### PRACTICAL 6

**AIM:** File System

* 1. List the file systems that are known by yoursystems.
  2. Create an ext2 file systems on the 200MBpartition.
  3. Create an ext3 file systems on one of the 300MB logicaldrives.
  4. Create an ext4 on the 400MBpartition.
  5. Set the reserved space for root on the ext3 file system to 0percent.
  6. Verify your work with fdisk anddf.
  7. Perform a file system check on all the new filesystems.

### Theory:

A file system is a way of organizing files on your partition. Besides file-based storage, file systems usually include directories and access control, and contain meta information about files like access times, modification times and file ownership.

The properties (length, character set, ...) of filenames are determined by the file system you choose. Directories are usually implemented as files, you will have to learn how this is implemented! Access control in file systems is tracked by user ownership (and group owner- and membership) in combination with one or more access control lists.

A Linux file system is a structured collection of files on a disk drive or a partition. A partition is a segment of memory and contains some specific data. In our machine, there can be various partitions of the memory. Generally, every partition contains a file system.

The general-purpose computer system needs to store data systematically so that we can easily access the files in less time. It stores the data on hard disks (HDD) or

some equivalent storage type. There may be below reasons for maintaining the file system:

* Primarily the computer saves data to the RAM storage; it may lose the data if it gets turned off. However, there is non-volatile RAM (Flash RAM and SSD) that is available to maintain the data after the powerinterruption.
* Data storage is preferred on hard drives as compared to standard RAMas RAM costs more than disk space. The hard disks costs are dropping gradually comparatively theRAM.

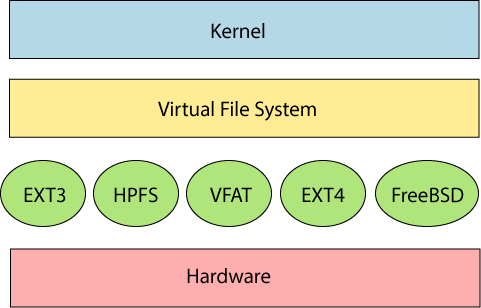
The [Linux](https://www.javatpoint.com/linux-tutorial) file system contains the following sections:

* The root directory(/)
* A specific data storage format (EXT3, EXT4, BTRFS, XFS and soon)
* A partition or logical volume having a particular filesystem.

**Linux File System Structure:**

* Linux file system has a hierarchal file structure as it contains a rootdirectory and its subdirectories. All other directories can be accessed from the root directory. A partition usually has only one file system, but it may have more than one filesystem.
* A file system is designed in a way so that it can manage and provide space for non-volatile storage data. All file systems required a namespace that is a naming and organizational methodology. The namespace defines the naming process, length of the file name, or a subset of characters that can be usedfor the file name. It also defines the logical structure of files on a memory segment, such as the use of directories for organizing the specific files. Once a namespace is described, a Metadata description must be defined for that particularfile.
* The data structure needs to support a hierarchical directory structure; this structure is used to describe the available and used disk space for aparticular block. It also has the other details about the files such as file size, date & time of creation, update, and lastmodified.
* Also, it stores advanced information about the section of the disk, suchas partitions andvolumes.
* The advanced data and the structures that it represents contain the information about the file system stored on the drive; it is distinctand independent of the file systemmetadata.

Linux file system contains two-part file system software implementation architecture. Consider the below image:



The file system requires an API (Application programming interface) to access the function calls to interact with file system components like files and directories.

[API](https://www.javatpoint.com/api-full-form)facilitates tasks such as creating, deleting, and copying the files. It facilitates an algorithm that defines the arrangement of files on a file system.

The first two parts of the given file system together called a **Linux virtual file system**. It provides a single set of commands for the kernel and developers to

access the file system. This virtual file system requires the specific system driver to give an interface to the file system.

**Linux File System Features**

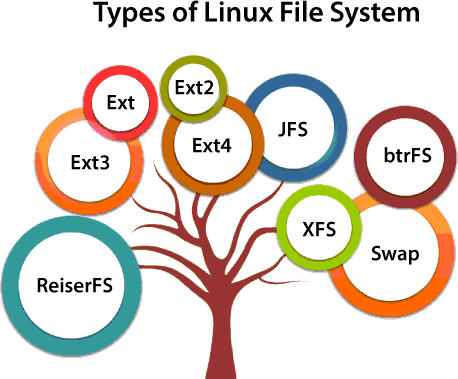
In Linux, the file system creates a tree structure. All the files are arranged as a tree and its branches. The topmost directory called the **root (/) directory**. All other directories in Linux can be accessed from the root directory.

Some key [features of Linux](https://www.javatpoint.com/linux-features) file system are as following:

* **Specifying paths:** Linux does not use the backslash (\) to separate the components; it uses forward slash (/) as an alternative. For example, as in Windows, the data may be stored in C:\ My Documents\ Work, whereas, in Linux, it would be stored in /home/ My Document/Work.
* **Partition, Directories, and Drives:** Linux does not use drive letters to organize the drive as Windows does. In Linux, we cannot tell whether we are addressing a partition, a network device, or an "ordinary" directory and aDrive.
* **Case Sensitivity:** Linux file system is case sensitive. It distinguishes between lowercase and uppercase file names. Such as, there is a difference between test.txt and Test.txt in Linux. This rule is also applied for directories and Linuxcommands.
* **File Extensions:** In Linux, a file may have the extension '.txt,' but it is not necessary that a file should have a file extension. While working withShell, it creates some problems for the beginners to differentiate between files and directories. If we use the graphical file manager, it symbolizes the files andfolders.
* **Hidden files:** Linux distinguishes between standard files and hidden files, mostly the configuration files are hidden in Linux OS. Usually, we don't need to access or read the hidden files. The hidden files in Linux are represented by a dot (.) before the file name (e.g., .ignore). To accessthe files, we need to change the view in the file manager or need to use a specific command in theshell.

**Types of Linux File System**

When we install the Linux operating system, Linux offers many file systems such as **Ext, Ext2, Ext3, Ext4, JFS, ReiserFS, XFS, btrfs,** and **swap**.



Let's understand each of these file systems in detail:

1. **Ext, Ext2, Ext3 and Ext4 filesystem**

The file system Ext stands for **Extended File System**. It was primarily developed for **MINIX OS**. The Ext file system is an older version, and is no longer useddue to somelimitations.

**Ext2** is the first Linux file system that allows managing two terabytes of data.Ext3 is developed through Ext2; it is an upgraded version of Ext2 and contains backward compatibility. The major drawback of Ext3 is that it does not support servers because this file system does not support file recovery and disksnapshot.

**Ext4** file system is the faster file system among all the Ext file systems. It is a very compatible option for the SSD (solid-state drive) disks, and it is the default file system in Linux distribution.

1. **JFS FileSystem**

JFS stands for **Journaled File System**, and it is developed by **IBM for AIX Unix**. It is an alternative to the Ext file system. It can also be used in place of Ext4, where

stability is needed with few resources. It is a handy file system when [CPU](https://www.javatpoint.com/cpu-full-form) power is limited.

1. **ReiserFS FileSystem**

ReiserFS is an alternative to the Ext3 file system. It has improved performance and advanced features. In the earlier time, the ReiserFS was used as the default file system in SUSE Linux, but later it has changed some policies, so SUSE returned to Ext3. This file system dynamically supports the file extension, but it has some drawbacks in performance.

1. **XFS FileSystem**

XFS file system was considered as high-speed JFS, which is developed for parallel I/O processing. NASA still using this file system with its high storage server (300+ Terabyte server).

1. **Btrfs FileSystem**

Btrfs stands for the **B tree file system**. It is used for fault tolerance, repair system, fun administration, extensive storage configuration, and more. It is not a goodsuit for the productionsystem.

1. **Swap FileSystem**

The swap file system is used for memory paging in Linux operating system during the system hibernation. A system that never goes in hibernate state is required to have swap space equal to its [RAM](https://www.javatpoint.com/ram-full-form) size.

**Commands for file systems:-**



**man fs**

The manual page about filesystems is accessed by typing **man fs**.

[root@rhel65 ~]# man fs

### /proc/filesystems

The Linux kernel will inform you about currently loaded file system drivers in

### /proc/filesystems.

root@rhel53 ~#cat/proc/filesystems | grep -v nodev ext2

iso9660 ext3

### /etc/filesystems

The **/etc/filesystems** file contains a list of autodetected filesystems (in case the

**mount** command is used without the **-t** option. Help for this file is provided by **man mount**. [root@rhel65 ~]# man mount **common file systems**

### ext2 and ext3

Once the most common Linux file systems is the **ext2** (the second extended) file system. A disadvantage is that file system checks on ext2 can take a long time.

**ext2** was being replaced by **ext3** on most Linux machines. They are essentially the same, except for the **journaling** which is only present in ext3.

Journaling means that changes are first written to a journal on the disk. Thejournal is flushed regularly, writing the changes in the file system. Journaling keeps the file system in a consistent state, so you don't need a file system check after an unclean shutdown or powerfailure.

### creating ext2 and ext3

You can create these file systems with the **/sbin/mkfs** or **/sbin/mke2fs** commands. Use **mke2fs -j** to create an **ext3** file system.

You can convert an ext2 to ext3 with **tune2fs -j**. You can mount an ext3 file system as ext2, but then you lose the journaling. Do not forget to run **mkinitrd** if you are booting from this device.

### ext4

The newest incarnation of the ext file system is named **ext4** and is available in the Linux kernel since 2008. **ext4** supports larger files (up to 16 terabyte) and larger file systems than **ext3** (and many more features).

Development started by making **ext3** fully capable for 64-bit. When it turned out the changes were significant, the developers decided to name it **ext4**.

### xfs

Redhat Enterprise Linux 7 will have **XFS** as the default file system. This is a highly scalable high-performance file system.

**xfs** was created for **Irix** and for a couple of years it was also used in **FreeBSD**. It is supported by the Linux kernel, but rarely used in dsitributions outside of the Redhat/CentOS realm.

### vfat

The **vfat** file system exists in a couple of forms : **fat12** for floppy disks, **fat16** on **ms-dos**, and **fat32** for larger disks. The Linux **vfat** implementation supports all of these, but vfat lacks a lot of features like security and links. **fat** disks can be read by every operating system, and are used a lot for digital cameras, **usb** sticks and to exchange data between different OS'ses on a home user's computer.

### iso 9660

**iso 9660** is the standard format for cdroms. Chances are you will encounter this file system also on your hard disk in the form of images of cdroms (often with the .iso extension). The **iso 9660** standard limits filenames to the 8.3 format. The Unix world didn't like this, and thus added the **rock ridge** extensions, which allows for filenames up to 255 characters and Unix-style file-modes, ownership and symbolic links. Another extensions to **iso 9660** is **joliet**, which adds 64 unicode characters to

the filename. The **el torito** standard extends **iso 9660** to be able to boot from CD- ROM's.

### udf

Most optical media today (including cd's and dvd's) use **udf**, the Universal Disk Format.

### swap

All things considered, swap is not a file system. But to use a partition as a **swap partition** it must be formatted and mounted as swap space.

### gfs

Linux clusters often use a dedicated cluster filesystem like GFS, GFS2, ClusterFS,

...

### and more...

You may encounter **reiserfs** on older Linux systems. Maybe you will see Sun's **zfs**

or the open source **btrfs**. This last one requires a chapter on itself.

### /proc/filesystems

The **/proc/filesystems** file displays a list of supported file systems. When you mount a file system without explicitly defining one, then mount will first try to probe **/etc/filesystems** and then probe **/proc/filesystems** for all the filesystems without the **nodev** label. If **/etc/filesystems** ends with a line containing only an asterisk (\*) then both files are probed.

paul@RHELv4u4:~$ cat/proc/filesystems nodev sysfs

nodev rootfs

nodev bdev

nodev proc

nodev sockfs nodev binfmt\_misc nodev usbfs nodev usbdevfs nodev futexfs nodev tmpfs

nodev pipefs

nodev eventpollfs nodev devpts

ext2

nodev ramfs nodev hugetlbfs

iso9660

nodev relayfs

nodev mqueue nodev selinuxfs

ext3 nodev rpc\_pipefs

nodev vmware-hgfs nodev autofs paul@RHELv4u4:~$

### putting a file system on a partition

We now have a fresh partition. The system binaries to make file systems can be found with ls.

[root@RHEL4b ~]# ls -lS /sbin/mk\*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| -rwxr-xr-x 3root | root | 34832 | Apr | 24 | 2006 | /sbin/mke2fs |
| -rwxr-xr-x 3root | root | 34832 | Apr | 24 | 2006 |  |
| /sbin/mkfs.ext2 |  |  |  |  |  |  |
| -rwxr-xr-x 3root | root | 34832 | Apr | 24 | 2006 |  |
| /sbin/mkfs.ext3 |  |  |  |  |  |  |
| -rwxr-xr-x 3root | root | 28484 | Oct | 13 | 2004 |  |
| /sbin/mkdosfs |  |  |  |  |  |  |
| -rwxr-xr-x 3root | root | 28484 | Oct | 13 | 2004 |  |
| /sbin/mkfs.msdos |  |  |  |  |  |  |
| -rwxr-xr-x 3root | root | 28484 | Oct | 13 | 2004 |  |
| /sbin/mkfs.vfat |  |  |  |  |  |  |
| -rwxr-xr-x 1root | root | 20313 | Apr | 10 | 2006 |  |
| /sbin/mkinitrd |  |  |  |  |  |  |
| -rwxr-x--- 1root | root | 15444 | Oct | 5 | 2004 |  |
| /sbin/mkzonedb |  |  |  |  |  |  |
| -rwxr-xr-x 1root | root | 15300 | May | 24 | 2006 |  |
| /sbin/mkfs.cramfs |  |  |  |  |  |  |
| -rwxr-xr-x 1root | root | 13036 | May | 24 | 2006 | /sbin/mkswap |
| -rwxr-xr-x 1root | root | 6912 | May | 24 | 2006 | /sbin/mkfs |
| -rwxr-xr-x 1root | root | 5905 | Aug | 3 | 2004 |  |

/sbin/mkbootdisk

[root@RHEL4b ~]#

It is time for you to read the manual pages of **mkfs** and **mke2fs**. In the example below, you see the creation of an **ext2 file system** on /dev/sdb1. In real life, you might want to use options like -m0 and -j.

root@RHELv4u2:~# mke2fs /dev/sdb1 mke2fs 1.35 (28-Feb-2004) Filesystem label=

OS type: Linux

Block size=1024 (log=0) Fragment size=1024 (log=0) 28112 inodes, 112420 blocks

5621 blocks (5.00%) reserved for the super user First data block=1

Maximum filesystem blocks=67371008

14 block groups

8192 blocks per group, 8192 fragments per group 2008 inodes per group

Superblock backups stored on blocks:

8193, 24577, 40961, 57345, 73729

Writing inode tables: done

Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 37 mounts or

180 days, whichevercomesfirst. Use tune2fs -c or -i tooverride.

### tuning a file system

You can use **tune2fs** to list and set file system settings. The first screenshotlists the reserved space for root (which is set at fivepercent).

[root@rhel4 ~]# tune2fs -l /dev/sda1 | grep -i "block count"

Blockcount: 104388

Reservedblockcount: 5219

[root@rhel4 ~]#

This example changes this value to ten percent. You can use tune2fs while the file system is active, even if it is the root file system (as in this example).

[root@rhel4 ~]# tune2fs -m10 /dev/sda1 tune2fs 1.35 (28-Feb-2004)

Setting reserved blocks percentage to 10 (10430 blocks) [root@rhel4 ~]# tune2fs -l /dev/sda1 | grep -i "block count"

Blockcount: 104388

Reservedblockcount: 10430

[root@rhel4 ~]#

### checking a file system

The **fsck** command is a front end tool used to check a file system for errors.

[root@RHEL4b ~]# ls /sbin/\*fsck\*

/sbin/dosfsck /sbin/fsck /sbin/fsck.ext2

/sbin/fsck.msdos

/sbin/e2fsck /sbin/fsck.cramfs /sbin/fsck.ext3

/sbin/fsck.vfat [root@RHEL4b ~]#

The last column in **/etc/fstab** is used to determine whether a file system should be checked at boot-up.

[paul@RHEL4b ~]$ grep ext /etc/fstab

|  |  |  |
| --- | --- | --- |
| /dev/VolGroup00/LogVol00  defaults 1 1 | / | ext3 |
| LABEL=/boot | /boot | ext3 |
| defaults 12 |  |  |
| [paul@RHEL4b ~]$ |  |  |

Manually checking a mounted file system results in a warning from fsck.

[root@RHEL4b ~]# fsck /boot fsck 1.35 (28-Feb-2004)

e2fsck 1.35 (28-Feb-2004)

/dev/sda1 is mounted.

WARNING!!! Running e2fsck on a mounted filesystemmay cause

SEVERE filesystem damage.

Do you really want to continue (y/n)? no check aborted.

But after unmounting fsck and **e2fsck** can be used to check an ext2 file system.

[root@RHEL4b~]#fsck /boot fsck 1.35(28-Feb-2004)

e2fsck 1.35 (28-Feb-2004)

/boot: clean, 44/26104 files, 17598/104388 blocks [root@RHEL4b ~]# fsck -p /boot

fsck 1.35 (28-Feb-2004)

/boot: clean, 44/26104 files, 17598/104388 blocks [root@RHEL4b ~]# e2fsck -p /dev/sda1

/boot: clean, 44/26104 files, 17598/104388 blocks

**File System Outputs:-**

1. **List the filesystems that are known by yoursystem.**

man fs

cat /proc/filesystems

cat /etc/filesystems (not on all Linux distributions)

1. **Create an ext2 filesystem on the 200MBpartition.**

mke2fs /dev/sdc1 (replace sdc1 with the correct partition)

1. **Create an ext3 filesystem on one of the 300MB logicaldrives.**

mke2fs -j /dev/sdb5 (replace sdb5 with the correct partition)

1. **Create an ext4 on the 400MBpartition.**

mkfs.ext4 /dev/sdb1 (replace sdb1 with the correct partition)

1. **Set the reserved space for root on the ext3 filesystem to 0percent.**

tune2fs -m 0 /dev/sdb5

1. **Verify your work with fdisk anddf.**

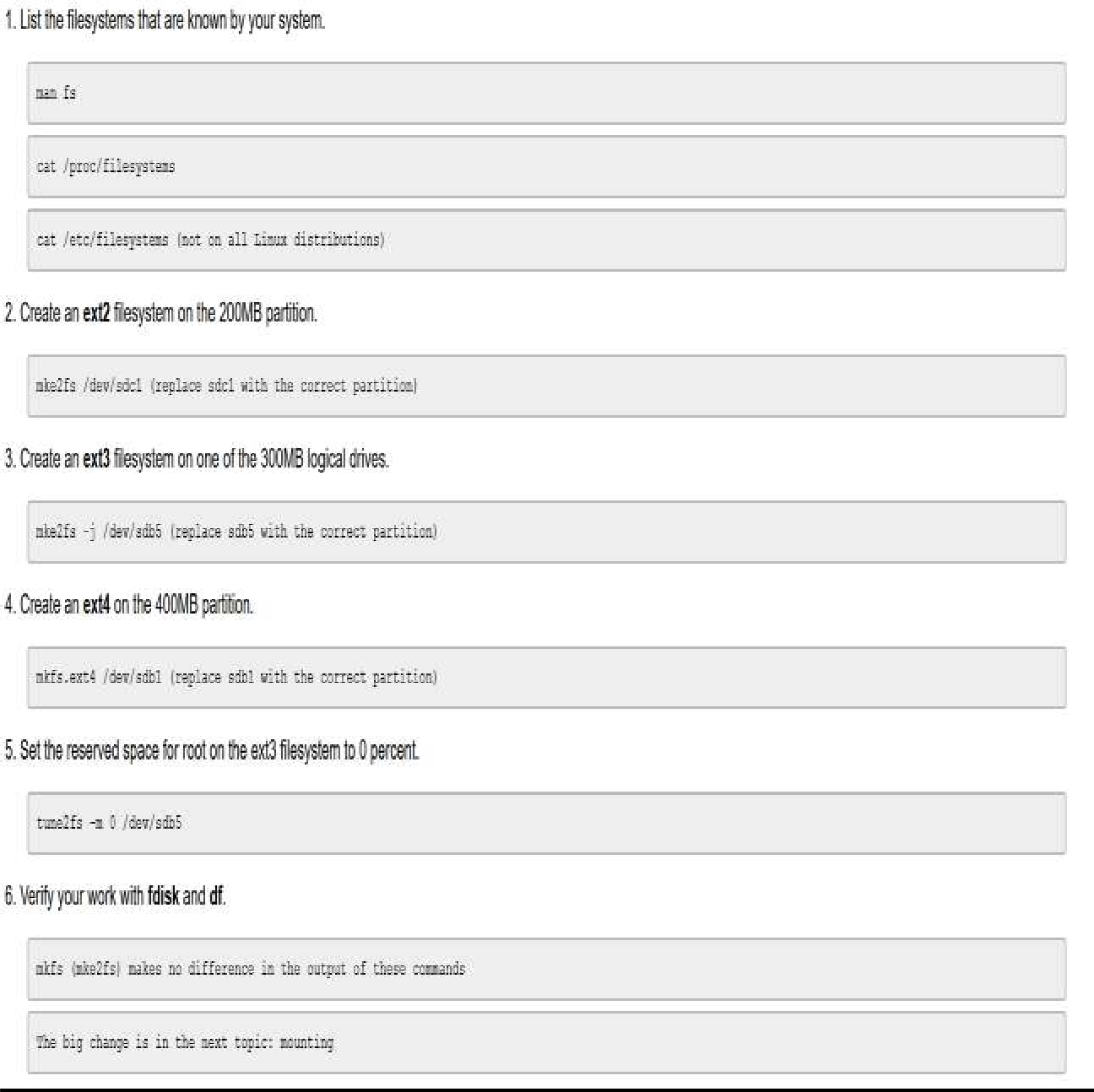
mkfs (mke2fs) makes no difference in the output of these commands

The big change is in the next topic: mounting

1. **Perform a file system check on all the new filesystems.**

fsck /dev/sdb1 fsck /dev/sdc1 fsck /dev/sdb5

**IMAGE OUTPUT(P.T.O):**





### CONCLUSION:

We have successfully studied about file system in linux and executed command in linux.

**AIM:-**

**PRACTICAL 7**

1. Schedule two jobs with at , display the at queue and remove thejob
2. As normal user , use crontab –e to schedule a script to run every four minutes
3. As ,root display the crontab file of normaluser
4. As a normal user again remove our crontabfile
5. Take a Look at the cron files and directories in /etc and understand them what is the runparts commanddoing

### THEORY:-

The **crontab** is a list of commands that you want to run on a regular schedule, and also the name of the command used to manage that list. Crontab standsfor―crontable,‖becauseitusesthejobscheduler*cron*toexecutetasks; *cron*itselfisnamedafter―chronos,‖theGreekwordfortime.*cron*isthesystem process which will automatically perform tasks for you according to a set schedule. The schedule is called the crontab, which is also the name of the program used to edit that schedule.

**Linux Crontab Format**

MIN HOUR DOM MON DOW CMD

**Crontab Fields and Allowed Ranges (Linux Crontab Syntax)**

|  |  |  |
| --- | --- | --- |
| **Field**  MIN | **Description**  Minute field | **Allowed Value**  0 to 59 |
| HOUR | Hour field | 0 to 23 |
| DOM | Day of Month | 1-31 |
| MON | Month field | 1-12 |
| DOW | Day Of Week | 0-6 |
| CMD | Command | Any command to be executed. |

**What does the crontab command do?**

On Unix-like operating systems, the **crontab command** opens the **cron** table for editing. The **cron** table is the list of tasks scheduled to run at regular timeintervals

on the system. The daemon which reads the **crontab** and executes the **commands**

at the right time is called **cron.**

### Why Use Cron Jobs?

Server admins have been using cron jobs for a long time. But since the target audience of this article is web developers, let's look at a few use cases of cron jobs that are relevant in this area:

* + If you have a membership site, where accounts have expiration dates, you can schedule cron jobs to regularly deactivate or delete accounts that are past their expiration dates.
  + You can send out daily newslettere-mails.
  + If you have summary tables (or materialized views) in your database, they can be regularly updated with a cron job. For example you may store every web page hit in a table, but another summary table may contain daily traffic summaries.
  + You can expire and erase cached data files in a certaininterval.
  + You can auto-check your website content for broken links and have a report e-mailed to yourselfregularly.
  + You can schedule long-running tasks to run from a command line script, rather than running it from a web script. Like encoding videos, or sending out masse-mails.
  + You can even perform something as simple as fetching your most recent Tweets, to be cached in a textfile.

### OUTPUTS:

AS NORMAL USER , USE CRONTAB –E TO SCHEDULE A SCRIPT TO RUN EVERY FOUR MINUTES

AS ,ROOT DISPLAY THE CRONTAB FILE OF NORMAL USER:

AS A NORMAL USER AGAIN REMOVE OUR CRONTAB FILE:

TAKE A LOOK AT THE CRON FILES AND DIRECTORIES IN /ETC AND UNDERSTAND THEM WHAT IS THE RUNPARTS COMMAND DOING:

These are all the directories in the folder /etc. The directories related to cron are:

1. cron.d
2. cron.daily
3. cron.hourly
4. cron.monthly
5. cron.weekly
6. crontab

In Kali Linux, when every script present in the above directories is executed we get the following output:

### WHAT IS THE RUNPARTS COMMAND DOING? :

Basically, [run-parts(8)](http://manpages.ubuntu.com/manpages/natty/man8/run-parts.8.html) takes a directory as an argument.

It will run every script that is found in this directory. For example, if you do a listing of /etc/cron.hourly, you'll see that it's a directory where you can put executable files to be run every hour.

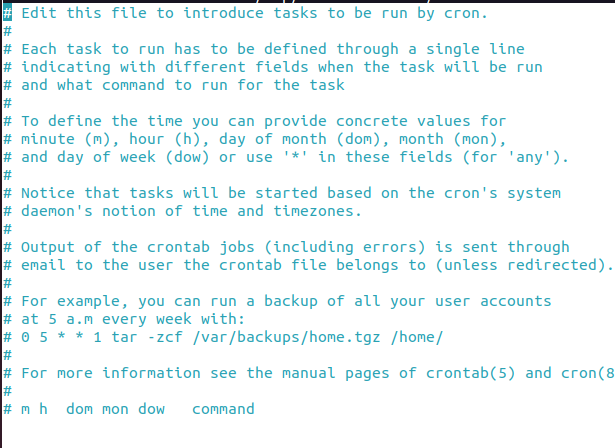
As you can see, in cron it's used for convenience, since you only have to specify one directory and everything in that directory will be executed. This makes it easy to maintain scripts in one of the etc/cron\* directories.

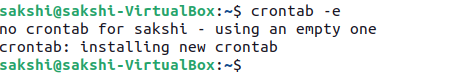
**Output:**

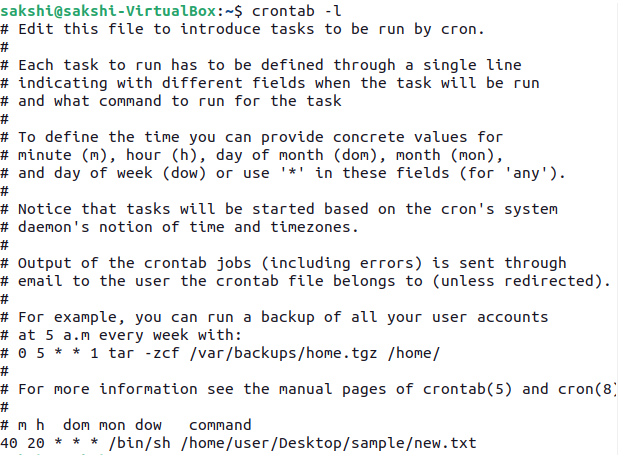
**OUTPUT :**

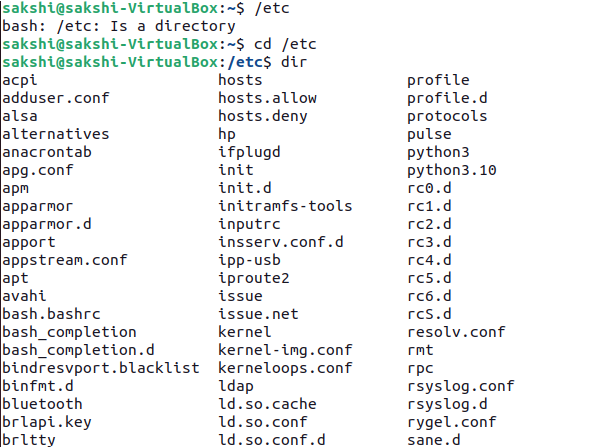
**OUTPUT :**

****







****

### CONCLUSION:

We have successfully used cron command to schedule a job and thus we know how to schedule any job in linux.

**Practical Number : 8 Date :**

**AIM :** Memory Management

i). Use dmesg to find the total amount of memory in your computer.

ii). Use free to display memory usage in kilobytes (then in megabytes).

iii). On a virtual machine, create a swap partition (you might need an extra virtual disk for this).

iv). Add a 20 megabyte swap file to the system.

v). Put all swap spaces in /etc/fstab and activate them. Test with a reboot that they are mounted.

vi). Use free to verify usage of current swap.

**Theory :**

Linux memory management subsystem is responsible, as the name implies, for managing the memory in the system. This includes implementation of virtual memory and demand paging, memory allocation both for kernel internal structures and user space programs, mapping of files into processes address space and many other cool things.

In Linux, the physical memory is called memory. When the physical memory fills up, Linux intelligently moves the less frequently accessed data from the memory to a specific part of the disk (HDD or SSD). This part of the disk is called swap.

When there is no available free physical memory, some less frequently access data is moved to the swap. This frees up the physical memory and thus saves the system from crashing.

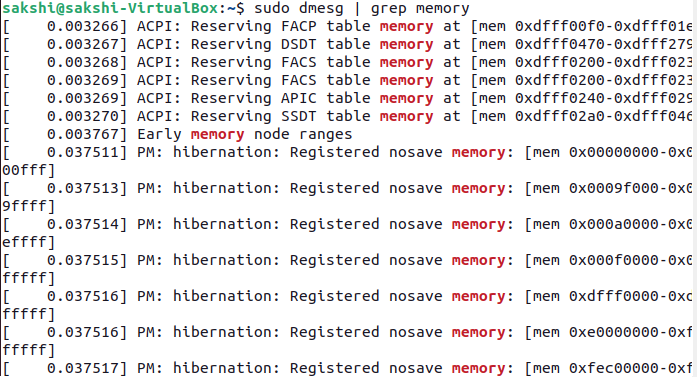
Swap disk is very slow compared to RAM or physical memory. If a Linux system uses swap space extensively, the system may become very slow and unresponsive. So, a Linux system shouldn’t use the swap space. We want to avoid it as much as possible. When a Linux system starts to fill up the swap space, it is a sign that the Linux system needs more physical memory. It’s a good idea to add more RAM or physical memory to the system.

1.Use dmesg to find the total amount of memory in your computer.:

dmesg command shows you the last status messages reported by your OS kernel, and since every boot procedure includes scanning the hardware and confirming the devices and resources recognized by the kernel, you can see some basic information by using dmesg.

Syntax: ~$ dmesg | grep Memory

For our purpose, we need to filter out the memory status:

****

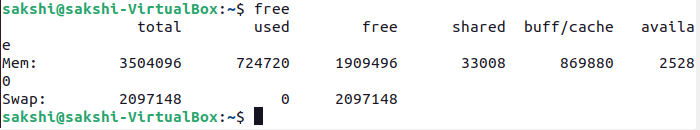
2. use to display memory usage in kilobytes (then in megabytes)

While using LINUX there might come a situation when you are willing to install a new application (big in size) and you wish to know for the amount of free memory available on your system. In LINUX, there exists a command line utility for this and that is free command which displays the total amount of free space available along with the amount of memory used and swap memory in the system, and also the buffers used by the kernel. free is a command that displays the total memory usage information of the system. free is shipped with almost all the Linux distribution by default.

syntax: ~$ free [OPTIONS]

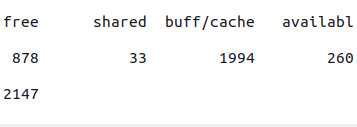
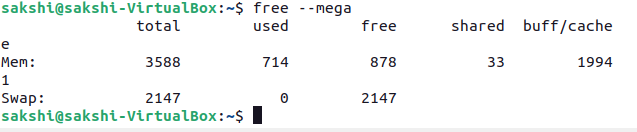
OPTIONS : refers to the options compatible with free command.

You can check memory usage with the free command as follows:



If we want to display the memory and swap usage information in megabytes, then run the free command with the –mega option as follows:

Syntax: $ free --mega



3. On a virtual machine, create a swap partition:

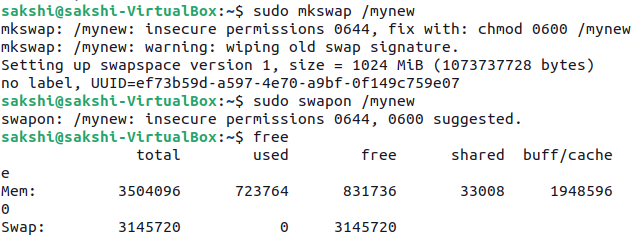
Linux provides for two types of swap space. By default, most Linux installations create a swap partition, but it is also possible to use a specially configured file as a swap file. A swap partition is just what its name implies—a standard disk partition that is designated as swap space by the mkswap command.

A swap file can be used if there is no free disk space in which to create a new swap partition or space in a volume group where a logical volume can be created for swap space. This is just a regular file that is created and preallocated to a specified size. Then the mkswap command is run to configure it as swap space. I don’t recommend using a file for swap space unless absolutely necessary.

Syntax: $ sudo dd if=/dev/zero of=/swapfile bs=1024 count=2097152;

$ sudo chmod 600 /mynew(file nmae)

$ sudo mkswap /(file\_name) mynew

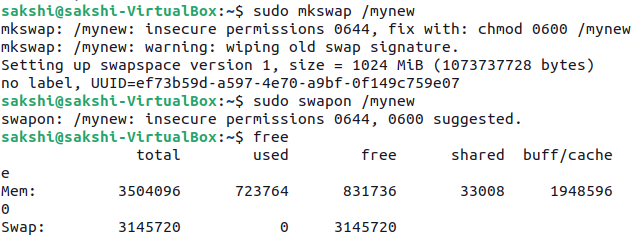


4. Add a 1024 megabyte swap file to the system :

Swap is a space on a disk that is used when the amount of physical RAM memory is full. When a Linux system runs out of RAM, inactive pages are moved from the RAM to the swap space.

Swap space can take the form of either a dedicated swap partition or a swap file. Typically, when running Ubuntu on a virtual machine, a swap partition is not present, and the only option is to create a swap file. Syntax: $ sudo swapon /(file name) mynew ---- to set file in swaping portion.

Syntax: $ sudo swapon /(file name) mynew ---- to set file in swaping portion.

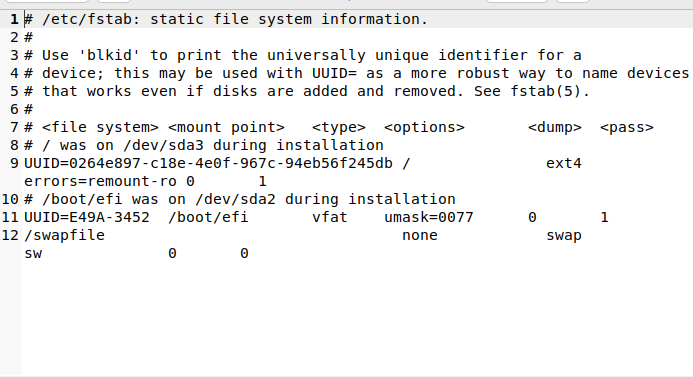


5. Put all swap spaces in /etc/fstab and activate them:

One of the easiest way of increasing the responsiveness of your server and guarding against out-of-memory errors in applications is to add some swap space.

To add all swap spaces in fstab we uesd following command: $ sudo gedit(compiler) /etc/fstab





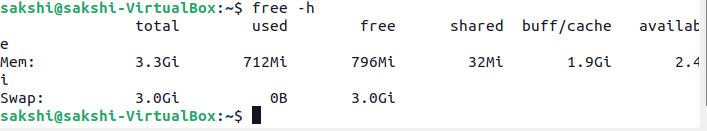
6.Use free to verify usage of current swap:

You can check the total, used and free swap space with the help of free command. The result will be displayed in kilobytes. The command looks like the following.

Syntax :

$ free

If you want to check the swap space in human-readable format with the closest possible unit, use the -h switch as follows.



**Conclusion:** I study about memory management, dmesg command, free command and file swapping in this practical.