CSL373/CSL633 Major Exam
Operating Systems
Sem II, 2014-15

Answer all 11 questions (14 pages) Max. Marks: 62

1.	
2.	
3.	
4.	
5.	
6.	
7.	
8.	
9.	
10.	
11.	

For **True/False** questions, please provide a brief justification (1-2 sentences) for your answer. No marks will be awarded for no/incorrect justification. We will penalize you if you provide extraneous information not related to the question being asked.

1. If the disk is fully utilized (i.e., providing near-full disk bandwidth), does that imply that the scheduler is doing a good job interleaving CPU-bound and I/O-bound jobs? [4]

2. Persistent data can be stored either on the local disk of a machine or on the disk of a remote machine (e.g., NFS). In one such system which stores persistent data on remote disk, it also caches the file data in local memory, so that it does not need to go to the remote disk on each read access.

a. Should the filesystem block size be smaller or larger than the block-size of a local filesystem? (Possible answers: smaller, larger, same). Briefly explain. [3]

b. The filesystem engineer varies the filesystem block size from 512B to 4MB, and notices that her application performance first increases (with increasing block size) and then decreases (with increasing block size). The application primarily performs reads on the filesystem. Can you explain this behaviour (both increase and decrease)? What do you think is the likely inflection point and why? (Hint: inflection point is the block size at which the application performance was highest). Try and justify your answer by picking a realistic example of the application and the system configuration. [6]

c. Should the filesystem engineer implement a write-back cache or a write-through cache? Can you think of a workload where write-through cache would perform better than a write-back cache? Clearly state your assumptions and answer in 1-2 sentences. [4]

d. In your opinion, what should be a good crash-consistency semantics for this remote filesystem? Justify your answer by considering tradeoffs between performance and crash-consistency semantics. What support would you provide for the application such that it can make assumptions against crash consistency? After stating your solution, clearly enumerate the different scenarios (e.g., what crashes, when does it crash, how recovery happens). [6]

3. What is the problem if you run a database server as a user-process over a UNIX-like kernel (e.g., Linux)? How can an exokernel design solve this problem? Your answer should be clear and precise. [2 + 4].

4. In which of these situations can an intruder be able to compromise the security of the system. Assume that sensitive (private) information lives in the registers, memory, and the disk of the system. A security compromise means that the intruder can gain access to this sensitive information.

a. Intruder has access to the raw disk device storing the passwords [1]

b. Intruder has access to the power supply of a system and has a user (non-root) account on the machine. Access to power supply means that she can turn the machine on/off (through power supply) at any time, and any number of times. Assume that the filesystem implements an asynchronous crash consistency model, whereby the filesystem data is flushed to disk only at periodic intervals. You can also assume a specific filesystem (e.g., ext3) while answering this question. You may also define your own simple filesystem to answer this question. Your answer may depend on the choice of filesystem. Notice that the intruder already has a user account, but could compromise security if she could get access to sensitive data of another user/root user. [6]

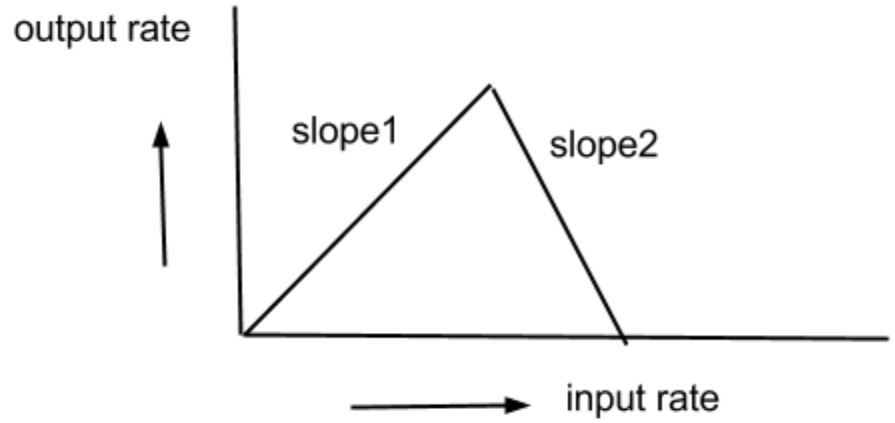
5. True/False: An ext3 transaction can be made to contain an arbitrarily large number of operations? If true, explain why. If false, name at least three different criteria on which the maximum size of an ext3 transaction would depend [4]

6. Why do we need separate “close” and “commit” operations for an ext3 log? Briefly explain in 1-3 sentences. Incomplete answers will not receive marks. [4]

7. Explain in one sentence, why RCU is better than reader-writer locks? [2]

8. Explain in 1-2 sentences why transactional memory (or transactions in general) are often considered better than locks? [2]

9. During receive livelock, the system may behave as follows (as also discussed during lecture):



What determines slope of line 1 (slope1)? What determines slope of line 2 (slope2)? [5]

10. The following code attempts to implement an atomic stack. Does it work? If yes, give a short intuitive correctness argument why it works. If no, state how it is vulnerable to race conditions. Assume the stack is initialized correctly. [6]

```
struct stack {
    struct lock head_lock;   // used to lock head
    struct elem *head;       // stack top
    lock_t count_lock;       // used to lock count
    int count;
};
struct elem {
    struct elem *next;
     /* other data fields. */
};
void push(struct stack *s, struct elem *e) {
    acquire(&s->head_lock);
    e->next = s->head;
    s->head = e;
    release(&s->head_lock);

    acquire(&s->count_lock);
    s->count++;
    release(&s->count_lock);
}
struct elem *pop(struct stack *s) {
    struct elem *e;
    int acquired;

    for (acquired = 0; !acquired; ) {
        acquire(&s->count_lock);
        if (s->count) {
            s->count--;
            acquired = 1;
        }
        release(&s->count_lock);
    }
    acquire(&s->head_lock);
    e = s->head;
    s->head = e->next;
    release(&s->head_lock);
    return e;
}
```

11. True / False : Increasing the size of the buffer cache should always reduce the (average) number of disk accesses. Assume everything else (e.g., workload, hardware, OS logic, etc.) is kept same, only buffer cache size is increased. [3]