Linux Disk Management Cheat Sheet

File systems

A file system is a storage space for files. There are several file system formats that are compatible with Linux, including:

File system format	Description
ext3	A long-standing standard Linux format
ext4	A newer standard Linux format
fat16	Older MS-DOS file format
fat32	Newer Windows format
NTFS	Another Windows format
XFS	A format created by SGI for high-capacity servers
APFS	A format used by Mac OS X
swap	A swap partition, not for writing files
iso9660	Standard format for CD-ROMs and DVD-ROMs

If you're installing Linux and aren't sure which file system format to use, ext4 is a safe choice.

mkfs

To format a partition, use the mkfs command. The most common usage is:

```
mkfs -t FORMAT DEVICE
```

The FORMAT argument is the file system type (like ext4 or XFS), and the DEVICE argument refers to the partition being formatted.

For example, to create a file system in the ext4 format on the second partition on device sda, you'd run:

```
mkfs -t ext4 /dev/sda2
```

df

To see how much space is free on a file system, use the df command.

```
df <OPTIONS> <FILE>
```

Run with no arguments, df will list all file systems and their remaining space in kilobytes. If you specify a file or directory, df only reports on the file system containing that file or directory. Commonly used options for df include:

Option	Description
-h	Display the free space in a human-readable format (like 2M for 2 megabytes)
-l	Display information for only local file systems (not remotely mounted shares)
-T	List the file system type

For example, to see free space on the volume that contains the file “file.txt” in human-readable format, run:

```
df -h file.txt
```

du

The du command lets you see how much space a file or directory takes up in a file system.

```
du <OPTIONS> FILE
```

If a file is specified, du will return the size of that file in kilobytes. For a directory, du will list the files in the directory with their sizes by default.

Common options for du include:

Option	Description
-c	Include a line with the total disk usage for all listed files
-h	Display the disk space in a human-readable format (like 2M for 2 megabytes)
-s	Report only the total size for a directory
-t SIZE	Exclude files under the specified size (like “10M” or “1G”)

For example, to see how much disk space a directory takes up in a human-readable format, run:

```
du -hs directoryname
```

To list all files over 500 megabytes in the /home directory and the total disk space they account for, run:

```
du -c -t 500M /home
```

Mount points

Linux makes file systems accessible by mounting them attached to directories called *mount points*. Any directory can act as a mount point. Directories that might have their own mount points include:

Mount point	Common use
/	Default file system
/boot	Files used in the boot process, like the Linux kernel
/home	Use home directories
/var	Storage for files that can vary in size, like log files and email
/media	Common mount point for ejectable media like DVDs
/mnt	A catchall directory for temporarily mounting file systems

A system that resides entirely on a single partition will have only one mount point, "/", to hold everything. When a directory has its own mount point it's usually for system security or stability reasons, like keeping /var on a separate disk partition so that a runaway log won't fill the whole system's disk space - only its own partition.

mount

The mount command will list the mount points on a system when run without any arguments:

```
mount
```

Or it will mount a file system defined in the /etc/fstab file if just a directory is specified:

```
mount <OPTIONS> DIRECTORY
```

Or it can be used to mount a file system to a mount point.

```
mount <OPTIONS> -t FSTYPE DEVICE DIRECTORY
```

Where FSTYPE is the file system type (like ext4 or FAT32), the DEVICE argument is the identifier for the device containing the file system, and the DIRECTORY argument is the directory to be used as a mount point.

For example, to format the first partition of device sdb as ext4 as /var/db, you'd run:

```
mount -t ext4 /dev/sdb1 /var/db
```

To mount a CD-ROM on device sr0 to /media as read-only, you'd run:

```
mount -r -t iso9660 /dev/sr0 /media
```

umount

To unmount a mounted file system, use the umount command.

```
umount DIRECTORY | DEVICE
```

You can specify either the directory used as the mount point or the identifier for the device being unmounted. When you're done with a CD-ROM, for example, you might run:

```
umount /media
```

The system will refuse to unmount a file system when a file it contains is in use, like when a process is writing a log file to that file system.
If you know that file system won't stay busy for long, you can use -l for a "lazy" unmount - that hides the file system's directory structure, then completes the unmount when the file system is no longer busy.

```
umount -l /mnt/thing1
```

If the file system won't unmount because it can't verify its status (like a remote fileshare that's no longer available), you can try to force the unmount with the -f option:

```
umount -f /mnt/fileshare
```

/etc/fstab

You can specify file systems to be mounted at boot time in the /etc/fstab file. You can also use this file to define mount points that aren't automatically mounted, like defining a mount point for a DVD-ROM drive.

The format used in /etc/fstab is:

```
DEVICE MOUNTPOINT FSTYPE MOUNTOPTS DUMP FSCK
```

- DEVICE: The device identifier (like a device file or label).
- MOUNTPOINT: The directory that acts as a mount point for the file system.
- FSTYPE: The file system type (ext4, swap, iso9660, etc.).
- MOUNTOPTS: Mount options. Can be "defaults" or a comma-separated list of options.
- DUMP: Used by the dump command, which is rarely used anymore. Set to 0 unless you know you're using dump, in which case set it to 1.
- FSCK: Priority for file system checks at boot time. A value of 0 means don't check, 1 should be used for the root file system ("/"), and other file systems to be checked should have a value of 2 here.

Common options used in /etc/fstab include:

Mount option	Description
defaults	Shortcut for a set of default options, including rw, suid, exec, auto, nouser, and async
auto	Mount the file system at boot time
noauto	Don't mount the file system at boot time
user	Allow regular users to mount the file system

nouser	Don't allow regular users to mount the file system (the default behavior)
async	Asynchronous disk writes (the default)
sync	Synchronous disk writes - increases wear on the drive but disk write requests are executed immediately
exec	Allow binaries on the file system to be executed
noexec	Disable execution of binaries stored on the file system
suid	Allow setuid and guid permissions on the file system
nosuid	Disable setuid and guid permissions on the file system
rw	Mount the file system as readable and writeable
ro	Mount the file system as read-only

Mounting /home from /dev/sdb1 with the default options but setuid and guid permissions disabled would look like this in /etc/fstab:

```
/dev/sdb1 /home ext4 defaults,nosuid 0 2
```

An entry for an optical drive, which wouldn't be mounted at startup but you might want regular users to access, might look like this:

```
/dev/sr0 /cdrom iso9660 ro,noauto,user 0 0
```

Swap

Swap is used as temporary memory space - if a process hasn't been accessed for a while, the system can write its information from memory to swap in order to free up physical memory space. A swap partition is usually created when the system is installed. You generally shouldn't run a system without swap - with swap space available, the system won't halt when it runs out of physical memory, just get much slower as it shuffles memory contents to and from the disk. Recommendations for swap size vary from half of the system's physical memory amount to twice that amount. In general, a higher proportion of swap is a good idea if a system has little physical memory.

You can format a partition to be used as swap space with the mkswap command:

```
mkswap DEVICE
```

After it's formatted, you tell the system to use the new swap space with the swapon command:

```
swapon DEVICE
```

Formatting and partitions

A.MBR
- Limited to 2 TB
- Four partitions (4 primary or 3 primary/1 extended)

1.BIOS
2.fdisk

B.GPT
1.UEFI
2.parted
3.gdisk

C./dev/sda1 breakdown

Disk identifiers

A.Device files
B.UUID and GUID
C.Labels
1.e2label
D.Listing Block devices
1.lsblk
2.lsblk -f
3.blkid