Working on the CS department hosts

Key takeaways

1. No root needed: your dotfiles' user-space half gives you a full environment.
2. One NFS home follows you, but a tmux session stays on its original host.
3. Mind the 15 GB quota: run /lusr/bin/chkquota early; ~/.cache is the usual hog.

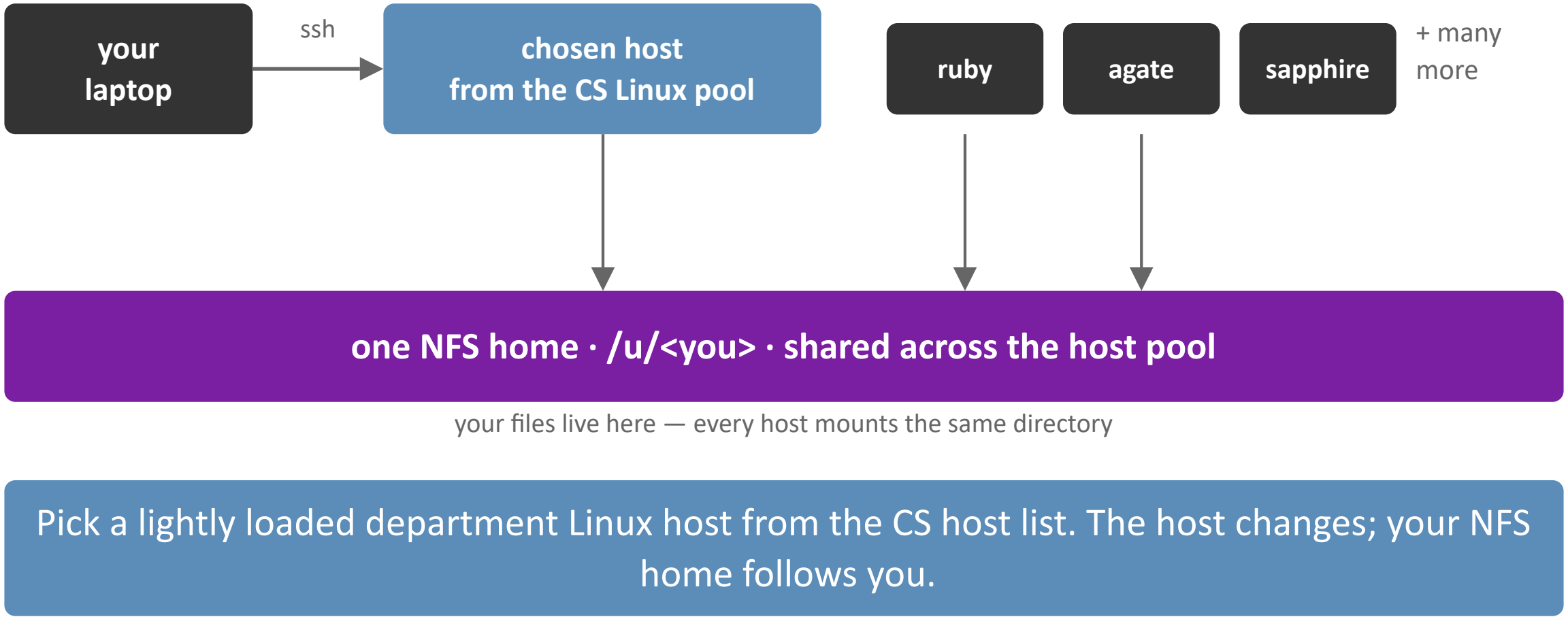
Work on a machine you do not own

Shared Linux machines for development, testing, and performance work

- Use department Linux hosts when your laptop is not ready, when you need a stable Linux shell, or when you want to test away from your own machine.
- You have no root there — and you don’t need it. The user-space half of your dotfiles does the work.
- This deck: choose a host from the pool, clone with gh, run setup, watch your disk quota, and keep track of which host owns your tmux session.

The lay of the land

Many Linux hosts, one shared home directory



Pick a lightly loaded department Linux host from the CS host list. The host changes; your NFS home follows you.

Pick from the pool, not one host

Use the status list or cshosts; do not pile the class onto one machine

Host class	How to choose	Examples from the public status list
Public Linux workstations	status page: up, low users, low load	agate, alexandrite, amber, aquamarine
More public workstations	same home directory; different CPU/load	beryl, calcite, diamond, emerald
More public workstations	record the exact hostname you used	garnet, jade, ruby, sapphire

```
ssh <csid>@agate.cs.utexas.edu      # one example, not the default
hostname -f                         # record where your session lives
cshosts help                        # host-list command on CS machines
```

Sources: UTCS Public Computer Labs page and Public UNIX Host Status page.

What’s already here vs. what you add

The host carries the heavy tools; setup adds a few small ones, no root

Already on the host — Ubuntu 24.04

- git, gh — clone + auth, ready to go
- tmux — sessions that outlive your login
- node — runs Claude Code
- rg, jq, python3, emacs, vim

You add — no root, into ~/.local

- uv — Python tooling
- fd, fzf — file + fuzzy search
- age, sops — secrets
- Claude Code — via the host’s node

You can’t sudo — but git, gh, tmux, node, and rg are commonly available. setup adds course-specific user-space binaries into ~/.local.

First mile — clone your repo, then run setup

git and gh are already installed, so this is short

```
# one-time: authenticate gh (opens a device-code login)
gh auth login

# clone YOUR private repo into ~/dotfiles
gh repo clone utac-f26/dotfiles-<github-login> ~/dotfiles

# install the user-space toolchain + link your config
cd ~/dotfiles && ./setup
```

- Replace <github-login> with your GitHub username. If a chosen host is missing gh, pick another department Linux host or use the README's HTTPS + Personal Access Token clone.
- setup installs into ~/.local (no root), generates your age key, and writes .setup-receipt.json when it finishes.

Disk quota — the real constraint

Undergrad home quota is 15 GB, and it cannot be raised

15 GB	20 GB	✕
undergrad (under)	**grad**	**cannot be increased**

- Where it goes: ~/.cache (uv's cache) and Claude Code's node install are the big two; browser cache, .vscode-server, and mail add up.

```
/lusr/bin/chkquota                    # your limit and % used
du -ahd1 ~ | sort -h                  # find the biggest directories
uv cache prune                        # reclaim uv's cache
rm -rf ~/.cache ~/.vscode-server      # safe — they get recreated
```

source: cs.utexas.edu/facilities/policies/disk-usage

tmux across the lab

Your files follow you; a tmux session stays on the one host you started it on

Your files — one NFS home

ruby ✓ agate ✓ sapphire ✓

the same /u/<you> on every host

your files follow you everywhere

Your tmux session — one host

start it on ruby:
attach from ruby ✓ from agate ✕

come back to the SAME host

Reconnect to the same host where you started the tmux session. remote-run pins jobs to one ‘worker’ session for exactly this reason.

Common gotchas

The four that trip people up first

- setup can’t authenticate
  - Run gh auth login BEFORE ./setup — the clone and setup both need GitHub access.
- New tools ‘command not found’ after setup
  - links wrote them into ~/.local/bin and your shell rc — open a new shell or `source ~/.zshrc`.
- ‘Disk quota exceeded’ mid-install
  - /lusr/bin/chkquota, clear ~/.cache, then re-run ./setup — it’s idempotent and picks up where it left off.
- Your tmux session vanished
  - You logged into a different host. Check your notes, reconnect to the host where tmux started, then tmux attach.

Key takeaways

1. No root, no problem: the user-space half of your dotfiles gives you your whole environment on a machine you don’t own.
2. One NFS home across the host pool — your files follow you, but a tmux session stays on the host you started it on.
3. Mind the 15 GB: run /lusr/bin/chkquota early, and ~/.cache is the usual hog. The quota cannot be raised.