Welcome to COMP 630

Introduction and Review

Don Porter

Why Grad OS?

- Primary Goal: Demystify how computers work

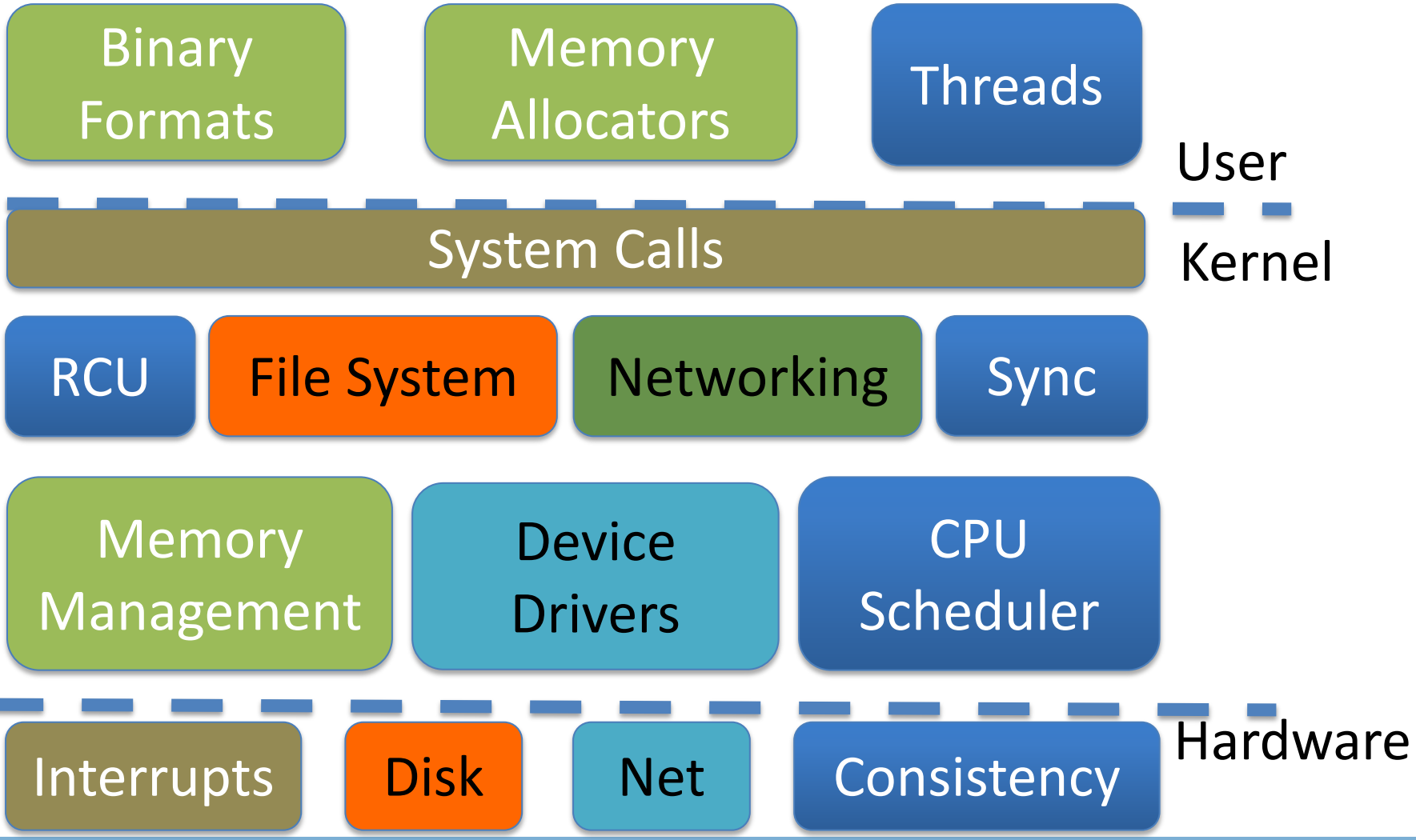
An example progression

- Undergrad OS:
  - High-level understanding of paging
  - Theoretical issues like fragmentation
- Grad OS Impl. (630): Build a pager
  - Solid understanding of how paging SW + HW work
- Advanced Grad OS (730): Read novel research papers
  - Do creative things with paging: virtualization, security, etc
  - Plan to do this next spring

This class: Learn by doing

- You will write major chunks of your own OS
  - Memory management, context switching, scheduler, file system, IPC, network driver, shell, etc.
  - Linux scheduler:
    - Difficult to understand just by reading source
    - Small modifications require first understanding the code
    - Impossible to replace/reimplement
  - No substitute for building it yourself!

A logical view of the OS



Labs, cont.

- This course is **coding intensive**
  - You should know C, or be prepared to remediate quickly
  - You will learn basic, inline x86 assembly
  - You must learn on your own/with team of up to 3 students
- The lab is difficult, but worthwhile
  - You will want to commemorate, with a T-shirt, tattoo, etc.

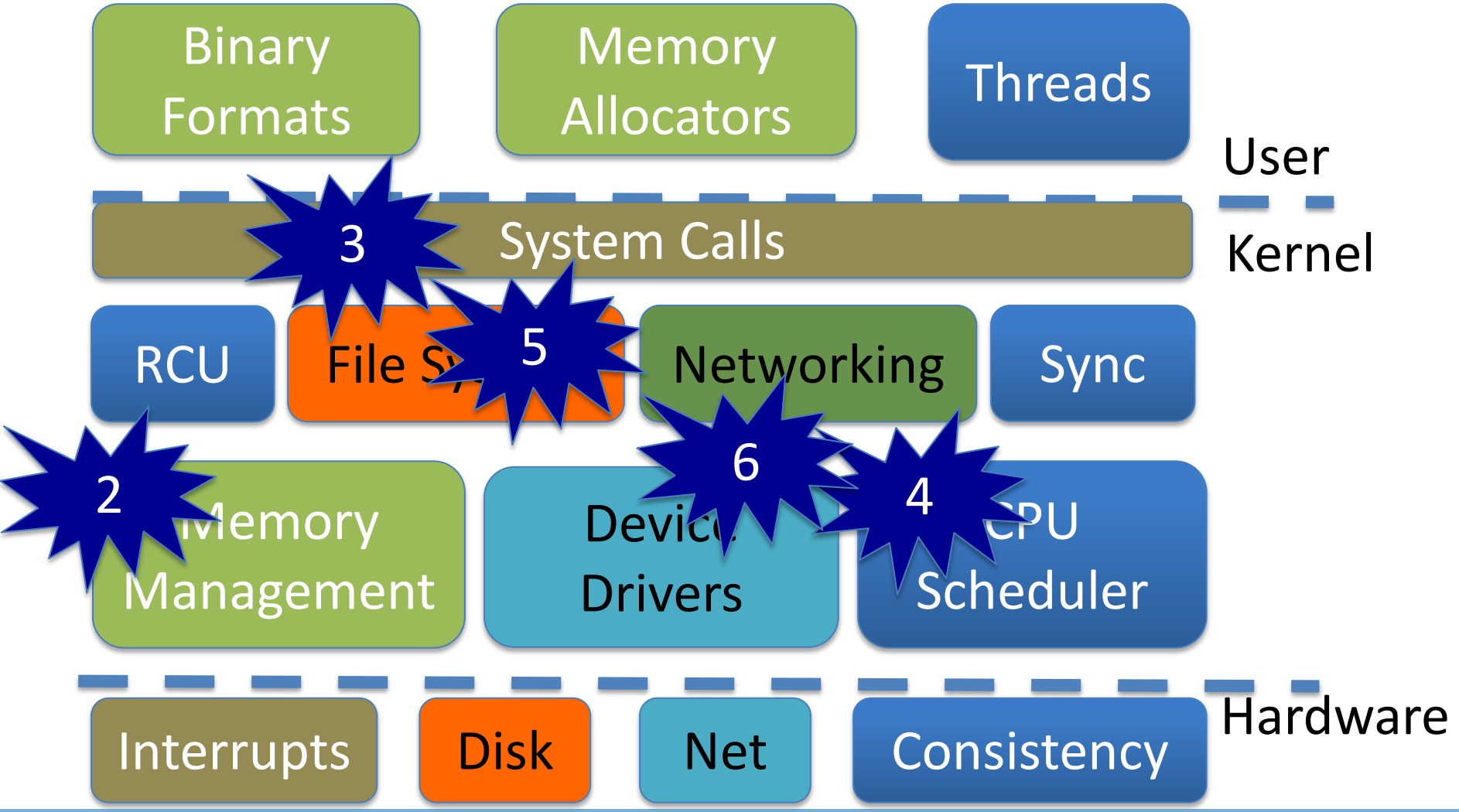
JOS

- Developed at MIT, used at several top schools
  - The "J" is for Josh Cates, not Java
- In C and Assembly, boots on real PC hardware
  - You get the skeleton code, fill in interesting pieces
- Build the right intuitions about real OSes
  - but with much simpler code

JOS 64

- You will actually implement a 64-bit variant of JOS
- Developed at by my TAs back at Stony Brook!
  - Primarily by Amit Arya and Abhinand Palicherla
  - Contributions also by: Vivek Kulkarni, Varun Agarwal, Chia-Che Tsai, Tao Zhang, Sagar Trehan, Jiahong Huang…
    - Some of these final projects or just contributions from a previous course
    - See your name here next year if you add a particularly useful feature!

JOS Labs


Binary Formats
Memory Allocators
Threads
User
3
System Calls
Kernel
RCU
File Sy
5
Networking
Sync
2
Memory Management
6
Devic
Drivers
4
CPU Scheduler
Interrupts
Disk
Net
Consistency
Hardware

Lab 6

- 3 Options
  1) Network device driver (guided assignment)
  2) Make JOS a hypervisor (guided assignment)
  3) Open-ended project
     - Add a significant feature to JOS
     - A research task on another system

Challenge Problems

- Each lab includes challenge problems, which you may complete for bonus points (generally 5—10 points out of 100)
  - Unwise to turn in a lab late to do challenge problems
  - Can complete challenge problems at any point in the semester---even on old labs
- Indicate any challenge problems completed in challenge.txt file

No Textbook

- You're welcome
- Several recommended texts
  - Several free on safari online site
  - Others at library
  - Required readings will mainly be papers you can print out

Readings

- My lectures aren't perfect; some concepts are subtle
  - Reading other words can be helpful for reinforcement and clarification
- You will learn more in class if you read before class
  - Can't ask the textbook questions
- ~7 papers will be posted and discussed over the course of the semester; these you should definitely read before class

Lectures

- Compare and contrast JOS with real-world OSes
  - Mostly Linux, some Windows or OS X, FreeBSD, etc.
- Supplement background on hardware programming
  - Common educational gap between OS and architecture

My Lecture Style

- I like participation and questions
- I can explain any concept in many ways, and explain missing background on the fly
  - …but I can't read your mind---I need to know if you don't understand something!

Administrative

- Syllabus, schedule, homework, etc. posted on course website
- www.cs.unc.edu/~porter/courses/comp630/s22

Recordings

- I usually record lectures for students to review later
  - I will share on Panopto/Sakai
- Recordings are **best effort**
  - Recordings may fail, be unwatchable, or get deleted by accident
  - Or be discontinued if too many students stop attending
    - I need your facial expressions and questions to know if lectures make sense
- Do not use this as a substitute for class attendance
  - Well, except for COVID absences…. I do have old videos in a pinch…

Guest Lectures

- Senior graduate students will give some lectures to gain teaching experience
- Professor Porter will review and critique guest lectures (in person or recorded) with guests

- Please:
  - Ask questions if something is unclear: in class or on piazza
  - Give Prof. Porter comments on guests (and his lectures)---positive and negative

Prerequisites

- Undergrad OS
  - In some cases, industry experience is ok
  - Worth brushing up if it has been a while
- C programming
- Basic Unix command-line proficiency
- See me if you have already done the JOS lab, or similar

Piazza

- This is the primary announcement medium
- And for discussions about course work
  - Do not post code here or other solutions
  - Goal: Everyone can learn from general questions
- Material discussed on piazza can be an exam question

- Details for piazza forum are on the course website

Other administrative notes

- Read syllabus completely
- 2 exams cover: lectures, labs, mailing list
- Every student will use walter.cs.unc.edu
  - Log in with your ONYEN
  - You may use your own computer, staff can't support it
- Private messages by email or piazza are fine, as needed

Special Offer!

- You can write your own exam questions
  - Send them to me in advance of the test, if I like them, I will use them
  - Do NOT share with anyone else

Lab Team (up to 3)

- Can work alone, but better with help
  - Some excellent students earned A's working alone
  - Many good students earned B's working alone
  - No need to be a hero
- Choose your own team
  - Piazza list good for finding them
- Same team for entire course
  - Changes only with instructor permission

Academic Integrity

- I take cheating very seriously. It can end your career.
- In a gray area, it is your job to stay on right side of line
- Never show your code to anyone except your partner and course staff
- Never look at anyone else's code (incl. other universities)
- Do not discuss code; do not debug each other's code
- Acknowledge students that give you good ideas

Integrity Homework

- Exercises applying course policies and ethics to several situations
- Due via gradescope on Wed, 1/19

Lateness

- Each group gets 72 late hours
  - List how many you use in slack.txt
  - Each day after these are gone costs a full letter grade on the assignment
- It is your responsibility to use these to manage:
  - Holidays, weddings, research deadlines, conference travel, Buffy marathons, release of the next Zelda game, etc.
- 4 Exceptions: illness (need doctor's note), death in immediate family, accommodation for disability, university approved religious accommodation

Lab 1 assigned

- Due Tues, 1/25 at 11:59 pm, eastern.
- Instructions on website
- Quick demo

Github Classroom

- Git/github are powerful and common industry tools
  - Create a github account if you don't have one already
  - We will use this to push out starter code
  - Recommend using it for group-internal code reviews
- We will use gradescope for submission/grading by selecting the branch you wish to submit
- Labs are auto-graded
  - Include 'make grade' you can run locally, so no surprises
- Bear with us as we work out any issues

Getting help

- Instructor keeps office hours
  - Note that “by appointment” means more time available on demand
- And piazza

Questions?

- We will use next Weds 1/19 for in-class hacking
- Remember:
  - Do academic honesty homework (due 1/19)
  - Lab 1 out (due 1/25)