Last updated on: 2020/01/30 03:28

File Hierarchy System

The Linux filesystem hierarchy starts with the **root** represented by a **/** character. Under the root can be found other directories containing task specific files. The hierarchy conforms to a standard called the **Linux File Hierarchy System**.

Directory Contents

Directory	Contents
/bin	Contains user programs such as ls, cp e.t.c.. Note that under RHEL 7 / CentOS7, this is a soft link (shorcut) to **/usr/bin**.
/boot	Contains bootloader files, kernels and initrd (INItial Ram Disk) files.
/dev	Contains nodes for accessing all the peripherals and devices connected to the system. The *udev* binary takes care of dynamically creating and deleting the relevant nodes automatically.
/etc	Contains static configuration files.
/home	Contains a directory for each registered user of the system except for root.
/lib	Contains common 32 bit libraries for applications and modules.
/lib64	Contains common 64 bit libraries for applications and modules.
/lost+found	Contains damaged file fragments found by the *fsck* command.
/media	Contains a folder for each of the mounted external file systems (CDRom DVDRom, USB Key e.t.c.).
/mnt	Contains a folder for each external file system mounted temporarily by root.
/opt	Contains optional application packages.
/proc	Contains a virtual file system that documents kernel and process status information as text files.
/root	The home directory of the root user.
/run	Replaces the /var/run directory. Note that in SLES 12, /var/run is a soft link (shorcut) to **/run**.
/sbin	Contains essential system administration binaires
/selinux	Contains a virtual file system used by SELINUX.
/srv	Contains site specific data **s**e**rv**ed by the system (www,ftp,databases e.t.c.).
/sys	Contains a virtual file system that describes devices for *udev*.

Directory	Contents
/tmp	Contains the temporary files created by the system and by applications.
/usr	Contains user commands in */usr/bin*, HOWTOs in */usr/share/doc*, manuals in */usr/share/man* and is the *Secondary Hierarchy* for read-only user data.
/var	Contains variable files. i.e. files that continually change such as log files and spool files.

Directory Structure

```
trainee@SLES11SP1:/> ls -l
total 101
drwxr-xr-x  2 root root  4096 25 sept. 15:48 bin
drwxr-xr-x  4 root root  1024 25 sept. 17:08 boot
drwxr-xr-x 16 root root  4020 25 sept. 17:08 dev
drwxr-xr-x 88 root root 12288 25 sept. 17:14 etc
drwxr-xr-x  3 root root  4096 25 sept. 15:55 home
drwxr-xr-x 13 root root  4096 25 sept. 15:49 lib
drwxr-xr-x  8 root root 12288 25 sept. 15:48 lib64
drwx------  2 root root 16384 25 sept. 15:43 lost+found
drwxr-xr-x  2 root root  4096  5 mai    2010 media
drwxr-xr-x  2 root root  4096  5 mai    2010 mnt
drwxr-xr-x  2 root root  4096  5 mai    2010 opt
dr-xr-xr-x 88 root root     0 25 sept. 17:07 proc
drwx------  6 root root  4096 25 sept. 17:14 root
drwxr-xr-x  3 root root 12288 25 sept. 15:50 sbin
drwxr-xr-x  2 root root  4096  5 mai    2010 selinux
drwxr-xr-x  4 root root  4096 25 sept. 15:43 srv
drwxr-xr-x 12 root root     0 25 sept. 17:07 sys
drwxrwxrwt  4 root root  4096 25 sept. 17:22 tmp
drwxr-xr-x 13 root root  4096 25 sept. 15:43 usr
drwxr-xr-x 15 root root  4096 25 sept. 15:44 var
```

```
trainee@SLES12SP1:/> ls -l
total 0
```

```
drwxr-xr-x   1 root root 1810 20 sept. 13:33 bin
drwxr-xr-x   1 root root 1096 21 sept. 04:19 boot
drwxr-xr-x  16 root root 3620 21 sept. 04:18 dev
drwxr-xr-x   1 root root 4746 21 sept. 04:18 etc
drwxr-xr-x   1 root root   14 20 sept. 13:34 home
drwxr-xr-x   1 root root 2906 20 sept. 13:32 lib
drwxr-xr-x   1 root root 4998 20 sept. 13:31 lib64
drwxr-xr-x   1 root root    0 21 sept.  2014 mnt
drwxr-xr-x   1 root root    0 21 sept.  2014 opt
dr-xr-xr-x 100 root root    0 20 sept. 13:47 proc
drwx------   1 root root  112 20 sept. 14:00 root
drwxr-xr-x  25 root root  640 21 sept. 04:18 run
drwxr-xr-x   1 root root 5044 20 sept. 13:33 sbin
drwxr-xr-x   1 root root    0 21 sept.  2014 selinux
drwxr-xr-x   1 root root   12 20 sept. 13:29 srv
dr-xr-xr-x  12 root root    0 20 sept. 13:47 sys
drwxrwxrwt   1 root root  102 21 sept. 04:18 tmp
drwxr-xr-x   1 root root  130 20 sept. 13:29 usr
drwxr-xr-x   1 root root  108 20 sept. 13:33 var
```

File Types

The three major file types under Linux are :

- Ordinary files,
- Directories,
- Special files or Devices.

Note that :

- Ordinary files can be anything from text files to binaries.

- The length of a file name is limited to 225 characters, including the file extension.
- Linux is case sensitive.
- If a file name starts with a dot (**.**), it is a hidden file.

The mount command

In order to be able to use external file systems, such as a CDRom or DVDRom, Linux needs to be informed of their availability. This is accomplished by using the **mount** command:

```
# mount /dev/<special_file> /mnt/<directory_name> [Enter]
```

where **/dev/<special_file>** is the file system to mount and **/mnt/<directory_name>** is the target directory where the mounted file system will be available to the system. The directory **/mnt/<directory_name>** must exist prior to using the **mount** command.

In the case where the **mount** command is used without options, the current mounted file systems are shown:

```
SLES11SP1:~ # mount
/dev/sda2 on / type ext3 (rw,acl,user_xattr)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
debugfs on /sys/kernel/debug type debugfs (rw)
udev on /dev type tmpfs (rw,mode=0755)
tmpfs on /dev/shm type tmpfs (rw,mode=1777)
devpts on /dev/pts type devpts (rw,mode=0620,gid=5)
/dev/sda1 on /boot type ext3 (rw,acl,user_xattr)
fusectl on /sys/fs/fuse/connections type fusectl (rw)
securityfs on /sys/kernel/security type securityfs (rw)
```

```
SLES12SP1:~ # mount
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
devtmpfs on /dev type devtmpfs (rw,nosuid,size=1931968k,nr_inodes=482992,mode=755)
```

```
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000)
tmpfs on /run type tmpfs (rw,nosuid,nodev,mode=755)
tmpfs on /sys/fs/cgroup type tmpfs (rw,nosuid,nodev,noexec,mode=755)
cgroup on /sys/fs/cgroup/systemd type cgroup
(rw,nosuid,nodev,noexec,relatime,xattr,release_agent=/usr/lib/systemd/systemd-cgroups-agent,name=systemd)
pstore on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime)
cgroup on /sys/fs/cgroup/cpuset type cgroup (rw,nosuid,nodev,noexec,relatime,cpuset)
cgroup on /sys/fs/cgroup/cpu,cpuacct type cgroup (rw,nosuid,nodev,noexec,relatime,cpuacct,cpu)
cgroup on /sys/fs/cgroup/memory type cgroup (rw,nosuid,nodev,noexec,relatime,memory)
cgroup on /sys/fs/cgroup/devices type cgroup (rw,nosuid,nodev,noexec,relatime,devices)
cgroup on /sys/fs/cgroup/freezer type cgroup (rw,nosuid,nodev,noexec,relatime,freezer)
cgroup on /sys/fs/cgroup/blkio type cgroup (rw,nosuid,nodev,noexec,relatime,blkio)
cgroup on /sys/fs/cgroup/perf_event type cgroup (rw,nosuid,nodev,noexec,relatime,perf_event)
cgroup on /sys/fs/cgroup/hugetlb type cgroup (rw,nosuid,nodev,noexec,relatime,hugetlb)
/dev/sda2 on / type btrfs (rw,relatime,space_cache,subvolid=259,subvol=/@/.snapshots/1/snapshot)
systemd-1 on /proc/sys/fs/binfmt_misc type autofs
(rw,relatime,fd=31,pgrp=1,timeout=300,minproto=5,maxproto=5,direct)
mqueue on /dev/mqueue type mqueue (rw,relatime)
debugfs on /sys/kernel/debug type debugfs (rw,relatime)
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime)
/dev/sda2 on /.snapshots type btrfs (rw,relatime,space_cache,subvolid=258,subvol=/@/.snapshots)
/dev/sda2 on /var/lib/mailman type btrfs (rw,relatime,space_cache,subvolid=269,subvol=/@/var/lib/mailman)
/dev/sda2 on /var/lib/mariadb type btrfs (rw,relatime,space_cache,subvolid=270,subvol=/@/var/lib/mariadb)
/dev/sda2 on /var/log type btrfs (rw,relatime,space_cache,subvolid=274,subvol=/@/var/log)
/dev/sda2 on /tmp type btrfs (rw,relatime,space_cache,subvolid=265,subvol=/@/tmp)
/dev/sda2 on /var/spool type btrfs (rw,relatime,space_cache,subvolid=276,subvol=/@/var/spool)
/dev/sda2 on /var/lib/named type btrfs (rw,relatime,space_cache,subvolid=272,subvol=/@/var/lib/named)
/dev/sda2 on /srv type btrfs (rw,relatime,space_cache,subvolid=264,subvol=/@/srv)
/dev/sda2 on /usr/local type btrfs (rw,relatime,space_cache,subvolid=266,subvol=/@/usr/local)
/dev/sda2 on /var/opt type btrfs (rw,relatime,space_cache,subvolid=275,subvol=/@/var/opt)
/dev/sda2 on /var/lib/pgsql type btrfs (rw,relatime,space_cache,subvolid=273,subvol=/@/var/lib/pgsql)
/dev/sda2 on /opt type btrfs (rw,relatime,space_cache,subvolid=263,subvol=/@/opt)
```

```
/dev/sda2 on /var/tmp type btrfs (rw,relatime,space_cache,subvolid=277,subvol=/@/var/tmp)
/dev/sda2 on /var/lib/mysql type btrfs (rw,relatime,space_cache,subvolid=271,subvol=/@/var/lib/mysql)
/dev/sda2 on /var/lib/libvirt/images type btrfs
(rw,relatime,space_cache,subvolid=268,subvol=/@/var/lib/libvirt/images)
/dev/sda2 on /var/crash type btrfs (rw,relatime,space_cache,subvolid=267,subvol=/@/var/crash)
/dev/sda2 on /home type btrfs (rw,relatime,space_cache,subvolid=262,subvol=/@/home)
/dev/sda2 on /boot/grub2/x86_64-efi type btrfs (rw,relatime,space_cache,subvolid=261,subvol=/@/boot/grub2/x86_64-
efi)
/dev/sda2 on /boot/grub2/i386-pc type btrfs (rw,relatime,space_cache,subvolid=260,subvol=/@/boot/grub2/i386-pc)
```

[stextbox id='black' image='null'] **Important** : Note that with SLES 11, the default filesystem is **ext3**whereas with SLES 12, the default filesystem is **btrfs**. Please see the unit **Managing Disks, Swap Space and Filesystems** for further coursework concerning ext3 and btrfs filesystems. [/stextbox]

Command Line Switches

The following switches can be used with the mount command:

```
SLES12SP1:~ # mount --help

Usage:
 mount [-lhV]
 mount -a [options]
 mount [options] [--source] <source> | [--target] <directory>
 mount [options] <source> <directory>
 mount <operation> <mountpoint> [<target>]

Options:
 -a, --all               mount all filesystems mentioned in fstab
 -c, --no-canonicalize   don't canonicalize paths
 -f, --fake              dry run; skip the mount(2) syscall
 -F, --fork              fork off for each device (use with -a)
 -T, --fstab <path>      alternative file to /etc/fstab
 -h, --help              display this help text and exit
 -i, --internal-only     don't call the mount.<type> helpers
 -l, --show-labels       lists all mounts with LABELs
 -n, --no-mtab           don't write to /etc/mtab
 -o, --options <list>    comma-separated list of mount options
 -O, --test-opts <list>  limit the set of filesystems (use with -a)
 -r, --read-only         mount the filesystem read-only (same as -o ro)
 -t, --types <list>      limit the set of filesystem types
     --source <src>      explicitly specifies source (path, label, uuid)
     --target <target>   explicitly specifies mountpoint
 -v, --verbose           say what is being done
 -V, --version           display version information and exit
 -w, --rw, --read-write  mount the filesystem read-write (default)

 -h, --help     display this help and exit
 -V, --version  output version information and exit

Source:
 -L, --label <label>     synonym for LABEL=<label>
 -U, --uuid <uuid>       synonym for UUID=<uuid>
 LABEL=<label>           specifies device by filesystem label
 UUID=<uuid>             specifies device by filesystem UUID
 PARTLABEL=<label>       specifies device by partition label
 PARTUUID=<uuid>         specifies device by partition UUID
 <device>                specifies device by path
 <directory>             mountpoint for bind mounts (see --bind/rbind)
 <file>                  regular file for loopdev setup

Operations:
 -B, --bind              mount a subtree somewhere else (same as -o bind)
 -M, --move              move a subtree to some other place
 -R, --rbind             mount a subtree and all submounts somewhere else
 --make-shared           mark a subtree as shared
 --make-slave            mark a subtree as slave
 --make-private          mark a subtree as private
 --make-unbindable       mark a subtree as unbindable
 --make-rshared          recursively mark a whole subtree as shared
 --make-rslave           recursively mark a whole subtree as slave
 --make-rprivate         recursively mark a whole subtree as private
 --make-runbindable      recursively mark a whole subtree as unbindable

For more details see mount(8).
```

The /etc/fstab file

In the case where the **mount** command is used with the **-a** option, all mount points specified in the **/etc/fstab** file are mounted:

```
SLES11SP1:~ # cat /etc/fstab
/dev/disk/by-id/ata-VBOX_HARDDISK_VB62af9a29-d9a982d5-part3 swap                 swap       defaults
0 0
/dev/disk/by-id/ata-VBOX_HARDDISK_VB62af9a29-d9a982d5-part2 /                    ext3       acl,user_xattr
1 1
/dev/disk/by-id/ata-VBOX_HARDDISK_VB62af9a29-d9a982d5-part1 /boot                ext3       acl,user_xattr
1 2
proc                 /proc                proc       defaults              0 0
sysfs                /sys                 sysfs      noauto                0 0
debugfs              /sys/kernel/debug    debugfs    noauto                0 0
usbfs                /proc/bus/usb        usbfs      noauto                0 0
devpts               /dev/pts             devpts     mode=0620,gid=5       0 0
```

```
SLES12SP1:~ # cat /etc/fstab
UUID=db743358-c2d6-47f6-97d7-e7a9c650f0c5 swap swap defaults 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b / btrfs defaults 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /boot/grub2/i386-pc btrfs subvol=@/boot/grub2/i386-pc 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /boot/grub2/x86_64-efi btrfs subvol=@/boot/grub2/x86_64-efi 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /home btrfs subvol=@/home 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /opt btrfs subvol=@/opt 0 0
```

```
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /srv btrfs subvol=@/srv 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /tmp btrfs subvol=@/tmp 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /usr/local btrfs subvol=@/usr/local 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /var/crash btrfs subvol=@/var/crash 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /var/lib/libvirt/images btrfs subvol=@/var/lib/libvirt/images 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /var/lib/mailman btrfs subvol=@/var/lib/mailman 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /var/lib/mariadb btrfs subvol=@/var/lib/mariadb 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /var/lib/mysql btrfs subvol=@/var/lib/mysql 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /var/lib/named btrfs subvol=@/var/lib/named 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /var/lib/pgsql btrfs subvol=@/var/lib/pgsql 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /var/log btrfs subvol=@/var/log 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /var/opt btrfs subvol=@/var/opt 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /var/spool btrfs subvol=@/var/spool 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /var/tmp btrfs subvol=@/var/tmp 0 0
UUID=6b7e374a-ae42-4f93-b6aa-d288dfbbb74b /.snapshots btrfs subvol=@/.snapshots 0 0
```

Understanding the /etc/fstab file

Each line in **/etc/fstab** has 6 fields :

Field 1	**Field 2**	**Field 3**	**Field 4**	**Field 5**	**Field 6**
Special file or UUID or Virtual File System	Mount Point	Filesystem Type	Comma separated list of options	Used by the dump command (1 = dump, 0 or empty = do not dump)	The order in which the *fsck* command checks the disks/partitions at boot time

The **UUID** (*Universally Unique Identifier*) is a randomly generated 128 bit string that is automatically generated by the system when a filesystem is created on the partition.

Mountpoint Options

The most important mount point options are as follows:

Option	Filesystem	Description	Default Value
defaults	All	Use default options: rw, suid, dev, exec, auto, nouser, and async.	N/A [1]
auto/noauto	All	Do or do not mount when “mount -a” is given.	auto
rw/ro	All	Mount the filesystem read-write/read-only.	rw
suid/nosuid	All	Allow/disallow set-user-identifier or set-group-identifier bits to take effect.	suid
dev/nodev	All	Interpret/do not interpret character or block special devices on the filesystem.	dev
exec/noexec	All	Permit/do not permit execution of binaries.	exec
sync/async	All	All I/O to the filesystem should be done synchronously/asynchronously.	async
user/nouser	All	Allow/disallow a user to mount. The mount point is read from the /etc/fstab file. Only the user that mounted the filesystem can unmount it.	N/A
users	All	Allow every user to mount and unmount the filesystem.	N/A
owner	All	Allow device owner to mount.	N/A
atime/noatime	POSIX	Do not use noatime feature, then the inode access time is controlled by kernel defaults/Do not update inode access times on this filesystem	atime
uid=value	Non-Linux filesystems	Set the owner of the root of the filesystem.	root
gid=value	Non-Linux filesystems	Set the group of the root of the filesystem.	N/A
umask=value	Non-Linux filesystems	Set the umask. The default is the umask of the current process. The value is given in octal.	N/A
dmask=value	Non-Linux filesystems	Set the umask applied to directories only. The value is given in octal.	Current processes' umask
fmask=value	Non-Linux filesystems	Set the umask applied to regular files only. The value is given in octal.	Current processes' umask

The umount command

To unmount a file system, you need to use the **umount** command. For example:

```
# umount /mnt/target_directory [Entrée]
```

Command Line Switches

The following switches can be used with the umount command:

```
SLES12SP1:~ # umount --help

Usage:
 umount [-hV]
 umount -a [options]
 umount [options] <source> | <directory>

Options:
 -a, --all               unmount all filesystems
 -A, --all-targets       unmount all mountpoints for the given device in the
                           current namespace
 -c, --no-canonicalize   don't canonicalize paths
 -d, --detach-loop       if mounted loop device, also free this loop device
     --fake              dry run; skip the umount(2) syscall
 -f, --force             force unmount (in case of an unreachable NFS system)
 -i, --internal-only     don't call the umount.<type> helpers
 -n, --no-mtab           don't write to /etc/mtab
 -l, --lazy              detach the filesystem now, clean up things later
 -O, --test-opts <list>  limit the set of filesystems (use with -a)
 -R, --recursive         recursively unmount a target with all its children
 -r, --read-only         in case unmounting fails, try to remount read-only
 -t, --types <list>      limit the set of filesystem types
 -v, --verbose           say what is being done

 -h, --help     display this help and exit
 -V, --version  output version information and exit

For more details see umount(8).
```

Unix File Systems

Each file system contains the following :

- superblock
- inodes
- data blocks

Superblock

The superblock contains :

- the block size,
- the size of the file system,
- the number of mounts for the file system,
- a pointer to the root of the file system,
- pointers to the free inodes,
- pointers to free data blocks.

Linux maintains multiple redundant copies of the superblock in every file system.

For example, to view the primary and backup superblock locations on ext filesystems, use the following command:

```
SLES11SP1:~ # mount | grep ext
/dev/sda2 on / type ext3 (rw,acl,user_xattr)
/dev/sda1 on /boot type ext3 (rw,acl,user_xattr)
SLES11SP1:~ # dumpe2fs /dev/sda1 | grep -i superblock
dumpe2fs 1.41.9 (22-Aug-2009)
  Primary superblock at 1, Group descriptors at 2-2
  Backup superblock at 8193, Group descriptors at 8194-8194
  Backup superblock at 24577, Group descriptors at 24578-24578
  Backup superblock at 40961, Group descriptors at 40962-40962
  Backup superblock at 57345, Group descriptors at 57346-57346
  Backup superblock at 73729, Group descriptors at 73730-73730
```

To repair an ext file system using a backup superblock use the following command :

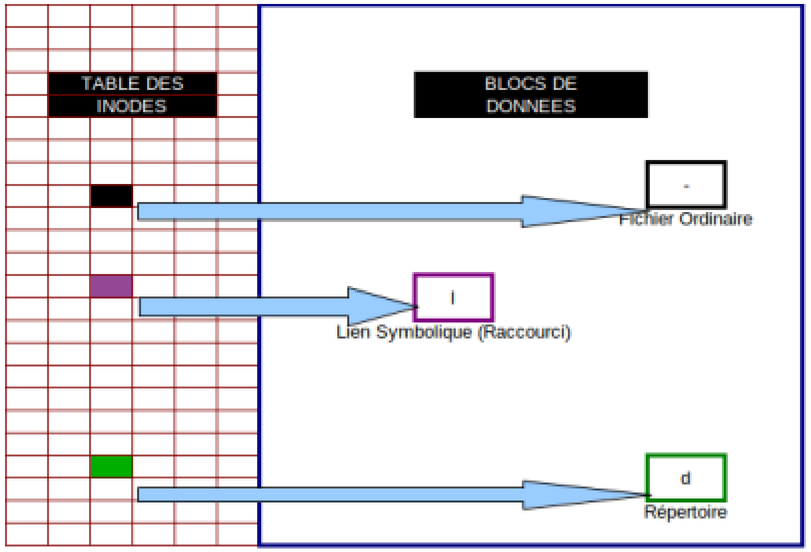
```
# e2fsck -f -b 8193 /dev/sda1 [Enter]
```

Inodes

Each file is represented by an **inode**. An inode contains the following information:

- the file type : **-**, **d**, **l**, **b**, **c**, **p**, **s**,
- file permissions, for example : **rwx rw- r-**,
- the number of hard links,
- the UID of the file creator or the current UID attributed by the **chown** command,
- the GID of the creating process or the current GID attributed by the **chgrp** command,
- the file size in bytes,
- the date of the last modification of the file's inode content : **ctime**,
- the date of the last modification of the file contents : **mtime**,
- the date of the last access : **atime**,
- allocation addresses that point to the data blocks used by the file.

For example:



Execute the following command:

```
SLES12SP1:~ # ls -ld /dev/console /dev/sda1 /etc /etc/passwd
crw------- 1 root root 5, 1 Sep 28 10:37 /dev/console
brw-rw---- 1 root disk 8, 1 Sep 28 10:37 /dev/sda1
drwxr-xr-x 1 root root 4746 Sep 28 10:38 /etc
-rw-r--r-- 1 root root 1335 Sep 20 13:34 /etc/passwd
```

The first character of each line indicates the file type:

- **-** - an ordinary file,
- **d** - a directory,
- **l** - a symbolic link,
- **b** - a bloc type peripheral,
- **c** - a character type peripheral,
- **p** - a named pipe for communication between processes,
- **s** - a network socket.

To see the inode numbers, execute the previous command with, in addition, the **-i** option:

```
SLES12SP1:~ # ls -ldi /dev/console /dev/sda1 /etc /etc/passwd
 4306 crw------- 1 root root 5, 1 Sep 28 10:37 /dev/console
 6871 brw-rw---- 1 root disk 8, 1 Sep 28 10:37 /dev/sda1
  257 drwxr-xr-x 1 root root 4746 Sep 28 12:02 /etc
58930 -rw-r--r-- 1 root root 1335 Sep 20 13:34 /etc/passwd
```

Data Blocks

File data is stored in data blocks. In the case of a directory, the data block contains a table referencing the inodes and the names of the contents of the directory.

The name of the file is stored in the parent directory's data block and not in the inode. This means that a file can be referenced by one or more different names. To add a name to a data block, you need to create what is called a **hard link**.

Hard (Physical) Links

A hard link is created by using the **ln** command.

```
SLES12SP1:~ # cd /tmp; mkdir inode; cd inode; touch file1; ls -ali
total 0
442 drwxr-xr-x 1 root root  10 Sep 28 12:23 .
256 drwxrwxrwt 1 root root 112 Sep 28 12:23 ..
443 -rw-r--r-- 1 root root   0 Sep 28 12:23 file1
```

file1 shows an inode number of **443** and a single name, indicated by the number **1** in the third column:

```
443 -rw-r--r-- 1 root root   0 Sep 28 12:23 file1
```

now create the hard link and check the result:

```
SLES12SP1:/tmp/inode # ln file1 file2
```

```
SLES12SP1:/tmp/inode # ls -ali
total 0
442 drwxr-xr-x 1 root root  20 Sep 28 12:24 .
256 drwxrwxrwt 1 root root 112 Sep 28 12:23 ..
443 -rw-r--r-- 2 root root   0 Sep 28 12:23 file1
443 -rw-r--r-- 2 root root   0 Sep 28 12:23 file2
```

Now you can see two lines, one for file1 and a second for file2:

```
443 -rw-r--r-- 2 root root   0 Sep 28 12:23 file1
443 -rw-r--r-- 2 root root   0 Sep 28 12:23 file2
```

file1 and **file2** are referenced by the same inode. As a result the number of names has been increased to two in the thrid column.

[stextbox id='black' image='null'] **Important** - Hard links cannot be created across file system boundaries. A hard link can only be created if the source file exists. [/stextbox]

Soft (Symbolic) Links

A soft link is a shortcut to a file or directory. A soft link is created using the same **ln** command with the **-s** option.

```
SLES12SP1:/tmp/inode # ln -s file1 file3
SLES12SP1:/tmp/inode # ls -ali
total 4
442 drwxr-xr-x 1 root root  30 Sep 28 12:26 .
256 drwxrwxrwt 1 root root 112 Sep 28 12:23 ..
443 -rw-r--r-- 2 root root   0 Sep 28 12:23 file1
443 -rw-r--r-- 2 root root   0 Sep 28 12:23 file2
444 lrwxrwxrwx 1 root root   5 Sep 28 12:26 file3 -> file1
```

Note here that the soft link is referenced by a separate inode.

[stextbox id='black' image='null'] **Important** - A soft link can be created across file system boundaries and can be created even when the source file does not exist. [/stextbox]

---

1) Not Applicable