Turbo boost your productivity on the cluster with **tmux**

Chris Want
cjwant@ualberta.ca

2022-11-23
WestDRI Webinar

Slides: https://tinyurl.com/westdri-tmux-2022
Video: link coming soon …

Digital Research Alliance of Canada | Alliance de recherche numérique du Canada

Outline (sort of…)

- Motivation / Problems
- What is **tmux**?
- How do I do things in **tmux**?
- … specifically, how do I make the most of **tmux** on the cluster?

(Those last two things are mushed together.)

Motivation / Problems

- I have an **SSH connection to the cluster that keeps dropping**. Each time it drops, I lose all of my work (including any interactive jobs that might be running).
- I connected to the cluster and am doing some work. I realize now that I need to run a command, but I already have another command running and I don't want to disturb it. **I think I'm going to have to SSH back into the cluster in a separate terminal** … but do I need to do this? A concrete example: running a monitoring command for a job (either batch or interactive).
- I am running something on the cluster, but my bus is here. I don't want to close my laptop and lose my work, but I really have to leave … . Concrete example: building software
- I want to share my working session with two computers.

tmux provides solutions to all of the above problems.

What is tmux?

tmux stands for “Terminal multiplexer”. You can think of a multiplexer as something that allows you to select multiple inputs to send information to a single output.

In this case, the inputs are (bash) sessions. The output is your terminal. You select what you want to see.

In concrete terms:

- tmux allows you to run multiple sessions in a single terminal
- tmux also allows you to detach and reconnect to sessions.

Note that tmux is installed on all of the Digital Research Alliance of Canada (**“the Alliance”**, nee Compute Canada) clusters.

Three important concepts …

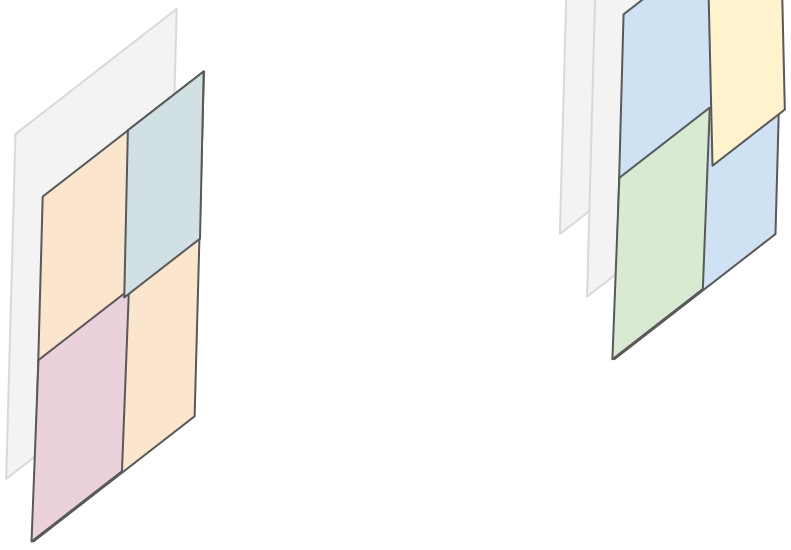
Sessions: these are running instances of tmux

Windows: these are one or more sets of windows in a running instance of tmux

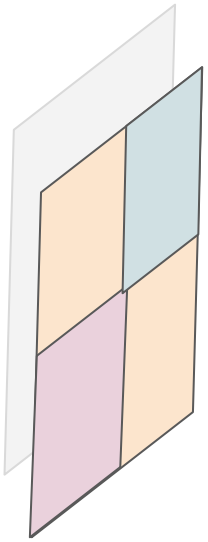
Panes: these are panels inside any particular window

Sessions have one or more windows

Windows have one or more panes



tmux session basics



Start: **tmux**

Exit: **exit** (ctrl-d)

List sessions: **tmux ls**

NOTE: all tmux commands start with the prefix **Ctrl-b**

Possibly the most important command for new users:

Help! **Ctrl-b ?**

Detach! **Ctrl-b d**

Reattach! **tmux attach** (useful when only one session)

Demo: unrequested detach!

First, an important note about login nodes!
Pay attention to which login node you connect to! To restore your session, you must get this same login node when reconnecting!

I'm building software, but my session has become detached because of the network, or because I needed to shutdown my laptop. How do I get back to work?

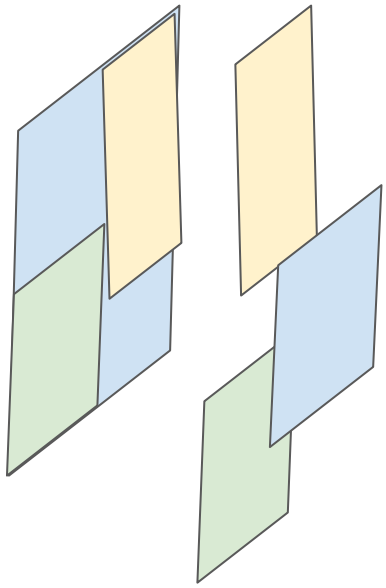
E.g., Suppose I decided it was a good idea to rebuild python:

```
eb --force Python-3.10.2-GCCcore-9.3.0.eb
```

This will take hours, which is not ideal if the network isn't stable.

tmux acts as an insurance policy.

tmux pane basics



Panes in **tmux** are essentially sub-windows that are tiled to fit on the terminal.

- Spliting your current window:
  - **Ctrl-b %** (vertical split)
  - **Ctrl-b "** (horizontal split)

- Switching between panes:
  - **Ctrl-b [Arrow keys]**
    (one, or more if you're quick)

- Resize: **Ctrl-b Ctrl-[Arrow keys]**

- Maximizing a pane: **Ctrl-b z** (toggle on/off)

- Killing the current pane: **exit** (or **Ctrl-d**)

Demo: monitoring a job

Goal: start a job then use some panes to attach to the job and monitor it

Attaching to a running job can be tricky, particularly when GPUs are involved and we attach more than one instance.

"*Breaking news: Slurm is picky and subtle"* – Mark Hahn

We submit a job with **sbatch** and get the job id (**JOB_ID** below) with **sq/squeue**

e.g. First attachment in one pane

srun --jobid JOB_ID --pty bash

Second attachment (no access to GPU) in another pane:

srun --jobid JOB_ID --overlap --gpus=0 --pty bash

Now we can monitor the jobs in the panes using for example:

- **htop**
- **watch -n 3 nvidia-smi**

More on sessions (multiple sessions)

By default, you can't start **tmux** inside of **tmux** … but you can detach and create another tmux session if you want.

Detach: **Ctrl-b d**

Start a new session: **tmux**

List the sessions: **tmux ls**

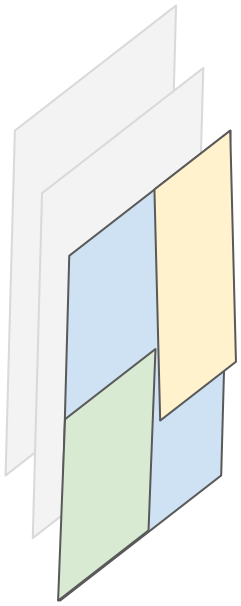
Notice that each session has a numerical identifier that can help you attach to it, e.g., **tmux attach -t 0**

But what about an identifier that is a bit more semantic?

Renaming the session: **Ctrl-B $** (enter new name, e.g. "Batch job")

Then to attach: **tmux attach -b "Batch job"**

Windows



Sometimes you might want to organize collections of panes.

Windows can help with that.

Create a new window: **Ctrl-b c**

Go to next window: **Ctrl-b n**

Go to previous window: **Ctrl-b p**

Rename a window: **Ctrl-b ,**

Browse all windows (in all sessions): **Ctrl-b w**

Demo: Interactive job

Interactive jobs are great for debugging a job that might be giving issues. It allows you to do experiments in an environment that matches your batch job. Once you have things working, you can submit your fixed program as a batch job.

We use **salloc** instead of **sbatch**, and include the job specifications on the command line, e.g.,

salloc --time=1:00:00 --ntasks=1 --mem-per-cpu=4000M

(Other options for account, number of CPUs, etc. can be included.)

With this setup, we can run the interactive job in one pane (or window), and modify source code in one-or-more others.

Another trick …

You can use tmux to s**hare the same terminal between two different computers** (simulated with different terminal windows). Just connect to the same login node, and attach to the same session. (Assumption: the user is the same on both computers.)

Note that you can (in theory) also share a tmux session among different users on the cluster. The setup is a bit complicated, and this may represent a security concern, so I'm not going to go into it today. (Obvious use case: show the new team member how to do stuff).

Gotchas!

Here are a few gotchas when running tmux:

- Multi pane copy/paste
  - The solution to most problems is usually "maximize pane first"
- Scrolling up in the terminal back buffer
  - Solution: **Ctrl-b [** (then **page up**, **q** to quit)
- On reconnect: "Where's my tmux session?"
  - Check that you reconnected to the same login node!
  - ssh to the correct login node as needed
- Doesn't always play well with SSH agent on reattach.
  - E.g., if you use SSH keys to access GitHub, and these keys are forwarded from your home machine via an SSH agent, your ability to access GitHub may be gone after you reattach.

Where should I run **tmux** when working on the cluster?
(Think of the pros and cons)

a) On my own computer?
b) On the login node of a cluster?
c) On a work node of a cluster as part of a job?

What I didn't talk about in this webinar …

- How to configure tmux
- Mouse support for tmux
- Plugins and extensions to make it look cool
- Many, many other ways of doing things!

Questions?

Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions
Questions Questions Question Questions

Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions
Questions Questions Questions Questions

Questions Questions Questions Questions Questions
Questions Questions Questions Questions Questions
Questions Questions Questions Questions Questions
Questions Questions Questions Questions Questions
Questions Questions Questions Questions Questions

Questions?