

Tutorial Link <https://course.testpad.chitkara.edu.in/tutorials/Disk Management in Linux II/62e8c040f90d2d345f891cce>

TUTORIAL

Disk Management in Linux II

Topics

- 1.1 How to Add Swap on CentOS & RHEL 8/7/6
- 1.2 How to Use 'fsck' to Repair File System Errors in Linux
- 1.3 When to Use fsck in Linux
- 1.4 fsck Available options
- 1.5 How to Run fsck to Repair Linux File System Errors
- 1.6 Understanding fsck exit codes
- 1.7 Repair Linux Filesystem Errors
- 1.8 Backup using dd command

How to Add Swap on CentOS & RHEL 8/7/6

It is a good practice to create swap on system during operating system installation. But in some cases if you don't have enabled swap on your system during installation and you are required to enable swap on system.

Swap is very useful for that system which required more RAM than physical available. If memory is full and system required more RAM to run applications properly it checks for swap space and transfers files there. In general terms swap is a part of hard disk used as RAM on system.

- Check System Swap – Before working make sure that system has already swap enabled. If there are no swap, you will get output header only.

```
swapon -s
```

- Create Swap File – Let's create a file to use for swap in system of required size. Before making file make sure you have enough free space on disk. Generally it recommends that swap should be equal to double of installed physical memory.
- If you have 4 GB physical RAM installed in your system, you can create swap of 8 GB in size.

```
fallocate -l 8G /swapfile  
chmod 600 /swapfile
```

- Make It Swap – Now make the swap usable file using mkswap command.

```
mkswap /swapfile
```

- Enable Swap – Now setup the swap for system using swapon command.

```
swapon /swapfile
```

- Now again check that swap is enabled or not. You will see results something like below.

```
sudo swapon -s
```

Filename	Type	Size	Used	Priority
/swapfile	file	8187150	0	-1

- Setup Swap Permanent – Append the following entry in /etc/fstab file to enable swap on system reboot.

```
vim /etc/fstab
/swapfile none swap sw 0 0
```

- Setup Kernel Parameter – Now change the swappiness kernel parameter as per your requirement. It tells the system how often system utilize this swap area.
- Edit /etc/sysctl.conf file and append following configuration in file.

```
vim /etc/sysctl.conf
vm.swappiness=10
```

- Now reload the sysctl configuration file

```
sysctl -p
```

At this point you have successfully enabled swap on your CentOS or RHEL system.

How to Use 'fsck' to Repair File System Errors in Linux

Filesystems are responsible for organizing how data is stored and recovered. One way or another, with time, filesystem may become corrupted and certain parts of it may not be accessible. If your filesystem develops such inconsistency it is recommend to verify its integrity.

This can be completed via system utility called **fsck (file system consistency check)**. This check can be done automatically during boot time or ran manually.

When to Use fsck in Linux

There are different scenarios when you will want to run fsck. Here are few examples:

- The system fails to boot.
- Files on the system become corrupt (often you may see input/output error).
- Attached drive (including flash drives/SD cards) is not working as expected.

fsck Available options

Fsck command needs to be run with superuser privileges or **root**. You can use it with different arguments. Their usage depend on your specific case. Below you will see some of the more important options:

- **-A** – Used for checking all filesystems. The list is taken from [/etc/fstab](#).
- **-C** – Show progress bar.
- **-l** – Locks the device to guarantee no other program will try to use the partition during the check.
- **-M** – Do not check mounted filesystems.
- **-N** – Only show what would be done – no actual changes are made.
- **-P** – If you want to check filesystems in parallel, including root.
- **-R** – Do not check root filesystem. This is useful only with '**-A**'.
- **-r** – Provide statistics for each device that is being checked.
- **-T** – Does not show the title.
- **-t** – Exclusively specify the filesystem types to be checked. Types can be comma separated list.

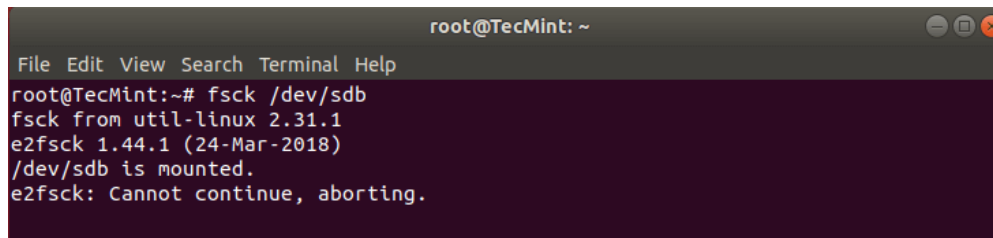
- -v – Provide description what is being done.

How to Run fsck to Repair Linux File System Errors

In order to run **fsck**, you will need to ensure that the partition you are going to check is not mounted. For the purpose of this article, I will use my second drive `/dev/sdb` mounted in `/mnt`.

Here is what happens if I try to run **fsck** when the partition is mounted.

```
# fsck /dev/sdb
```

A terminal window titled 'root@TecMint: ~' showing the command 'fsck /dev/sdb' being executed. The output indicates that the partition is already mounted and fsck cannot continue.

```
root@TecMint:~# fsck /dev/sdb
fsck from util-linux 2.31.1
e2fsck 1.44.1 (24-Mar-2018)
/dev/sdb is mounted.
e2fsck: Cannot continue, aborting.
```

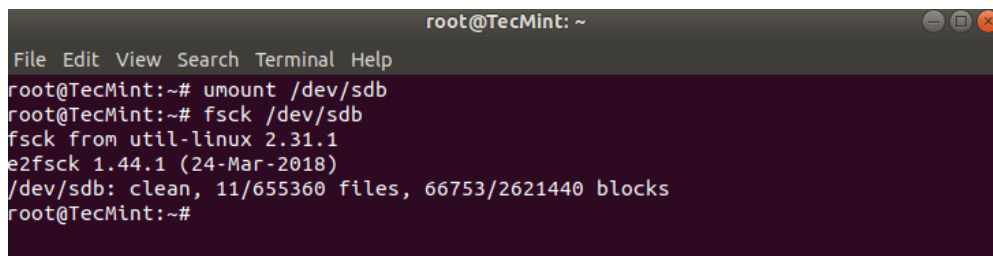
Run fsck on Mounted Partition

To avoid this unmount the partition using.

```
# umount /dev/sdb
```

Then **fsck** can be safely ran with.

```
# fsck /dev/sdb
```

A terminal window titled 'root@TecMint: ~' showing the sequence of commands to unmount the partition and then run fsck. The output shows the partition is clean.

```
root@TecMint:~# umount /dev/sdb
root@TecMint:~# fsck /dev/sdb
fsck from util-linux 2.31.1
e2fsck 1.44.1 (24-Mar-2018)
/dev/sdb: clean, 11/655360 files, 66753/2621440 blocks
root@TecMint:~#
```

Run fsck on Linux Partition

Understanding fsck exit codes

After running **fsck**, it will return an exit code. These codes can be seen in fsck's manual by running:

```
# man fsck

0      No errors
1      Filesystem errors corrected
2      System should be rebooted
4      Filesystem errors left uncorrected
8      Operational error
16     Usage or syntax error
32     Checking canceled by user request
128    Shared-library error
```

Repair Linux Filesystem Errors

Sometimes more than one error can be found on a filesystem. In such cases you may want **fsck** to automatically attempt to correct the errors. This can be done with:

```
# fsck -y /dev/sdb
```

The **-y** flag, automatically “yes” to any prompts from fsck to correct an error.

Similarly, you can run the same on all filesystems (without **root**):

```
$ fsck -AR -y
```

How to Run fsck on Linux Root Partition

In some cases, you may need to run **fsck** on the **root** partition of your system. Since you cannot run **fsck** while the partition is mounted, you can try one of these options:

- Force fsck upon system boot
- Run fsck in rescue mode

We will review both situations.

Force fsck Upon System Boot

This is relatively easy to complete, the only thing you need to do is create a file called **forcefsck** in the root partition of your system. Use the following command:

```
# touch /forcefsck
```

Then you can simply force or schedule a reboot of your system. During the next bootup, the **fsck** will be performed. If downtime is critical, it is recommended to plan this carefully, since if there are many used inodes on your system, **fsck** may take some extra time.

After your system boots, check if the file still exists:

```
# ls /forcefsck
```

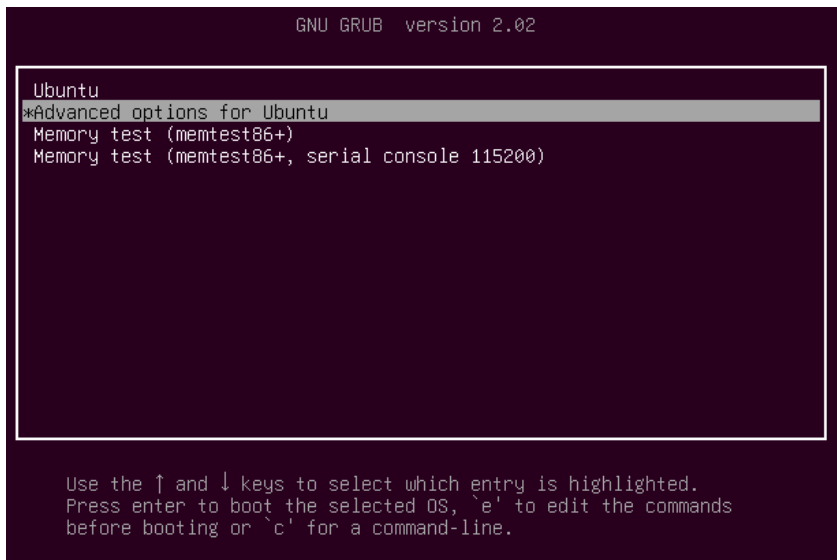
If it does, you may want to remove it in order to avoid **fsck** on every system boot.

Run fsck in Rescue Mode

Running **fsck** in **rescue mode** requires few more steps. First prepare your system for **reboot**. Stop any critical services like **MySQL/MariaDB** etc and then type.

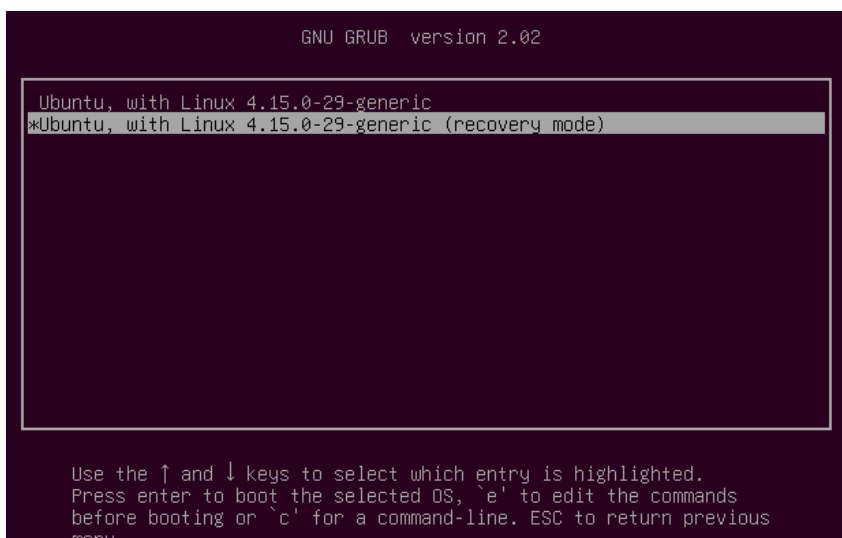
```
# reboot
```

During the boot, hold down the **shift** key so that the grub menu is shown. Select the “**Advanced options**”.



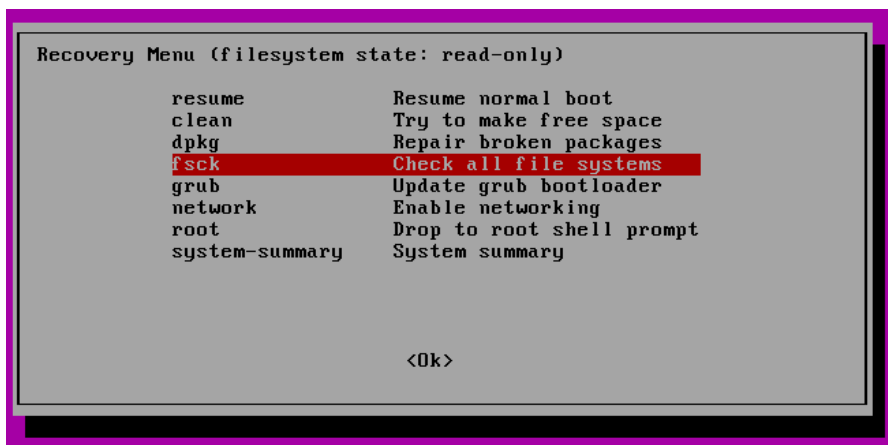
Grub Advance Options

Then choose “**Recovery mode**”.



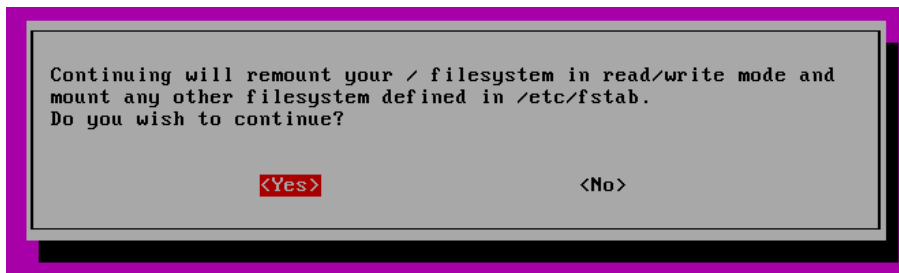
Select Linux Recovery Mode

In the next menu select “**fsck**”.



Select fsck Utility

You will be asked if you wish to have your / filesystem remounted. Select "yes".



Confirm Root Filesystem

You should see something similar to this.

```

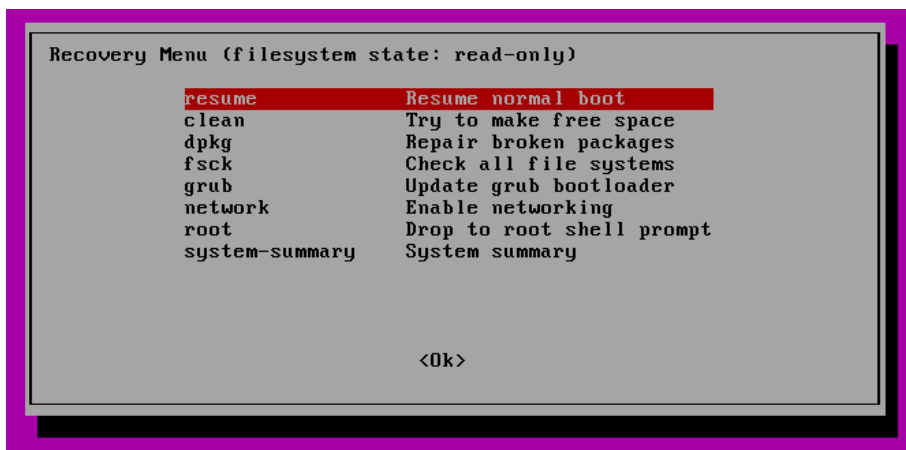
/lib/recovery-mode/recovery-menu: line 75: /etc/default/rcS: No such file or dir
ectory
fsck from util-linux 2.31.1
/dev/sda1: clean, 141762/786432 files, 1300329/3145216 blocks
Starting Remount Root and Kernel File Systems...
[ OK ] Started Remount Root and Kernel File Systems.
Starting Load/Save Random Seed...
Starting Flush Journal to Persistent Storage...
Activating swap /swapfile...
Starting udev Kernel Device Manager...

Finished, please press ENTER
[ OK ] Started Load/Save Random Seed.
[ OK ] Started Flush Journal to Persistent Storage.
[ OK ] Activated swap /swapfile.
[ OK ] Reached target Swap.
[ OK ] Started udev Kernel Device Manager.
[ OK ] Started Dispatch Password Requests to Console Directory Watch.
[ OK ] Reached target Local Encrypted Volumes.
[ OK ] Listening on Load/Save RF Kill Switch Status /dev/rfkill Watch.
[ OK ] Reached target Sound Card.

```

Running fsck Filesystem Check

You can then resume to normal boot, by selecting "Resume".



Select Normal Boot

Backup using dd command

dd is a command-line utility for Unix and Linux operating systems whose primary purpose is to convert and copy files.

On Linux, device drivers for hardware (such as hard disk drives) and special device files (such as `/dev/zero` and `/dev/random`) appear in the file system just like normal files. `dd` can also read and/or write from/to these files, provided that function is implemented in their respective drivers. As a result, `dd` can be used for tasks such as backing up the boot sector of a hard drive, and obtaining a fixed amount of random data. The `dd` program can also perform conversions on the data as it is copied, including byte order swapping and conversion to and from the ASCII and EBCDIC text encodings.

Some practical examples on dd command :

1. **To backup the entire Hard Disk:** To backup an entire copy of a hard disk to another hard disk connected to the same system, execute the `dd` command as shown. In this `dd` command example, the Linux device name of the source hard disk is `/dev/hda`, and device name of the target hard disk is `/dev/hdb`.

```
# dd if=/dev/sda of=/dev/sdb
```

- "if" represents input file, and "of" represents output file. So the exact copy of `/dev/sda` will be available in `/dev/sdb`.
- If there are any errors, the above command will fail. If you give the parameter `"conv=noerror"` then it will continue to copy if there are read errors.
- Input file and output file should be mentioned very carefully. Just in case, you mention source device in the target and vice versa, you might lose all your data.

- To copy, hard drive to hard drive using dd command given below, sync option allows you to copy everything using synchronized I/O.

```
# dd if=/dev/sda of=/dev/sdb conv=noerror, sync
```

2. **To backup a Partition :** You can use the device name of a partition in the input file, and in the output either you can specify your target path or image file

```
# dd if=/dev/hda1 of=~/partition.img
```

3. **To create an image of a Hard Disk :** Instead of taking a backup of the hard disk, you can create an image file of the hard disk and save it in other storage devices.

There are many advantages of backing up your data to a disk image, one being the ease of use. This method is typically faster than other types of backups,

enabling you to quickly restore data following an unexpected catastrophe. It creates the image of a hard disk /dev/hda.

```
# dd if=/dev/hda of=~/hdadisk.img
```

4. **To restore using the Hard Disk Image :** To restore a hard disk with the image file of an another hard disk, the following dd command can be used

```
# dd if=hdadisk.img of=/dev/hdb
```

The image file hdadisk.img file, is the image of a /dev/hda, so the above command will restore the image of /dev/hda to /dev/hdb.

5. **To create CDROM Backup :** dd command allows you to create an iso file from a source file. So we can insert the CD and enter dd command to create an iso file of

a CD content.

```
# dd if=/dev/cdrom of=tgsservice.iso bs=2048
```

dd command reads one block of input and process it and writes it into an output file. You can specify the block size for input and output file. In the above dd command

example, the parameter "bs" specifies the block size for the both the input and output file. So dd uses **2048bytes** as a block size in the above command.



CodeQuotient

Tutorial by codequotient.com | All rights reserved, CodeQuotient 2025