CS179F: Projects in Operating System

Introduction

Emiliano De Cristofaro and Lian Gao

Team

- Instructor: Emiliano De Cristofaro (emilianodc@cs.ucr.edu)
  - I am a Professor in CSE working on security, privacy, and cybersafety
  - Office hours: TBA
- TA: Gao Lian
  - PhD student in cybersecurity
  - Office hours in lab on Wednesdays (more details later)

Projects

- 5 projects in xv6-riscv, one every 2 weeks, each 20% of the final grade
    - Unix Uilities: sleep, find, xargs
    - Memory Allocation
    - Copy-On-Write
    - File System: large files and symbolic links
    - mmap

Projects (with deadlines)

- 5 projects in xv6-riscv, one every 2 weeks, each 20% of the final grade

• Unix Uilities: sleep, find, xargs	Oct 18th, 1:59:59pm
• Memory Allocation	Nov 1st, 1:59:59pm
• Copy-On-Write	Nov 15th, 1:59:59pm
• File System: large files and symbolic links	Nov 29th, 1:59:59pm
• mmap	Dec 13th, 1:59:59pm

Project “Rules”

- Each project should be finished **individually**, unless the class size increases unexpectedly
  - Discussions are fine and encouraged
  - TA and I are there for help, try Piazza first before email
  - Other “ways” to get coding done? E.g., Github Copilot?
- Late policy
  - 20% if within 48 hours
  - No grading beyond 48 hours (exceptions granted with evidence)

Class Material

- https://github.com/emidec/cs179f-fall23

Resources

- Operating Systems: Three Easy Pieces, Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau
- XV6: A Simple UNIX-like Teaching Operating System, Russ Cox, Frans Kaashoek, and Robert Morris
- Lions' Commentary on UNIX' 6th Edition, John Lions, Peer to Peer Communications. ISBN: 1-57398-013-7. 1st edition (June 14, 2000)
- A good guidance: https://pdos.csail.mit.edu/6.828/2023/labs/guidance.html

Class Schedule

- Lectures: Tuesdays 3:30-4:20pm, Watkins 1101 (Oct 3 - Dec 5)
- Labs: Wednesdays 6:00-8:50pm, Sproul Hall 2340 (Oct 4 - Dec 6)

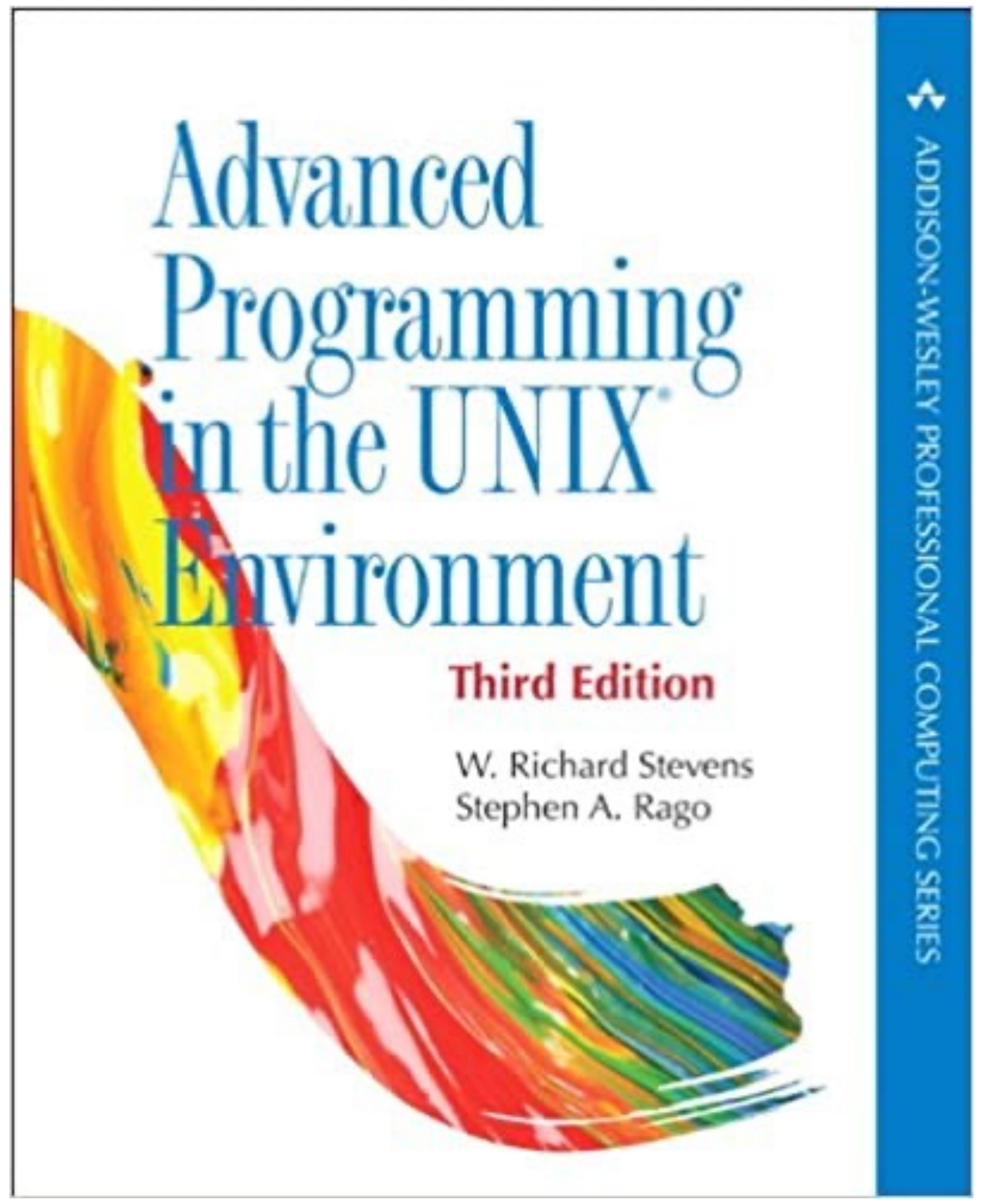
- **We don’t need both sessions every week. Let’s discuss options at the end of the lecture.**

Communication

- Piazza (https://piazza.com/ucr/fall2023/cs179f) as the main communication channel
  - Announcements, slides, projects, polls, etc.
  - Discussion and Q&A
- Canvas (https://elearn.ucr.edu/courses/110956)
  - For assignments and grades

Objectives

Why taking this class?


Advanced Programming in the UNIX® Environment
Third Edition
W. Richard Stevens
Stephen A. Rago
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

- Get familiar with system programming (lab 1)
  - Purpose of OS: help app developers achieve their goals
  - System programming: leverage what the OS provides (via syscalls) to implement ideas
    - For example: `sleep, ls, find, grep, xargs`
    - For example:

Objectives

Why taking this class?

- Have a better understanding of how OS works, why?
    - Trouble shooting
        - Why I can only open `NFILE` files?
        - Why I can’t have files larger than 268 blocks?
    - Performance: I have a bottleneck at reading files
    - Capabilities: what if the kernel doesn’t have what I want?

Objectives

Why taking this class?

- Have better understanding about OS concepts
  - Copy-on-write
  - Lazy allocation
  - Memory mapped files
  - inode

Objectives

Why taking this class?

- Further improve your problem solving skills
  1. Understand what is the task
     - What is the input and what is the output?
  2. Sketch your solution
     - What information is required? How to process the info?
  3. Prepare some test cases (from simple to complex/corner cases)
  4. Implement your solution and test it

Objectives

Why taking this class?

- Further improve your debugging skills
    - why `usertests` fail?!

Additional Objectives?

- Have a taste of AI assistants
- At the beginning of the quarter
    - Github Copilot, ChatGPT
- By the end of the quarter
    - GPT-4, Microsoft 365 Copilot, LLaMA (Meta), PaLM (Google)

Who Should be Worried?

Extra Credits?

- Re-implementation of attacks or defenses against/for systems/OS security
  - Papers published in top-tier security conferences

Environment — xv6

- We will use the XV6 operating system as a base for our projects
  - A re-implementation of Unix Version 6 for a modern RISC-V multiprocessor using ANSI C
- Familiarize yourself with XV6 on how it is organized and implemented:
  - Take a look at the online version of the Lions commentary
  - Look at the source code, etc.

Tools

- **xv6-riscv** (see previous slide/class GitHub)
- **qemu** (open source machine emulator and virtualizer)
- **labs code** (on class repo)

See README.md on the class repo (https://github.com/emidec/cs179f-fall23) for more info on how to set everything up

Note: currently having trouble with Mac (use Linux VM) and new versions of qemu (use v4 or v5, not v8)

Lab 1

- Implement the UNIX program sleep for xv6
- Write a simple version of the UNIX find program: find all the files in a directory tree with a specific name
- Write a simple version of the UNIX xargs program: read lines from the standard input and run a command for each line, supplying the line as arguments to the command

See class git repo / https://github.com/emidec/cs179f-fall23/blob/xv6-riscv-fall23/doc/lab1.md

Lab 1 — Util

- Quick reference:
  - $ make qemu   // compile and run xv6
  - $ make grade // test your solution with the grading program
  - $ ./grade-lab-util sleep
    - $ Make GRADEFLAGS=sleep grade
  - To quit qemu type: ctrl+a x

- To compile your program:
  - Add your program under /xv6-riscv/user named as <prog>.c
  - Modify UPROGS in Makefile accordingly

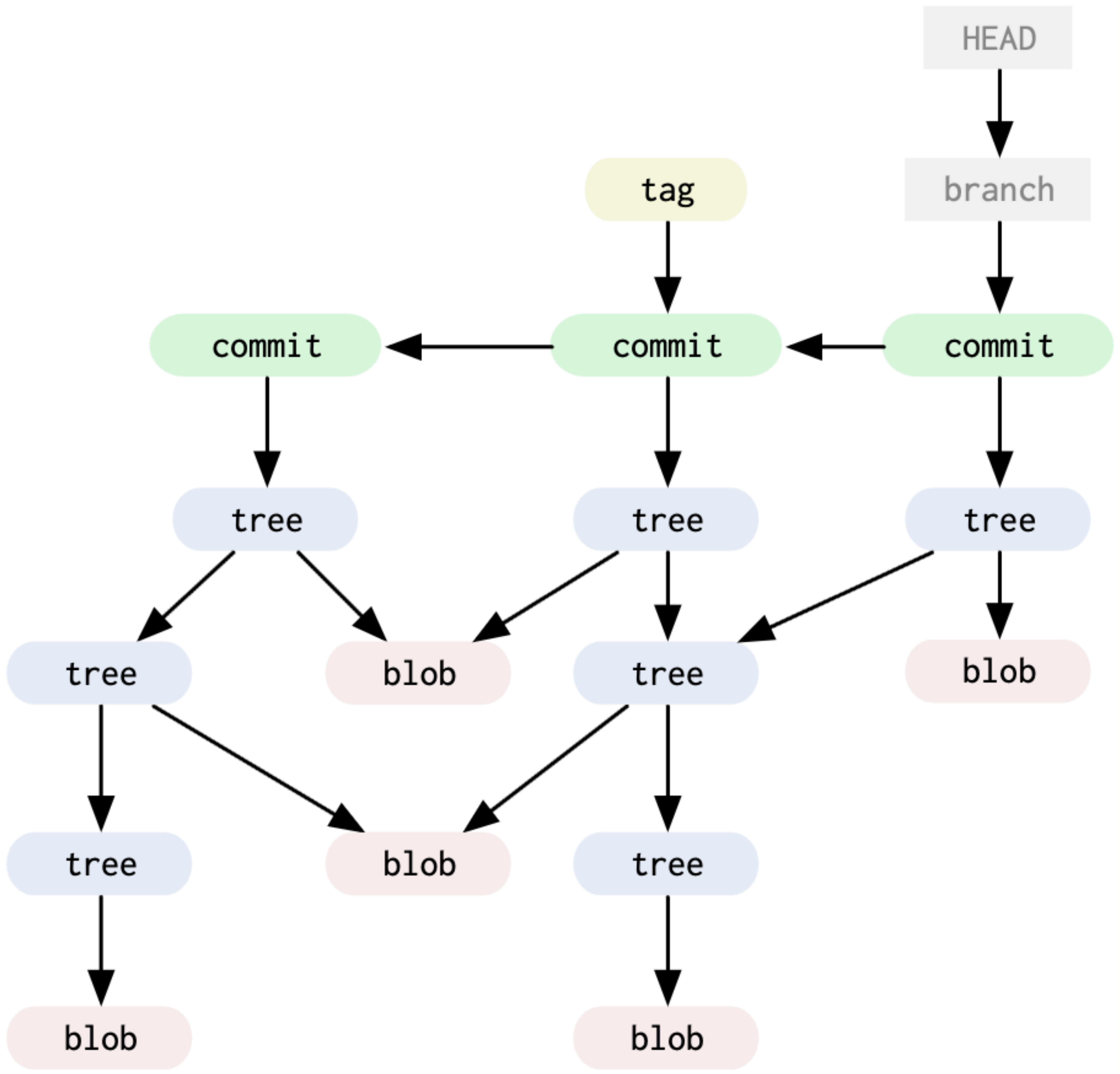
GIT

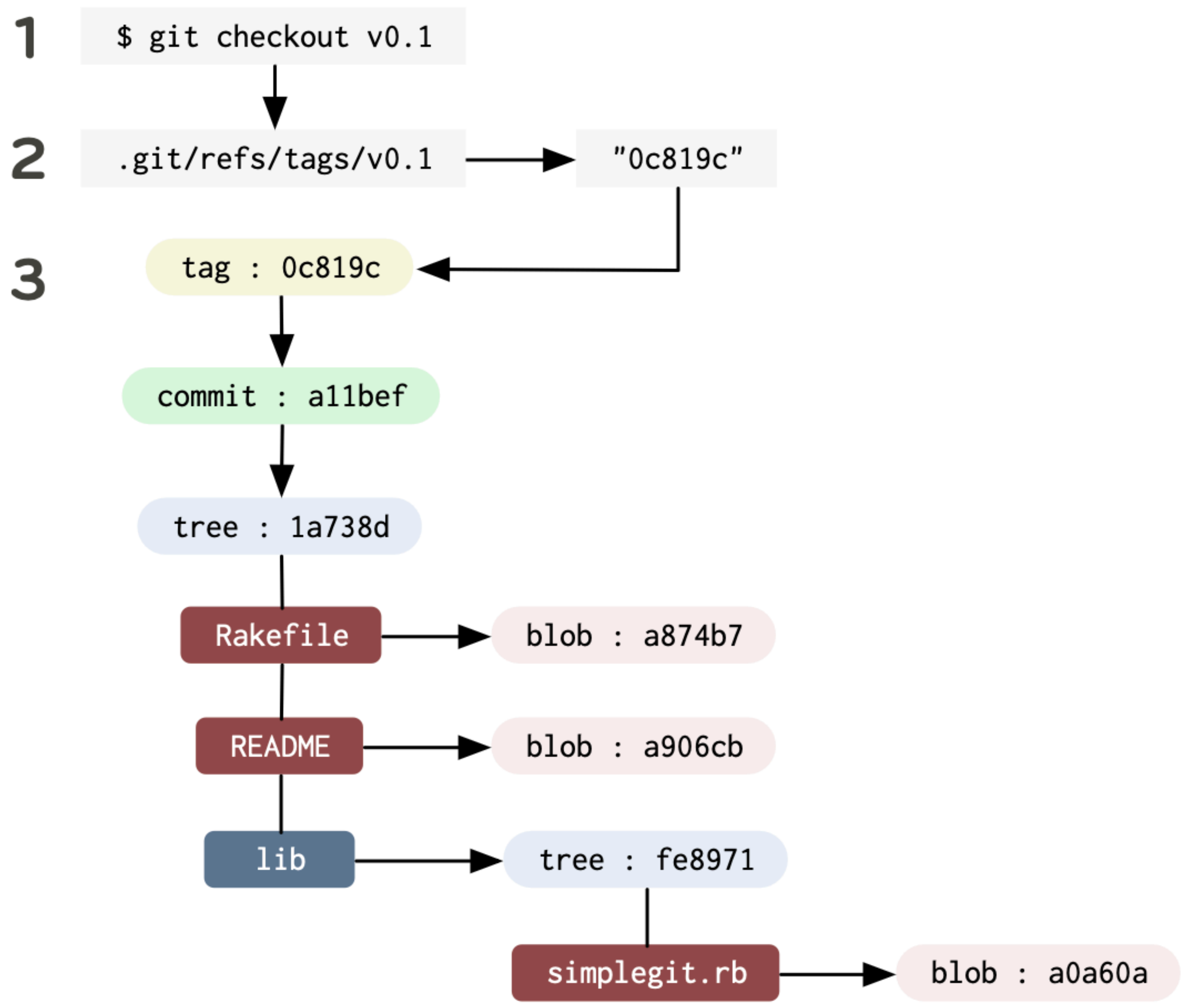
A quick introduction

- A version control system (and a file system)
  - Data is stored as `blobs` (files) and `trees` (directories)
  - `commit`: a commit is a reference (with a message/comment) to a `tree`, which represents the state of the project (fs), the ID of a `commit` is a SHA-1 hash
  - `refs`: named references to `commits/refs`, such as `HEAD`, `HEAD^1`, `TAGS`, `branches`
  - `remote refs`: references to a remote project (fs)

GIT

Structure and traversal





```
$ git checkout v0.1
```

GIT

Why we like it?

- Everything is self-contained, no central storage (subversion, CVS), no background services/daemons
  - Want to backup? Just copy the project directory
  - Access to remote branches? SSH is enough (e.g., `sledge:xv6`)
    - Fully distributed, excellent support for parallel development
- Fast, simple, support anything
  - You can use git to version control everything

GIT

Basic operations

- `clone`: copy the whole thing (directory)
- `checkout`: go to a commit
- `diff`: show what has changed
- `add`: what changes to be included in a commit
- `commit`: snapshot the state and add a message
- `log`: examine the history
- `stash`: temporarily save the current changes

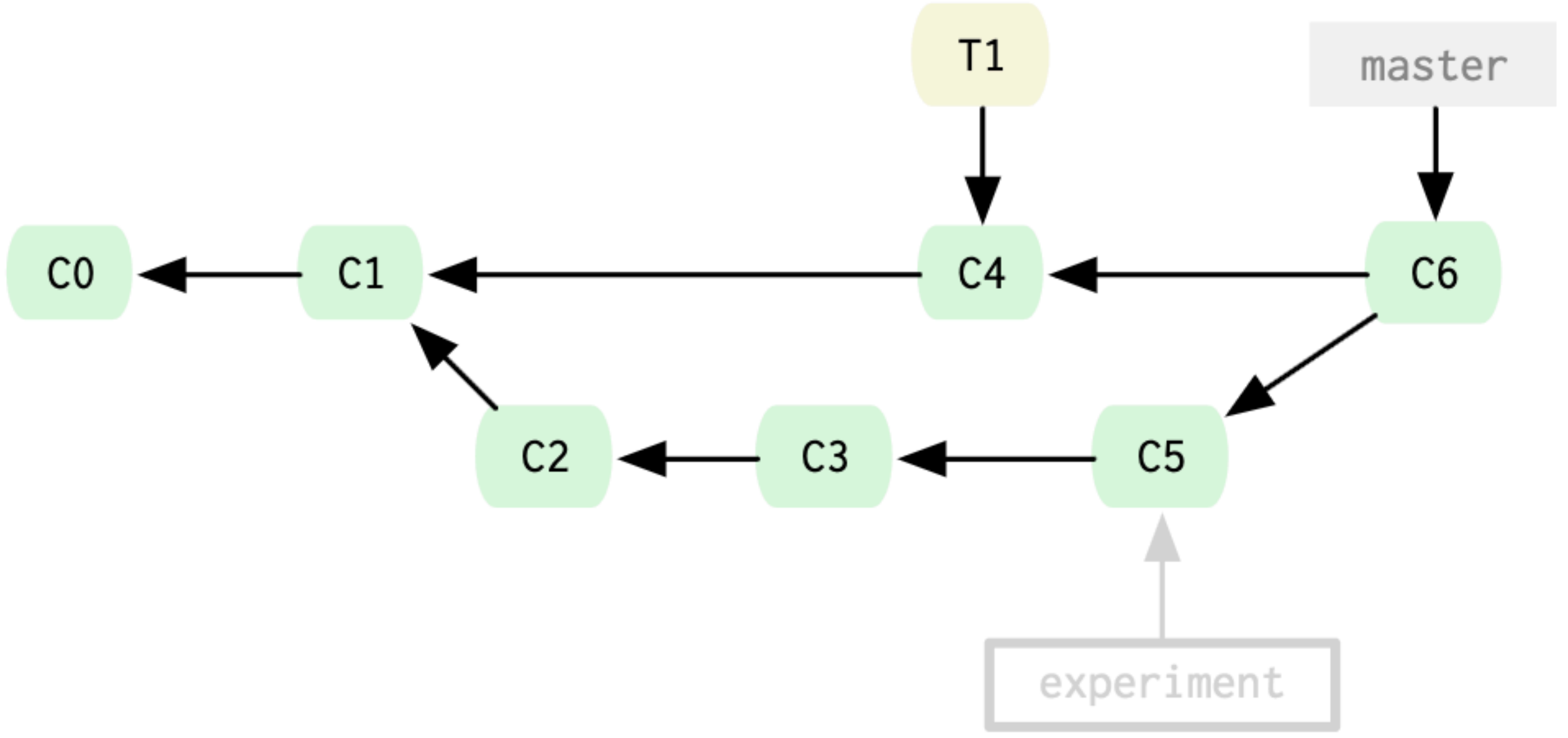
GIT

A bit more advanced operations

- branching: how to create a new branch?
  - `git checkout -b bname`: creates a new `ref` with name `bname`, unnamed commits is hard to go back
- revet changes: if mistakenly changed something, how to restore?
  - `git checkout/restore`
- revert changes: what if the unwanted changes have been committed?
  - `git checkout`: you can always go back to any snapshot (e.g., `HEAD^`, the previous commit)

GIT

A bit more advanced operations

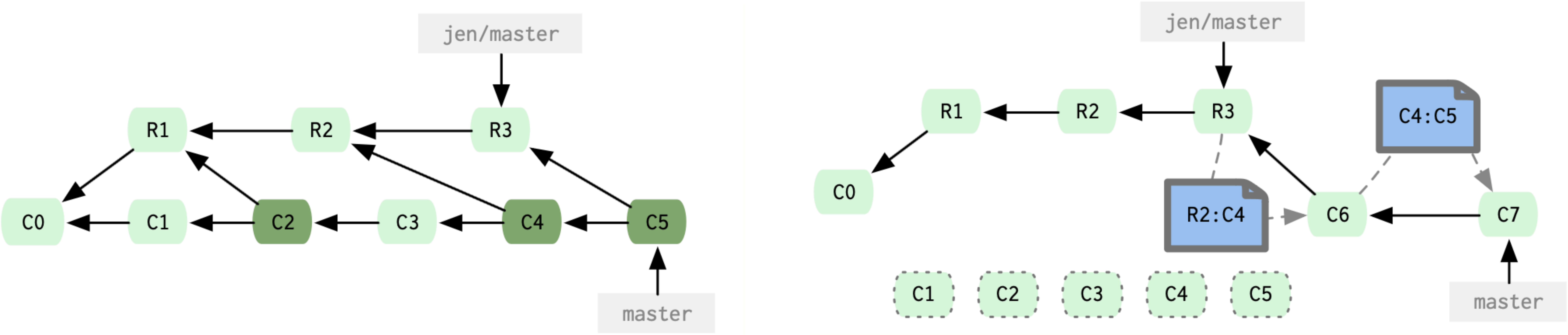


- merge: combine two snapshots
  - `git merge ref`: `ref` could be another (remote) branch, or just a commit
  - nothing changed? cool, it’s just a fast forward (moving the `HEAD` ref to point to the same commit)
  - CONFLICTS? git will try to automatically resolve as many conflicts as possible, the remaining ones have to be manually resolved

GIT

A bit more advanced operations

- `rebase`: combine changes, simplify the version history



GIT

Access to remote branches

- `fetch`: retrieve the new items (blobs, trees, refs, etc) from remote
- `pull`: fetch + merge
- `push`: store new items to remote

GITHUB

- A (mostly) free service to remotely store your git-managed projects
- And other features for project management
  - ISSUES: tracking problems and discuss solutions
  - WIKI: documentation
  - PULL REQUESTS: better managed merging
  - PROJECTS: planning and tracking
  - ACTIONS: automations, CI/CD