☑ 1. Last time
☑ 2. Final exam
☑ 3. Your questions
☑ 4. Wrap-up

---

2. Final exam
   - 110 minute exam
      - stay seated at 100 mins
      - closed book
      - TWO two-sided sheets allowed

Material
   - Readings
   - Labs
   - HWs $\longrightarrow$ see l12.txt
   - Classes
      [see midterm topic list]

Post-midterm topics (not guaranteed to be necessary or sufficient)
   virtual memory
      virtual memory on x86-64
         virtual address $[0000|\overset{\text{36 bits}}{\cdots\cdots}|\text{12 bits}]$
      entry in L1...L4 page tables

entry

[40 bits   more bits   bottom 3 bits]

protection (u/s | w/R | P/NP)

what's a TLB?

- page faults
  - mechanics
  - costs
  - uses
- page replacement policies (FIFO, LRU, CLOCK, OPT)
- thrashing
- mmap()

I/O
- architecture
- how CPUs and devices interact
  - mechanics
  - polling vs. interrupts
  - DMA vs. programmed I/O
- device drivers

synchronous vs. async I/O

context switches

User-level threading

Disks

- geometry
- performance
- interface
- scheduling (skipped in class, covered in book)

File systems
- basic objects: files, directories, metadata, links, inodes
- how does naming work?
- types of file layout
  - extents/contiguous, linked, index
  - classic Unix + FFS are variants of indexed
- analogy between inode and top-level page directory (aka L1 page table)
- tradeoffs
- performance

Crash recovery
- ad hoc
- copy-on-write (cow)
- journaling (redo logging, undo logging, undo+redo)
  WAL

RPC, client/server systems

case study: NFS

marquee user of RPC

mostly skipped in class, covered in book

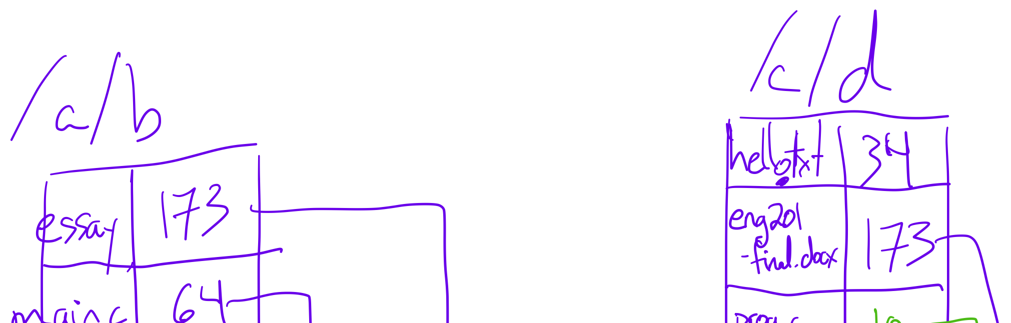
protection and security

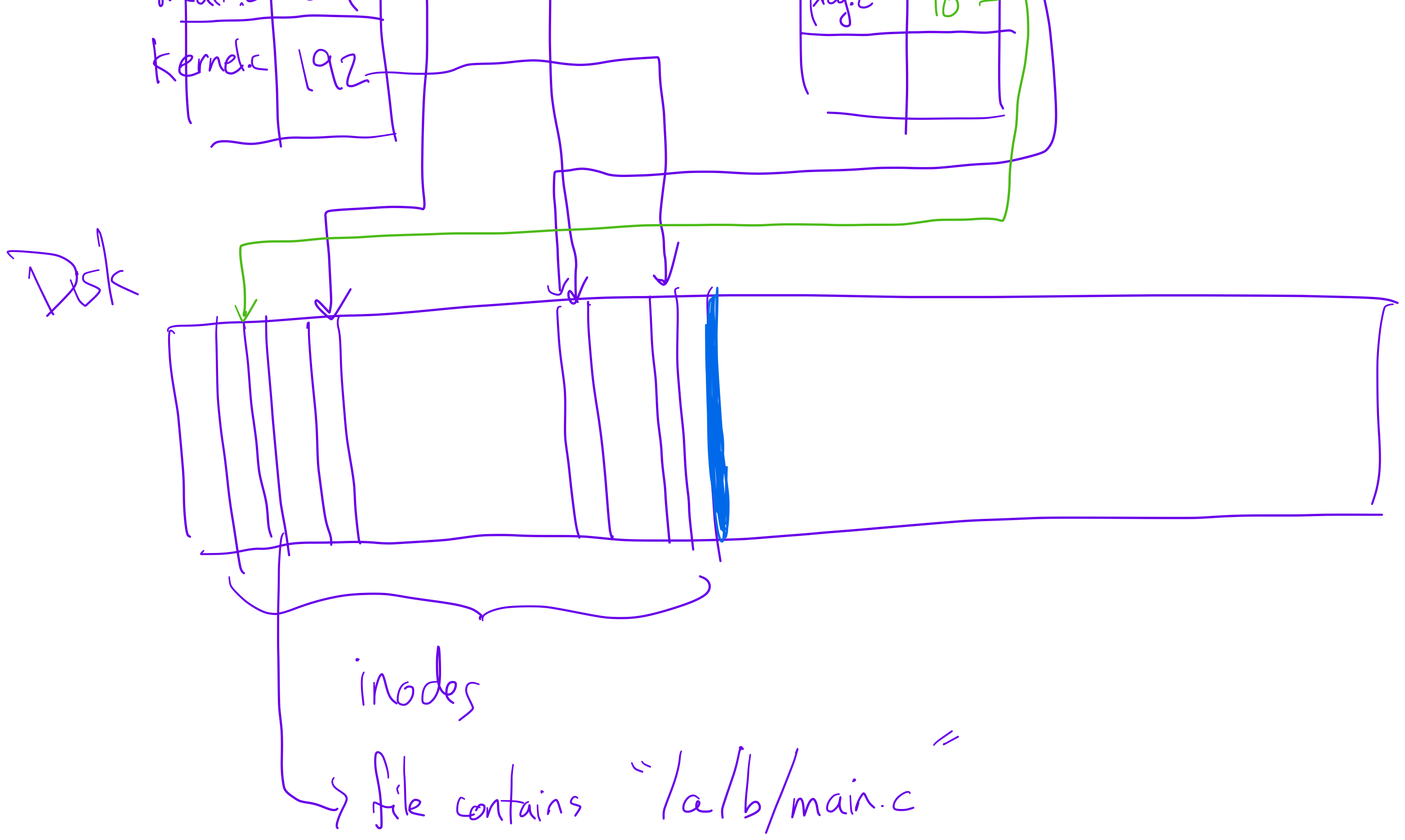
- stack smashing / buffer overflow
- Unix security model
  - access control, privileges, setuid, attacks
- trusting trust

bootup, from power-on

- static linking + loading is a key tool
- bootstrap process
  - H/W copies firmware into read/write mem
  - firmware is mini OS
  - runs bootloader program, which ultimately begins kernel
  - kernel invokes init(1)
  - init(1) invokes login(1)
  - login(1) lets you get a shell and begin executing programs

---



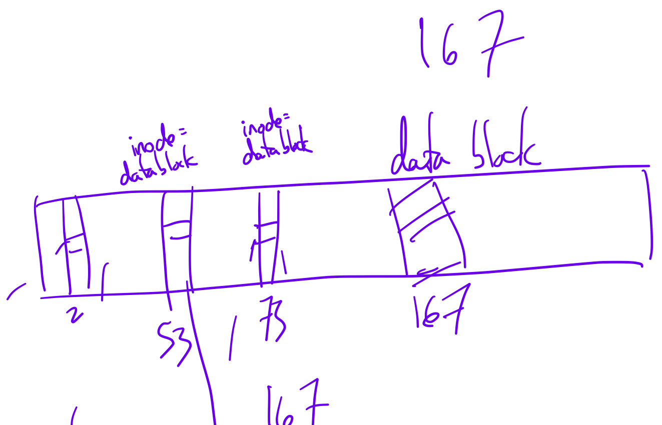


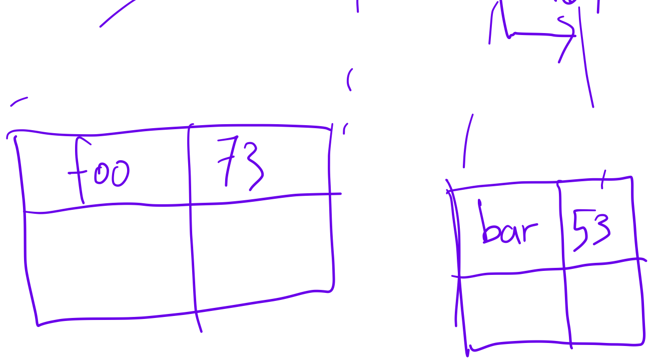
vm-map (pg-table, 0x200000, pa1, PGSIZE,
PTE_P,
a|b|c);

vm-map (" ", 0x300000, pa2, 5*PGSIZE, ...);
vm-map (" ", 0x301000, pa3, PGSIZE, ...);

/foo/bar

2.1 reads and writes





foo
73
bar
53