CS 6210 Fall 2022 Final – Test 3 (allotted time for canvas quiz = 120 minutes)

Name:_____GT Number:

Note:

1. **Write your name and GT number on each page.**
2. The test is **CLOSED BOOK** and **NOTES.**
3. Please provide the answers in the space provided. You can use scratch paper (provided by us) to figure things out (if needed) but you get credit **only** for what you put down in the space provided for each answer.
4. For conceptual questions, **concise bullets** (**not wordy sentences**) are preferred.
5. While it is NOT REQUIRED, where appropriate use figures to convey your points (a figure is worth a thousand words!)
6. **Illegible answers are wrong answers.**
7. **DON'T GET STUCK ON ANY SINGLE QUESTION…FIRST PASS: ANSWER QUESTIONS YOU CAN WITHOUT MUCH THINK TIME; SECOND PASS: DO THE REST.**

Good luck!

Question number		Points earned	Running total
1 (1 minute)	(Max: 1 point)		
2 (10 minutes)	(Max: 20 points)		
3 (9 minutes)	(Max: 9 points)		
4 (10 minutes)	(Max: 12 points)		
5 (12 minutes)	(Max: 13 points)		
6 (5 minutes)	(Max: 5 points)		
7 (5 minutes)	(Max: 9 points)		
8 (3 minutes)	(Max: 6 points)		
9 (3 minutes)	(Max: 6 points)		
10(10 minutes)	(Max: 10 points)		
11(5 minutes)	(Max: 5 points)		
12(5 minutes)	(Max: 6 points)		
Total (78 minutes)	(Max:100 points)		

1. **(1 minute, 1 point)** (you get 1 point regardless of your answer)
In Spring 2023 Kishore's Class CS 6211 – System Design for Cloud Computing (formerly offered as an 8803 in both the on-campus and OMS programs) is offered both on campus and for OMS. Gauging your interest in signing up for this: https://faculty.cc.gatech.edu/~rama/CS6211-External/ (Note: This is an intense hands-on project-oriented course) (circle one):

a) I did well in CS 6210, and I will sign up.
b) I did well in CS 6210, but I will not sign up.
c) I did not do well in this course, so I will not sign up.
d) I am not a fan of systems courses, so I will not sign up.
e) I am not a fan of Kishore's courses, so I will not sign up.

Potpourri (HIDDEN)

2. (10 minutes, 20 points) (HIDDEN)
(Answer True/False with justification. No credit without justification)

(+1) if the T/F false bit correct (ONLY if there is some accompanying justification)
(+1) if the justification is correct

(i) This question pertains to Coral Sloppy DHT which uses key-based routing. You are given the invariant that a node with node-id N = Key K is always up; and the node N has indicated that it is willing to host exactly ONE meta-data entry for Key K. Given this, a get(K) from a requesting node could result in the requestor not getting the "value" associated with "Key K".

Answer: False. Regardless of the key-based routing algorithm used the requestor will eventually hit the node N from which it will get the "value" associated with Key K.

Acceptable answer:
True. There is a network partition between the client and Node K.

(ii) A data center uses both partitioning the corpus of data into chunks and hosting the chunks on distinct servers to increase opportunity for parallel processing of an incoming request; it also uses replication of the data chunks in multiple servers for fault tolerance. Assume 20% of the servers fail, and these are exactly the servers that are used to hold replicas of the data chunks. Given this situation, there is no way the data center can give "full harvest" for an incoming query.

Answer: False. The failed nodes are holding the replicas. The primary copy servers are still available to provide a "full harvest".

(iii) Your friend has submitted a map-reduce job. Map is a sequential function that works on a given input shard. Her input data is split into 100 shards. Ignoring any set-up time, the actual time to perform the map function for a shard is T. Initially she had asked for 50 nodes for the map phase. Ignoring all set-up time, the map operations on the entire input will complete in 2T time units.

Answer: True. The 50 nodes will do double duty to complete the map phase.

(iv) This is continuation of the previous question. To speed up the map phase, she asks for 200 nodes. With this allocation, the map operations on the entire input will complete in T/2 time units.

Answer: False. The map function itself is indivisible, so 100 nodes complete the 100 shards for the map phase in T time units. The remaining 100 nodes are idle.

(v) Consider 3 processes running on a uniprocessor: P1 – high priority; P2 – low priority; P3 – medium priority. P1 makes a blocking RPC call to P2. Since P3 could potentially preempt P2, there is no way to ensure that P1's RPC takes a deterministic completion time.

Answer: False. The OS can bump up the priority of P2 to be that of the caller (in this case P1) thus ensuring that P2 will not get preempted by P3.

(vi) Temporal causality in a multithreaded application pertains to associating timestamps with data items produced by the threads.

Answer: False. It pertains to outputs of independent threads being related to one another along the time axis.

(vii) RioVista writes an "UNDO Log" at the beginning of the transaction. The UNDO Log is resilient to power failure.

Answer: True. The UNDO log is implicitly written into the Rio File cache which is battery backed and survives both power failure and software failure. Thus, crash recovery could restore the data segment to its original value (before the transaction) if the node crashed in the middle of the transaction.

(viii) Consider a multi-threaded process which is using the LRVM library for persistence support. Two threads simultaneously execute the following code: "begin-xaction; set-range(region R); modify data structures in R; end-xaction". Given the above execution, LRVM will create a composite redo-log that contains faithfully all the independent changes to the data structures in R made by both the threads.

Answer: False. This is a data race between the two threads. There is no guarantee what the redo-log will contain since LRVM is not responsible for ensuring mutual exclusion for the access to the data structures contained in the region.

(ix) Rio File cache uses a battery backed DRAM for the file cache. Therefore, writing to the disk from the file cache can be arbitrarily delayed. Such delaying would in fact reduce the total amount of disk writes.

Answer: True. Repeated writes to the same block of a file will not result in repeated disk writes; lifetime of many files is short and so they may actually get deleted before they have to be written to the disk.

(x) Key distribution problem is one of the main reason why AFS chose not to use public key crypto system.

Answer: True. For every user a pair of keys is needed. And these keys have to be populated in EVERY Virtue workstation in the entire campus. And the key DB has to be updated and kept consistent with the churn in the student population.

Internet-scale computing

3. (9 minutes, 9 points) (Giant-scale services)

You are building a planetary <key, value> store. The key is a unique-id for every person on the planet. The value is information about the person such as name, phone number, email-id, etc. You wish to provide high availability and fast access time to process queries to the <key-value> store.

Your design places one data center in every continent with an identical replica of the <key, value> store.

- The data center has 100,000 servers.
- The <key, value> store is partitioned into 10,000 equal shards and stored on 10,000 servers.
- Each shard is replicated in 10 servers.
- You intend to provide "full harvest" for each query and fully exploit the available parallelism for processing each query.
- A server assigned to a query is dedicated to that query for the duration of the query processing.

(a) (3 points) How many simultaneous queries can you process in each data center? Justify your answer.

Answer:
10. (+1.5)
This is the number of replicas available. (+1.5)

(b) (3 points) Assume that the query processing time is Ts for each shard, and the time for aggregating the intermediate results into the final result is Ta. What is the processing time for each query? Justify your answer.

Answer:
Ts+Ta. (+1.5)
All the 10,000 shards will be processed in parallel for a given query. (+1.5)

(c) (3 points) Assume that you perform rolling upgrade whenever there is a hardware upgrade. The rolling upgrade takes down 10% of the servers at a time in each phase. What strategy will you use in the rolling upgrade that ensures the data center remains available all the time? What will be reduction in DQ during every upgrade phase?

Answer:
In each phase, update a set of 10,000 nodes that contain the full replica of the <key-value> store. (+1)
This will leave the remaining 90,000 nodes available for processing incoming queries at full harvest. (+1)
The DQ will be reduced by 10% in each phase. In other words, the yield will be 0.9 (instead of 1). (+1)

4. (10 minutes, 10 points) (Map-reduce)
A map-reduce application consists of:

- 100 shards of data to be processed
- 20 distinct outputs to be produced
- CPU time to execute a map function: T_m
- CPU time to execute a reduce function: T_r
- I/O time to write the intermediate result by a mapper: T_i
- (Blocking) RPC time to fetch an intermediate result by a reducer: T_{rpc}
- I/O time to write out the result by a reducer: T_f

The map-reduce infrastructure uses 50 threads for the map function and 10 threads for the reduce function. Ignoring scheduling overheads by the infrastructure, compute the total execution time for the above application. Assume that the reduce phase starts ONLY after the map phase is complete.

(Show your work for partial credit)

Answer:
- 50 threads work in parallel each taking: Tm+Ti to finish (+2)
- Since there are 100 map instances total time for mapping: 2*(Tm+Ti) (+2)
- Each reducer has to do 100 blocking RPCs to get the intermediate results: 100*Trpc (+2)
- 10 reducers work in parallel to produce the final result each taking: 100*Trpc + Tr + Tf (+2)
- Since there are 20 reduce instances total time to complete the reduce work: 2*(100*Trpc+Tr+Tf) (+1)
- Total time = 2*[(Tm+Ti) + (100*Trpc+Tr+Tf)] (+1)

5. (12 minutes, 13 points) (CDN – Coral)

In the following

- Put(x, y) denotes putting the key-value pair (key = x and value = y)
- Get (x) denotes getting the value corresponding to the key = x

Assume a coral system in which the "l" value for a node is set to 3; "β" is infinite.

Consider the following scenario:

- Put (100, 1)
- Put (100, 9)
- Put (100, 8)
- Put (100, 6)
- Put (100, 5)
- Put (100, 4)
- Put (100, 3)
- Put (100, 2)
- Put (100, 7)
- Put (100, 10)

Assume that there have no prior puts with the key 100.

Assume that the Coral routing for all the above puts hit exactly the following nodes *en route*: 48, 70, 88

(a) (5 points) Show the occupancy of the key-value pairs in the system after the above puts have completed successfully

Answer: (+0.5 for each correct entry below)
Node 100: (100, 1); (100, 9); (100, 8)
Node 88: (100, 6); (100, 5); (100, 4)
Node 70: (100, 3); (100, 2); (100, 7)
Node 48: (100, 10)

(b) (2 points) Node Q executes Get(100). The Coral routing for this request takes the path (Q -> 50 -> 75 -> 88 -> 100). What does Node Q get back?

Answer:
Gets the three values (6, 5, 4) from Node 88

(have to avoid double jeopardy in grading)

(c) (2 points) A node P executes Get (100). The Coral routing for this request takes the path (P -> 35 -> 79 -> 90 -> 100). What does Node P get back?

Answer:
Gets the three values (1, 9, 8) from Node 100

(have to avoid double jeopardy in grading)

(d) (4 points) Node 100 is shut down.

(i) A node R executes Get (100). The Coral routing for this request takes the path (R -> 45 -> 75 -> 88 -> 100). What does Node R get Back?

Answer:
Gets the value (6, 5, 4) from Node 88

(have to avoid double jeopardy in grading)

(ii) A node S executes Get (100). The Coral routing for this request takes the path (S -> 45 -> 78 -> 90 -> 100). What does Node S get back?

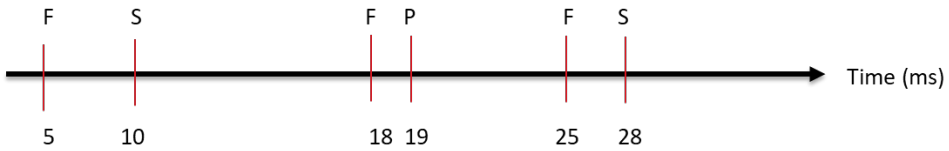
Answer:
Path tries to go all the way to 100 but gets nothing since 100 is dead.

Real-time and Multimedia

6. (5 minutes, 5 points) (TS-Linux)

Given the following timeline:

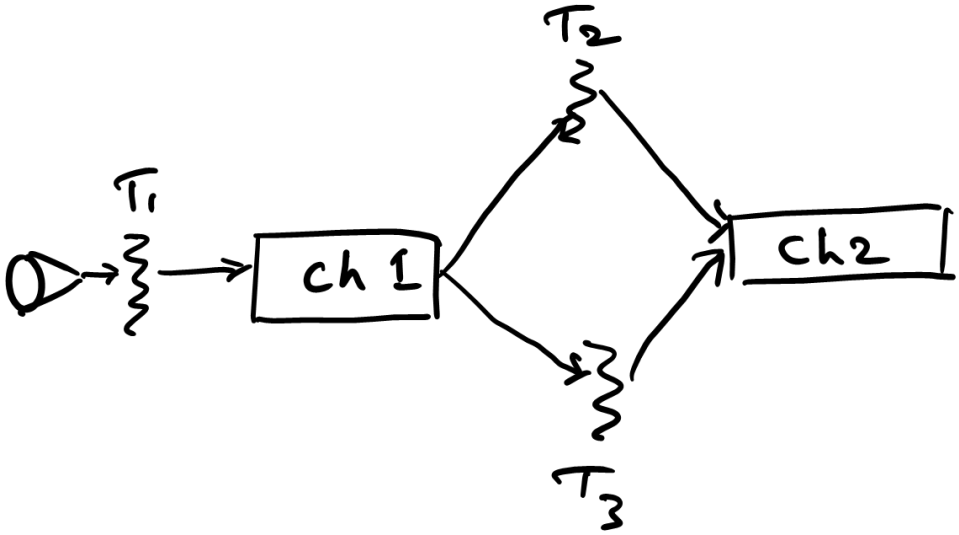
- P is periodic timer interrupts
- S is system calls
- F is firm timer interrupts (implemented using one-shot timers with an overshoot of 2 ms)



How many external interrupts are experienced by the OS? Justify your answer for any credit.

Answer:
There will be one firm timer interrupt at time = 5; (+1.5)
The firm timer interrupt at time = 18 will be combined with the periodic interrupt at time = 19; (+2)
There will be another firm timer interrupt at time = 25. (+1.5)

7. (5 minutes, 9 points) (PTS)



(a) (3 points) with reference to the above PTS graph, assume that Ch1 has items with the following timestamps: 10, 20, 30, 40, 50. Assume that ch2 has NO items presently.

Threads T2 and T3 concurrently execute the following code:

```
<item, ts> = get(ch1, "oldest");
processed-item  = process(item);
put(ch2, processed-item, ts);
```

What will be the content of Ch1 and Ch2 after the above code is executed?

Answer:
Ch1 contents remain unchanged; (+1)
Ch2 will contain two new items both with timestamps = 10 (+2)

(b) (3 points) Consider the same set up as above. Now assume that Ch2 already has 4 items with timestamps: 2, 4, 6, 8. The channel has a maximum capacity of 5 data items. The channel policy is to keep the most recent timestamped items. What will be the content of Ch1 and Ch2 after T2 and T3 execute the same code as in part (a)?

Answer:
Ch1 contents remain unchanged; (+1)

Ch2 will contain items with the following timestamps: 4, 6, 8, 10, 10; (+2)

Bonus point:
item with timestamp = 2 will be moved into the garbage list for Ch2. (+1)

(c) (3 points) What should the developer do to make sure there is no data loss?

Answer: The developer should provide the PTS runtime with a pickling handler for Ch2 that will allow PTS to automatically store the items retired from Ch2 into archival storage.
(+3 if the above sense is conveyed)
Partial credit: (+1) if the answer says that NOPERSIST flag should not be set for this channel. This is not a complete answer since the handlers are needed to store/restore from archival store.

Failures and Recovery

8. (6 minutes, 6 points) (LRVM)

(a) (3 min, 3 points) How does the "no flush" mode in end-transaction help in improving the performance of a server written on top of LRVM?

Answer: For recovery, the redo log has to be synchronously flushed to the disk. This means that the server will be blocked at end-transaction waiting for the I/O to be completed by LRVM. Since I/O is a long-latency operation this will result in significant performance loss for the server. Using "no flush" mode, the server is indicating to LRVM that it should NOT be blocked at end-transaction. Eliminating blocking I/O will make the server more performant.
Of course, it is now the responsibility of the server to explicitly perform log flushes but this is not in the critical path of the transaction execution, and could be done in parallel by another thread thus overlapping server execution with I/O.
(+3 if the answer says the server can continue execution in parallel with I/O)

Another acceptable answer with full credit:
Since LRVM does not do a log force at end-transaction, and it becomes the developer responsibility to persist the log to the disk at their own level or paranoia, it could potentially reduce the total amount of I/O performed and thus increase the performance (at the cost of vulnerability to data loss due to power failure or software crash).

(b) (3 min, 3 points) During crash recovery, the redo log is applied to the data segments to bring the server to a consistent state prior to the crash. LRVM chooses to apply the log to the affected data segments starting from the tail of the log rather than the head. Why?

Answer: It is conceivable the same records of the data segment will written in multiple transactions T1, T2, T3, ... in that order. T3 is

the most write to that data record. So the writes by transactions T1 and T2 are redundant and can be eliminated thus reducing the crash recovery time.
(+3 if redundant writes eliminated mentioned)

9. (6 minutes, 6 points) (RioVista)
LRVM and RioVista provide the same API for application developer. The difference is only in the implementation of the APIs in the two systems.

(a) (3 min, 3 points) Explain the difference in the implementation of the "set-range" call in the two systems.

Answer:
LRVM creates an in-memory UNDO record which is thrown away at the end of the transaction. (+1.5)
RioVista creates an UNDO log (which is implicitly written to the Rio File cache) which survives failures since Rio file cache is battery backed. (+1.5)

(b) (3 min, 3 points) What is the effect of the "no abort" mode in the begin-transaction call in RioVista? Explain your answer.

Answer:
This mode will be ignored by RioVista. (+1)
The reason is the UNDO log *must* be created in RioVista since the transaction is directly making modification to the data segment that is memory-mapped to the virtual memory of the server. So in the event of a crash during the transaction, the data segment has to be restored back to the original state *before* the transaction for which the UNDO log is necessary. (+2)

10. (10 minutes, 10 points) (Quicksilver)
You are implementing a key-value store (call it the "server") on top of the Quicksilver operating system. Your server entertains "get" and "put" calls from remote clients on a local area network (LAN). In processing a "put" and "get" calls the server would create windows on the client workstation and store/retrieve the key-value pair in persistent storage potentially calling other "storage" nodes on the LAN. The transaction tree created thusly for recovery purposes keeps track of breadcrumbs created by the client-server interactions. You have provided helper code so that the TM at each node touched by the client-server interactions records the ephemeral data structures and persistent data structures created by these interactions. You have also provided the cleanup code for getting rid of the breadcrumbs left behind by the client-server interactions.

(i) (2 points) What is the lifetime of the transaction tree created by the client-server interactions?

Answer: The transaction tree goes away (after taking the appropriate recovery action if needed) as soon as the "get/put" call is

successfully completed, or if the client process abnormally exits the program after issuing the "get/put" call.
(all or nothing)

(ii) (4 points) What protocol would you use in the cleanup code for the windows created on the client workstation? Why?

Answer:
One-phase commit protocol. (+2)
The windows are ephemeral data structures so notification to get rid of them (i.e., a single phase) from the coordinator TM is sufficient for the cleanup. (+2)

(iii) (4 points) What protocol would you use in the cleanup code for the server to successfully complete the "put" operation? Why?

Answer:
Two-phase commit protocol. (+2)
The coordinator must get agreement from the storage nodes (in the first phase) touched by the server for the "put" operation, and only if all the storage nodes "vote" to commit, the coordinator will successfully complete the "put" operation (in the second phase). (+2)

Security

11. (5 minutes, 5 points) (AFS)

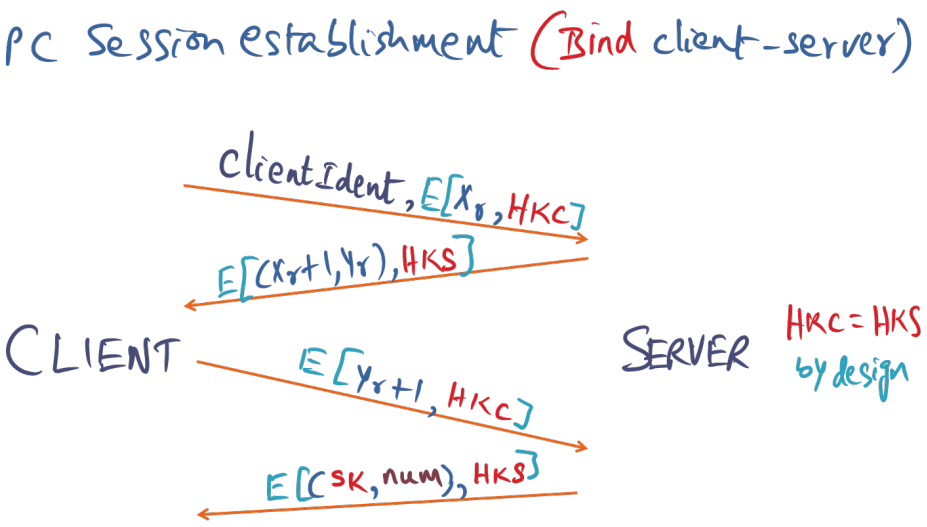
You and your friend are reviewing the design choices made in the Andrew File System (AFS). Your friend points out that after logging in both the client workstation (Virtue) and the file server (Vice) have the symmetric key (called the handshake key HKC) needed for encrypting and decrypting message between the client and server. Given this, she argues there is no need for the "bind" protocol which is the centerpiece of AFS. How would you counter her argument? **(succinct bullets please)**

Answer: Bind protocol is needed for addressing mutual suspicion. Both the client and server need to know that the other party is really who they are claiming to be. For example, if the HKC is compromised a person in the middle (who has the compromised password) could decrypt the initial message from the client. However, they do not know how to interpret the content (the fact that the bind protocol embeds a random number Xr that has to be incremented and sent back to the client). Therefore, even if a response is sent back to the client by the masquerader the client workstation will be able to find out that the reply did not come from the "real server". The same is true for Yr in the other direction.

(+5 if the above sense is conveyed in the answer)

12. (6 minutes, 6 points) (AFS)

Consider the following situation. You are a student in CMU in the 80's. You have logged into Vice from virtue workstation. Now you want to start an RPC session. Virtue on your behalf starts the bind protocol as shown below:



(i) (1 point) A person in the middle gets a hold of the client response "E[Yr+1, HKC]". They send the packet to the server. What would happen as a result?

Answer:
Any of the following gets 1 point:
The server will simply ignore the packet as a duplicate.
OR
It may also resend "E[(SK, num, HKS]" if it thinks the client side did not get the original response.

Bonus point (if they say this in addition):
In either case the person in the middle cannot do anything with it since they do not have the key HKC.
OR
This does not result in system malfunction. It simply is a form of a "Denial of Service" attack on the system.

(ii) (1 point) You are in the middle of an RPC session. You want to download a file. What key would you be using to make the file request? Why? (no credit without answering the why part).

Answer: SK is the key used. Once the session is established, Virtue uses the newly generated SK for THIS session in order not to over-expose HKC.
(all or nothing)

(iii) (1 point) While you are in the middle of the RPC session, the person in the middle who got a hold of your "E[Yr+1, HKC]" packet, successfully breaks the key. Now they have the key HKC. How does it affect your current RPC session? Explain your answer.

Answer: Virtue will be using SK for the duration of the current RPC session. Therefore, the fact that HKC is available to the person in the middle does not affect your current RPC session.
(all or nothing)

(iv) (3 points) You start a new RPC session (with the key HKC compromised as above). What could happen as a result?

Answer:

The person in the middle could successfully the first message decrypt "ClientIdent, E[Xr, HKC]" and respond to the client masquerading as the server. (+1.5)

However, Virtue is expecting a response in which Xr is incremented by 1. Since the person in the middle does not know the semantics of the protocol, their response is not consistent with the protocol. Virtue will reject the response and log you out. (+1.5)

Bonus point:
You can login to Virtue again and restart a new login session (which will create a new ClientIdent and HKC for the new login session).(+1)

(+3 if the above sense is conveyed)