CS 3214: Computer Systems
Lecture 6: File Descriptors + Pipe

Instructor: Huaicheng Li

Sept 8 2022

VIRGINIA TECH™

Announcements

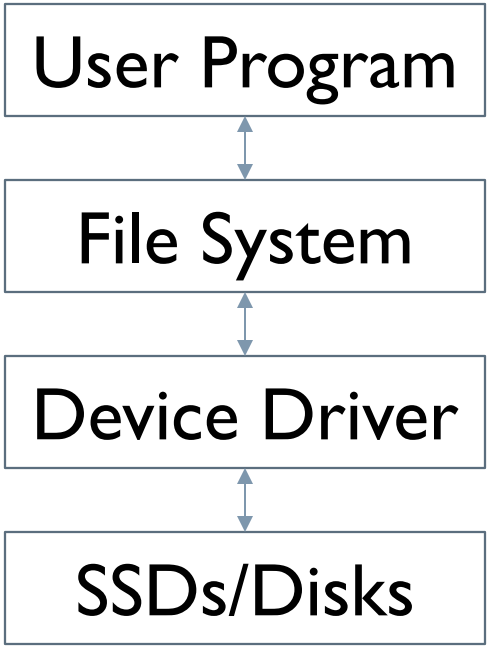
- Project 1 released, deadline: 9/28 11:59pm

Linux/Unix: Everything is a file

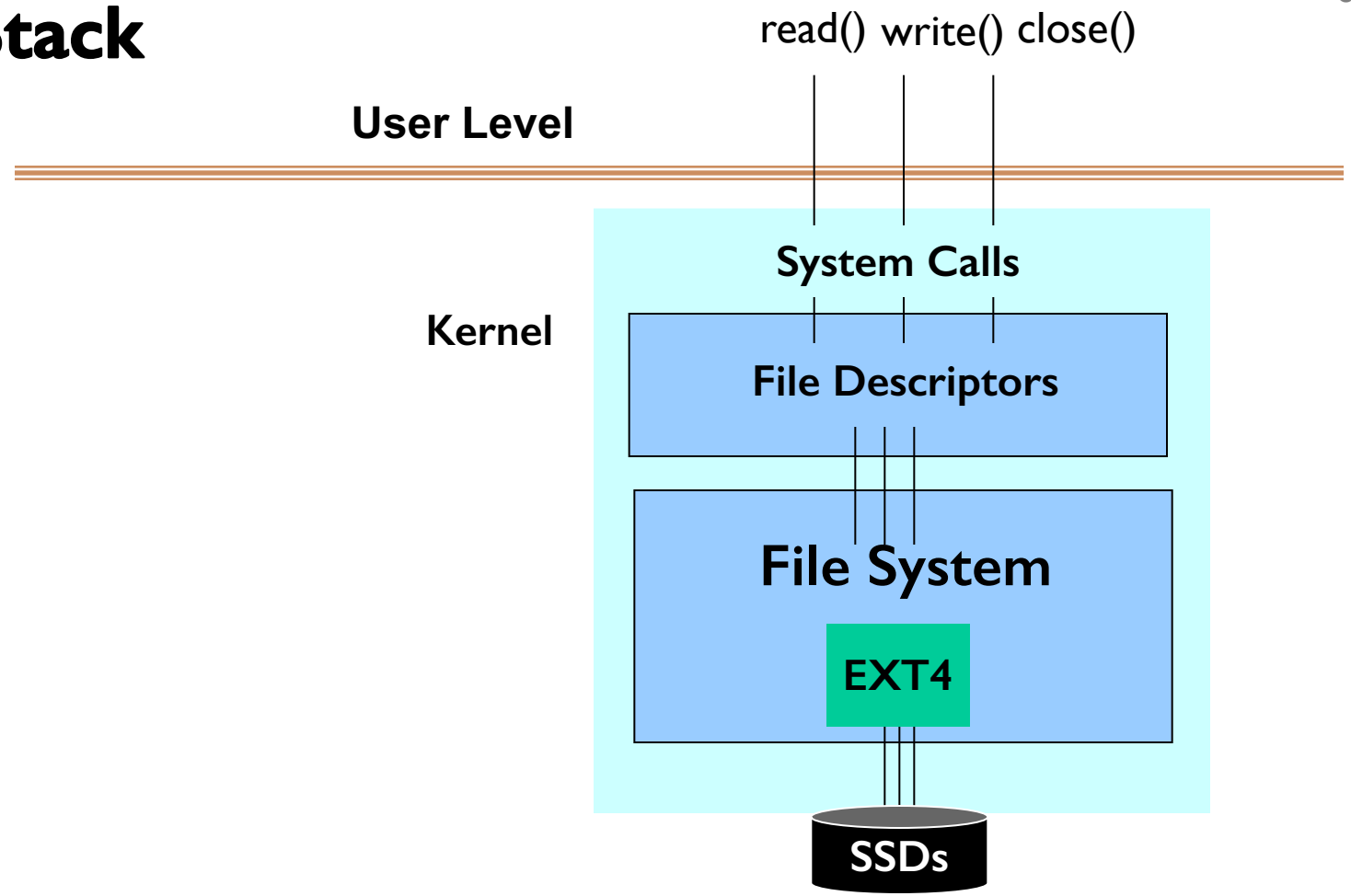
Standard Streams

- C: stdin (0), stdout (1), stderr (2)

- File path
  - Absolute path (e.g., /usr/bin/ls)
  - Relative path (e.g., ./a.out)
- File types
  - Regular
  - Block / character
  - Socket
  - Directory
  - Links
  - …
- File/Storage Stack



Storage Stack

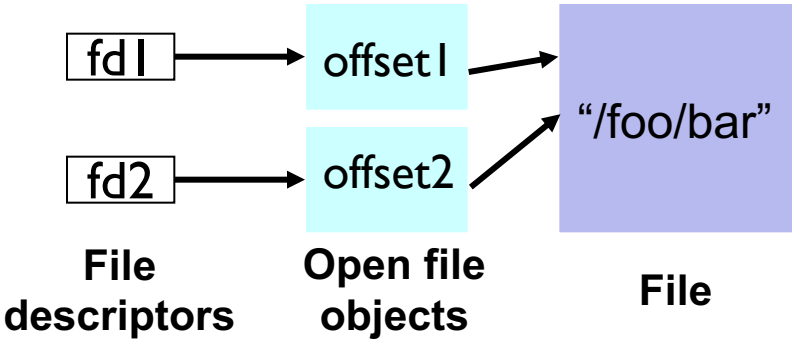


- *int fd = open(const char *path, int oflag, …);*

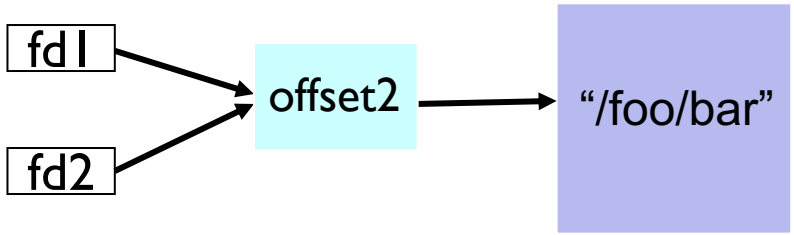
- *ssize_t ret = write(int fd, void *buf, size_t nbyte);*
- *ssize_t ret = read(int fd, void *buf, size_t nbyte);*
- *ssize_t ret = close(int fd);*

Accessing Open Files

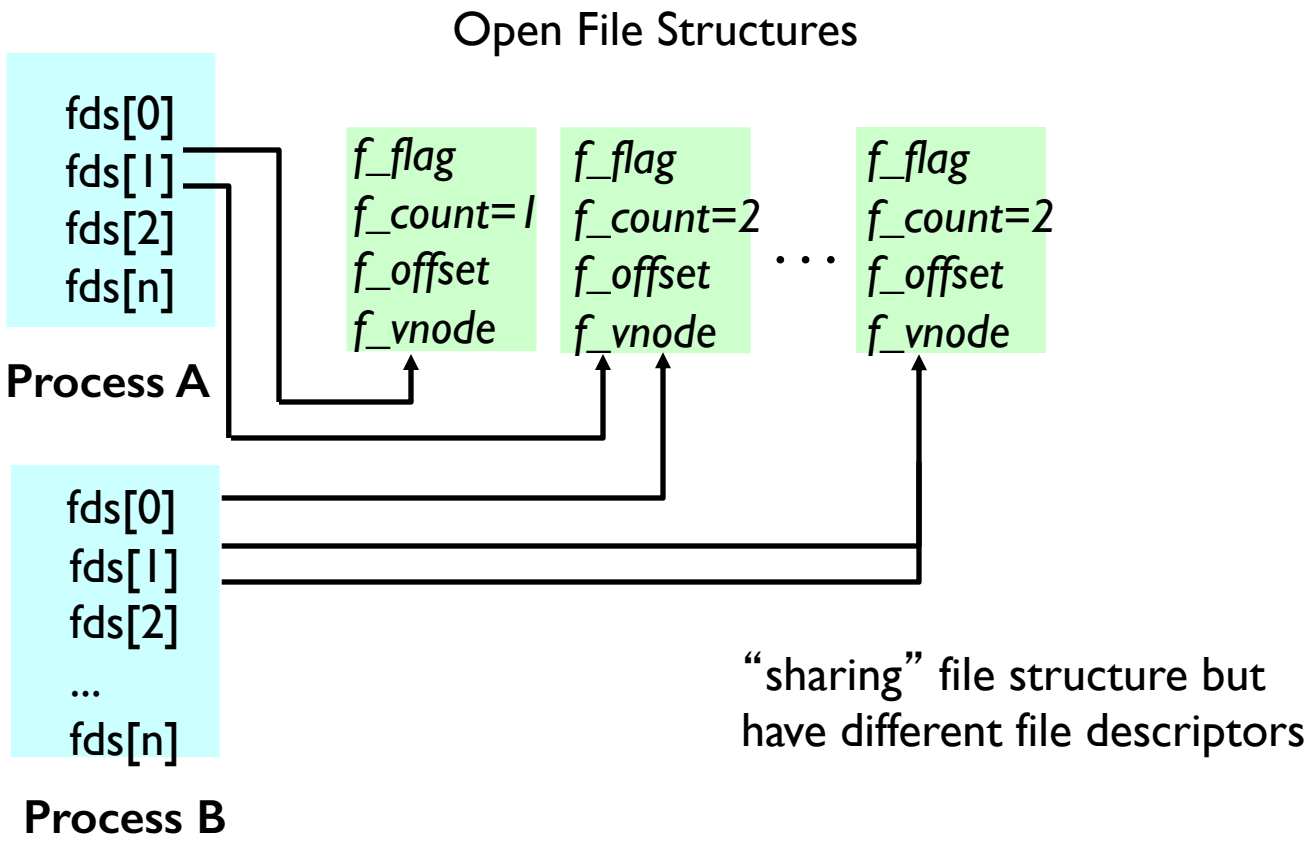
- Two opens of the same file yield independent sessions

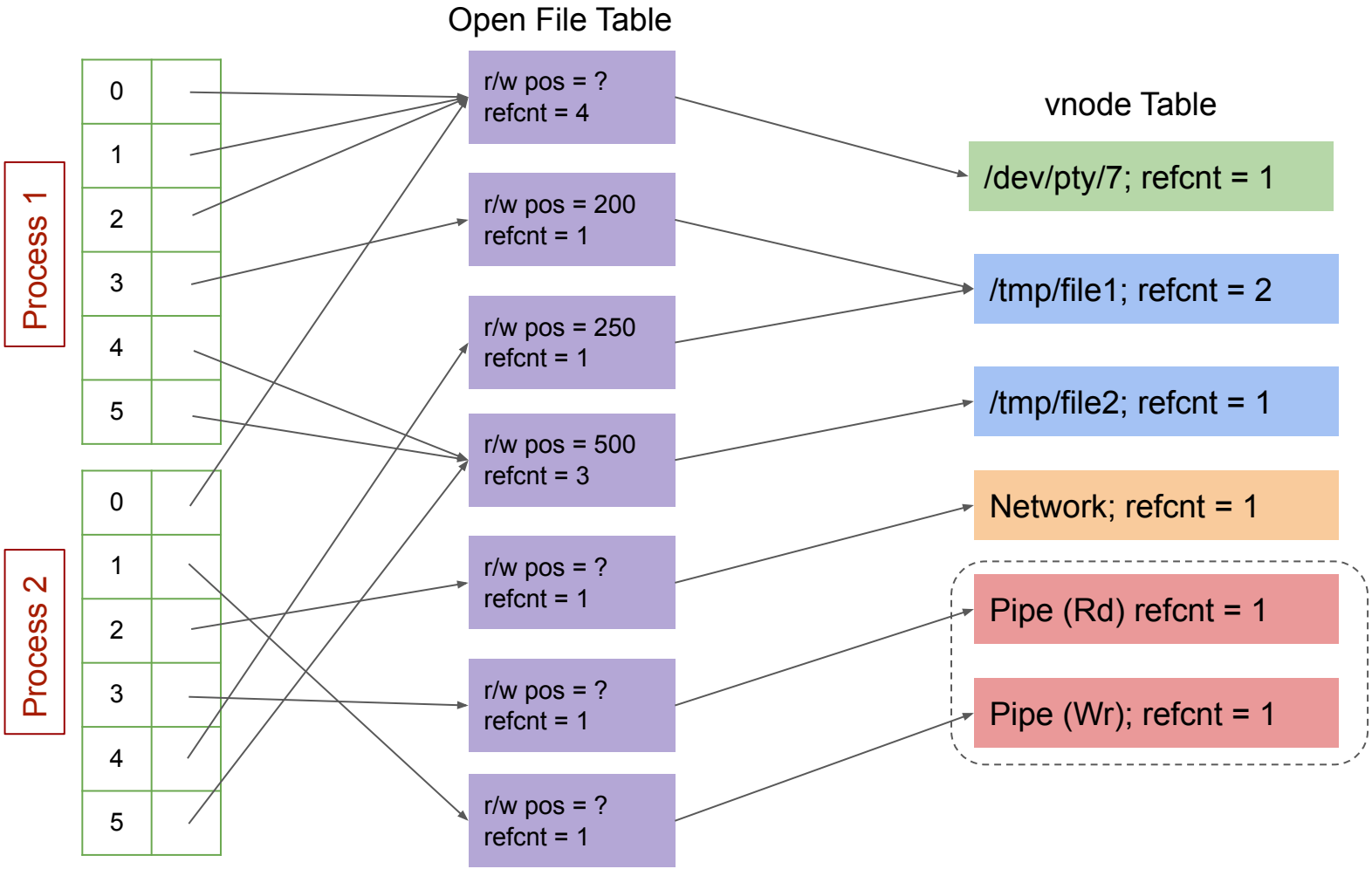


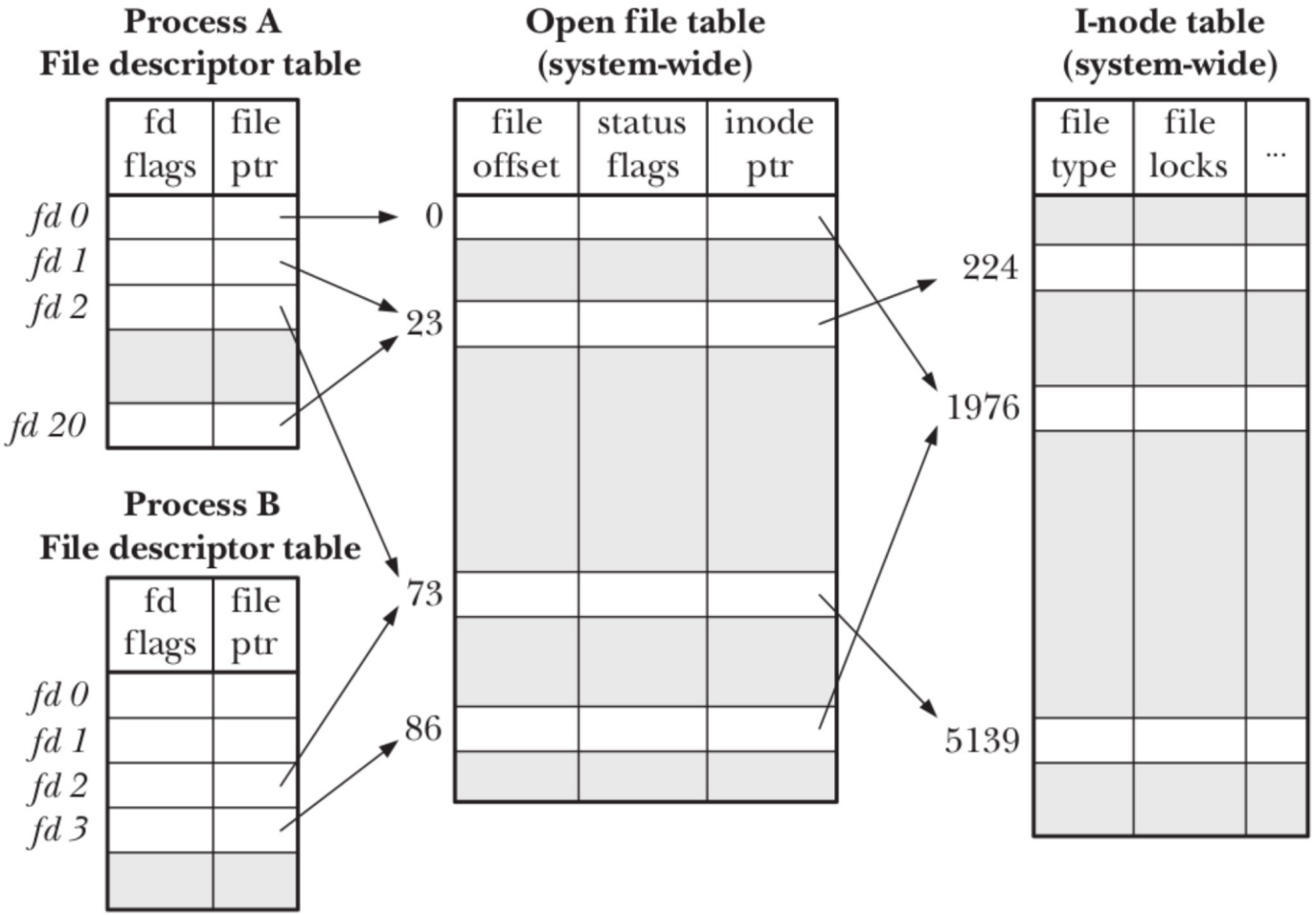
- Two opens of the same file yield independent sessions



Some associated structures in kernel space

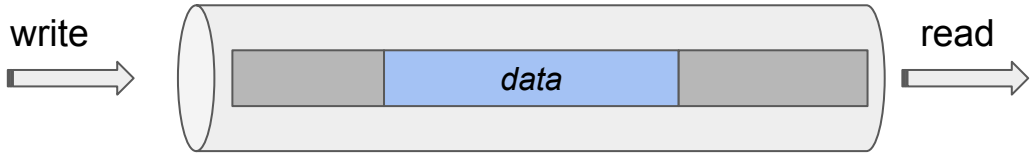



Open File Table
vnode Table
Process 1
Process 2
0
1
2
3
4
5
r/w pos = ?
refcnt = 4
r/w pos = 200
refcnt = 1
r/w pos = 250
refcnt = 1
r/w pos = 500
refcnt = 3
r/w pos = ?
refcnt = 1
r/w pos = ?
refcnt = 1
r/w pos = ?
refcnt = 1
/dev/pty/7; refcnt = 1
/tmp/file1; refcnt = 2
/tmp/file2; refcnt = 1
Network; refcnt = 1
Pipe (Rd) refcnt = 1
Pipe (Wr); refcnt = 1


Process A
File descriptor table
fd flags
file ptr
fd 0
fd 1
fd 2
fd 20
Process B
File descriptor table
fd flags
file ptr
fd 0
fd 1
fd 2
fd 3
Open file table (system-wide)
file offset
status flags
inode ptr
0
23
73
86
I-node table (system-wide)
file type
file locks
...
224
1976
5139

- ***close(fd):***
  - clear entry in file descriptor table, decrement refcount in open file table
  - if zero, deallocate entry in open file table and decrement refcount in vnode table
  - if zero, deallocate entry in vnode table and close underlying object
  - for certain objects (pipes, socket), closing the underlying object has important side effects that occur only if all file descriptors referring to it have been closed
- ***lseek(fd, offset, …)***
- ***dup(int fd):*** create a new file descriptor referring to the same file descriptor as fd, increment refcount
- ***dup2(int fromfd, int tofd):*** if tofd is open, close it. Then, assign tofd to the same open file entry as fromfd, increment refcount
- ***opendir(), closedir(), readdir(), …***
- On fork(), the child inherits a copy of the parent's file descriptor table (and the reference count of each open file table entries is incremented)
  On exit() (or abnormal termination), all entries are closed

Pipes



- Writers:
  - can store data in the pipe as long as there is space
  - blocks if pipe is full until reader drains pipe
- Readers:
  - drains pipe by reading from it
  - if empty, blocks until writer writes data
- Pipes provide a classic "bounded buffer" abstraction that
  - is safe: no race conditions, no shared memory, handled by kernel
  - provides flow control that automatically controls relative progress: e.g., if writer is BLOCKED, but reader is READY, it'll be scheduled. And vice versa.
  - Created unnamed; file descriptor table entry provide for automatic cleanup