CS 537 Handout Nov 3

- Review:
  - RAID 10 performance
  - RAID 4 performance
  - RAID 5 performance
- Abstraction: File
  - Linear array of bytes
  - Low-level name: inode number
  - High-level name: human-readable name
- Abstraction: Directory
  - maps human-readable name of files to low-level name
  - directories can contain other directories: dir tree
  - Root directory + separator to get paths
  - Absolute, relative paths
- Naming is important
  - Everything is a file in Unix
  - Contributed greatly to its success
- Interface
  - Read book, man pages for details!
  - open()
  - File descriptor
  - File offset
  - lseek(): does not do a disk seek! Only changes offset
  - fsync()
  - rename()
  - Metadata
  - Symbolic and hard links
  - Mounting a file system

Research

All File Systems Are Not Created Equal: On the Complexity of Crafting Crash-Consistent Applications. OSDI 2014 [Disclaimer: I worked on this :)]

Questions

1. How is lseek() related to a disk seek? Are they the same?
2. Why do we need fsync()? What could go wrong if we did not have that system call?
3. When you are creating a new file, you need to fsync() the file. Why do you need to fsync() the parent directory?
4. Why are symbolic links needed?
5. Why do need low-level names for files?
6. Why can symbolic links have dangling references but not hard links?
7. What are the different ways in which you can implement a directory?
8. Is a hierarchy the best way of organizing directories? What if you had just a list of files?
9. If process A passed a file descriptor to process B, can B use to read or write from a file?
10. How would you atomically update multiple blocks of a file?