Using the RAM disk block device with Linux

1) Overview

The RAM disk driver is a way to use main system memory as a block device. It is required for initrd, an initial filesystem used if you need to load modules in order to access the root filesystem (see Documentation/admin-guide/initrd.rst). It can also be used for a temporary filesystem for crypto work, since the contents are erased on reboot.

The RAM disk dynamically grows as more space is required. It does this by using RAM from the buffer cache. The driver marks the buffers it is using as dirty so that the VM subsystem does not try to reclaim them later.

The RAM disk supports up to 16 RAM disks by default, and can be reconfigured to support an unlimited number of RAM disks (at your own risk). Just change the configuration symbol BLK_DEV_RAM_COUNT in the Block drivers config menu and (re)build the kernel.

To use RAM disk support with your system, run './MAKEDEV ram' from the /dev directory. RAM disks are all major number 1, and start with minor number 0 for /dev/ram0, etc. If used, modern kernels use /dev/ram0 for an initrd.

The new RAM disk also has the ability to load compressed RAM disk images, allowing one to squeeze more programs onto an average installation or rescue floppy disk.

2) Parameters

2a) Kernel Command Line Parameters

ramdisk_size=N
: Size of the ramdisk.

This parameter tells the RAM disk driver to set up RAM disks of N k size. The default is 4096 (4 MB).

2b) Module parameters

rd_nr
: /dev/ramX devices created.

max_part
: Maximum partition number.

rd_size
: See ramdisk_size.

3) Using "rdev"

"rdev" is an obsolete, deprecated, antiquated utility that could be used to set the boot device in a Linux kernel image.

Instead of using rdev, just place the boot device information on the kernel command line and pass it to the kernel from the bootloader.

You can also pass arguments to the kernel by setting FDARGS in arch/x86/boot/Makefile and specify in initrd image by setting FDINITRD in arch/x86/boot/Makefile.

Some of the kernel command line boot options that may apply here are:

```
ramdisk_start=N
ramdisk_size=M
```

If you make a boot disk that has LILO, then for the above, you would use:

```
append = "ramdisk_start=N ramdisk_size=M"
```

4) An Example of Creating a Compressed RAM Disk

To create a RAM disk image, you will need a spare block device to construct it on. This can be the RAM disk device itself, or an unused disk partition (such as an unmounted swap partition). For this example, we will use the RAM disk device, "/dev/ram0".

Note: This technique should not be done on a machine with less than 8 MB of RAM. If using a spare disk partition instead of /dev/ram0, then this restriction does not apply.

a. Decide on the RAM disk size that you want. Say 2 MB for this example. Create it by writing to the RAM disk device. (This step is not currently required, but may be in the future.) It is wise to zero out the area (esp. for disks) so that maximal compression is achieved for the unused blocks of the image that you are about to create:

    ```
    dd if=/dev/zero of=/dev/ram0 bs=1k count=2048
    ```

b. Make a filesystem on it. Say ext2fs for this example:

```
mke2fs -vm0 /dev/ram0 2048
```

c. Mount it, copy the files you want to it (eg: /etc/* /dev/* ...) and unmount it again.

d. Compress the contents of the RAM disk. The level of compression will be approximately 50% of the space used by the files. Unused space on the RAM disk will compress to almost nothing:

```
dd if=/dev/ram0 bs=1k count=2048 | gzip -v9 > /tmp/ram_image.gz
```

e. Put the kernel onto the floppy:

```
dd if=zImage of=/dev/fd0 bs=1k
```

f. Put the RAM disk image onto the floppy, after the kernel. Use an offset that is slightly larger than the kernel, so that you can put another (possibly larger) kernel onto the same floppy later without overlapping the RAM disk image. An offset of 400 kB for kernels about 350 kB in size would be reasonable. Make sure offset+size of ram_image.gz is not larger than the total space on your floppy (usually 1440 kB):

```
dd if=/tmp/ram_image.gz of=/dev/fd0 bs=1k seek=400
```

g. Make sure that you have already specified the boot information in FDARGS and FDINITRD or that you use a bootloader to pass kernel command line boot options to the kernel.

That is it. You now have your boot/root compressed RAM disk floppy. Some users may wish to combine steps (d) and (f) by using a pipe.

> Paul Gortmaker 12/95

Changelog:

SEPT-2020 :

> Removed usage of "rdev"

10-22-04 :
Updated to reflect changes in command line options, remove obsolete references, general cleanup. James Nelson (james4765@gmail.com)

12-95 :
Original Document