ECE 471 – Embedded Systems
Lecture 6

Vince Weaver

`https://web.eece.maine.edu/~vweaver`

`vincent.weaver@maine.edu`

16 September 2024

Announcements

- HW#2 was posted
- People doing OK getting Pis setup?
  Can turn into almost a sysadmin task rather than embedded systems

Connecting Remotely to Pi

- Plain text, via ssh (or serial terminal)
  Things like "screen" and "tmux" can even hold connections open
- Graphical over network
  - VNC
  - Remote Desktop
  - X11 forwarding. Over ssh, use `ssh -Y` when connecting
    After many years X11 being discontinued for Wayland

which does not have built in display forwarding

Using the Linux Command-Line

- You can interact fully with Linux at a text-based command line (a shell prompt)
- Even in a GUI you can bring up a terminal emulator to do this
- The way we did things in the old days.
  Some of us still prefer the command line.
- Good to know as in some situations (server, sysadmin, HPC, embedded systems) you might only get command line access to a machine

Files and the Filesystem

- On Linux/UNIX "everything is a file"
- A file is a collection of bytes that you can randomly access
- They are usually stored in a filesystem
  - Gives a name and directory (folder) hierarchy so you can easily find the files (rather than having to remember the block offset on disk where it lives)
  - Various types of filesystems available
  - Yet another Operating System abstraction

- Do people even use filesystems anymore?
  - There's worry that on modern interfaces (like phones) files are just magically on the cloud and you find them by search, essentially magic
  - At least currently the files still actually live on a filesystem somewhere (unless they're in a database which is another story for another time)

Steps for doing the homework via command line

- Login
- Use `pwd` to print current working directory
- Use `mkdir` to create a directory for 471 files
- Use `cd` to enter the new directory
- Use `ls` to list files
- Use `wget` to download assignment file
- Use `tar` to decompress/unpack assignment
- Use `cd` again to enter directory

- Edit the file using an editor `nano` or similar
- Run `make` to build assignment
  Aside: look at contents of Makefile
- Run `./hello_world` to test what we built
  Aside: why do you need the `./`?
  As with lots of weird things, for security reasons
- Run `env` to look at the environment variables
- Run `shutdown` to shut down when done
  Does it require `sudo` privileges?

The Linux Shell

- The command prompt is just a program called a “shell”
- Default is bash, the “Bourne Again Shell” (more computer person humor).
- There are various shells available (bash, sh, zsh, csh, tcsh, ksh)
- You can select via `chfn`
- It’s just another program in C, started by `login`
- When you run a program it runs as a separate process, but when done it returns to the shell

Root Filesystem Layout

- Executables in `/bin`, `/usr/bin`
  Describe the on-going merge of /usr/bin with /bin
- System executables under `/sbin`, `/usr/sbin`
- Device nodes under `/dev`
- Config files under `/etc`
- Home directories under `/home`, also `/root`
- Temp Files under `/tmp`. Often wiped at reboot.
- Magic dirs under `/proc`, `/sys`
- Libraries under `/lib`, `/usr/lib`, sometimes `lib64` too

- Boot files under `/boot`
- `/usr` historically was where user files lived (now that's /home). A long time ago when low on disk space sysadmins put extra stuff in /usr and it stuck.
- `/opt` often commercial software installed there
- `/srv`, `/run`, `/var` these are where server programs store data
- `/media`, `/mnt` places to mount external disks like memory keys and CD roms
- `/lost+found` where the disk checker may store lost files it finds when fixing a disk after unclean shutdown

Interesting Config Files in /etc

- /etc/fstab – the filesystems to mount at boot time
- /etc/passwd – list of all users, world readable
- /etc/shadow – passwords stored here for security reasons
- /etc/hostname – name of the machine
- /etc/hosts – list of local machines, usually searched before resorting to DNS lookup over network
- /etc/resolv.conf – where your nameserver address is put
- /etc/sudoers – list of users allowed to use “sudo”
- /etc/network/interfaces – on Debian the network

settings are stored here

- /etc/rc* – what gets run at boot (prior to systemd)

Running things on Boot

- Useful for 471 projects
- Official solution is to write a systemd startup file (TODO: look up official name)
- In the old days you just stuck a shell script in /etc/rc*
- You can still do that with `rc.local` but not sure if that will keep working

Devices

Block vs Char devices

- /dev/sd* – SCSI (hard disks)
- /dev/mmc* – SD cards
- /dev/tty* – tty (teletype, logins, serial ports)
- /dev/zero
- /dev/full
- /dev/random , /dev/urandom
- /dev/loop

Network devices are an exception.

Interesting /proc Files

These files are not on disk, but “virtual” and created on-the-fly by the operating system when you request them.

- /proc/cpuinfo – info on cpu
- /proc/meminfo – memory info
- Each process (running program) has its own directory that has info about it

Processes

- Each program assigned its own number, a process id, often called a “pid”
- Can list processes with `ps -efa`
- Also can get real-time view of what’s going on in a system with `top`
- `htop` is a more advanced `top`

Common Commands

- `ls` : list files

  `ls -la` : list long output, show all (hidden) files. on Linux any file starting with . is hidden

  `ls -la /etc` : list all in /etc directory

  `ls *.gz` : show all ending in gz. * and ? are wildcards and can be used as regular expressions.

- `cd DIR` : change directories (folders)

  `cd ..` : go to parent directory

  `cd .` : go to current directory

`cd /` : go to root directory

`cd ~` : go to home directory

- `cat FILE` – dump file to screen (originally used to conCATenate files together but more commonly used to list files)
- `more / less` – list contents of file but lets you scroll through them. less more advanced version of more
- `exit / logout / control-D` – log out of the machine
- `df / du` – show disk space

`df -h` pretty-prints it

- `man command` – show documentation (manual) for a command. For example `man ls`
- `rm` remove file. CAREFUL! Especially famous `rm -rf`. In general on Linux you cannot undo a remove.
- `cp` copy file. CAREFUL! By default will overwrite the destination without prompting you.
- `mv` move file. CAREFUL! Can overwrite!
  `mv -i` will prompt before overwrite

- `tar` create archive file `tar cvf output.tar dir`
  `tar xzvf output.tar.gz` uncompresses a .tar.gz file
- `gzip / gunzip / bzip2 / bunzip2` compress/uncompre
  a file. gzip and bzip2 are two common formats, many more exist

Compiler / Devel Commands

- `make` – build a file based on list of dependencies in Makefile
- `gcc` – C compiler. Simplest something like this: `gcc -O2 -Wall -o hello hello.c`
- `g++` C++ `gfortran` Fortran
- `as`, `ld` – assembler and linker
- `gdb` – debugger

- `strace` – list system calls
- `git` – source code management

Other Commands

- `shutdown` – used to shutdown / reboot
- `last` – list last people to log in
- `su` / `sudo` – switch to root, run command as root
- `uptime` – how long machine has been up
- `date` – show the date

  as root you can use `date -s` to set the date

- `whoami` – who are you
- `write` / `wall` / `talk` – write to other users
- `finger` – get info on other users
- `w` / `who` – see who is logged in
- `wc` – count words/bytes/lines in a file
- `dmesg` – print system and boot messages (might need sudo)

- `ln` – link files together, sort of like a shortcut
  `ln -s goodbye.c hello.c` – symbolic link. also hard links

- `dd` – move disk blocks around, often used for creating disk images

- `mount` / `umount` – mount or unmount filesystems

- `mkfs.ext3` – make new filesystem

- `e2fsck` – filesystem check

- `ifconfig` / `route` – show and setup network config being replaced by `ip` tool

- `dpkg` / `apt-get update/upgrade/install` – Debian only package management

- `ssh` / `scp` – log into other machines, copy files remotely

- `lynx` – text-based web browser

- `reset` – clear the screen and reset settings (useful if you accidentally cat a binary file and end up with a screenful of garbage). Control-L also refreshes the screen

- `linux_logo` – my program
  Try `linux_logo -L 9 | less`

Editing files

Linux and UNIX have many, many editors available. Most famous are vi and emacs. On our board using nano might be easiest.

- `nano` – a simple text editor.
  `nano FILENAME` – edit a filename
  It shows the commands you can do at the bottom. `^O` means press control-O
  control-O : writes
  control-X : exits

control-W : searches

control-\\: search and replace

control-C : prints line number

Redirection and Pipes

- redirect to a file : `ls > output`
- redirect from a file : `wc < output`
- pipe from one command to another : `ls | wc`, `dmesg | less`
- re-direct stderr : `strace 2> output`

Suspend/Resume

- Press control-C to kill a job
- Press control-Z to suspend a job
- Type `bg` to continue it in the background
- Type `fg` to resume it (bring to foreground)
- Run with & to put in background to start with. (ie, `mpg123 music.mp3 &`).

Permissions

- user, group – use chgrp
- read/write/execute – use chmod

Shell Scripts

- Create a list of files in a dir
- Start with the shell, `#/bin/sh` (or perl, etc)
- Make executable `chmod +x myfile`

Command Line History

- Can press “tab” to auto-complete a command
- Can press “up arrow” to re-use previous commands
- Can use “control-R” to search for previous commands

Environment Variables

- `env`
- Varies from shell to shell.
- `export TERM=vt102`
- PATH, and why "." isn't in it. This is why you have to run self-compiled binaries as `./blah`