CSCI 460 Operating Systems

Follow-ups From Last Class / Review

Professor Travis Peters
Fall 2019

Questions From Last Class…

Discuss in groups; get up and use whiteboards around the room to write on!

Questions From Last Class…

Discuss in groups; get up and use whiteboards around the room to write on!

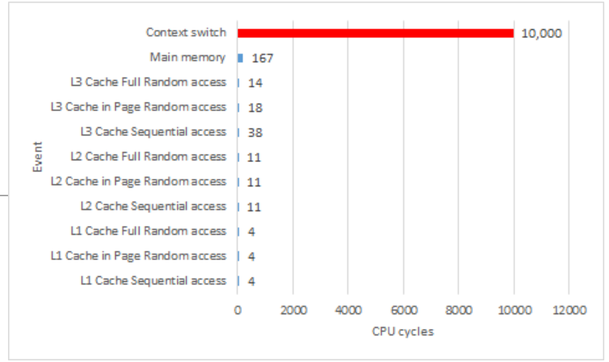
---

1. What are User Space and Kernel Space?

- Why do we have different “spaces” (“User Space” and “Kernel Space”)?
- Why is it expensive to “switch” between them?
- What is the canonical memory layout for User Space and Kernel Space

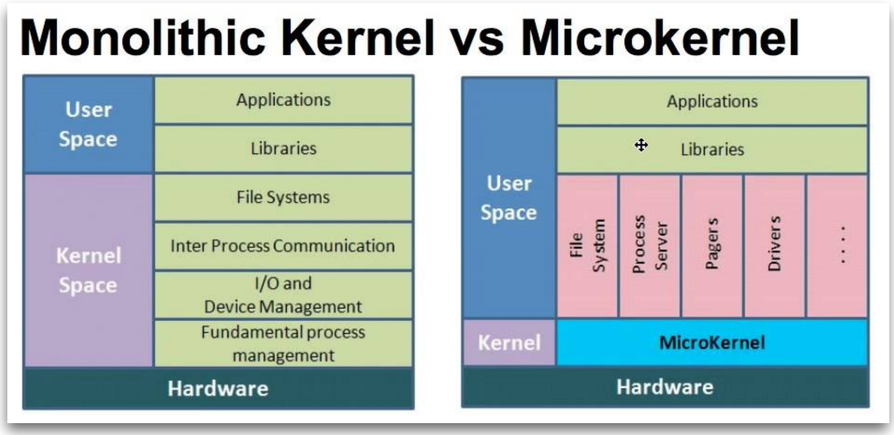
Questions From Last Class…

Discuss in groups; get up and use whiteboards around the room to write on!

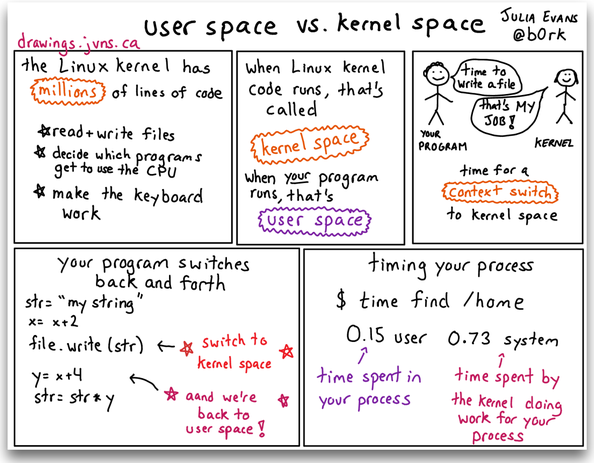


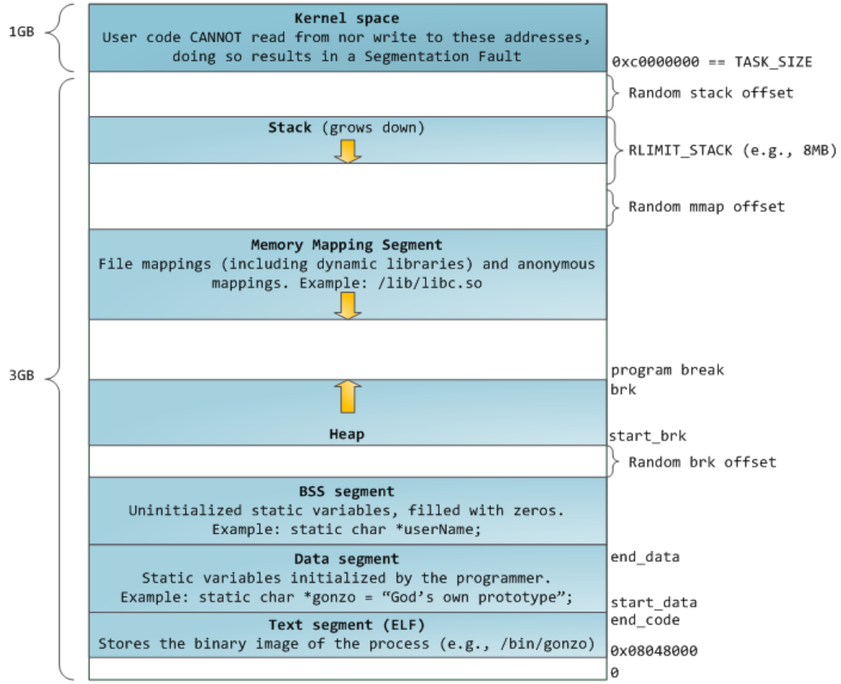
1. What are User Space and Kernel Space?

- Why do we have different “spaces” (“User Space” and “Kernel Space”)?
- Why is it expensive to “switch” between them?
- What is the canonical memory layout for User Space and Kernel Space



—https://i.ytimg.com/vi/4RVSgxrsxvE/maxresdefault.jpg





Questions From Last Class…

Discuss in groups; get up and use whiteboards around the room to write on!

---

2. Heaps (user heap vs. kernel heap)

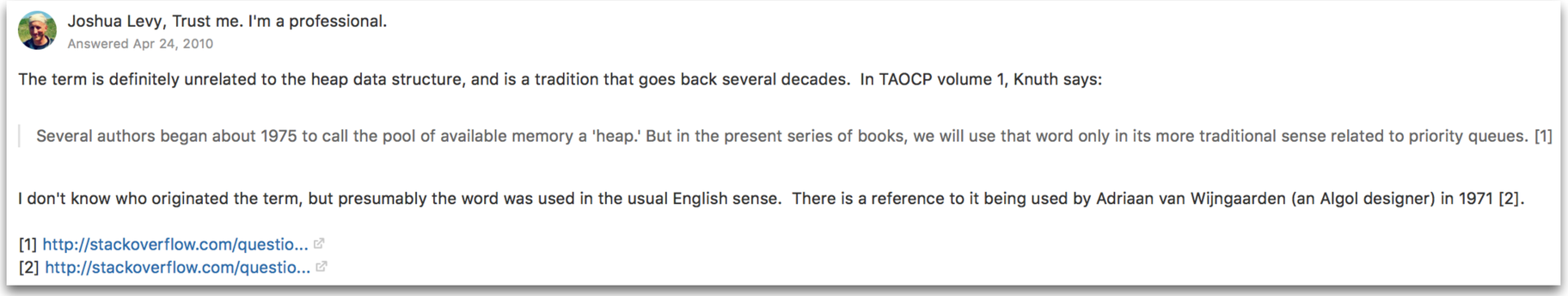
- How are heaps used in OSs?
- For that matter, what is the stack? And how is it used?

Questions From Last Class…

Discuss in groups; get up and use whiteboards around the room to write on!

2. Heaps (user heap vs. kernel heap)

- How are heaps used in OSs?
- For that matter, what is the stack? And how is it used?



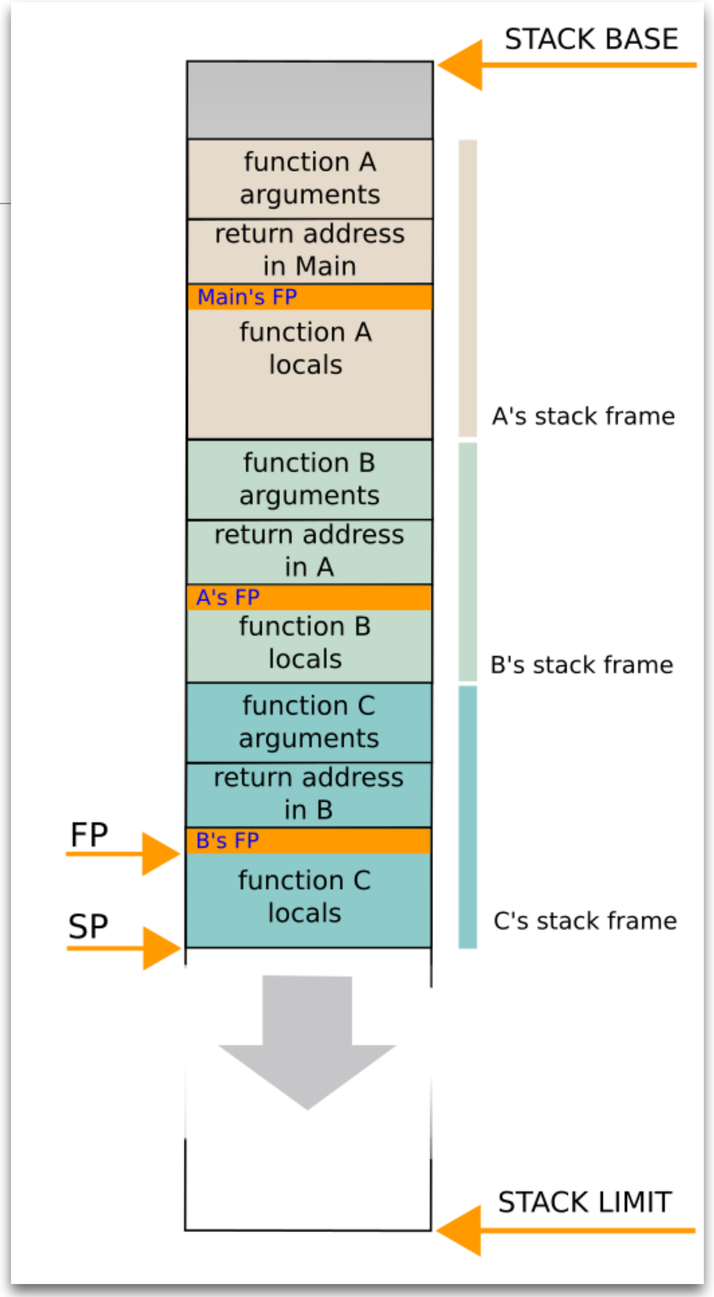
Joshua Levy, Trust me. I'm a professional.
Answered Apr 24, 2010

The term is definitely unrelated to the heap data structure, and is a tradition that goes back several decades. In TAOCP volume 1, Knuth says:

> Several authors began about 1975 to call the pool of available memory a 'heap.' But in the present series of books, we will use that word only in its more traditional sense related to priority queues. [1]

I don't know who originated the term, but presumably the word was used in the usual English sense. There is a reference to it being used by Adriaan van Wijngaarden (an Algol designer) in 1971 [2].

[1] http://stackoverflow.com/questio...
[2] http://stackoverflow.com/questio...



Questions From Last Class…

Discuss in groups; get up and use whiteboards around the room to write on!

---

3. Fork—why would you `fork()`?

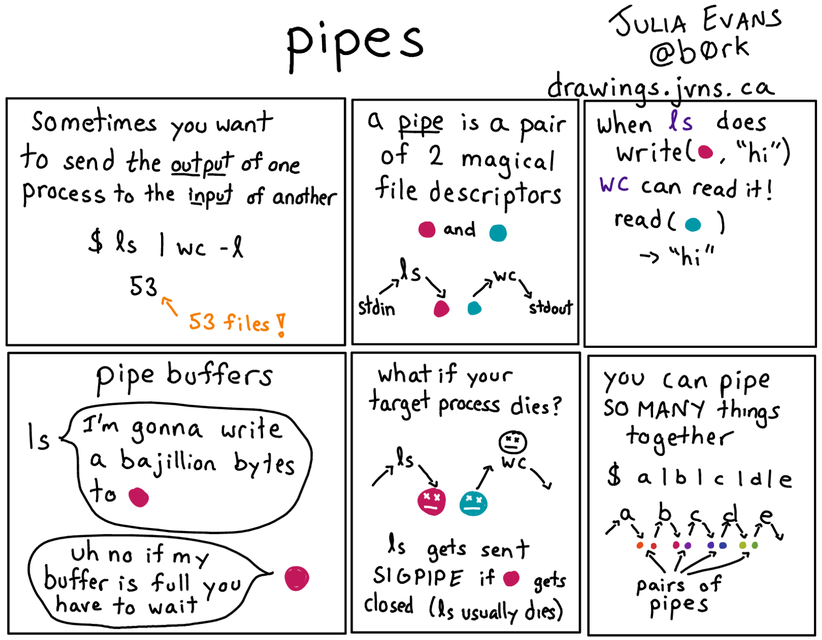
- What does `fork()` do? Why is that useful?
- Pros and cons of `fork()`?
- Alternatives to `fork()`?

Questions From Last Class…

Discuss in groups; get up and use whiteboards around the room to write on!

3. Fork—why would you `fork()`?

- What does `fork()` do? Why is that useful?
- Pros and cons of `fork()`?
- Alternatives to `fork()`?



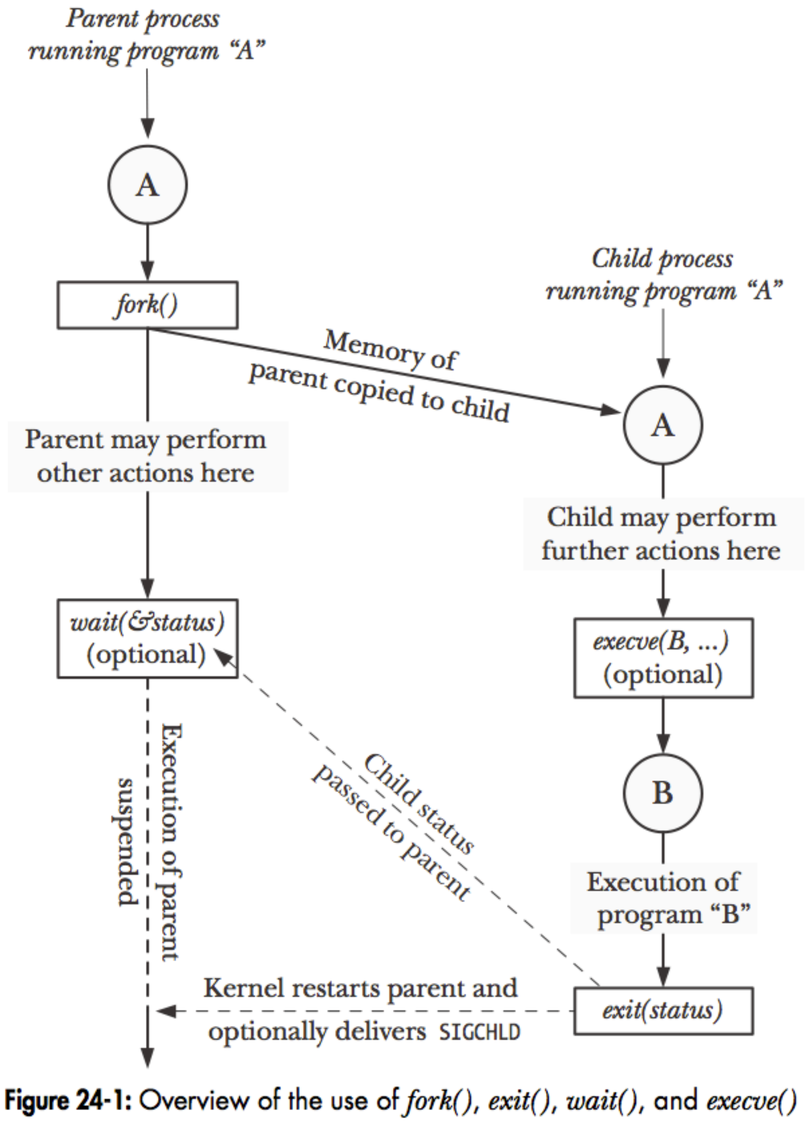
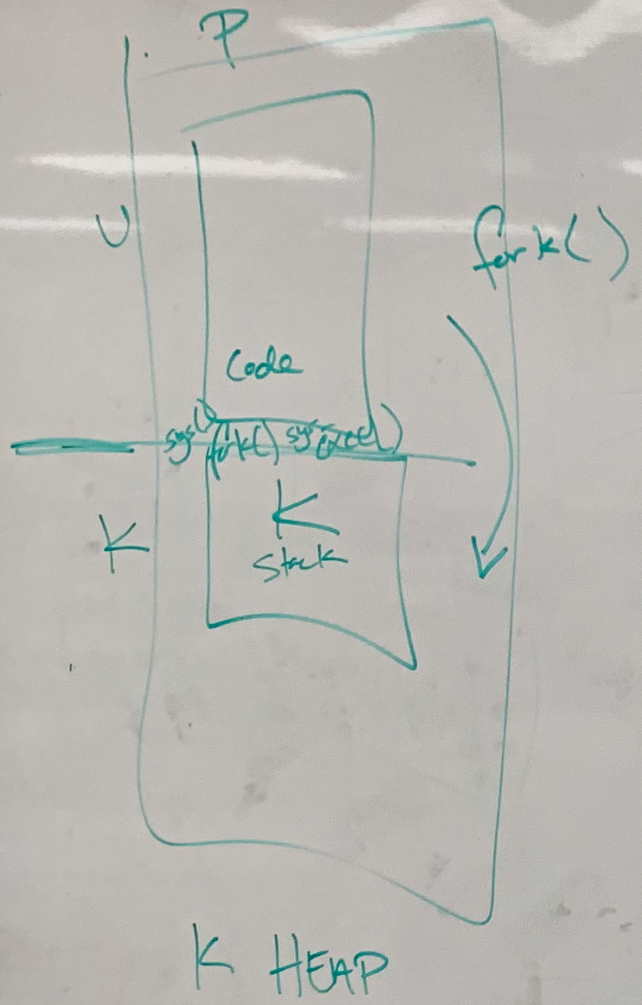


Figure 24-1: Overview of the use of *fork()*, *exit()*, *wait()*, and *execve()*

Photos of whiteboards after class

(Unfortunately, some got erased before I could snap a photo…)



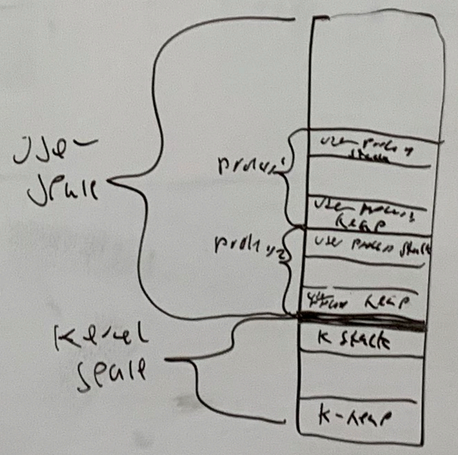
f(a,b,c)

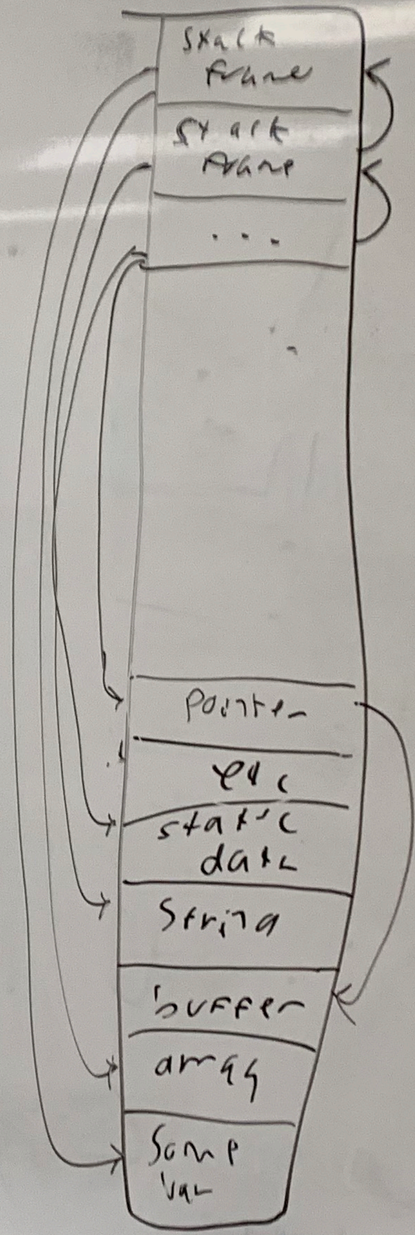
user space
- Restricted access to system resources

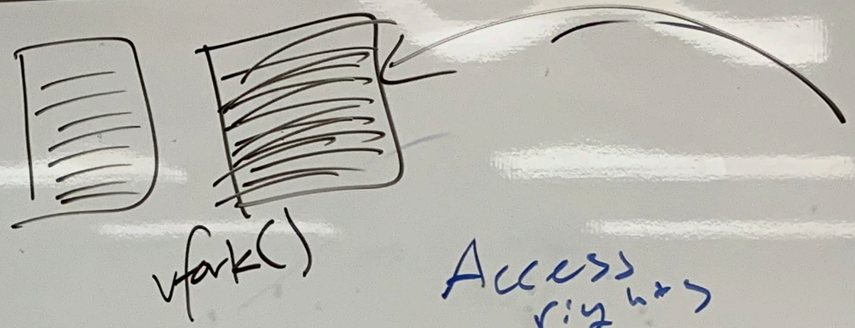
PCB
Proc State
Stack
Kernel Stack
K State

Kernel Space
- Full access to system resources

x addr_func
a
b
c
func() context switches are expensive because
: Main memory operations like
: stack & function args,
: registers & cached buffers,
: are flushed.

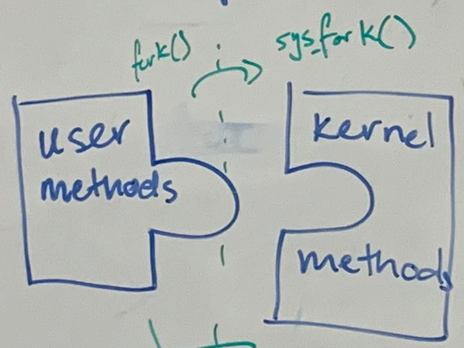


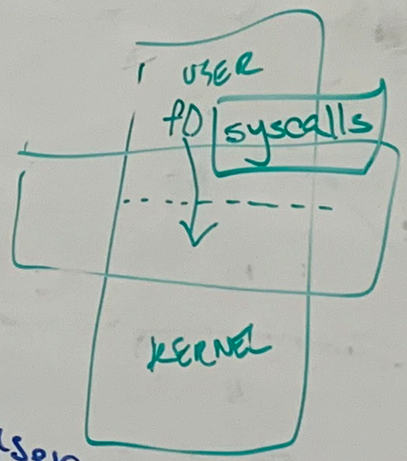


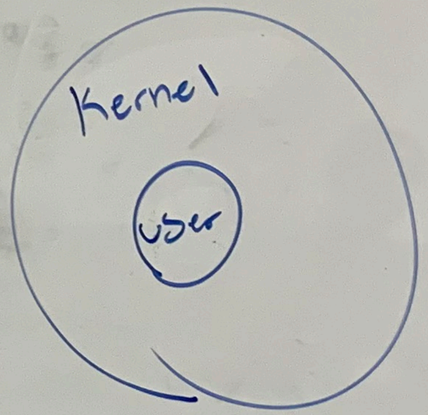


Access rights

- permissions







unprivileged instructions

privileged instructions
eg interact w/device

- User space vs. kernel space

Syscalls →

- Why?
  - for control & security purposes
- Expensive to switch
  - not the same memory space
  - mode change of CPU
  - ensure user space cannot execute/access certain privileged code

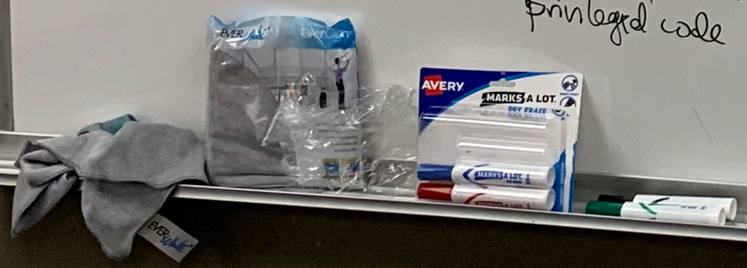
fork() parent: init it=1
- duplicate new process
- check if you're a child (return code)
- id - alternative: exec()
- PCB * security issue! forkbomb
- cons: strange, pro: origin for backtracing

User space

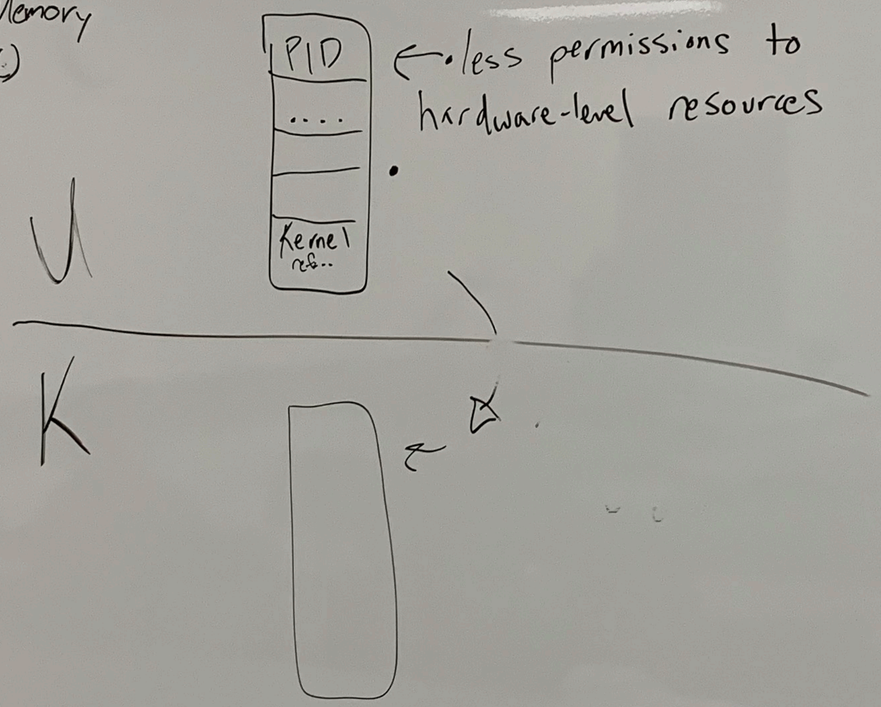
Kernel space

Heap:
- information to store in dynamic memory that you can access in any order

Stack:
- cannot access in any order
- completed calls



Heap
- Allocates Dynamic Memory
  - Malloc() + Calloc()
- User heap
- Kernel heap



init ↖?

p1 ↖

p2