Exercise 17.1: Using a File as a Disk Partition Image

The exercises in this section will make simple use of **mkfs** for formatting filesystems, and **mount** for mounting them at places in the root filesystem tree. These commands will be explained in detail in the next session.

For the purposes of the exercises in this course you will need unpartitioned disk space. It need not be large, certainly one or two GB will suffice.

If you are using your own native machine, you either have it or you do not. If you do not, you will have shrink a partition and the filesystem on it (first!) and then make it available, using **gparted** and/or the steps we have outlined or will outline.

Or you can use the **loop device** mechanism with or without the **parted** program, as we will do in the first two exercises in this section.

Please Note

If you have real physical unpartitioned disk space you do not **need** to do the following procedures, but it is still a very useful learning exercise.

We are going to create a file that will be used as a container for a full hard disk partition image, and for all intents and purposes can be used like a real hard partition. In the next exercise we will show how to put more than one partition on it and have it behave as an entire disk.

1. Create a file full of zeros 1 GB in length:

   ```
   $ dd if=/dev/zero of=imagefile bs=1M count=1024
   ```

   You can make a much smaller file if you like or do not have that much available space in the partition you are creating the file on.

2. Put a filesystem on it:

   ```
   $ mkfs.ext4 imagefile
   ```

   ```
   mke2fs 1.42.9 (28-Dec-2013)
   imagefile is not a block special device.
   Proceed anyway? (y,n) y
   Discarding device blocks: done
   .....
   ```

   Of course you can format with a different filesystem, doing **mkfs.ext3**, **mkfs.vfat**, **mkfs.xfs** etc.

3. Mount it somewhere:

   ```
   $ mkdir mntpoint
   $ sudo mount -o loop imagefile mntpoint
   ```

   You can now use this to your heart's content, putting files etc. on it.

4. When you are done unmount it with:

   ```
   $ sudo umount mntpoint
   ```

An alternative method to using the `loop` option to mount would be:

```
$ sudo losetup /dev/loop2 imagefile
$ sudo mount /dev/loop2 mntpoint
 ....
$ sudo umount mntpoint
$ sudo losetup -d /dev/loop2
```


We will discuss **losetup** in a subsequent exercise, and you can use `/dev/loop[0-7]` but you have to be careful they are not already in use, as we will explain.

You should note that using a loop device file instead of a real partition can be useful, but it is pretty worthless for doing any kind of measurements or benchmarking. This is because you are placing one filesystem layer on top of another, which can only have a negative effect on performance, and mostly you just use the behavior of the underlying filesystem the image file is created on.

