

**Note:** This is just a list of topics and things to think about from the latter half of the course. Keep in mind that you are also responsible for the content in the first half of the course as well.

1. Paging
  - a. Know basics
  - b. I.e. how to get address widths, what the levels are, etc.
  - c. 2 level paging
  - d. Including 2 level page tables and segmenting with paging.
2. Translation Lookaside Buffers
  - a. What is a TLB?
    - i. How fast are they?
    - ii. How big are they?
    - iii. How are entries searched?
    - iv. What exactly does it do?
      1. Be able to list exactly/draw a flow chart of what happens when there is a page fault with a TLB!
  - b. Where a TLB fits into the paging scheme
    - i. Why do we use them?
    - ii. Be able to draw a complete picture of the paging process using the TLB!
  - c. What principle does the TLB exploit in order to work so well with so few entries?
    - i. How does this work?
  - d. What extra overhead does the TLB add to context switches?
3. Page Faults and Demand Paging
  - a. What is the principle of . . .
    - i. Temporal Locality?
    - ii. Spatial Locality?
  - b. Is there something the user could do when writing programs to minimize page faults?
  - c. What is the typical layout of a Page Table Entry (PTE), and what is each bit for?
  - d. What is a core map?
  - e. What size should pages be?
    - i. What are some pros/cons of large pages?
    - ii. Can we page the operating system?
  - f. After a page fault, can a process be restarted directly?
    - i. How many page faults can occur in one instruction?
  - g. What is a . . .
    - i. Page fetch algorithm?
    - ii. Page replacement algorithm?

- iii. Page placement algorithm?
  - h. What is demand paging?
    - i. What support is needed for this?
    - ii. Why is it used?
    - iii. How does it work?
- 4. Page Replacement Algorithms
  - a. Random
    - i. What is this?
    - ii. Why isn't it used?
  - b. FIFO
    - i. How does this work?
    - ii. What are its limitations?
      - 1. What is Belady's Anomaly—i.e. does giving this algorithm more pages always improve its performance?
  - c. LRU
    - i. What is this?
    - ii. What are some limitations in implementing it?
    - iii. How do we approximate it?
  - d. FINUFO/Clock/Nth Chance
    - i. What is this an approximation of?
    - ii. What aspect of PTE layout does this algorithm take advantage of?
    - iii. How does it work?
    - iv. Drawbacks?
    - v. Do we need hardware support to implement this at all?
  - e. MIN/OPT
    - i. What is the exact definition of an optimal page replacement algorithm?
    - ii. Why can't we implement it?
  - f. Be able to draw a flowchart of how paging works with page replacement, TLBs, and any flavor of multi-level paging.
  - g. How do we evaluate paging algorithms?
- 5. Thrashing and Working Sets
  - a. What is thrashing?
  - b. How do we fix this problem?
    - i. There are several solutions!
  - c. What is a working set?
    - i. Why is it useful?
    - ii. When is it useful?
  - d. What is the difference between global and local replacement?
  - e. What is the Working Set Paging algorithm?
    - i. How is it implemented?
      - 1. Can we do it exactly?
  - f. What is a balance set?
    - i. Why is it useful?
    - ii. When is it useful?
- 6. Disk Structure

- a. What is a platter?
    - i. How many are there?
  - b. What is a track?
    - i. How many are there?
  - c. What is a cylinder?
    - i. How many are there per track?
  - d. What is a sector?
  - e. What is a block?
  - f. What are some of the trends in disk design?
  - g. How can disks fail?
7. I/O Optimization
- a. How do we make file operations fast?
    - i. What is the importance of block size?
  - b. What characteristics of disk hardware most effect performance?
    - i. Know all of these!
  - c. What if there is more than one outstanding disk request?
    - i. How should we schedule them?
      - 1. FCFS
      - 2. SSTF
      - 3. SCAN
      - 4. C-SCAN
    - ii. Know the differences, pros, and cons of each one of these
    - iii. Are they all fair?
    - iv. Are any of them optimal?
  - d. What is rotational scheduling?
  - e. What are some problems we can run into when reading specific blocks?
    - i. What is a solution to some of these?
  - f. How can we minimize seek distance?
  - g. What is a disk cache for?
    - i. How does it help performance?
    - ii. What are some of the drawbacks of a disk cache?
  - h. What can we do at the file system level to increase performance?
    - i. What is. . .
      - 1. Pre-fetching?
      - 2. Data Reorganization?
      - 3. Data Replication?
    - ii. What are the pros and cons of each of these?
  - i. What is RAID?
8. File System Structure
- a. What is a file?
  - b. How are the blocks of a file laid out on disk?
    - i. Contiguous
      - 1. Pros?
      - 2. Cons?
    - ii. Linked List
      - 1. Pros?

- 2. Cons?
    - iii. Indexed Files
      - 1. Pros?
      - 2. Cons?
    - iv. Multilevel Indexed Files
      - 1. Pros?
      - 2. Cons?
  - c. How are files accessed?
  - d. What are the primary issues file systems should address?
  - e. What are the usual operating characteristics of a file system?
    - i. Most files large or small?
    - ii. More reads or writes?
    - iii. What kind of file IO is most prevalent?
  - f. How do we find the blocks of a file?
    - i. What is a File Descriptor? (iNode)
      - 1. What is recorded here?
      - 2. What is it used for?
      - 3. Where should the iNodes be placed on disk?
    - ii. What is NOT in a File Descriptor?
  - g. How do we allocate a new block for a file?
    - i. How do we find the block?
    - ii. How do we allocate it?
  - h. What is a directory?
    - i. What's stored in a directory?
    - ii. How does a disk transverse /one/two/three/four?
9. File Access Control
- a. What is an Access Control Matrix/List?
    - i. What are the rows?
    - ii. What are the columns?
    - iii. Where is this stored?
  - b. What is a Capability?
    - i. Where are they stored?
  - c. What are the disadvantages/advantages between the two?
    - i. Can we combine them?
10. Networks and Communication
- a. What is a Broadcast Network?
    - i. What are some of the issues that must be overcome for these to work well?
    - ii. What are some examples of this type of network?
  - b. What is a Point-to-Point Network?
    - i. What do these offer that Broadcast doesn't?
    - ii. What are some examples of Point-to-Point Networks?
  - c. What is latency?
    - i. Transmission versus Set-up
  - d. What is bandwidth?
  - e. What is the bandwidth-delay product?

- i. What does it tell us? Why is it useful?
- f. Ethernet
  - i. How does it work?
  - ii. Does it add anything to packets?
  - iii. Is it reliable?
  - iv. What is a MAC address?
- g. IP
  - i. Defining characteristics?
  - ii. What's in the header?
  - iii. Is it reliable?
- h. TCP
  - i. What does this offer that IP doesn't?
  - ii. How does it go about offering it?
  - iii. Is it slower/faster to transmit data than UDP?
  - iv. How does TCP control congestion?
- i. UDP
  - i. Defining characteristics?
  - ii. Why would you ever want to use this on top of TCP?
  - iii. Why use this at all, instead of straight IP?
- j. What are the seven ISO layers?
- k. What's the difference between circuit switching, packet switching, and "message" switching?
- l. What about network links makes them inherently unreliable?
  - i. Dropped packets
  - ii. Network congestion
- m. How is a packet routed through the internet?
  - i. What protocol is used?
  - ii. Is the internet reliable? Why/why not?

## 11. Distributed Systems

- a. What is a distributed system?
  - i. What are some of the goals in their design?
- b. Why are they useful?
- c. What is the General's Paradox?
- d. What is Two Phase Commit?
  - i. Why is it used?
  - ii. Why is it NOT used?
- e. What is Remote Procedure Call?
  - i. How is this different from local procedure call?
  - ii. Why do we want to use RPC?
  - iii. What are the issues that must be addressed in order to get this to work?
  - iv. What is 'marshalling'?
  - v. What are some problems with RPC?
- f. What is a distributed file system?
  - i. What if we made a system with no caching?
    - 1. How would this work?

- 2. What would be the issues?
- 3. What problems would this solve?
- ii. What is NFS?
  - 1. What are the key characteristics of NFS?
  - 2. What is really nice about NFS?
  - 3. What are the problems with it?
- iii. What is AFS?
  - 1. What are the key characteristics of AFS?
  - 2. What's nice about it?
  - 3. What are its problems?

## 12. Protection and Security

- a. What is the goal of protection?
  - i. Why not just lock your computer in a room, unplugged?
- b. What are the three aspects to a protection mechanism?
  - i. Authentication
    - 1. Passwords
      - a. Paradox – should they be long or short?
      - b. Salt
  - ii. Authorization determination
    - 1. Access lists versus capabilities
  - iii. Access enforcement
- c. What are some common weaknesses?
  - i. Abuse of valid privileges
  - ii. Imposter or Trojan Horse
  - iii. Listener / Eavesdropping
  - iv. Spoiler
  - v. Trap doors
- d. What are some countermeasures?
  - i. Logging
  - ii. Minimum privilege
  - iii. Correctness proofs
  - iv. Callbacks
  - v. Consistency / Plausibility Checks
- e. How does public/private key encryption work?
  - i. How do we go about establishing secure communications with a server?

## 13. Encryption

- a. What are challenges of implementation in practice?
  - i. Key distribution
  - ii. True randomness
- b. What are examples of crypto systems in use?
  - i. DES
  - ii. RSA
  - iii. PGP
  - iv. AES
  - v. Clipper Chip

#### 14. Internet Security

- a. What is a computer virus?
- b. What is a computer worm?
  - i. How different from a virus?
- c. How would one go about writing one of these?
- d. What is TCP Connection hijacking?
- e. How do we improve security in general?
  - i. What are the major problems here?