Learning Objectives

Learners will be able to...

Use the following commands:
- gzip
- bzip2
- zip
- tar
- xz
- cpio
- dd

Compression

What is file compression?

File compression is a way to reduce the size of a file, or set of files. The advantages of compressed files are that they require less storage space and in the case of files that will be downloaded or uploaded, their reduced size reduces transmission time.

Two types of compression

There are two types of compression, **lossy** and **lossless**.

Lossy compression

With **lossy** compression some of the original data is discarded and the file cannot be reconstructed to its original form. This form of compression is often used on images and sound files, for example **jpeg** and **mp3** formats. For image files the number of colors might be reduced and for sound files the number of samples. This type of compression can result in loss of quality.

An example of image lossy compression is **chroma subsampling**, color information is reduced by averaging sets of adjacent color values and setting them all to the same value.

Audio compression algorithms remove sounds that are outside of human hearing capacity, this technique is called **perceptual audio coding**.

Lossless compression

With **lossless compression** methods the original file can be reconstructed from the compressed file. In a nutshell, these methods recognize repeated patterns, replace them with a smaller placeholder and then keep the actual pattern in one place to be later substituted when the file is uncompressed.

The `gzip` compression tool

The name **gzip** refers to both the tool that does the compression and the file format of the resulting file. This tool has been around since 1992 and is still being maintained. The compression algorithm used by `gzip` is known as

DEFLATE. The two main advantages of **DEFLATE** are speed, and memory efficiency, during the compression process. It only compresses a single file and the resulting file is given a .gz file extension.

A few useful options for this command:
Option	Purpose
————	————-
`-d`	Decompress
`-k`	Keep input files
`-l`	List compressed size, uncompressed size, ratio

You can see all options by referring to the `man` page:

```
man gzip
```

Before `gzip` (notice the file size of Lesmiserables.txt is 3369045):

You can also view the file size by typing `ls -l Lesmiserables.txt`

1. Zip the large text file:

```
gzip Lesmiserables.txt
```

After `gzip` (notice the file size is now 1290580, much smaller!):

---

2. Get information about the compression using the `-l` option. It tells you about the compressed size, the uncompressed size and the ratio of the two. The file size was reduced by 61.7 percent.

```
gzip -l Lesmiserables.txt.gz
```

3. Now unzip the file - `gunzip` is equivalent to `gzip -d`

```
gzip -d Lesmiserables.txt.gz
```

4. Now take a look at the file and see that it is once again the exact same size as before being compressed.

```
ls -l Lesmiserables.txt
```

The `bzip2` compression tool

The `bzip2` compression tool was first introduced in 1996. The algorithm used by `bzip2` is called **Burrows–Wheeler transform** (BWT), another name for it is **block-sorting compression**. Compared to the gzip compression utility, `bzip2` outputs a smaller compressed file but due to the complexity of the algorithm, it takes much longer for the process to complete. It also requires more memory during the compression process than `gzip`. It only compresses a single file and the resulting file is given a .bz2 file extension.

Try `bzip2`:

```
bzip2 Lesmiserables.txt
```

Check the file size of the zipped file:

```
ls -l Lesmiserables.txt.bz2
```

The `bzip2` results in a smaller file but the time difference for the compression as compared to `gzip` is insignificant with a file of this size.

Unzip the file:

```
bzip2 -d Lesmiserables.txt.bz2
```

Check the file size after unzipping:

```
ls -l Lesmiserables.txt
```

View the `man` page for more information on this utility:

```
man bzip2
```

The `xz` compression tool

The `xz` compression tool is the newest of the tools on this page, it was first released in 2009. The algorithm used by `xz` is called **LZMA2**, it has a greater compression ratio than the two previous tools. Like `bzip2` the greater compression ability comes at the expense of the speed of the process. In some cases it can take 4-5 times longer than `bzip2`. It only compresses a single file and the resulting file is given a .xz file extension.

Try `xz` and view the resulting file size:

```
xz Lesmiserables.txt
ls -l Lesmiserables.txt.xz
```

You can view statistics about the compressed file using the `-l` option:

```
xz -l Lesmiserables.txt.xz
```

Uncompress the file using the `-d` option:

```
xz -d Lesmiserables.txt.xz
```

You can view more information about the `xz` command on the `man` page for it:

```
man xz
```

Archiving

Archiving is the process of combining multiple files or directories into one single file. Archiving is useful when you are backing data up or sharing it. For example, if you have multiple files you want to send to someone it is more efficient to turn them into a single archive, send them one file and they can extract the files when they receive it.

The `tar` command

The word `tar` is short for **Tape Archive** and the command refers back to the early days when backups were made to magnetic tape drives.

Common `tar` flags:

Flag	Function
`-c`	Create an archive
`-f`	Use an archive file
`-r`	Append to an archive
`-t`	List contents of an archive
`-v`	Verbose output
`-x`	Extract contents of an archive
`-z`	Compress the archive using gzip

View all the options for `tar`:

```
man tar
```

The directory `images` on the left contains 53 files. If you wanted to send them to someone in an email, attaching all the files would be cumbersome. Instead you can use `tar` to create an archive file and send that.

info

Note: `tar` is recursive by default

It will archive all files in the hierarchy of whatever you supply as the input argument.

COMMAND	FLAGS	OUTPUT FILENAME	FILES/DIR
tar	-cvf	images.tar	images

The tar command tar, flags, archive name, input files

- This is the most common invocation:

```
tar -cvf images.tar images
```

- We can list the contents of the `images.tar` file:

```
tar -tvf images.tar
```

- You can also create an archive and zip it with one command:

```
tar -zcvf images.tar.gz images
ls -l images.*
```

- Notice that the compressed file is smaller
- You can extract a single file - first we will rename the images directory because by default it extracts to the same hierarchy

```
mv images imagesbak
tar -xvf images.tar images/concat.png
ls -l images
```

- Extract a compressed archive to a different directory - the directory must already exist

```
mkdir extracted
tar -zxvf images.tar.gz -C extracted
```

The `cpio` command

The `cpio` command is used to copy files into and out of archives. The name refers to its functionality, “copy in copy out”. It is similar to `tar` but there are some distinct differences:
1. `cpio` is not recursive by default
1. `cpio` will not overwrite newer data file whereas the default for `tar` is to overwrite

Common `cpio` flags:

Flag	Purpose
`-o`	copy out mode copies files into an archive
`-i`	copy in mode copies files out of an archive
`-p`	copy pass copies files from one directory tree to another

- View all the options for `cpio`:

```
man cpio
```

- Archive all the files in the images directory to a file named `imagedir.cpio`

```
cd images
ls | cpio -o > imagedir.cpio
ls -l imagedir.cpio
```

The `zip` and `unzip` commands

The `zip` command archives and compresses a set of files. It is a popular cross-platform tool and that makes it ideal for sharing files with people on different systems.

- `zip` is not recursive by default, if you want it to zip everything in a directory you need to specify the `-r` flag. First we need to get out of the `images` directory.

```
cd ..
zip -r images.zip imagesbak
ls -l
```

- You'll notice that the .zip file is smaller than the .tar file, that's because `zip` compresses by default.
- To unzip the archive:

```
unzip images.zip
```

Notice that it won't overwrite automatically, it prompts to see what you want to do.

Use `man zip` and `man unzip` for more information about these commands.

Backup

The `dd` command

The `dd` command can be used to convert and copy files. You might use it to create a bootable usb version of your Linux operating system or to backup files. It uses a different command line syntax than most other Linux commands. Rather than **-option** the syntax looks like this - **-option=value**.

Try this command:

```
dd if=trydd.txt of=outdd.txt conv=ucase
ls outdd.txt
```

- This command copies the contents of `trydd.txt` to `outdd.txt` using the conversion `ucase` which turned it all to upper case characters.
- Learn more about the `dd` command:

```
man dd
```