup VirtIO drivers on Linux VM

May 2023, Version 1.1

Sharath Kumar
EMEA Senior Cloud Specialist Engineer

This step is needed if the exported VM is intended to launch in Paravirtualized mode on OCI prior Customer bring their on-premises Linux VM's.

Without this step the Linux dracut can't find an appropriate boot disk and the boot process fails with the messages like this:

```
dracut-initqueue[504]: Warning: dracut-initqueue timeout - starting timeout scripts
dracut-initqueue[504]: Warning: dracut-initqueue timeout - starting timeout scripts
dracut-initqueue[504]: Warning: dracut-initqueue timeout - starting timeout scripts
....
dracut-initqueue[504]: Warning: could not boot.
dracut-initqueue[504]: Warning: /dev/disk/by-id/md-uuid-2fdc509e:8dd05ed3:c2350cb4:ea5a620d do
Starting Dracut Emergency Shell...
Warning: /dev/disk/by-id/md-uuid-2fdc509e:8dd05ed3:c2350cb4:ea5a620d does not exist
```

In the most cases the needed virtio/qemu drivers or modules are presented in the kernel but sometimes it may occur that some of them are absent. And in this case it's needed to check it and add the missed drivers/modules.

To run "Paravirtualized" Instances on OCI, add paravirtualized device support (virtio_blk and virtio_scsi) by building the virtio drivers into the VM's initrd/initramfs.

Because this action works only on machines with a Linux kernel of version 3.4 or later, check that the system is running a modern kernel:

uname -a

Check virtio modules/drivers

Run "lspci |grep Virtio"

```
# sudo lspci |grep Virtio
00:03.0 Ethernet controller: Red Hat, Inc. Virtio network device
```

Run "lsmod|grep virtio"

```
# sudo lsmod|grep virtio
virtio_net 53248 0
net_failover 20480 1 virtio_net
virtio_scsi 24576 2
```

This is the possible output for the commands, but it may differ from output on user machine. Here the 2 VirtIO modules SCSI and Net are used.

Run "lsinitrd|less"

find the line "qemu" under part "dracut modules":

```
dracut modules:
bash
systemd
systemd-initrd
nss-softokn
rngd
i18n
network-legacy
network
ifcfg
drm
qemu
plymouth
prefixdevname
kernel-modules
kernel-modules-extra
kernel-network-modules
resume
rootfs-block
terminfo
udev-rules
biosdevname
dracut-systemd
usrmount
base
fs-lib
microcode_ctl-fw_dir_override
shutdown
```

If there is no "qemu" you have to do the steps below.

Add the qemu module

There are two general ways to do this: add qemu support using the tool dracut or editing the file /etc/dracut.conf.

Add the module qemu using the tool "dracut"

```
# dracut --logfile /var/log/Dracut.log --force --add qemu
```

Edit the file /etc/dracut.conf

Add the line aka "add_dracutmodules+=qemu" into the file /etc/dracut.conf:

```
# cat /etc/dracut.conf
# PUT YOUR CONFIG IN separate files
# in /etc/dracut.conf.d named ".conf"
# SEE man dracut.conf(5) for options
#omit_dracutmodules+=plymouth
add_dracutmodules+=qemu
```

Finally, the initrd must be rebuilt to pick up new setting after editing the file:

```
# dracut --logfile /var/log/Dracut.log --force
```

Verify the virtio drivers are present

Run again lsinitrd

```
# lsinitrd |grep qemu
```

Now user should see the output line with "qemu".

Setup qemu on Ubuntu\Debian

Compile VirtIO device support into the kernel

To compile VirtIO device support into the kernel, place the following lines in your kernel configuration file (usually it's /boot/config-* file):

```
CONFIG_VIRTIO=y
CONFIG_VIRTIO_BLK=y
CONFIG_VIRTIO_BLK_SCSI=y
CONFIG_SCSI_VIRTIO=y
```

And recompile the kernel.

In case of EFI system, to load VirtIO support as modules at boot time (without recompiling the kernel), place the following lines in loader.conf :

```
virtio_load="YES"
virtio_blk_load="YES"
virtio_scsi_load="YES"
```

Create initramfs file

This approach is the same one as described above for RHEL. If the tool dracut is absent then install it via "apt install dracut-core". Next repeat the steps for RHEL/OL.