How can we make kernel more reliable?

1. Modularize it so that when one component fails, the entire kernel doesn’t crash.
2. Use programming languages with stronger promises. 
   → rust guarantees memory safety (no data races)

How do we isolate fault for each component? Process is a unit of fault domain, so maybe move each component into their own user process!

→ so far all the OS components we’ve learned runs in kernel mode
→ OS = kernel
   mono- or this kernel
→ all components share the same address space & same failure domain
What are some challenges if we move an OS component into a user process?

1. Availability
   - What if the fs process terminates? No one can use filesys anymore?

2. Communication
   - How do we request service from the fs process? IPC?
   - Lots of IPC = high overhead

3. Access Permission
   - Access to certain HW still requires us to be in the kernel mode!
   - Needs both a kernel part for privileged access & a user process part for policies & designs

Can do this for other OS components as well!
- This approach is called microkernel
- Keeps the kernel small & implement most OS components in user space
Example of Microkernel in use: Google's Snap

Based off of Linux, moves network stack into a userspace process!

https://research.google/pubs/pub48630/