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..
/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.
/sbin	Contains essential system administration binaires.
/selinux	Contains a virtual file system used by SELINUX.
/snap	**Ubuntu 16.04 only**. Used in conjunction with the new **Snap** packages.
/srv	Contains site specific data **s**e**rv**ed by the system (www,ftp,databases e.t.c.).

Directory	Contents
/sys	Contains a virtual file system that describes devices for *udev*.
/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@ubuntu1404:/$ ls -l
total 92
drwxr-xr-x   2 root root  4096 sept. 27  2014 bin
drwxr-xr-x   3 root root  4096 sept. 27  2014 boot
drwxrwxr-x   2 root root  4096 sept. 27  2014 cdrom
drwxr-xr-x  15 root root  4120 sept. 25 15:17 dev
drwxr-xr-x 131 root root 12288 sept. 25 15:17 etc
drwxr-xr-x   3 root root  4096 sept. 27  2014 home
lrwxrwxrwx   1 root root    33 sept. 27  2014 initrd.img -> boot/initrd.img-3.13.0-32-generic
drwxr-xr-x  23 root root  4096 sept. 27  2014 lib
drwx------   2 root root 16384 sept. 27  2014 lost+found
drwxr-xr-x   3 root root  4096 sept. 28  2014 media
drwxr-xr-x   2 root root  4096 avril 11  2014 mnt
drwxr-xr-x   3 root root  4096 sept. 28  2014 opt
dr-xr-xr-x 102 root root     0 sept. 25 15:17 proc
drwx------   2 root root  4096 oct.  14  2014 root
drwxr-xr-x  20 root root   700 sept. 25 15:17 run
drwxr-xr-x   2 root root 12288 sept. 28  2014 sbin
drwxr-xr-x   2 root root  4096 juil. 22  2014 srv
dr-xr-xr-x  13 root root     0 sept. 25 15:17 sys
drwxrwxrwt   4 root root  4096 sept. 25 15:17 tmp
drwxr-xr-x  10 root root  4096 juil. 22  2014 usr
drwxr-xr-x  13 root root  4096 juil. 23  2014 var
```

```
lrwxrwxrwx   1 root root    30 sept. 27  2014 vmlinuz -> boot/vmlinuz-3.13.0-32-generic
```

```
trainee@ubuntu1604:/$ ls -l
total 100
drwxr-xr-x   2 root root  4096 mai    3 07:47 bin
drwxr-xr-x   3 root root  4096 mai    3 07:49 boot
drwxrwxr-x   2 root root  4096 mai    3 07:25 cdrom
drwxr-xr-x  19 root root  4180 août  18 12:39 dev
drwxr-xr-x 129 root root 12288 sept. 25 11:12 etc
drwxr-xr-x   3 root root  4096 mai    3 07:27 home
lrwxrwxrwx   1 root root    32 mai    3 07:31 initrd.img -> boot/initrd.img-4.4.0-21-generic
drwxr-xr-x  22 root root  4096 mai    3 07:47 lib
drwxr-xr-x   2 root root  4096 avril 21 00:07 lib64
drwx------   2 root root 16384 mai    3 07:17 lost+found
drwxr-xr-x   2 root root  4096 avril 21 00:07 media
drwxr-xr-x   2 root root  4096 avril 21 00:07 mnt
drwxr-xr-x   3 root root  4096 mai    3 08:14 opt
dr-xr-xr-x 114 root root     0 août  18 12:39 proc
drwx------   4 root root  4096 mai    3 08:33 root
drwxr-xr-x  24 root root   820 sept. 25 11:40 run
drwxr-xr-x   2 root root 12288 mai    3 07:51 sbin
drwxr-xr-x   2 root root  4096 avril 19 16:31 snap
drwxr-xr-x   2 root root  4096 avril 21 00:07 srv
dr-xr-xr-x  13 root root     0 août  18 12:38 sys
drwxrwxrwt   9 root root  4096 sept. 25 11:17 tmp
drwxr-xr-x  11 root root  4096 avril 21 00:13 usr
drwxr-xr-x  14 root root  4096 avril 21 00:19 var
lrwxrwxrwx   1 root root    29 mai    3 07:31 vmlinuz -> boot/vmlinuz-4.4.0-21-generic
```

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:

```
trainee@ubuntu1404:/$ mount
/dev/sda1 on / type ext4 (rw,errors=remount-ro)
proc on /proc type proc (rw,noexec,nosuid,nodev)
sysfs on /sys type sysfs (rw,noexec,nosuid,nodev)
none on /sys/fs/cgroup type tmpfs (rw)
none on /sys/fs/fuse/connections type fusectl (rw)
none on /sys/kernel/debug type debugfs (rw)
none on /sys/kernel/security type securityfs (rw)
```

```
udev on /dev type devtmpfs (rw,mode=0755)
devpts on /dev/pts type devpts (rw,noexec,nosuid,gid=5,mode=0620)
tmpfs on /run type tmpfs (rw,noexec,nosuid,size=10%,mode=0755)
none on /run/lock type tmpfs (rw,noexec,nosuid,nodev,size=5242880)
none on /run/shm type tmpfs (rw,nosuid,nodev)
none on /run/user type tmpfs (rw,noexec,nosuid,nodev,size=104857600,mode=0755)
none on /sys/fs/pstore type pstore (rw)
systemd on /sys/fs/cgroup/systemd type cgroup (rw,noexec,nosuid,nodev,none,name=systemd)
```

```
root@ubuntu1604:~# mount
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
udev on /dev type devtmpfs (rw,nosuid,relatime,size=230832k,nr_inodes=57708,mode=755)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000)
tmpfs on /run type tmpfs (rw,nosuid,noexec,relatime,size=50028k,mode=755)
/dev/sda1 on / type ext4 (rw,relatime,errors=remount-ro,data=ordered)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
tmpfs on /run/lock type tmpfs (rw,nosuid,nodev,noexec,relatime,size=5120k)
tmpfs on /sys/fs/cgroup type tmpfs (ro,nosuid,nodev,noexec,mode=755)
cgroup on /sys/fs/cgroup/systemd type cgroup
(rw,nosuid,nodev,noexec,relatime,xattr,release_agent=/lib/systemd/systemd-cgroups-agent,name=systemd,nsroot=/)
pstore on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime)
cgroup on /sys/fs/cgroup/hugetlb type cgroup (rw,nosuid,nodev,noexec,relatime,hugetlb,nsroot=/)
cgroup on /sys/fs/cgroup/memory type cgroup (rw,nosuid,nodev,noexec,relatime,memory,nsroot=/)
cgroup on /sys/fs/cgroup/freezer type cgroup (rw,nosuid,nodev,noexec,relatime,freezer,nsroot=/)
cgroup on /sys/fs/cgroup/cpuset type cgroup (rw,nosuid,nodev,noexec,relatime,cpuset,nsroot=/)
cgroup on /sys/fs/cgroup/pids type cgroup (rw,nosuid,nodev,noexec,relatime,pids,nsroot=/)
cgroup on /sys/fs/cgroup/blkio type cgroup (rw,nosuid,nodev,noexec,relatime,blkio,nsroot=/)
cgroup on /sys/fs/cgroup/net_cls,net_prio type cgroup (rw,nosuid,nodev,noexec,relatime,net_cls,net_prio,nsroot=/)
cgroup on /sys/fs/cgroup/perf_event type cgroup (rw,nosuid,nodev,noexec,relatime,perf_event,nsroot=/)
cgroup on /sys/fs/cgroup/devices type cgroup (rw,nosuid,nodev,noexec,relatime,devices,nsroot=/)
cgroup on /sys/fs/cgroup/cpu,cpuacct type cgroup (rw,nosuid,nodev,noexec,relatime,cpu,cpuacct,nsroot=/)
systemd-1 on /proc/sys/fs/binfmt_misc type autofs
```

```
(rw,relatime,fd=22,pgrp=1,timeout=0,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)
fusectl on /sys/fs/fuse/connections type fusectl (rw,relatime)
tmpfs on /run/user/1000 type tmpfs (rw,nosuid,nodev,relatime,size=50028k,mode=700,uid=1000,gid=1000)
```

Command Line Switches

The following switches can be used with the mount command:

```
root@ubuntu1604:~# mount --help

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

Mount a filesystem.

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
 -i, --internal-only     don't call the mount.<type> helpers
 -l, --show-labels       show also filesystem 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
 -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:

```
root@ubuntu1404:~# cat /etc/fstab
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/sda1 during installation
UUID=70eb8bc5-1759-433d-9797-9342a7b82cb2 /               ext4    errors=remount-ro 0       1
# swap was on /dev/sda5 during installation
UUID=85017f2f-081d-464e-ad83-52c3c895a113 none            swap    sw              0       0
```

```
root@ubuntu1604:~# cat /etc/fstab
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/sda1 during installation
UUID=c27fce7f-cc8a-4c6f-b19b-d929a4d570f2 /               ext4    errors=remount-ro 0       1
# swap was on /dev/sda5 during installation
UUID=68f67549-63f1-4833-b792-3566455bbe95 none            swap    sw              0       0
```

```
[root@centos5 ~]# cat /etc/fstab
LABEL=/                 /                       ext3    defaults        1 1
LABEL=/boot             /boot                   ext3    defaults        1 2
tmpfs                   /dev/shm                tmpfs   defaults        0 0
devpts                  /dev/pts                devpts  gid=5,mode=620  0 0
sysfs                   /sys                    sysfs   defaults        0 0
proc                    /proc                   proc    defaults        0 0
LABEL=SWAP-sda3         swap                    swap    defaults        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

Option	Filesystem	Description	Default Value
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:

```
root@ubuntu1604:~# umount --help
```

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

Unmount filesystems.

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:

```
root@ubuntu1604:~# mount | grep ext
/dev/sda1 on / type ext4 (rw,relatime,errors=remount-ro,data=ordered)
root@ubuntu1604:~# dumpe2fs /dev/sda1 | grep -i superblock
dumpe2fs 1.42.13 (17-May-2015)
  Primary superblock at 0, Group descriptors at 1-1
  Backup superblock at 32768, Group descriptors at 32769-32769
  Backup superblock at 98304, Group descriptors at 98305-98305
  Backup superblock at 163840, Group descriptors at 163841-163841
  Backup superblock at 229376, Group descriptors at 229377-229377
  Backup superblock at 294912, Group descriptors at 294913-294913
  Backup superblock at 819200, Group descriptors at 819201-819201
  Backup superblock at 884736, Group descriptors at 884737-884737
  Backup superblock at 1605632, Group descriptors at 1605633-1605633
```

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

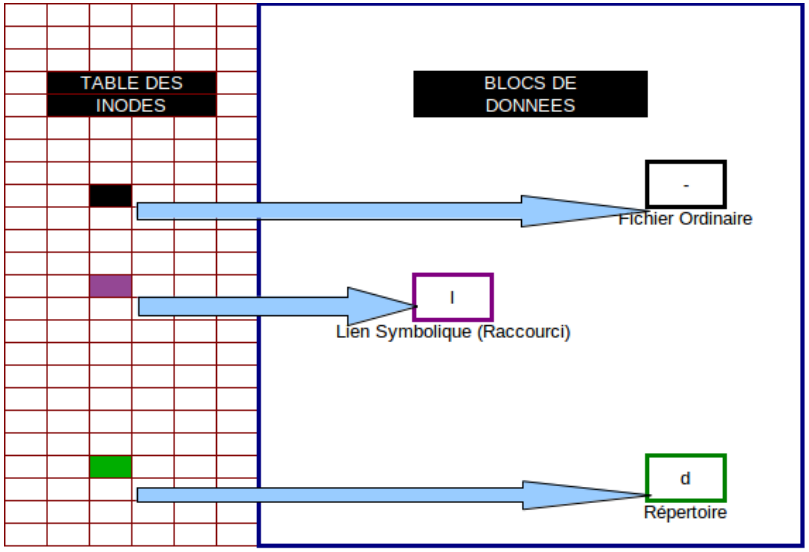
```
# e2fsck -f -b 32768 /dev/sda1 [Enter]
```

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:

```
root@ubuntu1604:~# ls -ld /dev/console /dev/sda1 /etc /etc/passwd
crw-------   1 root root  5, 1 sept. 28 10:31 /dev/console
brw-rw----   1 root disk  8, 1 sept. 28 10:31 /dev/sda1
drwxr-xr-x 129 root root 12288 sept. 28 10:41 /etc
-rw-r--r--   1 root root  2296 mai    3 08:08 /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:

```
root@ubuntu1604:~# ls -ldi /dev/console /dev/sda1 /etc /etc/passwd
    14 crw-------   1 root root  5, 1 sept. 28 10:31 /dev/console
   376 brw-rw----   1 root disk  8, 1 sept. 28 10:31 /dev/sda1
390913 drwxr-xr-x 129 root root 12288 sept. 28 10:41 /etc
396002 -rw-r--r--   1 root root  2296 mai    3 08:08 /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.

```
root@ubuntu1604:~# cd /tmp; mkdir inode; cd inode; touch file1; ls -ali
total 8
521308 drwxr-xr-x  2 root root 4096 sept. 29 10:26 .
390918 drwxrwxrwt 10 root root 4096 sept. 29 10:26 ..
521340 -rw-r--r--  1 root root    0 sept. 29 10:26 file1
```

file1 shows an inode number of **521340** and a single name, indicated by the number **1** in the third column:

```
521340 -rw-r--r--  1 root root    0 sept. 29 10:26 file1
```

now create the hard link and check the result:

```
root@ubuntu1604:/tmp/inode# ln file1 file2
```

```
root@ubuntu1604:/tmp/inode# ls -ali
total 8
521308 drwxr-xr-x  2 root root 4096 sept. 29 10:27 .
390918 drwxrwxrwt 10 root root 4096 sept. 29 10:26 ..
521340 -rw-r--r--  2 root root    0 sept. 29 10:26 file1
521340 -rw-r--r--  2 root root    0 sept. 29 10:26 file2
```

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

```
521340 -rw-r--r--  2 root root    0 sept. 29 10:26 file1
521340 -rw-r--r--  2 root root    0 sept. 29 10:26 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.

```
root@ubuntu1604:/tmp/inode# ln -s file1 file3
root@ubuntu1604:/tmp/inode# ls -ali
total 8
521308 drwxr-xr-x  2 root root 4096 sept. 29 10:32 .
390918 drwxrwxrwt 10 root root 4096 sept. 29 10:26 ..
521340 -rw-r--r--  2 root root    0 sept. 29 10:26 file1
521340 -rw-r--r--  2 root root    0 sept. 29 10:26 file2
521342 lrwxrwxrwx  1 root root    5 sept. 29 10:32 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]

<html>

</html>

---

1)

Not Applicable