Introduction

Amir Masoumzadeh

CSI 424/524

Aug. 26, 2025

UNIVERSITY AT ALBANY
State University of New York

Welcome to Computer Security!

Prof. Amir Masoumzadeh (amasoumzadeh@albany.edu)

- Office Hours: Tuesday/Thursday 4:30pm–5:30pm, CS 209 (on Podium), or by appointment

TA Sree Ashrit Dande (sdande@albany.edu)

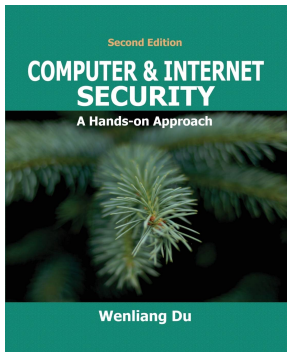
- Office Hours: Wednesday, ? (on Podium), or by appointment

Learning modules

1. Security basics: threats, principles, policies, and cryptography
2. Software security: OS and code level
3. Web security: threats specific to web applications
4. Network security: securing network protocols

Course structure

- Tools
  - Course page: for syllabus and schedule
  - GitHub: for lectures, assignments, and assignment questions
  - Brightspace: for in-class exercises and grades
- Textbook & Readings



  - Additional readings (links on the schedule)
  - Read assigned chapters/readings before each lecture
- Textbook (end-of-chapter) exercises
  - Posted in `book-exercises` repository
  - Review after each lecture
  - Discuss at beginning of following lecture

Course structure (cont.)

- About 12 Lab assignments
- Midterm exam (tentative date listed on the schedule)
- Final exam (date listed on the schedule)
- Project
  - Preferably in teams of up to three
  - Required for CSI 524
  - Optional for CSI 424 (worth 10% bonus)
  - Instructions will be posted in a few weeks

Lab environment

- Windows/Linux/Intel-Mac: Use a prepared virtual machine (VM) that has all required tools/configurations
  - Using VirtualBox (free)
- Apple Silicon Mac: Build an ARM64 virtual machine and deploy the lab tools/configurations
  - Using VMware Fusion (free for personal use)
- Setup instructions in `lab01` and `guides` repositories
  - Note: All your lab reports must show your personal prompt as instructed in `lab01`
  - Re-deploy a fresh VM if your setup become corrupted, but make sure you set it up again according to `lab01`
- Other options (with caveats)
  - Linux OS other than the one provided (you will need to configure it yourself)
  - Deploy the VM on a cloud provider (costs money)
  - Your primary Unix-like OS (risking corrupting your primary OS)

Lab assignments

- Released typically a week in advance of the deadline
- Start early. The least you can do is taking a look at it on day 1
- Ask questions
  - Create issues in **your assignment repository** on GitHub or meet us)
- Discuss freely with your classmates
- Submit individually based on your work on your own VM setup
- We collect your repository from GitHub at the deadline
- Grade will be recorded on Brightspace and announced
  - You have 5 business days to create an issue in your assignment repository if there is any issue with your grade
  - No re-grading after this 5-day period

Lecture activities and participation

- We will often demonstrate concepts using sample codes
  - I encourage you to have your VM ready during class and try them yourself too
- There will be in-class exercises (can be submitted only in class)

Policy highlights

- No tolerance for academic dishonesty
- No late submission (generally)
  - Passed 11:59pm on the day of the deadline and you will receive no point
  - All sort of software and hardware problems may occur
  - Plan ahead and do not leave it to last minute
- Can ask for one-time late lab submission
  - If granted, you can submit 3 days late
- Lowest lab grade will be dropped
- Up to 10% of in-class exercises will be dropped
- 5-day period to post any issue regarding a grade

How to succeed in this class

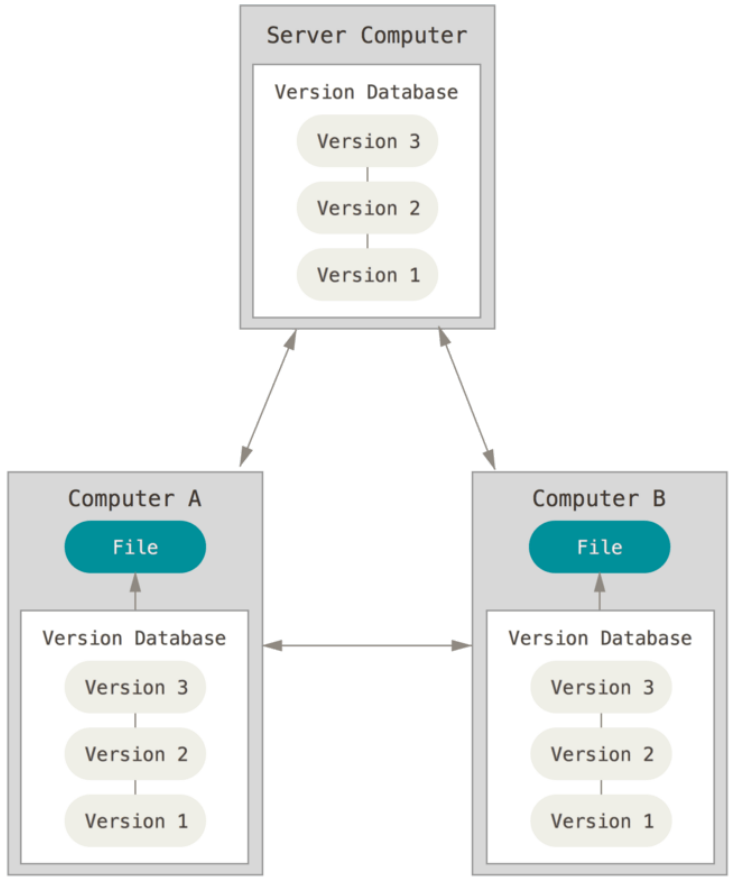
- Read ahead and exercise!
  - Book chapters are easy to read
  - Practice what you read as you read it (or during/after class)
  - Read the freaking manual!

credit: xkcd.com

- Start your assignments early
- Get help from us!
  - Office hours, GitHub, email, ...

Git and GitHub

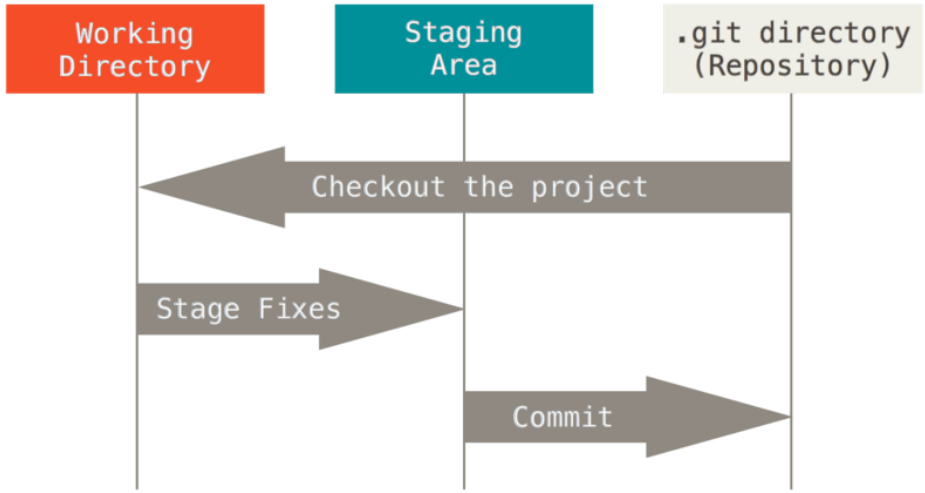
- Git is a distributed version control system
- GitHub is a cloud solution for storing git repositories that acts as one of the distributed nodes



Git states

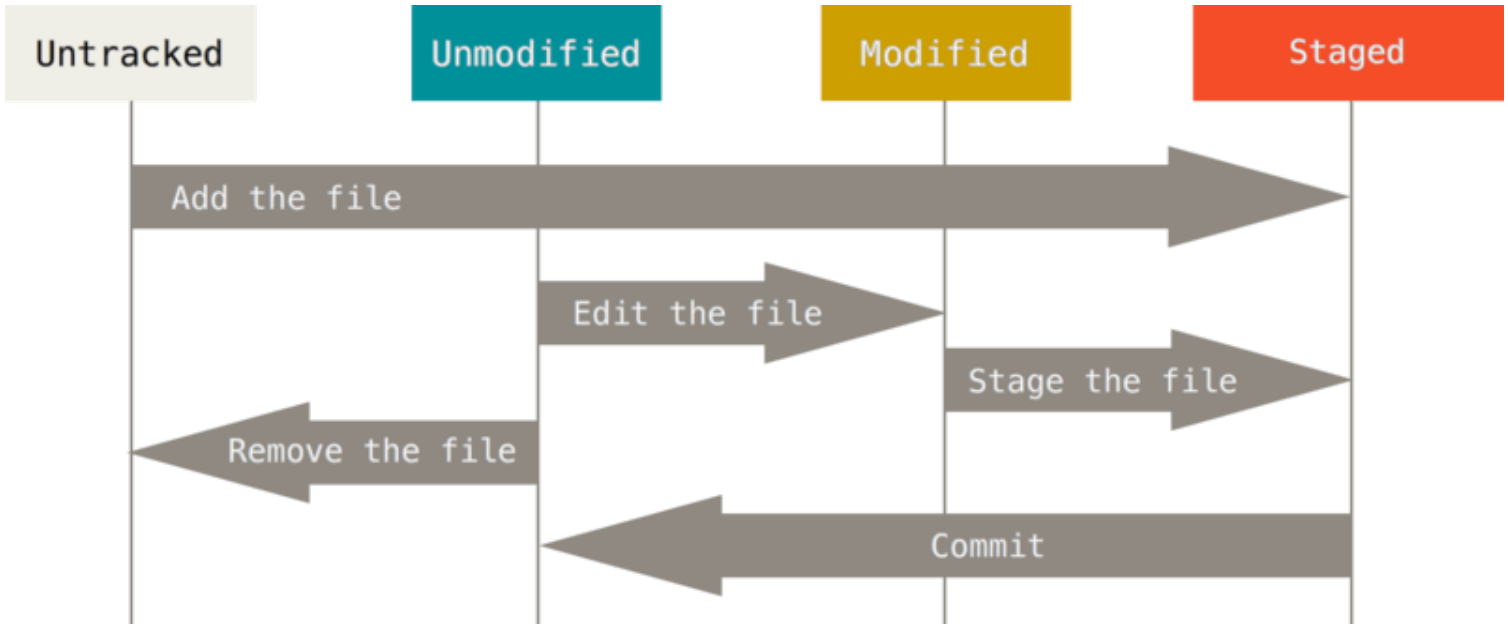
On your local machine files can be:

- In your local repo (committed)
- Checked out and modified, but not yet committed (working copy)
- In-between, in a "staging" area
  - Staged files are ready to be committed
  - A commit saves a snapshot of all staged state



Basic Git life cycle

- Modify files in your working tree
- Stage files (selectively), adding snapshots of them to your staging area
  - Also, track files in your working directory
- Commit, which takes staged files and stores that snapshot permanently to your Git directory



Basic Git life cycle: commands

Note: These are git sub-commands

- `init`: create an empty Git repo
- `status`: show working tree status
- `add`: stage a file
- `commit`: record staged changes to repo

Other commands you may use less frequently:

- `log`: show commit logs
- `diff`: show changes between commits, commit & working tree, etc.
- `rm`: remove files from working tree and index
- `checkout`: switch branches or restore working tree files

Git: working with remote repositories

- `clone`: clone a repo (create a local duplicate)
- `remote`: manage set of tracked repos
- `fetch`: download objects from remote repo
- `pull`: fetch and merge with local repo
- `push`: update remote repo

Using Git for lab assignments

- On release date of lab
  1. There will be a repository available with that lab name (e.g., `lab01`) on GitHub
  2. Follow the link in repository. It will create a private repo will for you for that specific homework: e.g., `lab01-username`
- For working on your homework, you
  1. Clone your homework repo (`lab01-username` , not `lab01`)
  2. Create/edit assignment files
  3. Add changes
  4. Commit
  5. Push (and go to 2 if needed)
- At the deadline, we
  1. Clone your repo to our machine
  2. Grade it