上海交通大学试卷（期末）

（ 2021 至 2022 学年 第 1 学期）

班级号____________________ 学号_________________________ 姓名__________

课程名称 计算机系统工程

成绩__________

Problem 1: Security (25’)

1. Alice sets up an email server on the public cloud. The email server was written in C++, a memory-unsafe language, so Alice uses the compiler to enforce Stack Canaries. How do Canaries defend against stack overflow attacks? (4’)

Stack Canaries protect against overwrites of a return address by inserting a magic value onto the stack after the return address, which is then checked before returns.

2. If there is a bug causing stack overflow and is known by an attacker, how can the attacker bypass Canaries of the email server? (4’)

He can guess Canary by endless crashes, as the email server restarts after crashes.

3. Shadow Stack is another way to defend against stack overflow attacks. Please briefly describe how it works and compare Shadow Stack with Stack Canaries. (Hint: from the aspects of security, performance, etc.) (4')

Shadow stacks store the return address in an isolated memory. Upon returning, the integrity of the return address is checked against the protected copy.

4. Alice finds that with Shadow Stack, the return address will be pushed and popped twice for each function call. She thinks the program can use the

original stack only for argument passing, and the shadow stack only for returning, so the return address will only be pushed and popped once. Will it make any trouble? (4')

Trouble: compatibility implications.

5. Alice uses hardware enclaves to protect the Email Server on the public cloud. What security properties do hardware enclaves provide (two)? How can Alice know she is using a real enclave? (5')

1) Isolation and 2) Attestation. Alice sends a nonce to the enclave; enclave signs the nonce using a private attestation key; she verifies using a public attestation key.

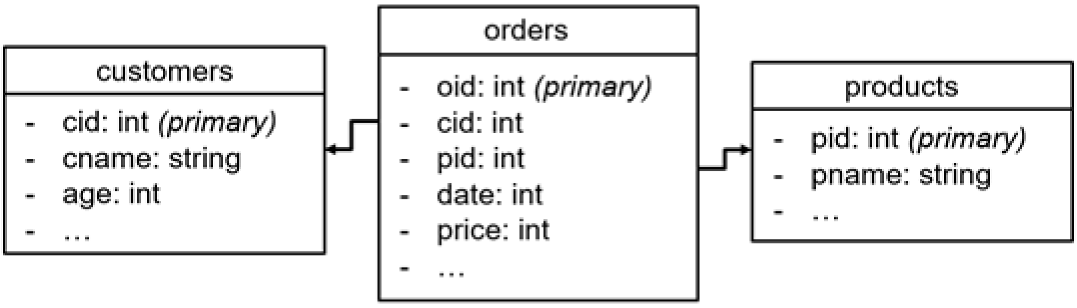
6. Alice installs a spam filter plugin to the Email Server. She uses a sandbox to ensure the spam filter can ONLY read her "email" file and write the "report" file (no rename nor network provided). Can you design a **side channel** to leak information? Hint: suppose another user, Bob, shares the same filesystem. (4’)

```
$ whoami
bob
$ ls -l /tmp
total 228K
-r--------    1 alice   alice      119K Jan 4 13:10 email
-rw-------   1 alice   alice        36K Jan 4 15:00 report
$ cat /tmp/email
cat: /tmp/email: Permission denied
$ cat /tmp/report
cat: /tmp/report: Permission denied
```

The spam filter *writes* the "report" file in a distinguishable way, periodically updating the 'size' attribute: fast incremental indicates 0, others 1.

Problem 2: Database (24’)

Xiaohong uses a **relational DBMS** called NaiveDB to manage the data produced by an online web store. To manage the information of customers, products and orders, Xiaohong created three tables as follows.



customers
- cid: int *(primary)*
- cname: string
- age: int
- ...

orders
- oid: int *(primary)*
- cid: int
- pid: int
- date: int
- price: int
- ...

products
- pid: int *(primary)*
- pname: string
- ...

1. Xiaohong wants to get the average age of the customers who bought ipad. NaiveDB provides two kinds of interfaces to retrieve data: KV interface and SQL. Xiaohong has tried both interfaces to get the result as follows.

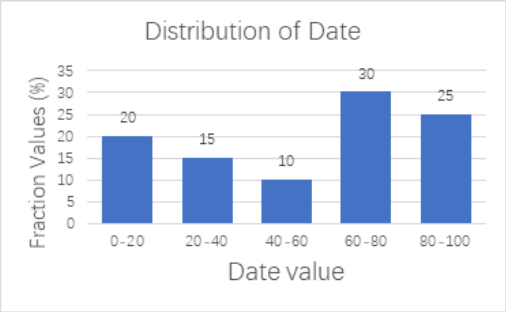
```
sum = 0
count = 0
for oid, row in orders:
   pname = products.get(row.pid).pname
   if (pname == 'ipad'):
       age = customers.get(row.cid).age
       sum += age
       count += 1
print (sum/count)
```

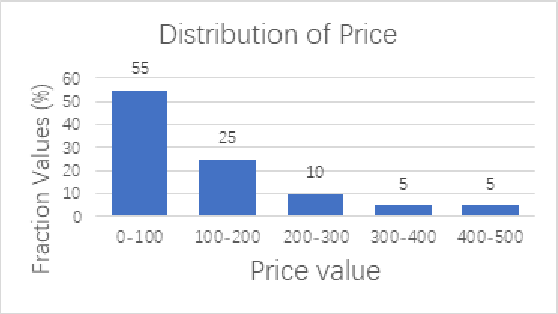
```
select avg(c.age)
from orders as o, custormers as c,
             products as p
where o.cid = c.cid
        and o.pid = p.pid
        and p.pname = 'ipad';
```

The KV interface is procedural, and the SQL interface is declarative. What are the benefits of such a declarative design? (5')

SQL hides implementation details. SQL is more expressive.

2. Indexing





```
select count(*) from orders where date > x and price > y;
```

To speed up the queries, Xiaohong created two indexes on `**orders.price**`, and `**orders.date**`. However, NaiveDB can only use **a single index** to accelerate the query above. The key distribution---how many keys are in a given key range is listed in the above figure. For example, the data that has date between 0-20 comprises 20% of the total data. Which index is better when x=40 and y=100 using the data distribution above? Why? (4')

(a) index `price` is better. (2')

(b) P(date>40)=65%, P(price>100)=45%.

3. OS can manage the cached pages of a file, and automatically move pages between memory and disk. However, NaiveDB manages a buffer pool for the cached pages by itself instead of using the buffer provided by OS. Why? Please describe at least two reasons. (5')

(a) OS does not know the database workload.

(b) OS processing overheads.

(c) unexpected buffer eviction due to other co-located applications.

4. The page size of the buffer pool in NaiveDB can be configured by the user. Xiaohong configured the page size to 16KB. Assumes that the NaiveDB uses B+Tree as its storage data structure. What are the benefits of using a large page size? What are the drawbacks? (Hint: the hardware can guarantee atomic write for 4KB page.) (5')

(a) benefit: shallow tree

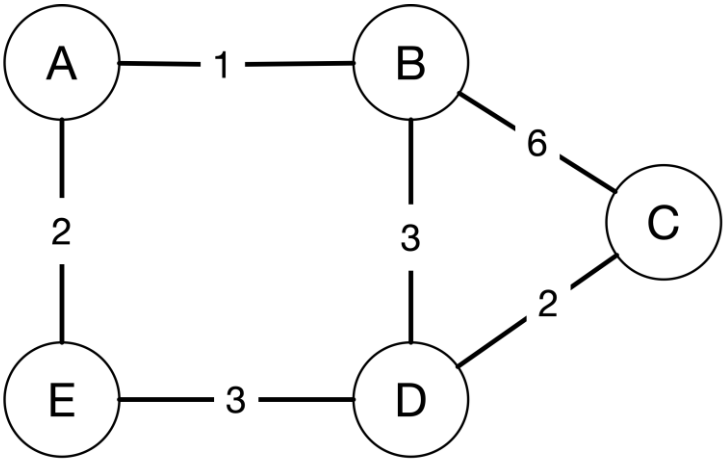
(b) drawback: need WAL

5. Xiaohong observes that the application mostly appends to the `orders` table. Assumes that the NaiveDB uses B+Tree to index the oid field, and the oid is monotonically increasing. If the NaiveDB uses the buffer pool to cache the index pages, which replacement policy can use to accelerate such a workload? Please briefly describe your reason. (5')

cache the right nodes of the tree

Problem 3: Network (26’)

1. Consider the following network graph using distance-vector routing. Initially, every node knows the cost to its neighbors. Please write down node A’ s initial routing table, as well as its routing table after one round of advertisement. Does A have the knowledge of the minimal cost to all nodes after one round? Please briefly explain your reason. (5’)



Initial:

dst	route	cost
B	A-B	1
E	A-E	2

After one round of advertisement:

dst	route	cost
B	A-B	1

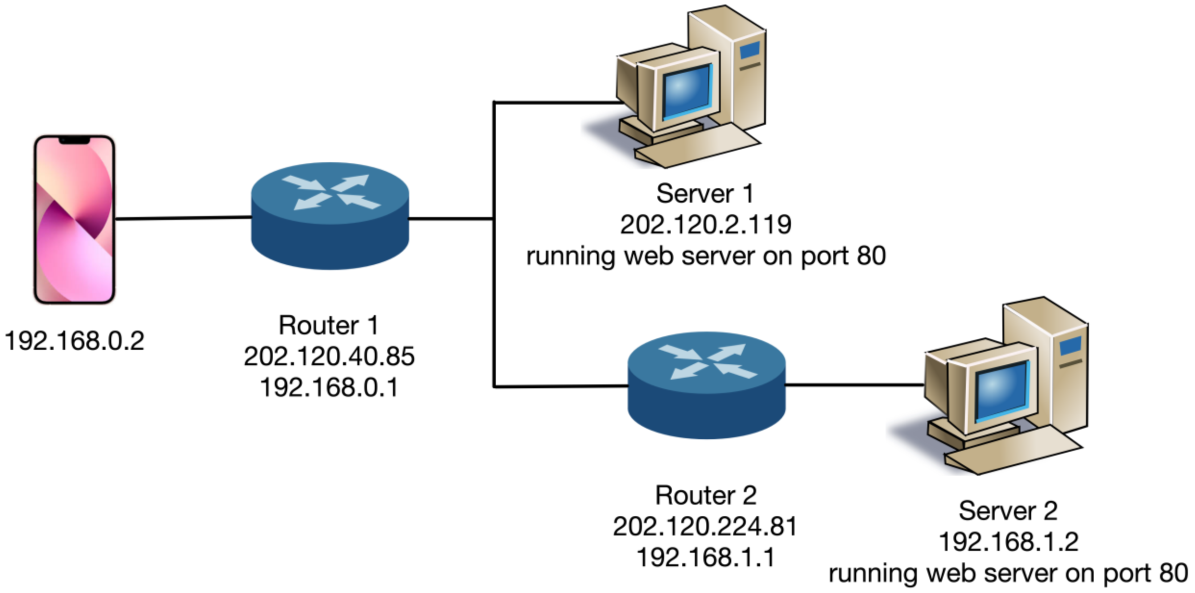
C	A-B	7
D	A-B	4
E	A-E	2

Node A does not contain the minimal cost to node C yet.

2. The INFINITY problem is a well known issue of the distance-vector routing algorithm. Please explain what it is, and provide a possible solution. (4’)

When a node A has no route to destination B, it will advertise a cost of INFINITY to B. But because the order in which advertisements are sent matters, sometimes nodes can incorrectly think there's a route when there isn't one. This can last for up to INFINITY steps (usually 16 or 32). Solution: split horizon.

3. Consider the network topology in the following figure and the servers use NAT (Network address translation) to provide a private network. The mobile phone and server 2 access the network through two routers, which bridges the private network and the internet. Can the mobile phone behind router 1 issue HTTP requests to fetch data from server 1 and server 2 without special configuration, respectively? If so, how is the connection established and what is the client IP address that the server see? If not, why? (4’)



Server 1: yes, through NAT, 202.120.40.85; Server 2: no, NAT gateway does not contain an entry to forward request to server 2 (since the request is not initiated from server 2).

4. Imaging two clients (S1 and S2) are sending TCP requests through the same network link, which uses AIMD (Additive Increase, Multiplicative Decrease) for congestion control. No other clients are concurrently using the link, and it is the only bottleneck, which has a capacity of 24 packets/s and latency of 500ms. At some time after slow start, the congestion window of S1 is 5 packets and S2 is 3 packets. Please list the congestion windows of S1 and S2 after 1 RTT, 2

RTT, ..., 10 RTT. When not divisible, round down (e.g., 3/2=1). (5’)

RTT	0										
S1	5										
S2	3										

RTT	0	1	2	3	4	5	6	7	8	9	10
S1	5	6	7	8	4	5	6	7	3	4	5
S2	3	4	5	6	3	4	5	6	3	4	5

5. Distributed data store follows the CAP theorem, which states that a data store can only provide two of the following three properties: consistency, availability and partition tolerance. Please explain which property is sacrificed in DNS, and give an example. (4’)

Consistency. If the DNS record is changed, the old value might still be cached.

6. A hash table typically takes O(1) operations for lookup. Can a distributed hash table (DHT) using consistent hashing provide the same guarantee? If not, what is the time complexity, and where does the gap exist? (4’)

No. O(log n). A DHT needs to first locate which node the entry belongs to. The resolving procedure is O(log n).

Problem 4: Distributed Transactions (25’)

1. Suppose you have a distributed database system and run transactions spanning multiple servers. The troublemaker Till plans to trigger multiple failures in your system, and those failures may let the system violate ACID semantics. To ensure correctness, the system must support multiple-site atomicity. **Please briefly describe what does multiple-site atomicity means** (3').

All or nothing in distributed transactions. When crash or abort occurs, all-or-nothing can always be guaranteed.

Two-phase commit (2PC) is a standard protocol to achieve multiple-site atomicity. The coordinator and workers (servers that store the data) communicate with each other to reach a consensus on the final transaction execution results. **Please explain what is 2PC and how it achieves multiple-site atomicity** (3').

2PC protocol allows the crashes in each phase, and trigger corresponding policies (resending

COMMIT, redo log etc.) to recover from the crash.

2. Till introduces the following failure scenarios. Please state whether the transaction could commit or abort, and give you reasons. Assume that your system is running 2PC.

1) One worker crashes during execution stage (i.e., before prepare) of the transaction. **Will this transaction commit / abort ? Give your reasons.**(3')

Abort. Worker does not finish the tx.

2) One of the coordinator's PREPARE message is lost (and the coordinate never reties). **Will this transaction commit / abort ? Give your reasons.**(3')

Abort. The coordinator won't receive all of the response, timeout and abort the tx.

3) One worker crashes after replying yes (prepared) in PREPARE phase, but before receiving the COMMIT message. All of other workers reply yes successfully and never crash. **Will this transaction commit / abort ? Give your reasons.**(3')

COMMIT. The coordinator will send COMMIT again until worker recovers.

4) The coordinator crashes after sending some of (not all of them) the COMMIT messages.**Will this transaction commit / abort ? Give your reasons.**(3')

COMMIT. The coordinator has log the commit. Thus resend messages after recovery

3. A data object (e.g., file) can be replicated to multiple sites to ensure fault tolerance. In the class, we've learned quorum-based algorith for consistency, i.e., each later read can retrieve the latest write. To achieve so, each operation has to obtain a read quorum (Q_r) or a write quorum (Q_w) to read or write a data item, respectively. Assuming the total number of replicas is N and answer the following questions.

1) Please describe the quorum rules in detail and explain how quorum helps in replica consistency. (4')

Qr + Qw > N.

To ensure the Qr and Qw could have overlap(at least one replica) that the Read operation could always obtain the latest version of data. **In addition, the quorum system can adjust the performance of reads and writes. By adjusting Qr and Qw, the system can better support Read or Write requests.**

2) Assume $N = 7$. **Please construct an example of** Q_r and Q_w that **is in favor of updating, and briefly explain your choice.** (3')

Qr = 5, Qw = 3

我承诺，我将严格遵守考试纪律。

承诺人：__________

题号									
得分									
批阅人(流水阅卷教师签名处)									