上 海 交 通 大 学 试 卷（A 卷）

（2014 至 2015 学年　第 2 学期）

班级号_____ 学号_____ 姓名_____

课程名称_____计算机系统工程_____ 成绩_____

Problem-1: Network (25')

We decided to provide a online payment service to earn some market sharing from Mr. Ma. Fortunately, most of the payment system has been done and you should **only focus on the network** part. The current network is implemented simply by wrapping any requests from the applications and sending them out with at-least-once assurance.

1. This system is not quite popular at first. Many of our users left us and deleted their accounts and some of them left us with an error message when deleting:

```
*** Error in `...': double free or corruption (fasttop): 0x0000000000c9e010 ***
```

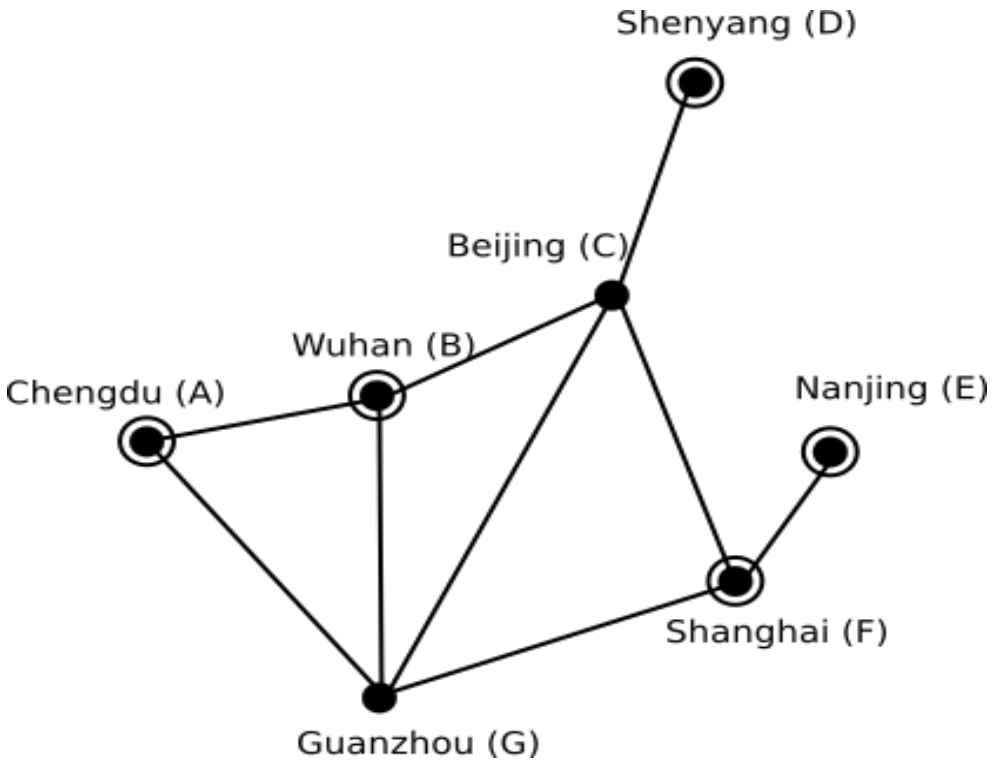
It is found that the critical code of deleting an account is like this:

```
extern map<string, struct account *> accounts;
      // ... something you don't care
      void remove_account(string user) {
            struct account *acc = accounts[user];
            free(acc);
      }
```

The account struct of existing user will always be valid and the users can only delete account *once*. Please explain in detail how this happened and how to solve it (hint: from the aspect of **network**) (4')?

2. The network cannot send out any request larger than 1500 bytes, which means we have to break a large request into some fragments. Please explain which layer puts this limit and give the reasons why the limitation is necessary. (4')

3. We try to provide high-availability service. A general idea is to distribute servers to different cities. So we finally put all our servers across 5 cities (the cycled nodes):



a. Internet Service Providers (ISP) from these cities deploy **path vector exchange** algorithm for routing and assume that every node spend the same time in each round of advertising and path selection steps. After 2 rounds of advertising and path selection, what is the path vector of Shanghai? How many advertising messages are needed to finish all rounds of exchanges (a node will only advertise messages when the state changes)? (5')

b. As shown in figure above, there is no server located in Beijing, which means account information may be distributed to different cities, i.e., Wuhan, Shenyang and Shanghai. To achieve fault tolerance, we will also synchronize data across different cities **periodically**. Someday, our optical fabric between Beijing and Shenyang is cut off. Can we simply redirect all requests to duplicated servers located in other cities to achieve high-availability? If not, why and what is the appropriate method? (4')

4. Assume that the SJTU TeleNet Corporation provides us a fixed bandwidth of 800,000 bytes per second for each user, and every request is a 1000-byte packet. Also, you can seem the network as a duplex which means you can achieve 800,000 bytes per second on both direction.

a. For simplicity, we first adopted a **lock-step** protocol with a fixed window. To fully exploit the network, we set the minimum window size as 160 packets within one session. Please give the first segment's round-trip time. (4')

b. Under the consideration of services' consuming rate, we then decided to adopt the **TCP** protocol. Assume that the round-trip time is exactly 1 millisecond (1 ms = 10^{-3} s) all the time, and the first time of **multiplicative decrease (whose decrease fraction is 0.5)** after **slow start** is 7 ms after sending first packet. Then after another 36 ms, the second multiplicative decrease occurs. Please give the packet consuming rate of server. (4')

Problem-2: Fault Tolerance (28')

1. Mary stores her data on a fault-tolerant storage system, which has 4 1-TB disks that together provide 4/3 TB of aggregate space, and can tolerate any two (but no more) disk failures, e.g., each piece of data has three replicas. Each disk has a MTTF of 10^5 days (24 * 10^5 hours), and you can assume:

- a disk fails as a whole, and
- all disk failures are independent.

a. The MTTF of a single disk is 24 * 10^5 hours, and thus the MTTF of one failure of the four disks is 6 * 10^5 hours. We can conclude that the MTTF decreases when there are more replications. Then, why do we use replications to improve reliability? How do replications make the whole system more reliable? (4')

b. Without repair, what's the MTTF of the system? (3')

c. It will take Mary 22 hours to buy a new disk from www.jd.com and 2 hours to initialize the content, and she will fix the disk as soon as there is a disk failure. What's the new MTTF of the system? (3')

d. According to your answer above, do you really believe that the system can be in use for so long? Which factors do you think will actually affect the system's lifetime? List as many as you can. (3')

2. Keeping three replicas takes up a lot of disk space. Ben was inspired by RAID, and designed a new system, design 2, to save space taking advantage of parity.

Design 1	Design 2
disk0: | a | b | c | e | f | …	disk0: | # | b | d | e | …
disk1: | a | b | d | e | f | …	disk1: | a | # | d | f | …
disk2: | a | c | d | e | g | …	disk2: | a | c | # | f | …
disk3: | b | c | d | f | g | …	disk3: | b | c | e | # | …

Layout of the original design. "a, b, c, …" represent disk contents. "#" represents a parity.

a. Do you agree with the following statements? If yes, please give your explanation. If no, please give a concrete case (8'):

   A. Design-2 has a higher performance.
   B. Design-2 is as reliable as the former one, e.g. it can survive the same number of disk failures as design 1.
   C. Design-2 put a heavier burden on cpu.
   D. Design-2 has a higher capacity.

b. Please help Ben modify design-2 to make it more reliable. Furthermore, it should be able to hold the same amount of contents as design-2. (3')

c. RAID promises high reliability. For a file system provided with a RAID, are logfiles still needed? Give your reasons. (4')

Problem-3: Security (26')

1. Ben is going to build a website. He heard that the cloud computing is cheap, so he decided to use virtual machine of ali-cloud (aka aliyun) to host his website. Which of the following are true about virtual machines? (8')

a. **(True/False)** Suppose Program A and Program B are two independent programs running inside the same guest OS. The virtual machine monitor prevents bugs in Program A from crashing Program B.
b. **(True/False)** Suppose Guest OS A and Guest OS B are two guest operating systems running on a single physical machine. The virtual machine monitor prevents bugs in Guest OS A from crashing Guest OS B.
c. **(True/False)** There is no reason to use virtual machines if the guest operating system is a microkernel rather than a monolithic kernel.
d. **(True/False)** Using virtual machines ensures that no bug can crash an entire physical machine.

2. Ben wants to use WordPress as the template of the website, apache as the web engine, and MySQL as the database. All are deployed on a virtual machine running Linux. He picked a nice domain name for his website: www.benben.com, and uses DNSPOD as his DNS server.

a. Please state the threat model for Ben (trusted and untrusted components). (4')

b. Sometimes Ben has to login to his website using some untrusted computers. Could you please give Ben some advices on how to design the login process? (hint: can we **only** use password?) (4')

c. Unfortunately, Ben doesn't care much about security. You want to warn him by attacking his website. What's your plan? (Hint: just list the steps you are trying to do) (4')

3. Ben finds that his website is attacked by some hacker. He asks help from you and shows you all the source code. After several days mining, you finally see some suspicious code here:

```
// auth2-chall.c in OpenSSH

input_userauth_info_response(...)
{
    unsigned int nresp;

    ...
    nresp = packet_get_int();
    if (nresp > 0) {
        response = xmalloc(nresp*sizeof(char*));
        for (i = 0; i < nresp; i++)
            response[i] = packet_get_string(NULL);
    }
}
```

And here:

```
int bufcpy(char *buf1, unsigned int len1, char *buf2, unsigned int len2)
{
   char mybuf[256];

   if((len1 + len2) > 256) {
       return -1;
   }

   memcpy(mybuf, buf1, len1);
   memcpy(mybuf + len1, buf2, len2);

   do_some_stuff(mybuf);

   return 0;
}
```

Assume that all the code are running on a 32-bit x86 machine. Please show the vulnerabilities on the above two code snippets. (6')

Problem-4: Consistency (21')

Ben wants to bring knowledge to every corner in the world. So he is establishing a chain second-hand book store "Book 0ops" in mountain villages. Each store, with infinite robot clerks, is located somewhere in the deep mountain. So the only way to communicate is to send clerks between stores. Clerks in the same store share memory, so you can take them along with the store as a whole.

Customers may do following operations:

- **buy(i, S, A)** : buy i books titled S at store A
- **sell(i, S, A)** : sell i books titled S at store A
- **query(S, A)** : ask at store A for how many books titled S Book 0ops totally has at the moment

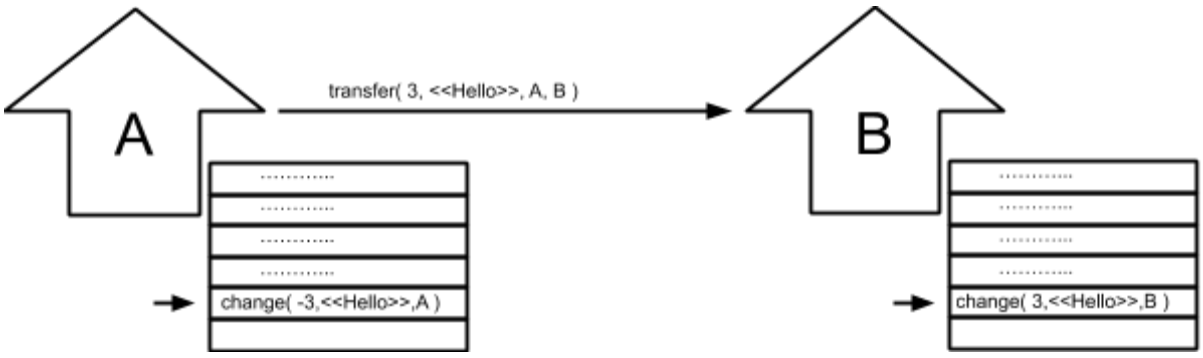
Each clerk can do only one of following operations each time:

- **transfer(i, S, A, B)** : carry i books of title S from store A to store B and return without communication
- **inform(A, B, lst)** : take an operation record list ***lst*** from store A to store B, tell the record list to store B, and return

Although these operations look complex, clerks only care about changes in book number. So each operation leaves records in the following form:

- **change(i, S, A)** : books titled S in store A changed by i, positive i means increase in number while negative i means decrease in number

E.g., **transfer()** will leave a record at each store:



You are an accountant in "Book 0ops", the only operation in your control is **inform()**
You have to help Ben make a protocol to ensure:

- Every **query(S, A)** returns a result as correct as possible

You can also assume:

- Clerks travel separately and each travel takes an uncertain period of time.
- Operations result in a negative book number of any title at any store are invalid and will not happen.

1. In the beginning, there were only two stores in operation. So Ben proposed a simple protocol to you: send a single clerk back and forth between two stores, taking message with all information visible to it. Operations were assumed to happen in the order they were visible to each store. Ben wonders whether this could guarantee a strict consistency.

a. What is the definition of strict consistency? What does it mean in this specific case? (4')

b. Soon enough, Ben found that sometimes a **query()** returned a number twice as large as real number. Please explain how this could happen. (4')

2. As "Book 0ops" grew, more and more stores were established. Ben decided to make some change. Ben noticed that, with a proper protocol, it is guaranteed that when a period of time T passes after an operation K, all **query()** will be aware of K. Ben wanted to make T as short as possible.

The second protocol Ben gave was like this:

- On the completion of *each* operation, clerks are sent to all stores to inform them of the operation.

a. To estimate T, Ben first needed to know the RTT (round trip time) between two stores. Calculate this for Ben according to the RTT of travelling clerks. Give a flowchart or pseudo code to show how. Explain the meaning of every variable you use. (4')

b. You found that all travels have almost the same RTT : 10 mins. With N stores, what is T now? (3')

c. Based on 2.b, if **no** two **inform()** operations are allowed to carry the same record **at the same time**, that is, **inform()**s are sent one after another,what's T now? (3')

d. Based on 2.c, if clerks can take some selected parameters instead of all records during inform(), and taking less information enables them to travel faster (RTT is 6 mins), can you give a protocol that works better? What's T under your protocol? (3')

上 海 交 通 大 学 试 卷（A 卷）

（2014 至 2015 学年　第 2 学期）

班级号_____ 学号_____ 姓名_____

课程名称_____计算机系统工程_____ 成绩_____

本页为答题纸，**请勿拆开**.

Problem-1 Network

1. This results from at-least-once assurance in network. Solutions are two: one by using at-most-once or exactly-once (sequence number and check duplication); one by setting the pointer to null.
2. One important reason is that the longer your packets are, the more expensive to resent a corrupted packet. Other reasons can be found in Principle of Computer System Design: an Introduction.
3.
    a. The path vector includes all cities except Chengdu. Messages for every city depends on edges, rounds depends on longest path.
    b. The replicated data may be stale, so we need a round of synchronization first.
4.
    a. 160 / (800,000 / 1000)
    b. ((2^7) / 2 + 36) / 1

Problem-2

1

a

MTTF of the whole system increases, because it only depends on a subset of all the replications.

Repair helps.

b

MTTF(system): T/4 + T/3 + T/2 = T* 13/12 = 26 * 10^5 (hours)

c

MTTF(system): T/4 * T / (3 * R) * T / (2 * R) = T^3/(24*R^2) = 10^15 (hours)

d

No. Human errors, natural disasters and so on.

2

a

A No. Concurrent write is not supported in the new design. example: write a and b.

B No. If disk 1 and 2 fail, content of a can not be recovered.
C Yes.
D Yes.
b
#abc
c#ab
de#f
fde#
c
Yes. RAID promises durability of the disks, while log provides atomicity of transactions. With the help of log, modification in main memory can be both atomic and durable.

Problem-3
1.a.F b.T c.F d.F

2.a.trust:apache, MYSQL, DNSPOD ...
untrust: the users …

b.Eg:Use two steps to confirm the identify of users instead of only passwords. Or Use one time password. …

c.Eg:Sql injection. Or analysis all the open source code, find the bug and exploit it to attack the website.

3.a.Change the unsigned to signed integer and then give xmalloc a huge parameter.
b.Integer overflow because of “len1 + len2”.

Problem-4

1. a. Strict consistency means every read to an object get the result of the latest write to this object. In this case, strict consistency means every query operation get the book number after the latest operation to any book in any store.
b. sell(100, S, A) → inform(A, B, lst) (lst includes the sell operation) → buy (50, S, A) → query(B, S)

2. a.
new_clerk_arrive_handler()
{
rtt = rtt * (α-1)/α + new_rtt * 1/α

}
rtt : the estimated rtt
new_rtt : rtt of arriving clerk
α : update factor, usually set to 8
new_clerk_arrive_handler : each time a clerk arrives, this is called
b. T = 5 mins
c. T = 5(n-1) mins
d. Only carry book title so that the receiver stores won't respond to relevant query immediately. Instead, they will send a clerk for the detail and then respond.
T = 3(n-1) mins