archiving files and directories

- An archive file captures metadata and contents of multiple files and directories as a single file
- often incorporates compression
- example file formats include:
  - **ar** - used for libraries of relocatable binaries (.o files)
  - **shar** - usually software packages, Unix-like systems
    - self-extracting shell-script!
  - **tar** - general purpose, Unix-like systems
  - **zip** - general purpose, many platforms, includes compression
  - **deb** - software packages, Debian-family Linux distributions

tar - archive/unarchive files and directories

```
# capture files in assignment directory tree
# -c create an archive
# -f archive filename
# -z compress with gzip
$ tar -zcf assignment.tar.gz assignment
$ cp assignment.tar.gz /tmp
$ cd /tmp
# extract files from archive
# -x create an archive
# -v (verbose) - print filenames when extracting
# -f archive filename
$ tar -xvf assignment.tar.gz
...
```

compressing files

- xz/unxz (-J option to tar)
  - algorithm Lempel– Ziv– Markovchain
  - good level of compression
  - slow to compresss but uncompression fast
- bzip2/bunzip2 (-j option to tar)
  - algorithm: Burrows—Wheeler algorithm
  - faster to compress than xz but compression level not as good
- gzip/gunzip (-z option to tar)
  - algorithm: DEFLATE
  - compression level not as good as bzip2
  - very widely available on Unix-like machines
  - used for HTTP compression

curl - interact with web-servers

- curl lets you interact from command line with web and other servers

```
# fetch a file
$ curl -O https://cgi.cse.unsw.edu.au/~cs2041/examples.zip
# get other info
$ curl I https://unsw.edu.au
HTTP/1.1 200 OK
Server: Apache/2.4.34 (Red Hat) OpenSSL/1.0.1e-fips PHP/5.6.25
X-Powered-By: PHP/5.6.25
# send data to web server
$ curl -X PUT -H 'content-type: txt/plain' https://google.com
# send cookies to web server
$ curl -b 'id=42' https://google.com
....
```

- curl has many other options
- wget provide similar functionality
  - Andrew likes `curl` better than `wget` - but little difference for most tasks

ssh - encrypted remote login

- ssh was written by Finnish university student Tatu Ylönen in 1995
- quickly adopted as an internet standard

```
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/andrewt/.ssh/id_rsa):
# leaves private key in $HOME/.ssh/id_rsa
# leaves public key in $HOME/.ssh/id_rsa.pub
$ cat $HOME/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAxL+t
....
# keep this file secret
$cat $HOME/.ssh/id_rsa
-----BEGIN RSA PRIVATE KEY-----
....
```

- Add public key to `$HOME/.ssh/authorized_keys` to allow for access without password.
- Can also run commands remotely:

```
$ ssh z1234567@login.cse.unsw.edu.au ls -las
```

rsync - efficiently copies files & directories

rsync efficiently copies files & directories locally or between machines (using ssh)

```
# mirror a directory tree in your CSE account
#    -a preserves metadata & copies recursively
#    -P shows progress
$ rsync -aP assignment/ login.cse.unsw.edu.au:assignment_backup/
```

If you run rsync command again it will only copy files which have changed.

If only a part of large file changed, will copy only the change (delta).

Many options, see `man rsync`

rsync - example copying only changed file

```
# create a directory a containing 1000 files
$ mkdir a
$ for i in $(seq 1 1000);do echo $i >a/$i; done
# copy directory a to director b
$ rsync -a a/ b/
# change one file
$ echo hello andrew >a/42.txt
# rsync will only copy changed files
$ rsync -av a/ b/
sending incremental file list
./
42.txt

sent 1,347 bytes  received 38 bytes  2,770.00 bytes/sec
total size is 305  speedup is 0.22
```

rsync - example copying only changed part of large file

```
# create a 100mb file
$ dd if=/dev/random bs=1M count=100 of=100_mb_file
# takes 25 seconds to copy it to CSE (40Mbps NBN)
$ time rsync 100_mb_file login.cse.unsw.edu.au:100_mb_file
real    0m24.943s
# repeat the rsync without changing the file - very fast
$ time rsync 100_mb_file login.cse.unsw.edu.au:100_mb_file
real    0m0.782s
# change a few bytes of the file
$ echo hello andrew >>100_mb_file
# rsync still fast
$ time rsync 100_mb_file login.cse.unsw.edu.au:100_mb_file
real    0m0.846s
```

Tools for Managing Processes

- *process* is an instance of an executing program
- on Unix-like systems each process had a unique number (pid)
  - pid's are smallish non-negative integer
- **`ps`** ... show process information
- **`kill`** ... send a signal to a process
  - typically used to terminate a process
  - signal 9 always terminates process

```
# using kill to logout
$ ps
    PID TTY          TIME CMD
 344024 pts/2    00:00:00 bash
 346137 pts/2    00:00:00 ps
$ kill -9 344024
```

Tools for Managing Processes

- **pgrep** … print PIDs of processes matching selection criteria
  - **pkill** send a signal to processes matching selection criteria
- killall … also send a signal to a process with particular names
  - less powerful but more widefully available than **pkill**
- **top** … real-time monitoring of running process
  - or more easy to use **htop**

```
# find processes with python in their name
$ pgrep python
10787
20060
25975
27475
# kill  processes with 2 consecutive vowels (don't do this!)
$ pkill -9 '[aeiou][aeiou]'
# kill  processes with 2 consecutive vowels (don't do this!)
$ pkill -9 '[aeiou][aeiou]'
# kill all programs named teams or zoom
$ killall zoom teams
```

Linux Filesystem Layout

- **`/home`** - home directories for users on the system
- **`/bin`** - important system programs (scripts and binaries)
- **`/usr/`** - system programs and associated files
  - **`/usr/bin`** system programs
  - **`/usr/local/bin`** custom installed local programs
  - **`/usr/lib`** - libraries (linked with programs)
  - **`/usr/include`** - header files for C programs
  - ...
- **`/etc`** - holds configuration for system programs
- **`/opt`** - multi-operating system packages sometimes installed here
- **`/var`** - system files that regulary change, e.g.: log files, database files.
- **`/tmp`** - directory for temporary files - files removed on reboot

Linux Filesystem Layout

- **`/root`** - home directory for root user
- **`/boot`** - files need to boot operating system
- **`/dev`** - pathnames for hardware devices.
- **`/media`** - mount-point for removable device
- **`/proc`** - special filesystem with information about processes
- **`/sys`** - special filesystem with information about system

/dev - directory for devices

- Unix devices are manipulated by special files (usually) in **/dev**
  - e.g a disk might appear as `/dev/sda`

```
$ ls -l /dev
...
brw-rw---- 1 root disk      8,   0 May 21 08:38 sda
brw-rw---- 1 root disk      8,   1 May 21 08:38 sda1
...
crw-rw-rw- 1 root root      1,   3 May 21 08:38 null
...
crw-rw-rw- 1 root root      1,   8 May 21 08:38 random
...
crw--w---- 1 root tty       4,   0 May 21 08:38 tty0
...
crw-rw-rw- 1 root root      1,   5 May 21 08:38 zero
```

- **c** indicates character device - read/write bytes
- **b** indicates block devices - read/write blocks

some interesting “virtual” devices

- **`/dev/null`**
  - writes do nothing (are ignored)
  - reads return nothing
- **`/dev/zero`**
  - writes also do nothing, but use `/dev/null`
  - reads return bytes containing 0
- **`/dev/random` `/dev/urandom`**

some interesting “virtual” devices

```
$ xxd /dev/null
$ xxd /dev/zero|head
00000000: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000010: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000020: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000030: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000040: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000050: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000060: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000070: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000080: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000090: 0000 0000 0000 0000 0000 0000 0000 0000  ................
$ xxd /dev/random|head
00000000: 16a0 e6c3 16ac d43d a258 de1f 9ba8 a24b  .......=.X.....K
00000010: 5661 3488 a6e4 fa2f 4204 fb80 16ad e0ec  Va4..../B.......
00000020: 9eb2 e60b 7267 1250 1954 47a4 75bb 79bc  ....rg.P.TG.u.y.
00000030: 410c 334a b38d 4801 550a 2c83 35a3 e30d  A.3J..H.U.,.5...
00000040: d603 31b7 6062 6c54 3b6f 0e00 bd4a 765a  ..1.`blT;o...JvZ
00000050: cfe6 d39c 89ba 9a02 66cd 5044 f417 30da  ........f.PD..0.
00000060: eba5 df10 223a b963 7ad4 160c ae06 7660  ....":.cz.....v`
00000070: a4c4 352e 0252 615c 8e17 7b30 2e48 6fb3  ..5..Ra\..{0.Ho.
00000080: f9d5 d4f7 2913 73a9 3042 07de dde5 3537  ....).s.0B....57
00000090: 686c fba5 f41a b66a 7b68 2d48 3077 c672  hl.....j{h-H0w.r
```

fdisk - manipulate Disk partitions

- Disks are usually separated into separate regions called partitions.
  - allows parts of disk to be used for different purposes
- **fdisk** is a simple program to view or change partitions
- widely-used MBR partition format being replaced by GPT format
- see **gdisk** for GPT partitions
- gparted graphical tool handles both MBR and GPT

```
$ sudo fdisk -l /dev/sdi
Disk /dev/sdi: 28.66 GiB, 30752636928 bytes, 60063744 sectors
Disk model: Ultra
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x00000000

Device     Boot Start      End  Sectors  Size Id Type
/dev/sdi1          32 60063743 60063712 28.7G  c W95 FAT32 (LBA)
```

Beware: dangerous operation - have backups!

File System Formats

- **ext4** - mostly widely used Linux file-system
- **ext2/ext3** - older versions of ext4 - limited with less features
- **brtfs** - copy-on-write filesystem with interesting features
- **zfs** - filesystem which can span disks with interesting features
- **ntfs** default Windows file-system - can be accessed from Linux
- **vfat** - older Windows filesystem
  - widely used for removable devices such as SD cards and USB keys
- **nfs** - network file system used to provide remote access to file system
- **sshfs** - remote file system layered on ssh
  - you can use to access your CSE file at home

mkfs - create your own filesystem on a partition

```
# /dev/sdi is a 32Gb flash drive
$ ls -l /dev/sdi*
brw-rw---- 1 root disk 8, 96 Aug  4 12:47 /dev/sdi
brw-rw---- 1 root disk 8, 97 Aug  4 12:47 /dev/sdi1
$ mkfs /dev/sdi1
mke2fs 1.45.6 (20-Mar-2020)
Discarding device blocks: done
Creating filesystem with 262144 4k blocks and 100096 inodes
Filesystem UUID: 66028671-cece-47ff-804c-4a3b7f9f0ea5
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376
Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done
```

- **Beware**: overwrites (destroys) contents of disk - potential huge data loss!

mount - mount a file-system

- mount makes a file-system available below a point in the file-system
  - mount point must be an existing directory
- umount reverses this.

```
$ mkdir /tmp/i
$ sudo mount /dev/sdb1 /tmp/i
$ ls -l /tmp/i
..
$ umount /tmp/i
```

- distributions typically include a helper program to mount & unmount removable devices.

fsck - repair a file-system

- Power failure or other unexpected events may leave a filesystem in inconsistent state.
- **fsck** (file system check) checks and repairs a file-system.

```
# copy a random byte into raw file-system to simulate corruption
$ sudo dd if=/dev/random of=/dev/sdi1 bs=1 seek=1024 count=1
# filesystem will noy mount
$ sudo mount /dev/sdi1 /tmp/i
mount: /tmp/i: wrong fs type, bad option, bad superblock on /dev/sdi1, missing co
# repair file system
$ sudo fsck /dev/sdi1
e2fsck 1.46.4 (18-Aug-2021)
ext2fs_open2: The ext2 superblock is corrupt
fsck.ext2: Superblock invalid, trying backup blocks...
/dev/sdi1 was not cleanly unmounted, check forced.
Pass 1: Checking inodes, blocks, and sizes
...
# filesystem will now mount
$ sudo mount /dev/sdi1 /tmp/i
```

- File system should not be in use (unmounted)

/etc/fstab - filesystem configuration

- Configures file systems on device to be mounted when system starts.

```
$ cat /etc/fstab
# device    mount-point  fs-type  options
/dev/sda1      /          ext4     noatime,errors=remount-ro  1 1
/dev/sda2      none       swap     sw                         0 0
```

- Must include a root file-system on /
- Usually includes a swap device.

Often uses a unique label for devices because device names can change if hardware reconfigured, e.g. more disks added.

```
$ cat /etc/fstab
UUID=36bcedb9-de07-4de0-82c6-509000029f0e  /  ext4 defaults  1 1
```

/etc/passwd - user “database”

User information in **/etc/passwd**

Password hashes in **/etc/shadow**

Every user has unique number: **uid**

```
$ sed 2q /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
$ sudo sed 2q /etc/shadow
root:$6$YiSiP7Pehz8aoe...../:18379:0:99999:7:::
daemon:*:18362:0:99999:7:::
```

- better to not edit `/etc/passwd` & `/etc/shadow` directly
- instead use (distribution-specific) tools, e.g on Debian-like systems:
  - add users with **adduser**
  - remove users with **deluser**

/etc/group - group database

Group information in **/etc/group**

```
$ head /etc/group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:
tty:x:5:
```

- Each group has unique number: **gid**
- Better to not edit /etc/group directly
- instead use (distribution-specific) tools, e.g on Debian-like systems:
- Add users to groups with **adduser**
- Also **addgroup delgroup**

The root user

Many system actions require root (uid == 0)

su allows you to execute commands as root or any other user.

sudo allows command to be run as root

Use cautiously - easy to damage system with comands run as root.

Edit sudo config file `/etc/sudoers` with **`visudo`**

```
# Adding user to sudo group should allow them to run sudo
$ adduser andrewt sudo
```

Linux Distributions

- A distribution packages the Linux kernel with many other programs.
- Hundreds of linux distributions
- distributions used by CSE students include:
  - Debian - classic distro, used for CSE systems, runs on many architectures
  - Ubuntu - popular distro, based on Debian
  - Mint - based on Ubuntu
  - Arch - lightweight rolling release
  - Alpine - lightweight distro used in containers

The Debian Linux Distribution

- One of the oldest Linux distribution (1993)
- widely used & available for many platforms.
  - 10 CPU architecture offcially supported
  - 14+ more CPU architecture unofficially supported
- Stable - new release every 2 yrs.
- Many derivative distributions
- packages total 300+ million lines of code

Linux Packages

- A packages contains files that make up an application
- And build scripts to install/remove application.
- May contain metadata for managing the package.
- Used to install new applications onto a system
- Debian uses the **`.deb`** format (as do Ubuntu, Mint)
  - other distributions use other formats, e.g **`rpm`** (Red Hat) and **`pkg`** (Arch)
- distributions have their own package management tools
- Debian has
  - **`dpkg`** - low level tool - you probably don't need to use directly
  - **`apt`** - command-line tool - you probably want to use this
    - in scripts use **`apt-get`** (same options)
  - **`aptitude`** & **`synaptic`** - high level GUI tools

apt - Debian Package Manager

```
# update database of packages available
$ sudo apt update
# install a package + dependencies
$ sudo apt install <packagename>
# install a downloaded package file
$ sudo apt install ./package.deb
# uninstall package
$ sudo apt remove <packagename>
# update all packages
$ sudo apt dist-upgrade
# search for a package
$ sudo apt search <packagename>
# search for a package
$ sudo apt search <packagename>
# print package metadata including dependencies
$ sudo apt info <packagename>
```

dpkg - Debian Package Tool

- **dpkg** can be used to install indvidual **.deb** files
- **dpkg** is a low level tool, you mostly want **apt**
- **dpkg** can only check dependencies
- **apt** search package respositories for package
  - also downloads dependencies

```
$ curl -LO https://github.com/COMP1511UNSW/dcc/releases/download/2.7.8/dcc_2.7.8_
$ ls -l dcc_2.7.8_all.deb
-rw-r--r-- 1 andrewt 60180 Aug  4 11:18 dcc_2.7.8_all.deb
$ sudo apt install ./dcc_2.7.2_all.deb
```

dpkg - inspecting package metadata

```
$ apt info dcc
Package: dcc
Version: 2.7.8
Priority: optional
Section: devel
Maintainer: Andrew Taylor <andrewt@unsw.edu.au>
Installed-Size: 77.8 kB
Depends: python3 (>= 3.6), gdb (>= 7.12), clang (>= 7.0), valgrind (>= 1:3.13)
Homepage: https://github.com/COMP1511UNSW/dcc
Download-Size: 60.2 kB
APT-Manual-Installed: yes
APT-Sources: /tmp/2041/dcc_2.7.8_all.deb
Description: compiler for novice C programmers
 dcc compiles C programs using clang and adds explanations suitable
 for novice programmers to compiler messages novice programmers are
 likely to encounter and not understand.  dcc also adds code to the
 binary which detects run-time errors and print information likely
 to be helpful to novice programmers, including printing values of
 variable in lines used near where the run-time error occurred.
```

Linux containers

- Linux containers provided by a set of kernel features
- linux containers allow running a processes in a different “world”
- you can run a process with effectively:
  - different root directory
    - e.g. /usr/bin can look different to process
  - different namespace
    - e.g different process ids
    - e.g different view of network
  - different uid
  - specified resource limits
- allows a program to be run which needs different packages to those installed

Docker

- Docker popular set of tools for working with Linux containers
- uses a union file system in clever way
  - overlays images so base images can be resued
- can specify dependencies for a program and produce self-contained image
  - e.g. can specify program requires python-3.9
- can run image on any Linux, Windows or OSX system with Docker installed
  - independent of what software is installed on that platform
- **dockerhub** provides sharing similar to **github
- Docker great for many purposes, but heavyweight (requires daemon)
  - other container tools may be useful depending on needs
- good to experiment with over the term-break
  - containers hard to use at CSE
    - disk space issues if every student has containers
    - docker not available - requires root
    - udocker - only supports some containers
    - podman - doesn't work with NFS
  - easy to install on your own machine
    - install instructions: https://docs.docker.com/get-docker/

Running Shell Command Inside and Outside a Container

```
$ whoami
andrewt
$ pwd
/home/andrewt
$ grep andrewt /etc/passwd
andrewt:x:517:517:andrewt,,,:/home/andrewt:/bin/bash
$ docker run -it --rm debian:bookworm bash -l
root@fa3e8e5e1b38:/# whoami
root
root@fa3e8e5e1b38:/# pwd
/
root@fa3e8e5e1b38:/# find /|wc -l
57208
root@fa3e8e5e1b38:/# cat /etc/passwd
root:x:0:0:root:/root:/bin/ash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
...
root@fa3e8e5e1b38:/# grep andrewt /etc/passwd
```

Mounting A Directory Inside a Container

```
$ ls /home/andrewt|wc
     87      92     819
$ docker run -it --rm debian:bookworm bash -l
root@fa3e8e5e1b38:/# ls /home/andrewt|wc
     87      92     819
root@fa3e8e5e1b38:/# cd /home/andrewt
root@fa3e8e5e1b38:/#echo hello Andrew >hello.txt
root@fa3e8e5e1b38:/# exit
$ cat hello.txt
hello Andrew
```

Example Dockerfile

```
# use Debian bookworm as our base image
FROM debian:bookworm
# install packages and create some directories and a file
RUN \
    apt-get update &&\
    apt-get install -q -y apache2 curl &&\
    apt-get -q -y clean &&\
    mkdir -p /var/run/apache2 /var/log/apache2 /var/lock &&\
    echo hello Andrew >/var/www/html/hello.html
# specify default command
ENTRYPOINT \
    apachectl start &&\
    bash -l &&\
    apachectl stop
```

source code for Dockerfile

Example - Building and Running the Container

```
$ curl http://127.0.0.1/hello.html
curl: (7) Failed to connect to 127.0.0.1 port 80 after 0 ms: Connection refused
$ docker build -t my_apache .
$ docker run -it my_apache
root@544075dcfbf5:/# curl http://127.0.0.1/hello.html
hello Andrew
root@544075dcfbf5:/# exit
logout
$ curl http://127.0.0.1/hello.html
curl: (7) Failed to connect to 127.0.0.1 port 80 after 0 ms: Connection refused
$
```

Example - Making a Network Port visible Outside the Container

```
$ curl http://127.0.0.1:1080/hello.html
curl: (7) Failed to connect to 127.0.0.1 port 1080 after 0 ms: Connection refused
$ docker run -it -p 1080:80 my_apache
root@1699e0605ed6:/
```

In another window

```
$ curl http://127.0.0.1:1080/hello.html
hello Andrew
```

Example - Supplying Webpages from Outside the Container

```
$ mkdir /tmp/content
$ for i in $(seq 0 9);do echo $i > /tmp/content/$i.html; done
$ docker run -it -p 1080:80 -v /tmp/content:/var/www/html my_apache
root@1699e0605ed6:/
```

In another window

```
$ curl http://127.0.0.1:1080/5.html
hello Andrew
```