ECE 471 – Embedded Systems
Lecture 24

Vince Weaver

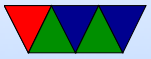
https://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

3 November 2025

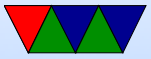
Announcements

- Demosplash
- HW#8 was posted, pick up hardware if you haven't
- Don't forget project ideas for Friday (7th)
  E-mail, one per group, with group members and topic
  Please e-mail even if I said it was OK in person



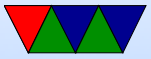
Current Events: Daylight Savings

- Clock math always hard for computers
- Can you hardcode start/stop days for Daylight Savings?
- Some states don't follow it (mostly AZ these days)
- Congress can (and has) changed it on fairly short notice
- Other countries this happens yearly
- You will often see OS / firmware updates because of this



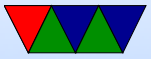
Avoiding DST problems

- Linux/UNIX often had system clock UTC + offset rather than trying to have it set to local time
- Old Windows/DOS would use local time which could cause problems



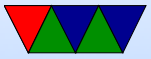
Embedded System Security: Voting

- Election Tuesday
- Are voting machines embedded systems?
- Can elections be hacked?
- Most reports of hacking are social engineering type (i.e. Foreign Country buys deceptive Facebook ads)

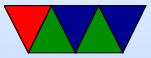


Voting Machine Types

- Vary a lot by state and even town (good is some ways, hard to target them all)
- Old fashioned: purely paper, machines with levers
- Like Maine: mark paper ballot, use embedded system to count, have paper trail if need recount
- Touch-screen voting (often encouraged for accessibility), OK if it prints paper ballot you can later hand count
- Fully electronic with no paper trail, difficult to audit
- Even worse: all internet voting

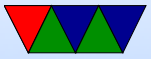


- Mail in votes have their own issues, make votes possibly non-anonymous so people can pressure/bribe you to vote a certain way
- Maine Question #1 in 2025



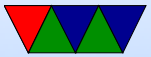
Voting Machine Hacking

- Some voting machines found trivial to hack. Running windows, with exposed USB connector.
- How did researchers get access to them. (eBay)
- Attacks often have to be local unless you happen to hack main database



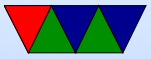
Why hard to audit/secure?

- Often are old and not tested well (Windows XP, only used once a year)
- Run by small teams in towns w/o much IT experience
- Often bought from lowest bidder
- Can you trust the software? What if notices it is Election Day and only then flips 1/10th the vote from Party A to Party B. Would anyone notice? What if you have source code?
- What if the OS does it. What if Windows had code that



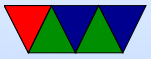
on Election Day looked for a radio button for Party A and silently changed it to Party B when pressed?

- OK you have and audit the source code. What about the compiler? (Reflections on Trusting Trust). What about the compiler that compiled the compiler?
- And of course the hardware, but that's slightly harder to implement but a lot harder to audit.



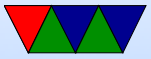
Summary of Shared vs Static Libraries

- Shared libraries, only need one copy of code on disk and in memory
  - Good for embedded system (less room needed)
  - Good for security updates (only need to update lib, not every program using it
- Static libraries, all libraries included
  - No dependencies
- These days maybe containers, docker, kubertenes
- Also Flatpack, Snap. Why? Stability, Know package



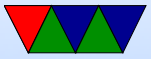
will work on all distributions, Not have to install dependencies

- Can use `ldd` to view library usage



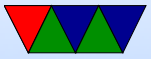
More on Firmware – Where to Draw Line?

- On desktop, the bootloader and kernel considered software
- On embedded system, if those are in flash or otherwise hard to access, would be firmware

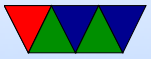


Embedded System Firmware

- Many devices are their own embedded systems these days.
  Lots of small things have microcontroller inside
- What is firmware?
- Code that runs on an embedded system
  - Is it only bare metal?
  - Could a full Operating System be included?
- Traditionally is a binary "blob"
- In ROM? Or upgradable? Why might you want to

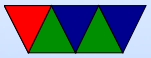


upgrade? (bug fixes, economy, etc.)



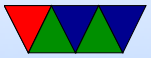
Binary Blobs

- What is in the firmware?
  - Can you modify it yourself?
- Can it contain a full operating system?
  - ThreadX on Pi, Minix on Intel servers?
- Where does it live?
  - Hardcoded in ROM?
  - Upgradable by flash?
  - RAM that's uploaded at boot?



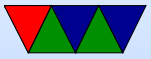
Non-persistent Firmware

- RAM that gets uploaded after boot (loses state on power off)
  What happens if your hard-drive or some other boot critical hardware uses this?
- Can be annoying if writing low-level code, you bought hardware but it might not work at all unless you upload a bunch of firmware to it each time you power up



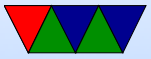
Open Source Firmware Issues

- Can you run a "completely" free computer with completely open software?
- Note the different uses of Free
  - Free as in Freedom/Liberty to use code as you wish
  - Free as in no cost to you
- Depends on how you classify firmware



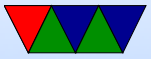
Linux Firmware Issues

- licensing issues
- can boot media (CD, USB) ship with firmware
- What if distro (like Debian) has rules against non-open binaries
- Can you netboot if the network card requires proprietary firmware (common problem with wifi cards)
- Weird workarounds, webcam that had MacOS driver so had to extract firmware from that and copy to Linux partition



Firmware Licensing Issues

- Is a computer system truly free software if binary blobs running (like on a Pi)
  - Aside, FSF has weird definition where they only get upset about updatable firmware
- Debian tried to have a firmware-free install by default, but had to give up as not practical (especially at install)
- If someone ships firmware that has GPL code in it, what are their responsibilities?



Offloading Work to Firmware

- Companies don't like sharing their code, which makes things like Linux drivers hard
- Can they put as much as possible into firmware with only a small stub in OS?
  - Benefit: OS does less work, is simpler
  - Downside: you have no idea what the code is doing
- On Linux, NVIDIA drivers famously contentious. Possibly they are now planning to push more and more code onto firmware on the card itself