Christopher Welch
Prof. Nielsen
CIS220.470
February 24, 2018

Chapter 2 Hands-On Projects (2-1 + 2-3-> 2-12)

Project 2-1

[student@COS-047 ~]$ man mount
MOUNT(8) System Administration MOUNT(8)

NAME
mount - mount a filesystem

SYNOPSIS
mount [-lhV]

mount -a [-fFnrsvw] [-t vfstype] [-O optlist]

mount [-fnrsvw] [-o option[,option]...] device|dir

mount [-fnrsvw] [-t vfstype] [-o options] device dir

DESCRIPTION
All files accessible in a Unix system are arranged in one big tree, the file hierarchy, rooted at /. These files can be spread out over several devices. The mount command serves to attach the filesystem found on some device to the big file tree. Conversely, the umount(8) command will detach it again.

The standard form of the mount command, is
mount -t type device dir

This tells the kernel to attach the filesystem found on device (which is of type type) at the directory dir. The previous contents (if any) and owner and mode of dir become invisible, and as long as this filesystem remains mounted, the pathname dir refers to the root of the filesystem on device.

If only directory or device is given, for example:

mount /dir

then mount looks for a mountpoint and if not found then for a device in the /etc/fstab file. It's possible to use --target or --source options to avoid ambivalent interpretation of the given argument. For example

mount --target /mountpoint

The listing and help.
The listing mode is maintained for backward compatibility only.

For more robust and definable output use findmnt(8), especially in your scripts. Note that control characters in the mountpoint name are replaced with '?'.

mount [-l] [-t type]

lists all mounted filesystems (of type type). The option -l adds the labels in this listing. See below.

The device indication.

Most devices are indicated by a file name (of a block special device), like /dev/sda1, but there are other possibilities. For example, in the case of an NFS mount, device may look like knuth.cwi.nl:/dir. It is possible to indicate a block special device using its filesystem LABEL or UUID (see the -L and -U options below) and partition PARTUUID or PARTLABEL (partition identifiers are supported for GUID Partition Table (GPT) and MAC partition tables only).

The recommended setup is to use tags (e.g. LABEL=<label>) rather than /dev/disk/by-{label,uuid,partuuid,partlabel} udev symlinks in the /etc/fstab file. The tags are more readable, robust and portable. The mount(8) command internally uses udev symlinks, so use the symlinks in /etc/fstab has no advantage over the tags. For more details see libblkid(3).

Note that mount(8) uses UUIDs as strings. The UUIDs from command line or fstab(5) are not converted to internal binary representation. The string representation of the UUID should be based on lower case characters.

The proc filesystem is not associated with a special device, and when mounting it, an arbitrary keyword, such as proc can be used instead of a device specification. (The customary choice none is less fortunate: the error message `none busy' from umount can be confusing.)

The /etc/fstab, /etc/mtab and /proc/mounts files.

The file /etc/fstab (see fstab(5)), may contain lines describing what devices are usually mounted where, using which options. The default location of the fstab(5) file could be overridden by --fstab <path> command line option (see below for more details).

The command

mount -a [-t type] [-O optlist]

(usually given in a bootscript) causes all filesystems mentioned in fstab (of the proper type and/or having or not having the proper options) to be mounted as indicated, except for those whose line contains the noauto keyword. Adding the -F option will make mount fork, so that the filesystems are mounted simultaneously.

When mounting a filesystem mentioned in fstab or mtab, it suffices to give only the device, or only the mount point.

The programs mount and umount maintain a list of currently mounted filesystems in the file /etc/mtab. If no arguments are given to mount, this list is printed.

The mount program does not read the /etc/fstab file if device

(or LABEL, UUID, PARTUUID or PARTLABEL) and dir are specified. For example:

mount /dev/foo /dir

If you want to override mount options from /etc/fstab you have to use:

mount device|dir -o <options>

and then the mount options from command line will be appended to the list of options from /etc/fstab. The usual behaviour is that the last option wins if there is more duplicated options.

When the proc filesystem is mounted (say at /proc), the files /etc/mtab and /proc/mounts have very similar contents. The former has somewhat more information, such as the mount options used, but is not necessarily up-to-date (cf. the -n option below). It is possible to replace /etc/mtab by a symbolic link to /proc/mounts, and especially when you have very large numbers of mounts things will be much faster with that symlink, but some information is lost that way, and in particular using the "user" option will fail.

The non-superuser mounts.

Normally, only the superuser can mount filesystems. However, when fstab contains the user option on a line, anybody can mount the corresponding system.

Thus, given a line

/dev/cdrom /cd iso9660 ro,user,noauto,unhide

any user can mount the iso9660 filesystem found on his CDROM using the command mount /dev/cdrom

or

mount /cd

For more details, see fstab(5). Only the user that mounted a filesystem can unmount it again. If any user should be able to unmount, then use users instead of user in the fstab line. The owner option is similar to the user option, with the restriction that the user must be the owner of the special file. This may be useful e.g. for /dev/fd if a login script makes the console user owner of this device. The group option is similar, with the restriction that the user must be member of the group of the special file.

The bind mounts.

Since Linux 2.4.0 it is possible to remount part of the file hierarchy somewhere else. The call is

mount --bind olddir newdir

or shortoption

mount -B olddir newdir

or fstab entry is:

/olddir /newdir none bind

After this call the same contents is accessible in two places. One can also remount a single file (on a single file). It's also possible to use the bind mount to create a mountpoint from a regular directory, for example:

mount --bind foo foo

The bind mount call attaches only (part of) a single filesystem, not possible submounts. The entire file hierarchy including submounts is attached a second place using

mount --rbind olddir newdir

or shortoption

mount -R olddir newdir

Note that the filesystem mount options will remain the same as those on the original mount point.

mount(8) since v2.27 (backported to RHEL7.3) allow to change the options by passing the -o option along with --bind for example:

mount --bind,ro foo foo

This feature is not supported by Linux kernel and it is implemented in userspace by additional remount mount(2) syscall. This solution is not atomic.

The alternative (classic) way to create a read-only bind mount is to use remount operation, for example:

mount --bind olddir newdir
mount -o remount,ro,bind olddir newdir

Note that read-only bind will create a read-only mountpoint (VFS entry), but the original filesystem superblock will still be writable, meaning that the olddir will be writable, but the newdir will be read-only.

It's impossible to change mount options recursively (for example with -o rbind,ro).

The move operation.

Since Linux 2.5.1 it is possible to atomically move a mounted tree to another place. The call is

mount --move olddir newdir

or shortoption

mount -M olddir newdir

This will cause the contents which previously appeared under olddir to be accessed under newdir. The physical location of the files is not changed. Note that the olddir has to be a mountpoint.

Note that moving a mount residing under a shared mount is invalid and unsupported. Use findmnt -o TARGET,PROPAGATION to see the current propagation flags.

The shared subtrees operations.

Since Linux 2.6.15 it is possible to mark a mount and its submounts as shared, private, slave or unbindable. A shared mount provides ability to create mirrors of that mount such that mounts and umounts within any of the mirrors propagate to the other mirror. A slave mount receives propagation from its master, but any not vice-versa. A private mount carries no propagation abilities. A unbindable mount is a private mount which cannot be cloned through a bind operation. Detailed semantics is documented in Documentation/filesystems/sharedsubtree.txt file in the kernel source tree.

Supported operations:

mount --make-shared mountpoint

mount --make-slave mountpoint

mount --make-private mountpoint

mount --make-unbindable mountpoint

The following commands allows one to recursively change the type of all the mounts under a given mountpoint.

mount --make-rshared mountpoint

mount --make-rslave mountpoint

mount --make-rprivate mountpoint

mount --make-runbindable mountpoint

mount(8) does not read fstab(5) when --make-* operation is requested. All necessary information has to be specified on command line.

Note that Linux kernel does not allow to change more propagation flags by one mount(2) syscall and the flags cannot be mixed with another mount options.

Since util-linux 2.23 mount command allows to use more propagation flags together and with another mount operations. This feature is EXPERIMENTAL. The propagation flags are applied by additional mount(2) syscalls after previous successful mount operation. Note that this use case is not atomic. The propagation flags is possible to specify in fstab(5) as mount options (private, slave, shared, unbindable, rprivate, rslave, rshared, runbindable).

For example

mount --make-private --make-unbindable /dev/sda1 /A

is the same as

mount /dev/sda1 /A

mount --make-private /A

mount --make-unbindable /A

COMMAND LINE OPTIONS

The full set of mount options used by an invocation of mount is determined by first extracting the mount options for the filesystem from the fstab table, then applying any options specified by the -o argument, and finally applying a -r or -w option, when present.

Command line options available for the mount command:

-V, --version
Output version.

-h, --help
Print a help message.

-v, --verbose
Verbose mode.

-a, --all
Mount all filesystems (of the given types) mentioned in fstab.

-F, --fork
(Used in conjunction with -a.) Fork off a new incarnation of mount for each device. This will do the mounts on different devices or different NFS servers in parallel. This has the advantage that it is faster; also NFS timeouts go in parallel. A disadvantage is that the mounts are done in undefined order. Thus, you cannot use this option if you want to mount both /usr and /usr/spool.

-f, --fake
Causes everything to be done except for the actual system call; if it's not obvious, this ``fakes'' mounting the filesystem. This option is useful in conjunction with the -v flag to determine what the mount command is trying to do. It can also be used to add entries for devices that were mounted earlier with the -n option. The -f option checks for existing record in /etc/mtab and fails when the record already exists (with regular non-fake mount, this check is done by kernel).

-i, --internal-only
Don't call the /sbin/mount.<filesystem> helper even if it exists.

-l, --show-labels
Add the labels in the mount output. Mount must have permission to read the disk device (e.g. be suid root) for this to work. One can set such a label for ext2, ext3 or ext4 using the e2label(8) utility, or for XFS using xfs_admin(8), or for reiserfs using reiserfstune(8).

-n, --no-mtab
Mount without writing in /etc/mtab. This is necessary for example when /etc is on a read-only filesystem.

-c, --no-canonicalize
Don't canonicalize paths. The mount command canonicalizes all paths (from command line or fstab) and stores canonicalized paths to the /etc/mtab file. This option can be used together with the -f flag for already canonicalized absolute paths.
-s Tolerate sloppy mount options rather than failing. This will ignore mount options not supported by a filesystem type. Not all filesystems support this option. This option exists for support

of the Linux autofs-based automounter.

--source src
If only one argument for the mount command is given then the argument might be interpreted as target (mountpoint) or source (device). This option allows to explicitly define that the argument is mount source.

-r, --read-only
Mount the filesystem read-only. A synonym is -o ro.

Note that, depending on the filesystem type, state and kernel behavior, the system may still write to the device. For example, Ext3 or ext4 will replay its journal if the filesystem is dirty. To prevent this kind of write access, you may want to mount ext3 or ext4 filesystem with "ro,noload" mount options or set the block device to read-only mode, see command blockdev(8).

-w, --rw, --read-write
Mount the filesystem read/write. This is the default. A synonym is -o rw.

-L, --label label
Mount the partition that has the specified label.

-U, --uuid uuid
Mount the partition that has the specified uuid. These two options require the file /proc/partitions (present since Linux 2.1.116) to exist.

-T, --fstab path
Specifies alternative fstab file. If the path is directory then the files in the directory are sorted by strverscmp(3), files that starts with "." or without .fstab extension are ignored. The option can be specified more than once. This option is mostly designed for initramfs or chroot scripts where additional configuration is specified outside standard system configuration.

Note that mount(8) does not pass the option --fstab to /sbin/mount.<type> helpers, it means that the alternative fstab files will be invisible for the helpers. This is no problem for normal mounts, but user (non-root) mounts always require fstab
to verify user's rights.

-t, --types vfstype
The argument following the -t is used to indicate the filesystem type. The filesystem types which are currently supported include: adfs, affs, autofs, cifs, coda, coherent, cramfs, debugfs, devpts, efs, ext, ext2, ext3, ext4, hfs, hfsplus, hpfs, iso9660, jfs, minix, msdos, ncpfs, nfs, nfs4, ntfs, proc, qnx4, ramfs, reiserfs, romfs, squashfs, smbfs, sysv, tmpfs, ubifs, udf, ufs, umsdos, usbfs, vfat, xenix, xfs, xiafs. Note that coherent, sysv and xenix are equivalent and that xenix and coherent will be removed at some point in the future — use sysv instead. Since kernel version 2.1.21 the types ext and xiafs do not exist anymore. Earlier, usbfs was known as usbdevfs. Note,

the real list of all supported filesystems depends on your kernel.

The programs mount and umount support filesystem subtypes. The subtype is defined by '.subtype' suffix. For example 'fuse.sshfs'. It's recommended to use subtype notation rather than add any prefix to the mount source (for example 'sshfs#example.com' is depreacated).

For most types all the mount program has to do is issue a simple mount(2) system call, and no detailed knowledge of the filesystem type is required. For a few types however (like nfs, nfs4, cifs, smbfs, ncpfs) ad hoc code is necessary. The nfs, nfs4, cifs, smbfs, and ncpfs filesystems have a separate mount program. In order to make it possible to treat all types in a uniform way, mount will execute the program /sbin/mount.TYPE (if that exists) when called with type TYPE. Since various versions of the smbmount program have different calling conventions, /sbin/mount.smbfs may have to be a shell script that sets up the desired call.

If no -t option is given, or if the auto type is specified, mount will try to guess the desired type. Mount uses the blkid library for guessing the filesystem type; if that does not turn up anything that looks familiar, mount will try to read the file /etc/filesystems, or, if that does not exist, /proc/filesystems. All of the filesystem types listed there will be tried, except for those that are labeled "nodev" (e.g., devpts, proc and nfs). If /etc/filesystems ends in a line with a single * only, mount will read /proc/filesystems afterwards. All of the filesystem types will be mounted with mount option "silent".

The auto type may be useful for user-mounted floppies. Creating a file /etc/filesystems can be useful to change the probe order (e.g., to try vfat before msdos or ext3 before ext2) or if you use a kernel module autoloader.

More than one type may be specified in a comma separated list. The list of filesystem types can be prefixed with no to specify the filesystem types on which no action should be taken. (This can be meaningful with the -a option.) For example, the command:

mount -a -t nomsdos,ext

mounts all filesystems except those of type msdos and ext.

--target dir

If only one argument for the mount command is given then the argument might be interpreted as target (mountpoint) or source (device). This option allows to explicitly define that the argument is mount target.

-O, --test-opts opts

Used in conjunction with -a, to limit the set of filesystems to which the -a is applied. Like -t in this regard except that it is useless except in the context of -a. For example, the command:

mount -a -O no_netdev

mounts all filesystems except those which have the option _net-
dev specified in the options field in the /etc/fstab file.

It is different from -t in that each option is matched exactly;
a leading no at the beginning of one option does not negate the
rest.

The -t and -O options are cumulative in effect; that is, the command

mount -a -t ext2 -O _netdev

mounts all ext2 filesystems with the _netdev option, not all
filesystems that are either ext2 or have the _netdev option
specified.

-o, --options opts
Options are specified with a -o flag followed by a comma sepa-
rated string of options. For example:

mount LABEL=mydisk -o noatime,nouser

For more details, see FILESYSTEM INDEPENDENT MOUNT OPTIONS and
FILESYSTEM SPECIFIC MOUNT OPTIONS sections.

-B, --bind
Remount a subtree somewhere else (so that its contents are
available in both places). See above.

-R, --rbind
Remount a subtree and all possible submounts somewhere else (so
that its contents are available in both places). See above.

-M, --move
Move a subtree to some other place. See above.

FILESYSTEM INDEPENDENT MOUNT OPTIONS

Some of these options are only useful when they appear in the
/etc/fstab file.

Some of these options could be enabled or disabled by default in the
system kernel. To check the current setting see the options in
/proc/mounts. Note that filesystems also have per-filesystem specific
default mount options (see for example tune2fs -l output for extN
filesystems).

The following options apply to any filesystem that is being mounted
(but not every filesystem actually honors them - e.g., the sync option
today has effect only for ext2, ext3, fat, vfat and ufs):

async All I/O to the filesystem should be done asynchronously. (See
also the sync option.)

atime Do not use noatime feature, then the inode access time is con-
trolled by kernel defaults. See also the description for stric-
tatime and relatime mount options.

noatime
Do not update inode access times on this filesystem (e.g., for faster access on the news spool to speed up news servers).

auto Can be mounted with the -a option.

noauto Can only be mounted explicitly (i.e., the -a option will not cause the filesystem to be mounted).

context=context, fscontext=context, defcontext=context and rootcontext=context
The context= option is useful when mounting filesystems that do not support extended attributes, such as a floppy or hard disk formatted with VFAT, or systems that are not normally running under SELinux, such as an ext3 formatted disk from a non-SELinux workstation. You can also use context= on filesystems you do not trust, such as a floppy. It also helps in compatibility with xattr-supporting filesystems on earlier 2.4.<x> kernel versions. Even where xattrs are supported, you can save time not having to label every file by assigning the entire disk one security context.

A commonly used option for removable media is context="system_u:object_r:removable_t".
Two other options are fscontext= and defcontext=, both of which are mutually exclusive of the context option. This means you can use fscontext and defcontext with each other, but neither can be used with context.

The fscontext= option works for all filesystems, regardless of their xattr support. The fscontext option sets the overarching filesystem label to a specific security context. This filesystem label is separate from the individual labels on the files. It represents the entire filesystem for certain kinds of permission checks, such as during mount or file creation. Individual file labels are still obtained from the xattrs on the files themselves. The context option actually sets the aggregate context that fscontext provides, in addition to supplying the same label for individual files.

You can set the default security context for unlabeled files using defcontext= option. This overrides the value set for unlabeled files in the policy and requires a filesystem that supports xattr labeling.

The rootcontext= option allows you to explicitly label the root inode of a FS being mounted before that FS or inode becomes visible to userspace. This was found to be useful for things like stateless linux.

Note that the kernel rejects any remount request that includes the context option, even when unchanged from the current context.

Warning: the context value might contain commas, in which case the value has to be properly quoted, otherwise mount(8) will interpret the comma as a separator between mount options. Don't

forget that the shell strips off quotes and thus double quoting is required. For example:

mount -t tmpfs none /mnt -o 'context="system_u:object_r:tmp_t:s0:c127,c456",noexec'

For more details, see selinux(8).

Defaults Use default options: rw, suid, dev, exec, auto, nouser, and async.

Note that the real set of the all default mount options depends on kernel and filesystem type. See the begin of this section for more details.

dev Interpret character or block special devices on the filesystem.

nodev Do not interpret character or block special devices on the file system.

diratime
Update directory inode access times on this filesystem. This is the default.

nodiratime
Do not update directory inode access times on this filesystem.

dirsync
All directory updates within the filesystem should be done synchronously. This affects the following system calls: creat, link, unlink, symlink, mkdir, rmdir, mknod and rename.

exec Permit execution of binaries.noexec Do not allow direct execution of any binaries on the mounted filesystem. (Until recently it was possible to run binaries anyway using a command like /lib/ld*.so /mnt/binary. This trick fails since Linux 2.4.25 / 2.6.0.)

group Allow an ordinary (i.e., non-root) user to mount the filesystem if one of his groups matches the group of the device. This option implies the options nosuid and nodev (unless overridden by subsequent options, as in the option line group,dev,suid).

iversion
Every time the inode is modified, the i_version field will be incremented.

noiversion
Do not increment the i_version inode field.

mand Allow mandatory locks on this filesystem. See fcntl(2).

nomand Do not allow mandatory locks on this filesystem.

_netdev
The filesystem resides on a device that requires network access (used to prevent the system from attempting to mount these filesystems until the network has been enabled on the system).

nofail Do not report errors for this device if it does not exist.

relatime
Update inode access times relative to modify or change time. Access time is only updated if the previous access time was earlier than the current modify or change time. (Similar to noatime, but doesn't break mutt or other applications that need to know if a file has been read since the last time it was modified.)

Since Linux 2.6.30, the kernel defaults to the behavior provided by this option (unless noatime was specified), and the strictatime option is required to obtain traditional semantics. In addition, since Linux 2.6.30, the file's last access time is always updated if it is more than 1 day old.

norelatime
Do not use relatime feature. See also the strictatime mount option. strictatime
Allows to explicitly requesting full atime updates. This makes it possible for kernel to defaults to relatime or noatime but still allow userspace to override it. For more details about the default system mount options see /proc/mounts.

nostrictatime
Use the kernel's default behaviour for inode access time updates.

suid Allow set-user-identifier or set-group-identifier bits to take effect.

nosuid Do not allow set-user-identifier or set-group-identifier bits to take effect. (This seems safe, but is in fact rather unsafe if you have suidperl(1) installed.)

silent Turn on the silent flag.

loud Turn off the silent flag.

owner Allow an ordinary (i.e., non-root) user to mount the filesystem if he is the owner of the device. This option implies the options nosuid and nodev (unless overridden by subsequent options, as in the option line owner,dev,suid).

remount
Attempt to remount an already-mounted filesystem. This is commonly used to change the mount flags for a filesystem, especially to make a readonly filesystem writable. It does not change device or mount point.

The remount functionality follows the standard way how the mount command works with options from fstab. It means the mount command doesn't read fstab (or mtab) only when a device and dir are fully specified.

mount -o remount,rw /dev/foo /dir

After this call all old mount options are replaced and arbitrary stuff from fstab is ignored, except the loop= option which is internally generated and maintained by the mount command.

mount -o remount,rw /dir

After this call mount reads fstab (or mtab) and merges these options with options from command line (-o).

ro Mount the filesystem read-only.

rw Mount the filesystem read-write.

sync All I/O to the filesystem should be done synchronously. In case of media with limited number of write cycles (e.g. some flash drives) "sync" may cause life-cycle shortening.

user Allow an ordinary user to mount the filesystem. The name of the mounting user is written to mtab so that he can unmount the filesystem again. This option implies the options noexec, nosuid, and nodev (unless overridden by subsequent options, as in the option line user,exec,dev,suid).

nouser Forbid an ordinary (i.e., non-root) user to mount the filesystem. This is the default.

users Allow every user to mount and unmount the filesystem. This option implies the options noexec, nosuid, and nodev (unless overridden by subsequent options, as in the option line users,exec,dev,suid).

x-* All options prefixed with "x-" are interpreted as comments or userspace applications specific options. These options are not stored to mtab file, send to mount.<type> helpers or mount(2) system call. The suggested format is x-<appname>.<option> (e.g. x-systemd.automount).

x-mount.mkdir[=<mode>]
Allow to make a target directory (mountpoint). The optional argument <mode> specifies the file system access mode used for mkdir (2) in octal notation. The default mode is 0755. This functionality is supported only for root users.

FILESYSTEM SPECIFIC MOUNT OPTIONS

The following options apply only to certain filesystems. We sort them by filesystem. They all follow the -o flag.

What options are supported depends a bit on the running kernel. More info may be found in the kernel source subdirectory Documentation/filesystems.

Mount options for adfs

uid=value and gid=value
Set the owner and group of the files in the filesystem (default: uid=gid=0).

ownmask=value and othmask=value
Set the permission mask for ADFS 'owner' permissions and 'other'

permissions, respectively (default: 0700 and 0077, respectively). See also /usr/src/linux/Documentation/filesystems/adfs.txt.

Mount options for affs

uid=value and gid=value

Set the owner and group of the root of the filesystem (default: uid=gid=0, but with option uid or gid without specified value, the uid and gid of the current process are taken).

setuid=value and setgid=value

Set the owner and group of all files.

mode=value

Set the mode of all files to value & 0777 disregarding the original permissions. Add search permission to directories that have read permission. The value is given in octal.

protect

Do not allow any changes to the protection bits on the filesystem.

usemp Set uid and gid of the root of the filesystem to the uid and gid of the mount point upon the first sync or umount, and then clear this option. Strange...

verbose

Print an informational message for each successful mount.

prefix=string

Prefix used before volume name, when following a link.

volume=string

Prefix (of length at most 30) used before '/' when following a symbolic link.

reserved=value

(Default: 2.) Number of unused blocks at the start of the device.

root=value

Give explicitly the location of the root block.

bs=value

Give blocksize. Allowed values are 512, 1024, 2048, 4096.

grpquota|noquota|quota|usrquota

These options are accepted but ignored. (However, quota utilities may react to such strings in /etc/fstab.)

Mount options for cifs

See the options section of the mount.cifs(8) man page (cifs-utils package must be installed).

Mount options for coherent

None.

Mount options for debugfs

The debugfs filesystem is a pseudo filesystem, traditionally mounted on /sys/kernel/debug. As of kernel version 3.4, debugfs has the following options:

uid=n, gid=n
Set the owner and group of the mountpoint.

mode=value
Sets the mode of the mountpoint.

Mount options for devpts

The devpts filesystem is a pseudo filesystem, traditionally mounted on /dev/pts. In order to acquire a pseudo terminal, a process opens /dev/ptmx; the number of the pseudo terminal is then made available to the process and the pseudo terminal slave can be accessed as /dev/pts/<number>.

uid=value and gid=value
This sets the owner or the group of newly created PTYs to the specified values. When nothing is specified, they will be set to the UID and GID of the creating process. For example, if t is a tty group with GID 5, then gid=5 will cause newly created
PTYs to belong to the tty group.

mode=value
Set the mode of newly created PTYs to the specified value. The default is 0600. A value of mode=620 and gid=5 makes "mesg y" the default on newly created PTYs.

newinstance
Create a private instance of devpts filesystem, such that indices of ptys allocated in this new instance are independent of indices created in other instances of devpts.

All mounts of devpts without this newinstance option share the same set of pty indices (i.e legacy mode). Each mount of devpts with the newinstance option has a private set of pty indices.

This option is mainly used to support containers in the linux kernel. It is implemented in linux kernel versions starting with 2.6.29. Further, this mount option is valid only if CONFIG_DEVPTS_MULTIPLE_INSTANCES is enabled in the kernel configuration.
To use this option effectively, /dev/ptmx must be a symbolic link to pts/ptmx. See Documentation/filesystems/devpts.txt in the linux kernel source tree for details.

ptmxmode=value

Set the mode for the new ptmx device node in the devpts filesystem.

With the support for multiple instances of devpts (see newinstance option above), each instance has a private ptmx node in the root of the devpts filesystem (typically /dev/pts/ptmx).

For compatibility with older versions of the kernel, the default

mode of the new ptmx node is 0000. ptmxmode=value specifies a more useful mode for the ptmx node and is highly recommended when the newinstance option is specified.

This option is only implemented in linux kernel versions starting with 2.6.29. Further this option is valid only if CONFIG_DEVPTS_MULTIPLE_INSTANCES is enabled in the kernel configuration.

Mount options for ext

None. Note that the `ext' filesystem is obsolete. Don't use it. Since Linux version 2.1.21 extfs is no longer part of the kernel source.

Mount options for ext2

The `ext2' filesystem is the standard Linux filesystem. Since Linux 2.5.46, for most mount options the default is determined by the filesystem superblock. Set them with tune2fs(8).

acl|noacl

Support POSIX Access Control Lists (or not).

bsddf|minixdf

Set the behaviour for the statfs system call. The minixdf behaviour is to return in the f_blocks field the total number of blocks of the filesystem, while the bsddf behaviour (which is the default) is to subtract the overhead blocks used by the ext2 filesystem and not available for file storage. Thus

```
% mount /k -o minixdf; df /k; umount /k
Filesystem   1024-blocks  Used Available Capacity Mounted on
/dev/sda6      2630655   86954  2412169      3%   /k
% mount /k -o bsddf; df /k; umount /k
Filesystem   1024-blocks  Used Available Capacity Mounted on
/dev/sda6      2543714      13  2412169      0%   /k
```

(Note that this example shows that one can add command line options to the options given in /etc/fstab.)

check=none or nocheck

No checking is done at mount time. This is the default. This is fast. It is wise to invoke e2fsck(8) every now and then, e.g. at boot time. The non-default behavior is unsupported (check=normal and check=strict options have been removed). Note that these mount options don't have to be supported if ext4 kernel driver is used for ext2 and ext3 filesystems.

debug Print debugging info upon each (re)mount.

errors={continue|remount-ro|panic}

Define the behaviour when an error is encountered. (Either ignore errors and just mark the filesystem erroneous and continue, or remount the filesystem read-only, or panic and halt the system.) The default is set in the filesystem superblock, and can be changed using tune2fs(8).

grpid|bsdgroups and nogrpid|sysvgroups

These options define what group id a newly created file gets. When grpid is set, it takes the group id of the directory in which it is created; otherwise (the default) it takes the fsgid

of the current process, unless the directory has the setgid bit set, in which case it takes the gid from the parent directory, and also gets the setgid bit set if it is a directory itself.

grpquota|noquota|quota|usrquota
The usrquota (same as quota) mount option enables user quota support on the filesystem. grpquota enables group quotas support. You need the quota utilities to actually enable and manage the quota system.

nouid32
Disables 32-bit UIDs and GIDs. This is for interoperability with older kernels which only store and expect 16-bit values.

oldalloc or orlov
Use old allocator or Orlov allocator for new inodes. Orlov is default.

resgid=n and resuid=n
The ext2 filesystem reserves a certain percentage of the available space (by default 5%, see mke2fs(8) and tune2fs(8)). These options determine who can use the reserved blocks. (Roughly: whoever has the specified uid, or belongs to the specified group.)

sb=n Instead of block 1, use block n as superblock. This could be useful when the filesystem has been damaged. (Earlier, copies of the superblock would be made every 8192 blocks: in block 1, 8193, 16385, ... (and one got thousands of copies on a big filesystem). Since version 1.08, mke2fs has a -s (sparse superblock) option to reduce the number of backup superblocks, and since version 1.15 this is the default. Note that this may mean that ext2 filesystems created by a recent mke2fs cannot be mounted r/w under Linux 2.0.*.) The block number here uses 1k units. Thus, if you want to use logical block 32768 on a filesystem with 4k blocks, use "sb=131072".

user_xattr|nouser_xattr
Support "user." extended attributes (or not).

Mount options for ext3
The ext3 filesystem is a version of the ext2 filesystem which has been enhanced with journaling. It supports the same options as ext2 as well as the following additions:

journal=update
Update the ext3 filesystem's journal to the current format.

journal=inum
When a journal already exists, this option is ignored. Otherwise, it specifies the number of the inode which will represent the ext3 filesystem's journal file; ext3 will create a new journal, overwriting the old contents of the file whose inode number is inum.

journal_dev=devnum
When the external journal device's major/minor numbers have

changed, this option allows the user to specify the new journal location. The journal device is identified through its new major/minor numbers encoded in devnum.

norecovery/noload
Don't load the journal on mounting. Note that if the filesystem was not unmounted cleanly, skipping the journal replay will lead to the filesystem containing inconsistencies that can lead to any number of problems.

data={journal|ordered|writeback}
Specifies the journaling mode for file data. Metadata is always journaled. To use modes other than ordered on the root filesystem, pass the mode to the kernel as boot parameter, e.g. rootflags=data=journal.

journal
All data is committed into the journal prior to being written into the main filesystem.

ordered
This is the default mode. All data is forced directly out to the main file system prior to its metadata being committed to the journal.

writeback
Data ordering is not preserved - data may be written into the main filesystem after its metadata has been committed to the journal. This is rumoured to be the highest-throughput option. It guarantees internal filesystem integrity, however it can allow old data to appear in files after a crash and journal recovery.

barrier=0 / barrier=1
This enables/disables barriers. barrier=0 disables it, barrier=1 enables it. Write barriers enforce proper on-disk ordering of journal commits, making volatile disk write caches safe to use, at some performance penalty. The ext3 filesystem does not enable write barriers by default. Be sure to enable barriers unless your disks are battery-backed one way or another. Otherwise you risk filesystem corruption in case of power failure.

commit=nrsec
Sync all data and metadata every nrsec seconds. The default value is 5 seconds. Zero means default.

user_xattr
Enable Extended User Attributes. See the attr(5) manual page.

acl Enable POSIX Access Control Lists. See the acl(5) manual page.

usrjquota=aquota.user|grpjquota=aquota.group|jqfmt=vfsv0
Apart from the old quota system (as in ext2, jqfmt=vfsold aka
version 1 quota) ext3 also supports journaled quotas (version 2 quota). jqfmt=vfsv0 enables journaled quotas. For journaled quo-

tas the mount options usrjquota=aquota.user and grpjquota=aquota.group are required to tell the quota system which quota database files to use. Journaled quotas have the advantage that even after a crash no quota check is required.

Mount options for ext4

The ext4 filesystem is an advanced level of the ext3 filesystem which incorporates scalability and reliability enhancements for supporting large filesystem.

The options journal_dev, noload, data, commit, orlov, oldalloc, [no]user_xattr [no]acl, bsddf, minixdf, debug, errors, data_err, grpid, bsdgroups, nogrpid sysvgroups, resgid, resuid, sb, quota, noquota, grpquota, usrquota usrjquota, grpjquota and jqfmt are backwardly compatible with ext3 or ext2.

journal_checksum

Enable checksumming of the journal transactions. This will allow the recovery code in e2fsck and the kernel to detect corruption in the kernel. It is a compatible change and will be ignored by older kernels.

journal_async_commit

Commit block can be written to disk without waiting for descriptor blocks. If enabled older kernels cannot mount the device. This will enable 'journal_checksum' internally.

barrier=0 / barrier=1 / barrier / nobarrier

This enables/disables the use of write barriers in the jbd code. barrier=0 disables, barrier=1 enables. This also requires an IO stack which can support barriers, and if jbd gets an error on a barrier write, it will disable again with a warning. Write barriers enforce proper on-disk ordering of journal commits, making volatile disk write caches safe to use, at some performance penalty. If your disks are battery-backed in one way or another, disabling barriers may safely improve performance. The mount options "barrier" and "nobarrier" can also be used to enable or disable barriers, for consistency with other ext4 mount options.

The ext4 filesystem enables write barriers by default.

inode_readahead_blks=n

This tuning parameter controls the maximum number of inode table blocks that ext4's inode table readahead algorithm will pre-read into the buffer cache. The value must be a power of 2. The default value is 32 blocks.

stripe=n

Number of filesystem blocks that mballoc will try to use for allocation size and alignment. For RAID5/6 systems this should be the number of data disks * RAID chunk size in filesystem blocks.

delalloc

Deferring block allocation until write-out time.

nodelalloc

Disable delayed allocation. Blocks are allocated when data is copied from user to page cache.

max_batch_time=usec
Maximum amount of time ext4 should wait for additional filesystem operations to be batch together with a synchronous write operation. Since a synchronous write operation is going to force a commit and then a wait for the I/O complete, it doesn't cost much, and can be a huge throughput win, we wait for a small amount of time to see if any other transactions can piggyback on the synchronous write. The algorithm used is designed to automatically tune for the speed of the disk, by measuring the amount of time (on average) that it takes to finish committing a transaction. Call this time the "commit time". If the time that the transaction has been running is less than the commit time, ext4 will try sleeping for the commit time to see if other operations will join the transaction. The commit time is capped by the max_batch_time, which defaults to 15000us (15ms). This optimization can be turned off entirely by setting max_batch_time to 0.

min_batch_time=usec
This parameter sets the commit time (as described above) to be at least min_batch_time. It defaults to zero microseconds. Increasing this parameter may improve the throughput of multi-threaded, synchronous workloads on very fast disks, at the cost of increasing latency.

journal_ioprio=prio
The I/O priority (from 0 to 7, where 0 is the highest priority) which should be used for I/O operations submitted by kjournald2 during a commit operation. This defaults to 3, which is a slightly higher priority than the default I/O priority.

abort Simulate the effects of calling ext4_abort() for debugging purposes. This is normally used while remounting a filesystem which is already mounted.

auto_da_alloc|noauto_da_alloc
Many broken applications don't use fsync() when replacing existing files via patterns such as

fd = open("foo.new")/write(fd,..)/close(fd)/ rename("foo.new", "foo")

or worse yet

fd = open("foo", O_TRUNC)/write(fd,..)/close(fd).
If auto_da_alloc is enabled, ext4 will detect the replace-via-rename and replace-via-truncate patterns and force that any delayed allocation blocks are allocated such that at the next journal commit, in the default data=ordered mode, the data blocks of the new file are forced to disk before the rename() operation is committed. This provides roughly the same level of guarantees as ext3, and avoids the "zero-length" problem that can happen when a system crashes before the delayed allocation blocks are forced to disk.

discard/nodiscard
Controls whether ext4 should issue discard/TRIM commands to the underlying block device when blocks are freed. This is useful for SSD devices and sparse/thinly-provisioned LUNs, but it is off by default until sufficient testing has been done.

nouid32
Disables 32-bit UIDs and GIDs. This is for interoperability with older kernels which only store and expect 16-bit values.

resize Allows to resize filesystem to the end of the last existing block group, further resize has to be done with resize2fs either online, or offline. It can be used only with conjunction with remount.

block_validity/noblock_validity
This options allows to enables/disables the in-kernel facility for tracking filesystem metadata blocks within internal data structures. This allows multi- block allocator and other routines to quickly locate extents which might overlap with filesystem metadata blocks. This option is intended for debugging purposes and since it negatively affects the performance, it is off by default.

dioread_lock/dioread_nolock
Controls whether or not ext4 should use the DIO read locking. If the dioread_nolock option is specified ext4 will allocate uninitialized extent before buffer write and convert the extent to initialized after IO completes. This approach allows ext4 code to avoid using inode mutex, which improves scalability on high speed storages. However this does not work with data journaling and dioread_nolock option will be ignored with kernel warning. Note that dioread_nolock code path is only used for extent-based files. Because of the restrictions this options comprises it is off by default (e.g. dioread_lock).

i_version
Enable 64-bit inode version support. This option is off by default.

Mount options for fat
(Note: fat is not a separate filesystem, but a common part of the msdos, umsdos and vfat filesystems.)

blocksize={512|1024|2048}
Set blocksize (default 512). This option is obsolete.

uid=value and gid=value
Set the owner and group of all files. (Default: the uid and gid of the current process.)

umask=value
Set the umask (the bitmask of the permissions that are not present). The default is the umask of the current process. The value is given in octal.
dmask=value

Set the umask applied to directories only. The default is the umask of the current process. The value is given in octal.

fmask=value
Set the umask applied to regular files only. The default is the umask of the current process. The value is given in octal.

allow_utime=value
This option controls the permission check of mtime/atime.

20 If current process is in group of file's group ID, you can change timestamp.

2 Other users can change timestamp.

The default is set from `dmask' option. (If the directory is writable, utime(2) is also allowed. I.e. ~dmask & 022)

Normally utime(2) checks current process is owner of the file, or it has CAP_FOWNER capability. But FAT filesystem doesn't have uid/gid on disk, so normal check is too inflexible. With this option you can relax it.

check=value
Three different levels of pickyness can be chosen:
r[elaxed]
Upper and lower case are accepted and equivalent, long name parts are truncated (e.g. verylongname.foobar becomes verylong.foo), leading and embedded spaces are accepted in each name part (name and extension).

n[ormal]
Like "relaxed", but many special characters (*, ?, <, spaces, etc.) are rejected. This is the default.

s[trict]
Like "normal", but names may not contain long parts and special characters that are sometimes used on Linux, but are not accepted by MS-DOS are rejected. (+, =, spaces, etc.)

codepage=value
Sets the codepage for converting to shortname characters on FAT and VFAT filesystems. By default, codepage 437 is used.

conv={b[inary]|t[ext]|a[uto]}
The fat filesystem can perform CRLF<-->NL (MS-DOS text format to UNIX text format) conversion in the kernel. The following conversion modes are available:
binary no translation is performed. This is the default.

text CRLF<-->NL translation is performed on all files.

auto CRLF<-->NL translation is performed on all files that don't have a "well-known binary" extension. The list of known extensions can be found at the beginning of fs/fat/misc.c (as of 2.0, the list is: exe, com, bin,

app, sys, drv, ovl, ovr, obj, lib, dll, pif, arc, zip, lha, lzh, zoo, tar, z, arj, tz, taz, tzp, tpz, gz, tgz, deb, gif, bmp, tif, gl, jpg, pcx, tfm, vf, gf, pk, pxl, dvi).

Programs that do computed lseeks won't like in-kernel text conversion. Several people have had their data ruined by this translation. Beware!

For filesystems mounted in binary mode, a conversion tool (fromdos/todos) is available. This option is obsolete.

cvf_format=module
Forces the driver to use the CVF (Compressed Volume File) module cvf_module instead of auto-detection. If the kernel supports kmod, the cvf_format=xxx option also controls on-demand CVF module loading. This option is obsolete.

cvf_option=option
Option passed to the CVF module. This option is obsolete.

debug Turn on the debug flag. A version string and a list of filesystem parameters will be printed (these data are also printed if the parameters appear to be inconsistent).

discard
If set, causes discard/TRIM commands to be issued to the block device when blocks are freed. This is useful for SSD devices and sparse/thinly-provisioned LUNs.

fat={12|16|32}
Specify a 12, 16 or 32 bit fat. This overrides the automatic FAT type detection routine. Use with caution!

iocharset=value
Character set to use for converting between 8 bit characters and 16 bit Unicode characters. The default is iso8859-1. Long filenames are stored on disk in Unicode format.

nfs If set, enables in-memory indexing of directory inodes to reduce the frequency of ESTALE errors in NFS client operations. Useful only when the filesystem is exported via NFS.

tz=UTC This option disables the conversion of timestamps between local time (as used by Windows on FAT) and UTC (which Linux uses internally). This is particularly useful when mounting devices (like digital cameras) that are set to UTC in order to avoid the pitfalls of local time.

quiet Turn on the quiet flag. Attempts to chown or chmod files do not return errors, although they fail. Use with caution!

showexec
If set, the execute permission bits of the file will be allowed only if the extension part of the name is .EXE, .COM, or .BAT. Not set by default.

sys_immutable

If set, ATTR_SYS attribute on FAT is handled as IMMUTABLE flag on Linux. Not set by default.

flush If set, the filesystem will try to flush to disk more early than normal. Not set by default.

usefree
Use the "free clusters" value stored on FSINFO. It'll be used to determine number of free clusters without scanning disk. But it's not used by default, because recent Windows don't update it correctly in some case. If you are sure the "free clusters" on FSINFO is correct, by this option you can avoid scanning disk.

dots, nodots, dotsOK=[yes|no]
Various misguided attempts to force Unix or DOS conventions onto a FAT filesystem.

Mount options for hfs

creator=cccc, type=cccc
Set the creator/type values as shown by the MacOS finder used for creating new files. Default values: '????'.

uid=n, gid=n
Set the owner and group of all files. (Default: the uid and gid of the current process.)

dir_umask=n, file_umask=n, umask=n
Set the umask used for all directories, all regular files, or all files and directories. Defaults to the umask of the current process.

session=n
Select the CDROM session to mount. Defaults to leaving that decision to the CDROM driver. This option will fail with anything but a CDROM as underlying device.

part=n Select partition number n from the device. Only makes sense for CDROMs. Defaults to not parsing the partition table at all.

quiet Don't complain about invalid mount options.

Mount options for hpfs

uid=value and gid=value
Set the owner and group of all files. (Default: the uid and gid of the current process.)

umask=value
Set the umask (the bitmask of the permissions that are not present). The default is the umask of the current process. The value is given in octal.

case={lower|asis}
Convert all files names to lower case, or leave them. (Default: case=lower.)

conv={binary|text|auto}
For conv=text, delete some random CRs (in particular, all followed by NL) when reading a file. For conv=auto, choose more or less at random between conv=binary and conv=text. For

conv=binary, just read what is in the file. This is the default.

nocheck
Do not abort mounting when certain consistency checks fail.

Mount options for iso9660
ISO 9660 is a standard describing a filesystem structure to be used on CD-ROMs. (This filesystem type is also seen on some DVDs. See also the udf filesystem.)

Normal iso9660 filenames appear in a 8.3 format (i.e., DOS-like restrictions on filename length), and in addition all characters are in upper case. Also there is no field for file ownership, protection, number of links, provision for block/character devices, etc.

Rock Ridge is an extension to iso9660 that provides all of these UNIX-like features. Basically there are extensions to each directory record that supply all of the additional information, and when Rock Ridge is in use, the filesystem is indistinguishable from a normal UNIX filesys-
tem (except that it is read-only, of course).

norock Disable the use of Rock Ridge extensions, even if available. Cf. map.

nojoliet
Disable the use of Microsoft Joliet extensions, even if avail-able. Cf. map.

check={r[elaxed]|s[trict]}
With check=relaxed, a filename is first converted to lower case before doing the lookup. This is probably only meaningful together with norock and map=normal. (Default: check=strict.)

uid=value and gid=value
Give all files in the filesystem the indicated user or group id, possibly overriding the information found in the Rock Ridge extensions. (Default: uid=0,gid=0.)

map={n[ormal]|o[ff]|a[corn]}
For non-Rock Ridge volumes, normal name translation maps upper to lower case ASCII, drops a trailing `;1', and converts `;' to `.'. With map=off no name translation is done. See norock. (Default: map=normal.) map=acorn is like map=normal but also apply Acorn extensions if present.

mode=value
For non-Rock Ridge volumes, give all files the indicated mode. (Default: read permission for everybody.) Since Linux 2.1.37 one no longer needs to specify the mode in decimal. (Octal is indicated by a leading 0.)

unhide Also show hidden and associated files. (If the ordinary files and the associated or hidden files have the same filenames, this may make the ordinary files inaccessible.)

block={512|1024|2048}
Set the block size to the indicated value. (Default:

block=1024.)

conv={a[uto]|b[inary]|m[text]|t[ext]}
(Default: conv=binary.) Since Linux 1.3.54 this option has no effect anymore. (And non-binary settings used to be very dangerous, possibly leading to silent data corruption.)

cruft If the high byte of the file length contains other garbage, set this mount option to ignore the high order bits of the file
length. This implies that a file cannot be larger than 16MB.

session=x
Select number of session on multisession CD. (Since 2.3.4.)

sbsector=xxx
Session begins from sector xxx. (Since 2.3.4.)

The following options are the same as for vfat and specifying them only makes sense when using discs encoded using Microsoft's Joliet extensions.

iocharset=value
Character set to use for converting 16 bit Unicode characters on CD to 8 bit characters. The default is iso8859-1.

utf8 Convert 16 bit Unicode characters on CD to UTF-8.

Mount options for jfs
iocharset=name
Character set to use for converting from Unicode to ASCII. The default is to do no conversion. Use iocharset=utf8 for UTF8 translations. This requires CONFIG_NLS_UTF8 to be set in the kernel .config file.
resize=value
Resize the volume to value blocks. JFS only supports growing a volume, not shrinking it. This option is only valid during a remount, when the volume is mounted read-write. The resize keyword with no value will grow the volume to the full size of the partition.

nointegrity
Do not write to the journal. The primary use of this option is to allow for higher performance when restoring a volume from backup media. The integrity of the volume is not guaranteed if the system abnormally ends.

integrity
Default. Commit metadata changes to the journal. Use this option to remount a volume where the nointegrity option was previously specified in order to restore normal behavior.

errors={continue|remount-ro|panic}
Define the behaviour when an error is encountered. (Either ignore errors and just mark the filesystem erroneous and continue, or remount the filesystem read-only, or panic and halt the system.)

noquota|quota|usrquota|grpquota

These options are accepted but ignored.

Mount options for minix

None.

Mount options for msdos

See mount options for fat. If the msdos filesystem detects an inconsistency, it reports an error and sets the file system read-only. The filesystem can be made writable again by remounting it.

Mount options for ncpfs

Just like nfs, the ncpfs implementation expects a binary argument (a struct ncp_mount_data) to the mount system call. This argument is constructed by ncpmount(8) and the current version of mount (2.12) does not know anything about ncpfs.

Mount options for nfs and nfs4

See the options section of the nfs(5) man page (nfs-utils package must be installed).

The nfs and nfs4 implementation expects a binary argument (a struct nfs_mount_data) to the mount system call. This argument is constructed by mount.nfs(8) and the current version of mount (2.13) does not know anything about nfs and nfs4.

Mount options for ntfs

iocharset=name

Character set to use when returning file names. Unlike VFAT, NTFS suppresses names that contain nonconvertible characters. Deprecated.

nls=name

New name for the option earlier called iocharset.

utf8 Use UTF-8 for converting file names.

uni_xlate={0|1|2}

For 0 (or `no' or `false'), do not use escape sequences for unknown Unicode characters. For 1 (or `yes' or `true') or 2, use vfat-style 4-byte escape sequences starting with ":". Here 2 give a little-endian encoding and 1 a byteswapped bigendian encoding.

posix=[0|1]

If enabled (posix=1), the filesystem distinguishes between upper and lower case. The 8.3 alias names are presented as hard links instead of being suppressed. This option is obsolete.

uid=value, gid=value and umask=value

Set the file permission on the filesystem. The umask value is given in octal. By default, the files are owned by root and not readable by somebody else.

Mount options for proc

uid=value and gid=value

These options are recognized, but have no effect as far as I can see.

Mount options for ramfs

Ramfs is a memory based filesystem. Mount it and you have it. Unmount it and it is gone. Present since Linux 2.3.99pre4. There are no mount options.

Mount options for reiserfs

Reiserfs is a journaling filesystem.

conv Instructs version 3.6 reiserfs software to mount a version 3.5 filesystem, using the 3.6 format for newly created objects. This filesystem will no longer be compatible with reiserfs 3.5 tools.

hash={rupasov|tea|r5|detect}

Choose which hash function reiserfs will use to find files within directories.

rupasov

A hash invented by Yury Yu. Rupasov. It is fast and preserves locality, mapping lexicographically close file names to close hash values. This option should not be used, as it causes a high probability of hash collisions.

tea A Davis-Meyer function implemented by Jeremy Fitzhardinge. It uses hash permuting bits in the name. It gets high randomness and, therefore, low probability of hash collisions at some CPU cost. This may be used if EHASHCOLLISION errors are experienced with the r5 hash.

r5 A modified version of the rupasov hash. It is used by default and is the best choice unless the filesystem has huge directories and unusual file-name patterns.

detect Instructs mount to detect which hash function is in use by examining the filesystem being mounted, and to write this information into the reiserfs superblock. This is only useful on the first mount of an old format filesystem.

hashed_relocation

Tunes the block allocator. This may provide performance improvements in some situations.

no_unhashed_relocation

Tunes the block allocator. This may provide performance improvements in some situations.

noborder

Disable the border allocator algorithm invented by Yury Yu. Rupasov. This may provide performance improvements in some situations.

nolog Disable journaling. This will provide slight performance improvements in some situations at the cost of losing reiserfs's fast recovery from crashes. Even with this option turned on, reiserfs still performs all journaling operations, save for actual writes into its journaling area. Implementation of nolog is a work in progress.

notail By default, reiserfs stores small files and `file tails' directly into its tree. This confuses some utilities such as LILO(8). This option is used to disable packing of files into the tree.

replayonly
Replay the transactions which are in the journal, but do not actually mount the filesystem. Mainly used by reiserfsck.

resize=number
A remount option which permits online expansion of reiserfs partitions. Instructs reiserfs to assume that the device has number blocks. This option is designed for use with devices which are under logical volume management (LVM). There is a special resizer utility which can be obtained from ftp://ftp.namesys.com/pub/reiserfsprogs.

user_xattr
Enable Extended User Attributes. See the attr(5) manual page.

acl Enable POSIX Access Control Lists. See the acl(5) manual page.

barrier=none / barrier=flush
This enables/disables the use of write barriers in the journaling code. barrier=none disables it, barrier=flush enables it. Write barriers enforce proper on-disk ordering of journal commits, making volatile disk write caches safe to use, at some performance penalty. The reiserfs filesystem does not enable write barriers by default. Be sure to enable barriers unless your disks are battery-backed one way or another. Otherwise you risk filesystem corruption in case of power failure.

Mount options for romfs
None.

Mount options for squashfs
None.

Mount options for smbfs
Just like nfs, the smbfs implementation expects a binary argument (a struct smb_mount_data) to the mount system call. This argument is constructed by smbmount(8) and the current version of mount (2.12) does not know anything about smbfs.

Mount options for sysv
None.

Mount options for tmpfs
size=nbytes
Override default maximum size of the filesystem. The size is given in bytes, and rounded up to entire pages. The default is half of the memory. The size parameter also accepts a suffix % to limit this tmpfs instance to that percentage of your physical RAM: the default, when neither size nor nr_blocks is specified, is size=50%

nr_blocks=

The same as size, but in blocks of PAGE_CACHE_SIZE

nr_inodes=
The maximum number of inodes for this instance. The default is half of the number of your physical RAM pages, or (on a machine with highmem) the number of lowmem RAM pages, whichever is the lower.

The tmpfs mount options for sizing (size, nr_blocks, and nr_inodes) accept a suffix k, m or g for Ki, Mi, Gi (binary kilo, mega and giga) and can be changed on remount.

mode= Set initial permissions of the root directory.

uid= The user id.
gid= The group id.

mpol=[default|prefer:Node|bind:NodeList|interleave|interleave:NodeList]
Set the NUMA memory allocation policy for all files in that instance (if the kernel CONFIG_NUMA is enabled) - which can be adjusted on the fly via 'mount -o remount ...'

default
prefers to allocate memory from the local node

prefer:Node
prefers to allocate memory from the given Node

bind:NodeList
allocates memory only from nodes in NodeList

interleave
prefers to allocate from each node in turn

interleave:NodeList
allocates from each node of NodeList in turn.

The NodeList format is a comma-separated list of decimal numbers and ranges, a range being two hyphen-separated decimal numbers, the smallest and largest node numbers in the range. For example, mpol=bind:0-3,5,7,9-15

Note that trying to mount a tmpfs with an mpol option will fail if the running kernel does not support NUMA; and will fail if its nodelist specifies a node which is not online. If your system relies on that tmpfs being mounted, but from time to time runs a kernel built without NUMA capability (perhaps a safe recovery kernel), or with fewer nodes online, then it is advisable to omit the mpol option from automatic mount options. It can be added later, when the tmpfs is already mounted on MountPoint, by 'mount -o remount,mpol=Policy:NodeList MountPoint'.

Mount options for ubifs
UBIFS is a flash file system which works on top of UBI volumes. Note that atime is not supported and is always turned off.

The device name may be specified as

ubiX_Y UBI device number X, volume number Y

ubiY UBI device number 0, volume number Y

ubiX:NAME
UBI device number X, volume with name NAME

ubi:NAME
UBI device number 0, volume with name NAME

Alternative ! separator may be used instead of :.

The following mount options are available:

bulk_read
Enable bulk-read. VFS read-ahead is disabled because it slows down the file system. Bulk-Read is an internal optimization. Some flashes may read faster if the data are read at one go, rather than at several read requests. For example, OneNAND can do "read-while-load" if it reads more than one NAND page.

no_bulk_read
Do not bulk-read. This is the default.

chk_data_crc
Check data CRC-32 checksums. This is the default.

no_chk_data_crc.
Do not check data CRC-32 checksums. With this option, the filesystem does not check CRC-32 checksum for data, but it does check it for the internal indexing information. This option only affects reading, not writing. CRC-32 is always calculated when writing the data.

compr={none|lzo|zlib}
Select the default compressor which is used when new files are written. It is still possible to read compressed files if mounted with the none option.

Mount options for udf

udf is the "Universal Disk Format" filesystem defined by the Optical Storage Technology Association, and is often used for DVD-ROM. See also iso9660.

gid= Set the default group.

umask= Set the default umask. The value is given in octal.

uid= Set the default user.

unhide Show otherwise hidden files.

undelete
Show deleted files in lists.

nostrict
Unset strict conformance.

iocharset
Set the NLS character set.

bs= Set the block size. (May not work unless 2048.)

novrs Skip volume sequence recognition.

session=
Set the CDROM session counting from 0. Default: last session.

anchor=
Override standard anchor location. Default: 256.
volume=
Override the VolumeDesc location. (unused)

partition=
Override the PartitionDesc location. (unused)

lastblock=
Set the last block of the filesystem.

fileset=
Override the fileset block location. (unused)

rootdir=
Override the root directory location. (unused)

Mount options for ufs
ufstype=value
UFS is a filesystem widely used in different operating systems. The problem are differences among implementations. Features of some implementations are undocumented, so its hard to recognize the type of ufs automatically. That's why the user must specify the type of ufs by mount option. Possible values are:

old Old format of ufs, this is the default, read only. (Don't forget to give the -r option.)

44bsd For filesystems created by a BSD-like system (NetBSD,FreeBSD,OpenBSD).
ufs2 Used in FreeBSD 5.x supported as read-write.

5xbsd Synonym for ufs2.

sun For filesystems created by SunOS or Solaris on Sparc.

sunx86 For filesystems created by Solaris on x86.

hp For filesystems created by HP-UX, read-only.

nextstep
For filesystems created by NeXTStep (on NeXT station) (currently read only).

nextstep-cd
For NextStep CDROMs (block_size == 2048), read-only.

openstep
For filesystems created by OpenStep (currently read

only). The same filesystem type is also used by Mac OS X.

onerror=value
Set behaviour on error:

panic If an error is encountered, cause a kernel panic.

[lock|umount|repair]
These mount options don't do anything at present; when an error is encountered only a console message is printed.

Mount options for umsdos
See mount options for msdos. The dotsOK option is explicitly killed by umsdos.

Mount options for vfat
First of all, the mount options for fat are recognized. The dotsOK option is explicitly killed by vfat. Furthermore, there are

uni_xlate
Translate unhandled Unicode characters to special escaped sequences. This lets you backup and restore filenames that are created with any Unicode characters. Without this option, a '?' is used when no translation is possible. The escape character is ':' because it is otherwise illegal on the vfat filesystem. The escape sequence that gets used, where u is the unicode character, is: ':', (u & 0x3f), ((u>>6) & 0x3f), (u>>12).

posix Allow two files with names that only differ in case. This option is obsolete.

nonumtail
First try to make a short name without sequence number, before trying name~num.ext.

utf8 UTF8 is the filesystem safe 8-bit encoding of Unicode that is used by the console. It can be enabled for the filesystem with this option or disabled with utf8=0, utf8=no or utf8=false. If `uni_xlate' gets set, UTF8 gets disabled.

shortname={lower|win95|winnt|mixed}

Defines the behaviour for creation and display of filenames which fit into 8.3 characters. If a long name for a file exists, it will always be preferred display. There are four modes: :

lower Force the short name to lower case upon display; store a long name when the short name is not all upper case.

win95 Force the short name to upper case upon display; store a long name when the short name is not all upper case.

winnt Display the shortname as is; store a long name when the short name is not all lower case or all upper case.

mixed Display the short name as is; store a long name when the short name is not all upper case. This mode is the

default since Linux 2.6.32.

Mount options for usbfs

devuid=uid and devgid=gid and devmode=mode

Set the owner and group and mode of the device files in the usbfs filesystem (default: uid=gid=0, mode=0644). The mode is given in octal.

busuid=uid and busgid=gid and busmode=mode

Set the owner and group and mode of the bus directories in the usbfs filesystem (default: uid=gid=0, mode=0555). The mode is given in octal.

listuid=uid and listgid=gid and listmode=mode

Set the owner and group and mode of the file devices (default: uid=gid=0, mode=0444). The mode is given in octal.

Mount options for xenix

None.

Mount options for xfs

allocsize=size

Sets the buffered I/O end-of-file preallocation size when doing delayed allocation writeout. Valid values for this option are page size (typically 4KiB) through to 1GiB, inclusive, in power-of-2 increments.

The default behaviour is for dynamic end-of-file preallocation size, which uses a set of heuristics to optimise the prealloca-
tion size based on the current allocation patterns within the file and the access patterns to the file. Specifying a fixed allocsize value turns off the dynamic behaviour.

attr2|noattr2

The options enable/disable an "opportunistic" improvement to be made in the way inline extended attributes are stored on-disk. When the new form is used for the first time when attr2 is selected (either when setting or removing extended attributes) the on-disk superblock feature bit field will be updated to reflect this format being in use.

The default behaviour is determined by the on-disk feature bit indicating that attr2 behaviour is active. If either mount option it set, then that becomes the new default used by the filesystem.

CRC enabled filesystems always use the attr2 format, and so will reject the noattr2 mount option if it is set.

barrier|nobarrier

Enables/disables the use of block layer write barriers for writes into the journal and for data integrity operations. This allows for drive level write caching to be enabled, for devices that support write barriers.

discard|nodiscard

Enable/disable the issuing of commands to let the block device reclaim space freed by the filesystem. This is useful for SSD

devices, thinly provisioned LUNs and virtual machine images, but may have a performance impact.

Note: It is currently recommended that you use the fstrim application to discard unused blocks rather than the discard mount option because the performance impact of this option is quite severe.

grpid|bsdgroups|nogrpid|sysvgroups
These options define what group ID a newly created file gets. When grpid is set, it takes the group ID of the directory in which it is created; otherwise it takes the fsgid of the current process, unless the directory has the setgid bit set, in which case it takes the gid from the parent directory, and also gets the setgid bit set if it is a directory itself.

filestreams
Make the data allocator use the filestreams allocation mode across the entire filesystem rather than just on directories configured to use it.

When ikeep is specified, XFS does not delete empty inode clusters and keeps them around on disk. When noikeep is specified, empty inode clusters are returned to the free space pool.

inode32|inode64
When inode32 is specified, it indicates that XFS limits inode creation to locations which will not result in inode numbers with more than 32 bits of significance.

When inode64 is specified, it indicates that XFS is allowed to create inodes at any location in the filesystem, including those which will result in inode numbers occupying more than 32 bits of significance.

inode32 is provided for backwards compatibility with older systems and applications, since 64 bits inode numbers might cause problems for some applications that cannot handle large inode numbers. If applications are in use which do not handle inode numbers bigger than 32 bits, the inode32 option should be specified.

largeio|nolargeio
If "nolargeio" is specified, the optimal I/O reported in st_blksize by stat(2) will be as small as possible to allow user applications to avoid inefficient read/modify/write I/O. This is typically the page size of the machine, as this is the granularity of the page cache.

If "largeio" specified, a filesystem that was created with a "swidth" specified will return the "swidth" value (in bytes) in st_blksize. If the filesystem does not have a "swidth" specified but does specify an "allocsize" then "allocsize" (in bytes) will be returned instead. Otherwise the behaviour is the same as if "nolargeio" was specified.

logbufs=value
Set the number of in-memory log buffers. Valid numbers range

from 2-8 inclusive.
The default value is 8 buffers.

If the memory cost of 8 log buffers is too high on small systems, then it may be reduced at some cost to performance on metadata intensive workloads. The logbsize option below controls the size of each buffer and so is also relevent to this case.

logbsize=value
Set the size of each in-memory log buffer. The size may be specified in bytes, or in kilobytes with a "k" suffix. Valid sizes for version 1 and version 2 logs are 16384 (16k) and 32768 (32k). Valid sizes for version 2 logs also include 65536 (64k), 131072 (128k) and 262144 (256k). The logbsize must be an integer multiple of the log stripe unit configured at mkfs time.

The default value for version 1 logs is 32768, while the default value for version 2 logs is MAX(32768, log_sunit).

logdev=deviceandrtdev=device
Use an external log (metadata journal) and/or real-time device. An XFS filesystem has up to three parts: a data section, a log section, and a real-time section. The real-time section is optional, and the log section can be separate from the data section or contained within it.

noalign
Data allocations will not be aligned at stripe unit boundaries. This is only relevant to filesystems created with non-zero data alignment parameters (sunit, swidth) by mkfs.

norecovery
The filesystem will be mounted without running log recovery. If the filesystem was not cleanly unmounted, it is likely to be inconsistent when mounted in "norecovery" mode. Some files or directories may not be accessible because of this. Filesystems mounted "norecovery" must be mounted read-only or the mount will fail.

nouuid Don't check for double mounted file systems using the file system uuid. This is useful to mount LVM snapshot volumes, and often used in combination with "norecovery" for mounting read-only snapshots.

noquota
Forcibly turns off all quota accounting and enforcement within the filesystem.

uquota/usrquota/uqnoenforce/quota
User disk quota accounting enabled, and limits (optionally) enforced. Refer to xfs_quota(8) for further details.

gquota/grpquota/gqnoenforce
Group disk quota accounting enabled and limits (optionally) enforced. Refer to xfs_quota(8) for further details.

pquota/prjquota/pqnoenforce
Project disk quota accounting enabled and limits (optionally)

enforced. Refer to xfs_quota(8) for further details.

sunit=value and swidth=value
Used to specify the stripe unit and width for a RAID device or a stripe volume. "value" must be specified in 512-byte block units. These options are only relevant to filesystems that were created with non-zero data alignment parameters.

The sunit and swidth parameters specified must be compatible with the existing filesystem alignment characteristics. In general, that means the only valid changes to sunit are increasing it by a power-of-2 multiple. Valid swidth values are any integer multiple of a valid sunit value.
Typically the only time these mount options are necessary if after an underlying RAID device has had it's geometry modified, such as adding a new disk to a RAID5 lun and reshaping it.

swalloc
Data allocations will be rounded up to stripe width boundaries when the current end of file is being extended and the file size is larger than the stripe width size.

wsync When specified, all filesystem namespace operations are executed synchronously. This ensures that when the namespace operation (create, unlink, etc) completes, the change to the namespace is on stable storage. This is useful in HA setups where failover must not result in clients seeing inconsistent namespace presentation during or after a failover event.

Mount options for xiafs
None. Although nothing is wrong with xiafs, it is not used much, and is not maintained. Probably one shouldn't use it. Since Linux version 2.1.21 xiafs is no longer part of the kernel source.

THE LOOP DEVICE
One further possible type is a mount via the loop device. For example, the command

mount /tmp/disk.img /mnt -t vfat -o loop=/dev/loop

will set up the loop device /dev/loop3 to correspond to the file /tmp/disk.img, and then mount this device on /mnt.

If no explicit loop device is mentioned (but just an option `-o loop' is given), then mount will try to find some unused loop device and use that, for example

mount /tmp/disk.img /mnt -o loop

The mount command automatically creates a loop device from a regular file if a filesystem type is not specified or the filesystem is known for libblkid, for example:

mount /tmp/disk.img /mnt

mount -t ext3 /tmp/disk.img /mnt

This type of mount knows about four options, namely loop, offset and sizelimit , that are really options to losetup(8). (These options can be used in addition to those specific to the filesystem type.)

Since Linux 2.6.25 is supported auto-destruction of loop devices and then any loop device allocated by mount will be freed by umount independently on /etc/mtab.

You can also free a loop device by hand, using `losetup -d' or `umount -d`.

RETURN CODES

mount has the following return codes (the bits can be ORed):

0 success

1 incorrect invocation or permissions

2 system error (out of memory, cannot fork, no more loop devices)

4 internal mount bug

8 user interrupt

16 problems writing or locking /etc/mtab

32 mount failure

64 some mount succeeded

The command mount -a returns 0 (all success), 32 (all failed) or 64 (some failed, some success).

NOTES

The syntax of external mount helpers is:
/sbin/mount.<suffix> spec dir [-sfnv] [-o options] [-t type.subtype]

where the <type> is filesystem type and -sfnvo options have same meaning like standard mount options. The -t option is used for filesystems with subtypes support (for example /sbin/mount.fuse -t fuse.sshfs).

FILES

/etc/fstab filesystem table

/etc/mtab table of mounted filesystems

/etc/mtab~ lock file

/etc/mtab.tmp temporary file

/etc/filesystems a list of filesystem types to try

ENVIRONMENT

LIBMOUNT_FSTAB=<path>
overrides the default location of the fstab file

LIBMOUNT_MTAB=<path>
overrides the default location of the mtab file

LIBMOUNT_DEBUG=0xffff
enables debug output

SEE ALSO
mount(2), umount(2), fstab(5), umount(8), swapon(8), findmnt(8), nfs(5), xfs(5), e2label(8), xfs_admin(8), mountd(8), nfsd(8), mke2fs(8), tune2fs(8), losetup(8)

BUGS
It is possible for a corrupted filesystem to cause a crash.

Some Linux filesystems don't support -o sync and -o dirsync (the ext2, ext3, fat and vfat filesystems do support synchronous updates (a la BSD) when mounted with the sync option).

The -o remount may not be able to change mount parameters (all ext2fs-specific parameters, except sb, are changeable with a remount, for example, but you can't change gid or umask for the fatfs).

It is possible that files /etc/mtab and /proc/mounts don't match. The first file is based only on the mount command options, but the content of the second file also depends on the kernel and others settings (e.g. remote NFS server. In particular case the mount command may reports unreliable information about a NFS mount point and the /proc/mounts file usually contains more reliable information.)

Checking files on NFS filesystem referenced by file descriptors (i.e. the fcntl and ioctl families of functions) may lead to inconsistent result due to the lack of consistency check in kernel even if noac is used.

The loop option with the offset or sizelimit options used may fail when using older kernels if the mount command can't confirm that the size of the block device has been configured as requested. This situation can be worked around by using the losetup command manually before calling mount with the configured loop device.

HISTORY
A mount command existed in Version 5 AT&T UNIX.

AUTHORS
Karel Zak <kzak@redhat.com>

AVAILABILITY
The mount command is part of the util-linux package and is available from ftp://ftp.kernel.org/pub/linux/utils/util-linux/.

util-linux January 2012 MOUNT(8)

```
[student@COS-047 ~]$ mount
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime,seclabel)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
devtmpfs on /dev type devtmpfs (rw,nosuid,seclabel,size=926016k,nr_inodes=231504,mode=755)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev,seclabel)
```

devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,seclabel,gid=5,mode=620,ptmxmode=000)
tmpfs on /run type tmpfs (rw,nosuid,nodev,seclabel,mode=755)
tmpfs on /sys/fs/cgroup type tmpfs (ro,nosuid,nodev,noexec,seclabel,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/net_cls,net_prio type cgroup (rw,nosuid,nodev,noexec,relatime,net_prio,net_cls)
cgroup on /sys/fs/cgroup/cpuset type cgroup (rw,nosuid,nodev,noexec,relatime,cpuset)
cgroup on /sys/fs/cgroup/pids type cgroup (rw,nosuid,nodev,noexec,relatime,pids)
cgroup on /sys/fs/cgroup/memory type cgroup (rw,nosuid,nodev,noexec,relatime,memory)
cgroup on /sys/fs/cgroup/perf_event type cgroup (rw,nosuid,nodev,noexec,relatime,perf_event)
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/hugetlb type cgroup (rw,nosuid,nodev,noexec,relatime,hugetlb)
cgroup on /sys/fs/cgroup/cpu,cpuacct type cgroup (rw,nosuid,nodev,noexec,relatime,cpuacct,cpu)
cgroup on /sys/fs/cgroup/blkio type cgroup (rw,nosuid,nodev,noexec,relatime,blkio)
configfs on /sys/kernel/config type configfs (rw,relatime)
/dev/mapper/centos-root on / type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
selinuxfs on /sys/fs/selinux type selinuxfs (rw,relatime)
systemd-1 on /proc/sys/fs/binfmt_misc type autofs
(rw,relatime,fd=33,pgrp=1,timeout=300,minproto=5,maxproto=5,direct)
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime,seclabel)
mqueue on /dev/mqueue type mqueue (rw,relatime,seclabel)
debugfs on /sys/kernel/debug type debugfs (rw,relatime)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw,relatime)
nfsd on /proc/fs/nfsd type nfsd (rw,relatime)
/dev/sda1 on /boot type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
fusectl on /sys/fs/fuse/connections type fusectl (rw,relatime)
tmpfs on /run/user/1001 type tmpfs (rw,nosuid,nodev,relatime,seclabel,size=188376k,mode=700,uid=1001,gid=1001)
gvfsd-fuse on /run/user/1001/gvfs type fuse.gvfsd-fuse (rw,nosuid,nodev,relatime,user_id=1001,group_id=1001)

Project 2-3

[student@COS-047 ~]$ echo $PS1
[\u@\h \W]\$
[student@COS-047 ~]$ PS1='\d \t>'
Sat Feb 24 17:09:25>
Sat Feb 24 17:10:15>PS1='\w>'
~>
~>PS1='$PWD>'
/home/student>
[student@COS-047 ~]$ PS1='$PWD>'
/home/student>

Project 2-4

[student@COS-047 ~]$ pwd
/home/student

Project 2-5

[student@COS-047 ~]$ PS1='$PWD>'
/home/student>cd /var/spool/mail
/var/spool/mail>cd
/home/student>

Project 2-6

```
[student@COS-047 ~]$ cd
[student@COS-047 ~]$ cd /home
[student@COS-047 home]$ cd student
[student@COS-047 ~]$
```

Project 2-7

```
[student@COS-047 ~]$ cd
[student@COS-047 ~]$ cd .
[student@COS-047 ~]$ cd ..
[student@COS-047 home]$ cd ..
[student@COS-047 /]$ pwd
/
[student@COS-047 /]$ cd
[student@COS-047 ~]$
```

Project 2-8

```
[student@COS-047 ~]$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
[student@COS-047 ~]$ cat > notes
[student@COS-047 ~]$ ls notes
notes
[student@COS-047 ~]$ ls /var
account  crash  games    lib   log   opt      spool   vmware
adm      db     gopher   local mail  preserve target  yp
cache    empty  kerberos lock  nis   run      tmp
[student@COS-047 ~]$ ls -l /dev
total 0
crw-rw----. 1 root video    10, 175 Jan 23 16:08 agpgart
crw-------. 1 root root     10, 235 Jan 23 16:08 autofs
drwxr-xr-x. 2 root root         140 Jan 23 16:08 block
drwxr-xr-x. 2 root root          60 Jan 23 16:08 bsg
crw-------. 1 root root     10, 234 Jan 23 16:08 btrfs-control
drwxr-xr-x. 2 root root          80 Jan 23 16:08 centos
drwxr-xr-x. 2 root root        2740 Feb 19 22:40 char
crw-------. 1 root root      5,   1 Jan 23 16:08 console
lrwxrwxrwx. 1 root root          11 Jan 23 16:08 core -> /proc/kcore
drwxr-xr-x. 6 root root         140 Jan 23 16:08 cpu
crw-------. 1 root root     10,  61 Jan 23 16:08 cpu_dma_latency
crw-------. 1 root root     10,  62 Jan 23 16:08 crash
drwxr-xr-x. 5 root root         100 Jan 23 16:08 disk
brw-rw----. 1 root disk    253,   0 Jan 23 16:08 dm-0
brw-rw----. 1 root disk    253,   1 Jan 23 16:08 dm-1
drwxr-xr-x. 2 root root         100 Jan 23 16:08 dri
crw-rw----. 1 root video    29,   0 Jan 23 16:08 fb0
lrwxrwxrwx. 1 root root          13 Jan 23 16:08 fd -> /proc/self/fd
crw-rw-rw-. 1 root root      1,   7 Jan 23 16:08 full
crw-rw-rw-. 1 root root     10, 229 Feb 19 22:40 fuse
crw-------. 1 root root     10, 228 Jan 23 16:08 hpet
```

```
drwxr-xr-x. 3 root root           0 Jan 23 16:08 hugepages
lrwxrwxrwx. 1 root root          25 Jan 23 16:08 initctl -> /run/systemd/initctl/fifo
drwxr-xr-x. 3 root root         220 Jan 23 16:08 input
crw-r--r--. 1 root root      1,  11 Jan 23 16:08 kmsg
srw-rw-rw-. 1 root root           0 Jan 23 16:08 log
crw-rw----. 1 root disk     10, 237 Jan 23 16:08 loop-control
crw-rw----. 1 root lp        6,   0 Jan 23 16:08 lp0
crw-rw----. 1 root lp        6,   1 Jan 23 16:08 lp1
crw-rw----. 1 root lp        6,   2 Jan 23 16:08 lp2
crw-rw----. 1 root lp        6,   3 Jan 23 16:08 lp3
drwxr-xr-x. 2 root root         100 Jan 23 16:08 mapper
crw-------. 1 root root     10, 227 Jan 23 16:08 mcelog
crw-r-----. 1 root kmem      1,   1 Jan 23 16:08 mem
drwxrwxrwt. 2 root root          40 Jan 23 16:08 mqueue
drwxr-xr-x. 2 root root          60 Jan 23 16:08 net
crw-------. 1 root root     10,  60 Jan 23 16:08 network_latency
crw-------. 1 root root     10,  59 Jan 23 16:08 network_throughput
crw-rw-rw-. 1 root root      1,   3 Jan 23 16:08 null
crw-------. 1 root root     10, 144 Jan 23 16:08 nvram
crw-------. 1 root root      1,  12 Jan 23 16:08 oldmem
crw-r-----. 1 root kmem      1,   4 Jan 23 16:08 port
crw-------. 1 root root    108,   0 Jan 23 16:08 ppp
crw-rw-rw-. 1 root tty       5,   2 Feb 24 18:37 ptmx
drwxr-xr-x. 2 root root           0 Jan 23 16:08 pts
crw-rw-rw-. 1 root root      1,   8 Jan 23 16:08 random
drwxr-xr-x. 2 root root          60 Jan 23 16:08 raw
lrwxrwxrwx. 1 root root           4 Jan 23 16:08 rtc -> rtc0
crw-------. 1 root root    253,   0 Jan 23 16:08 rtc0
brw-rw----. 1 root disk      8,   0 Jan 23 16:08 sda
brw-rw----. 1 root disk      8,   1 Jan 23 16:08 sda1
brw-rw----. 1 root disk      8,   2 Jan 23 16:08 sda2
crw-rw----. 1 root disk     21,   0 Jan 23 16:08 sg0
drwxrwxrwt. 2 root root         200 Feb 24 18:10 shm
crw-------. 1 root root     10, 231 Jan 23 16:08 snapshot
drwxr-xr-x. 2 root root          80 Jan 23 16:08 snd
lrwxrwxrwx. 1 root root          15 Jan 23 16:08 stderr -> /proc/self/fd/2
lrwxrwxrwx. 1 root root          15 Jan 23 16:08 stdin -> /proc/self/fd/0
lrwxrwxrwx. 1 root root          15 Jan 23 16:08 stdout -> /proc/self/fd/1
crw-rw-rw-. 1 root tty       5,   0 Jan 23 16:08 tty
crw--w----. 1 root tty       4,   0 Jan 23 16:08 tty0
crw--w----. 1 root tty       4,   1 Jan 23 16:08 tty1
crw--w----. 1 root tty       4,  10 Jan 23 16:08 tty10
crw--w----. 1 root tty       4,  11 Jan 23 16:08 tty11
crw--w----. 1 root tty       4,  12 Jan 23 16:08 tty12
crw--w----. 1 root tty       4,  13 Jan 23 16:08 tty13
crw--w----. 1 root tty       4,  14 Jan 23 16:08 tty14
crw--w----. 1 root tty       4,  15 Jan 23 16:08 tty15
crw--w----. 1 root tty       4,  16 Jan 23 16:08 tty16
crw--w----. 1 root tty       4,  17 Jan 23 16:08 tty17
crw--w----. 1 root tty       4,  18 Jan 23 16:08 tty18
crw--w----. 1 root tty       4,  19 Jan 23 16:08 tty19
crw--w----. 1 root tty       4,   2 Jan 23 16:08 tty2
crw--w----. 1 root tty       4,  20 Jan 23 16:08 tty20
crw--w----. 1 root tty       4,  21 Jan 23 16:08 tty21
crw--w----. 1 root tty       4,  22 Jan 23 16:08 tty22
crw--w----. 1 root tty       4,  23 Jan 23 16:08 tty23
crw--w----. 1 root tty       4,  24 Jan 23 16:08 tty24
```

```
crw--w----. 1 root tty       4,  25 Jan 23 16:08 tty25
crw--w----. 1 root tty       4,  26 Jan 23 16:08 tty26
crw--w----. 1 root tty       4,  27 Jan 23 16:08 tty27
crw--w----. 1 root tty       4,  28 Jan 23 16:08 tty28
crw--w----. 1 root tty       4,  29 Jan 23 16:08 tty29
crw--w----. 1 root tty       4,   3 Jan 23 16:08 tty3
crw--w----. 1 root tty       4,  30 Jan 23 16:08 tty30
crw--w----. 1 root tty       4,  31 Jan 23 16:08 tty31
crw--w----. 1 root tty       4,  32 Jan 23 16:08 tty32
crw--w----. 1 root tty       4,  33 Jan 23 16:08 tty33
crw--w----. 1 root tty       4,  34 Jan 23 16:08 tty34
crw--w----. 1 root tty       4,  35 Jan 23 16:08 tty35
crw--w----. 1 root tty       4,  36 Jan 23 16:08 tty36
crw--w----. 1 root tty       4,  37 Jan 23 16:08 tty37
crw--w----. 1 root tty       4,  38 Jan 23 16:08 tty38
crw--w----. 1 root tty       4,  39 Jan 23 16:08 tty39
crw--w----. 1 root tty       4,   4 Jan 23 16:08 tty4
crw--w----. 1 root tty       4,  40 Jan 23 16:08 tty40
crw--w----. 1 root tty       4,  41 Jan 23 16:08 tty41
crw--w----. 1 root tty       4,  42 Jan 23 16:08 tty42
crw--w----. 1 root tty       4,  43 Jan 23 16:08 tty43
crw--w----. 1 root tty       4,  44 Jan 23 16:08 tty44
crw--w----. 1 root tty       4,  45 Jan 23 16:08 tty45
crw--w----. 1 root tty       4,  46 Jan 23 16:08 tty46
crw--w----. 1 root tty       4,  47 Jan 23 16:08 tty47
crw--w----. 1 root tty       4,  48 Jan 23 16:08 tty48
crw--w----. 1 root tty       4,  49 Jan 23 16:08 tty49
crw--w----. 1 root tty       4,   5 Jan 23 16:08 tty5
crw--w----. 1 root tty       4,  50 Jan 23 16:08 tty50
crw--w----. 1 root tty       4,  51 Jan 23 16:08 tty51
crw--w----. 1 root tty       4,  52 Jan 23 16:08 tty52
crw--w----. 1 root tty       4,  53 Jan 23 16:08 tty53
crw--w----. 1 root tty       4,  54 Jan 23 16:08 tty54
crw--w----. 1 root tty       4,  55 Jan 23 16:08 tty55
crw--w----. 1 root tty       4,  56 Jan 23 16:08 tty56
crw--w----. 1 root tty       4,  57 Jan 23 16:08 tty57
crw--w----. 1 root tty       4,  58 Jan 23 16:08 tty58
crw--w----. 1 root tty       4,  59 Jan 23 16:08 tty59
crw--w----. 1 root tty       4,   6 Jan 23 16:08 tty6
crw--w----. 1 root tty       4,  60 Jan 23 16:08 tty60
crw--w----. 1 root tty       4,  61 Jan 23 16:08 tty61
crw--w----. 1 root tty       4,  62 Jan 23 16:08 tty62
crw--w----. 1 root tty       4,  63 Jan 23 16:08 tty63
crw--w----. 1 root tty       4,   7 Jan 23 16:08 tty7
crw--w----. 1 root tty       4,   8 Jan 23 16:08 tty8
crw--w----. 1 root tty       4,   9 Jan 23 16:08 tty9
crw-rw----. 1 root dialout   4,  64 Jan 23 16:08 ttyS0
crw-rw----. 1 root dialout   4,  65 Jan 23 16:08 ttyS1
crw-rw----. 1 root dialout   4,  66 Jan 23 16:08 ttyS2
crw-rw----. 1 root dialout   4,  67 Jan 23 16:08 ttyS3
crw-------. 1 root root     10, 239 Jan 23 16:08 uhid
crw-------. 1 root root     10, 223 Jan 23 16:08 uinput
crw-rw-rw-. 1 root root      1,   9 Jan 23 16:08 urandom
crw-------. 1 root root    250,   0 Jan 23 16:08 usbmon0
crw-rw----. 1 root tty       7,   0 Jan 23 16:08 vcs
crw-rw----. 1 root tty       7,   1 Jan 23 16:08 vcs1
crw-rw----. 1 root tty       7,   2 Jan 23 16:08 vcs2
```

```
crw-rw----. 1 root tty       7,   3 Jan 23 16:08 vcs3
crw-rw----. 1 root tty       7,   4 Jan 23 16:08 vcs4
crw-rw----. 1 root tty       7,   5 Jan 23 16:08 vcs5
crw-rw----. 1 root tty       7,   6 Jan 23 16:08 vcs6
crw-rw----. 1 root tty       7, 128 Jan 23 16:08 vcsa
crw-rw----. 1 root tty       7, 129 Jan 23 16:08 vcsa1
crw-rw----. 1 root tty       7, 130 Jan 23 16:08 vcsa2
crw-rw----. 1 root tty       7, 131 Jan 23 16:08 vcsa3
crw-rw----. 1 root tty       7, 132 Jan 23 16:08 vcsa4
crw-rw----. 1 root tty       7, 133 Jan 23 16:08 vcsa5
crw-rw----. 1 root tty       7, 134 Jan 23 16:08 vcsa6
drwxr-xr-x. 2 root root         60 Jan 23 16:08 vfio
crw-------. 1 root root     10,  63 Jan 23 16:08 vga_arbiter
crw-------. 1 root root     10, 137 Jan 23 16:08 vhci
crw-------. 1 root root     10, 238 Jan 23 16:08 vhost-net
crw-------. 1 root root     10,  58 Jan 23 16:08 vmci
crw-------. 1 root root     10,  57 Jan 23 16:08 vsock
crw-rw-rw-. 1 root root      1,   5 Jan 23 16:08 zero
```

A Device Special File

```
[student@COS-047 ~]$ ls -l /
total 32
-rw-r--r--.   1 root root    0 Jan 22  2017 1
lrwxrwxrwx.   1 root root    7 Jan 22  2017 bin -> usr/bin
dr-xr-xr-x.   4 root root 4096 Jan 22  2017 boot
drwxr-xr-x.  19 root root 3100 Jan 23 16:08 dev
drwxr-xr-x. 140 root root 8192 Feb 20 14:58 etc
drwxr-xr-x.   4 root root   35 Nov  5  2016 home
lrwxrwxrwx.   1 root root    7 Jan 22  2017 lib -> usr/lib
lrwxrwxrwx.   1 root root    9 Jan 22  2017 lib64 -> usr/lib64
drwxr-xr-x.   2 root root    6 Nov  5  2016 media
drwxr-xr-x.   2 root root    6 Nov  5  2016 mnt
drwxr-xr-x.   3 root root   15 Nov  5  2016 opt
dr-xr-xr-x. 208 root root    0 Jan 23 16:08 proc
dr-xr-x---.  17 root root 4096 Jan 11 00:57 root
drwxr-xr-x.  36 root root 1140 Feb 24 16:26 run
lrwxrwxrwx.   1 root root    8 Jan 22  2017 sbin -> usr/sbin
drwxr-xr-x.   2 root root    6 Nov  5  2016 srv
dr-xr-xr-x.  13 root root    0 Jan 23 16:08 sys
drwxrwxrwt.  21 root root 4096 Feb 24 18:37 tmp
drwxr-xr-x.  13 root root 4096 Jan 22  2017 usr
drwxr-xr-x.  22 root root 4096 Jan 23 16:08 var
```

```
[student@COS-047 ~]$ clear
[student@COS-047 ~]$ ls -a
.              .bashrc    Downloads      .mozilla  Templates
..             .cache     .esd_auth      Music     Videos
.bash_history  .config    .ICEauthority  notes
.bash_logout   Desktop    .lesshst       Pictures
.bash_profile  Documents  .local         Public
```

Project 2-9

```
[student@COS-047 ~]$ cat > first_name
Christopher
[student@COS-047 ~]$ cat > middle_name
Adam
[student@COS-047 ~]$ cat > last_name
Welch
[student@COS-047 ~]$ cat > full_name1.txt
Christopher Adam Welch
[student@COS-047 ~]$ cat > full_name22.txt
Christopher Adam Welch
[student@COS-047 ~]$ ls *name
first_name  last_name  middle_name
[student@COS-047 ~]$ ls full_name?.txt
full_name1.txt
[student@COS-047 ~]$ ls *.txt
full_name1.txt  full_name22.txt
```

Project 2-10

```
[student@COS-047 ~]$ cd
[student@COS-047 ~]$ mkdir dept_4540
[student@COS-047 ~]$ ls
dept_4540  Downloads      full_name22.txt  Music     Public
Desktop    first_name     last_name        notes     Templates
Documents  full_name1.txt  middle_name      Pictures  Videos
[student@COS-047 ~]$ cd dept_4540
[student@COS-047 dept_4540]$ cat > phones1
219:432:4567:Harrison:Joel
219:432:4587:Mitchell:Barbara
219:432:4589:Olson:Timothy
[student@COS-047 dept_4540]$ cat phones1
219:432:4567:Harrison:Joel
219:432:4587:Mitchell:Barbara
219:432:4589:Olson:Timothy
[student@COS-047 dept_4540]$ cd
[student@COS-047 ~]$ mkdir dept_4550
[student@COS-047 ~]$ ls
dept_4540  Documents   full_name1.txt   middle_name  Pictures   Videos
dept_4550  Downloads   full_name22.txt  Music        Public
Desktop    first_name  last_name        notes        Templates
[student@COS-047 ~]$ cd dept_4550
[student@COS-047 dept_4550]$ cat > phones2
219:432:4591:Moore:Sarah
219:432:4522:Polk:John
219:432:4501:Robinson:Lisa
[student@COS-047 dept_4550]$ cat phones2
219:432:4591:Moore:Sarah
219:432:4522:Polk:John
219:432:4501:Robinson:Lisa
[student@COS-047 dept_4550]$ clear
```

Project 2-11

```
[student@COS-047 dept_4550]$ cd
```

```
[student@COS-047 ~]$ mkdir corp_db
[student@COS-047 ~]$ cd corp_db
[student@COS-047 corp_db]$ cp ~/dept_4540/phones1 .
[student@COS-047 corp_db]$ cp ~/dept_4550/phones2 .
[student@COS-047 corp_db]$ ls
phones1  phones2
[student@COS-047 corp_db]$ cat phones1 phones2 > corp_phones
[student@COS-047 corp_db]$ clear


[student@COS-047 corp_db]$ more corp_phones
219:432:4567:Harrison:Joel
219:432:4587:Mitchell:Barbara
219:432:4589:Olson:Timothy
219:432:4591:Moore:Sarah
219:432:4522:Polk:John
219:432:4501:Robinson:Lisa
[student@COS-047 corp_db]$
```

Project 2-12

```
[student@COS-047 corp_db]$ cd
[student@COS-047 ~]$ clear

[student@COS-047 ~]$ chmod go+x ~
[student@COS-047 ~]$ chmod ugo+x ~/corp_db
[student@COS-047 ~]$ chmod o+w ~/corp_db/*
[student@COS-047 ~]$ ls -l ~/corp_db
total 12
-rw-rw-rw-. 1 student student 159 Feb 24 19:49 corp_phones
-rw-rw-rw-. 1 student student  84 Feb 24 19:47 phones1
-rw-rw-rw-. 1 student student  75 Feb 24 19:49 phones2
[student@COS-047 ~]$
```
