CS 6210 Spring 2024 Test 3

(120 min Canvas Quiz)

Max Points: 96

Internet Scale Computing [24 points]

Giant-Scale Services [8 points]

1. [2 points] Traditionally, database systems use IOPS as a metric (number of I/O operations per second) as a figure of merit. Modern cloud-based systems seldom use this metric. Explain why.

Answer:

- Giant-scale services that are the norm for the Cloud are network bound than I/O bound since replication and partitioning in the datacenter remove the I/O bottleneck.

2. [6 points] Assume that you are developing a giant scale service that has a corpus of data that is 150 TB. An individual server has a storage capacity of 10 TB, and you want to be able to serve up to 1000 simultaneous queries without any queuing delay, wherein each query requires a full harvest.

a. [3 points] How many machines will you need to meet these requirements? Explain your answer.

Answer:

- (+1.5) There will need to be 15 machines to handle a single concurrent request (since 150 TB / (10 TB/machine) = 15 machines)

- (+1.5) Since you need 15 machines/1 request, to have 1000 concurrent requests, there will need 15 * 1000 = 15000 machines.

b. [3 points] Assuming that a request is successful only with full harvest, what's the best-case and worse-case scenario as to how many simultaneous requests can be handled without any queuing delay if 15 machines went down?

Answer:

- (+1.5) Best-case scenario: the service can handle 999 concurrent requests if the servers are in the same server group that comprises an entire corpus of data.

- (+1.5) Worst-case scenario: the service can handle 985 concurrent requests if each of the 15 downed machines is a part of a different group of machines that comprises the corpus of data.

Map Reduce [8 points]

3. (4 points) For this problem assume that a mapper writes its intermediate results to a blob storage in the cloud; and a reducer gets it from the blob storage using RPC. The following items pertain to a given map-reduce application.

- Worker A is tasked with a map operation, which it completes successfully, storing R intermediate files to the blob storage.
- Worker B is assigned a reduce operation.
- Worker A experiences failure immediately after storing the intermediate files before informing the master that it has completed the assigned map task.

With succinct bullets explain what actions would be taken by the map-reduce framework.
Answer:

- Non-completion (i.e., no notification) and lack of "I am alive message" from Worker A will result in the master starting a new worker (say Worker C) for the mapper task assigned to Worker A (+1)
- Worker C completes its task (including storing the intermediate results in blob storage) and notifies the master (+1)
- Master notifies Worker B that it can get the intermediate results from the blob storage (+1)
- Worker B retrieves the intermediate files from blob storage in readiness for the reduce action. (+1)

4. [4 points] How does the MapReduce framework handle stragglers, and is there a risk of potential conflicts with this approach?

Ans: When a MapReduce operation is close to completion, the master schedules backup executions of the remaining in-progress tasks. The task is marked as completed in the scoreboard maintained by the master whenever either the primary or the backup execution completes. [+2 points]

If multiple tasks complete the same operation, the result is guaranteed to be correct owing to the idempotency of map/reduce operations. [+2 points]

CDN Coral [8 points]

5. [8 points] In the following problem, assume that

• 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 with node-ids 1, 2, 5, 10, 30, 75, 100, 150, 200 in which the "l" value for a node is set to 2; "β" is infinite.

• For simplicity, assume that all puts/gets in this problem using key-based routing go through node-ids: 30, 75, 100, 150, 200 irrespective of the source node of the request.

Charlie wants to use the Coral CDN to store and share a video featuring his adorable cat Apollo.

a. [3 points] To do this, Charlie issues a command Put(200, 10). What do 200 and 10 denote here? Where might this key-value pair be stored?

   Ans: Here, 200 is the content hash of the video (+1), 10 is Charlie's node-id, the node where the video is stored (+1)

   The key-value pair will be stored on a node whose node-id is closest to the key 200. In this case, node-id 200 exists, so the key-value pair will be placed there (+1)

b. [3 points] Charlie's cat video gets popular enough that his friends Alex and Noah offer to host a copy of the video on their nodes. First, Alex issues the command Put(200,1). Next, Noah issues the command Put(200,2). Where will the key-value pair <200,2> be placed? Explain your answer.

   Ans: The key-value pair <200,2> will be placed on node-id 150. (+1)

   +2 if the following sense is conveyed:

   - The request gets routed through node-ids 30, 75, 100, 150, 200
   - Node 200 is an exact match for the key, but it is full after Alex's request, so we go back to node 150 (the next non-full node in the route) and place the key-value pair there.

c. [2 points] Diana hears about Charlie's popular cat video and wants to see it. She issues the command Get(200) from node 5. What value(s) will she get back? Explain your answer.

   Ans: Diana will get back value 2 from node 150 (+1)

   This is because node 150 is the first node in the request's route that has information about key 200 (+1 if this sense is conveyed)

Real-time and Multimedia [12 points]

TS-Linux [4 points]

1. (4 points) Consider the implementation of firm timers in TS-Linux. With succinct bullets, discuss the tradeoffs associated with small and large values of overshoot parameter associated with firm timers.

Answer:

- The timer overshoot parameter allows making a trade-off between accuracy and overhead.
  - A small value of timer overshoot provides high timer resolution but increases overhead since the soft timing component of firm timers are less likely to be effective. (+2)

Conversely, a large value decreases timer overhead at the cost of increased maximum timer latency. (+2)

PTS [8 points]

2. [4 points] Explain two similarities and differences between PTS and Unix Sockets.

Similarities:
- Can be anywhere
- Can be accessed from anywhere
- Network-wide unique

Differences:
- PTS provides temporal causality using timestamps
- There can be multiple producers and consumers of a channel in PTS
- PTS handles both live and historical data

+1 – For each valid similarity
+1 – For each valid difference

3. [4 points] Consider three channels ch1, ch2, and ch3.

Ch1 contains elements with timestamps: 10, 20, 30, 40
Ch2 contains elements with timestamps: 10, 20, 30, 40
Ch3 is empty.

Consider a thread T which executes the following PTS code:

```
<item1,ts1> = Get(ch1, "oldest"); //returns the oldest element from ch1
<item2,ts2> = Get(ch2, "oldest"); //returns the latest element from ch2
Digest = Process(item1, item2); //code to process the two items fetched
Put (ch3, Digest, min(ts1, ts2)); // put digest with timestamp which is the minimum of ts1 and ts2
```

(a) [2 points] What is the timestamp associated with the above Put operation?

Ans: 10

+2 – All or nothing

(b) [2 points] In PTS, is there a way to get corresponding timestamped items from ch1 and ch2 using a single api call instead of invoking Get twice? If yes, explain. (No credit without reasoning)

Ans: Yes, by bundling streams using a group_get api.

+2 – All or nothing

Failures and Recovery [27 points]

LRVM [10 points]

1. (10 points) Consider that a region R has been mapped from disk using the LRVM primitive "map()" to the virtual address space 0x10000000 to 0x1000ffff. The metadata m1 is located at the virtual address 0x10001000 with size of 4 bytes and initial value 10. The metadata m2 is located at the virtual address 0x10005000 with size of 4 bytes and initial value 11. For all the cases discussed below, data is immediately persisted to disk upon flush.

Consider the threads T1 and T2 as described below.

Thread T1:

```
1: begin_transaction(tid1, mode); // no_restore is NOT set
2:    set_range(tid1, 0x10000000 /*addr*/, 0x2000 /*size*/);
3:    m1 = 12;
4:    m2 = 13;
5: end_transaction(tid1, mode); // no_flush is NOT set
6: no-operation;
```

Thread T2:

```
1: begin_transaction(tid2, mode); // no_restore is NOT set
2:    set_range(tid2, 0x10004000 /*addr*/, 0x2000 /*size*/);
3:    m2 = 16;
4: end_transaction(tid2, mode); // no_flush is NOT set
5: no-operation;
```

a. (2 points) Assume T1 is the only thread that is currently running. The application crashes when T1 is at line 6. What is the value of m1 upon crash recovery? Justify your answer. (No points without justification)

Answer: 12. Redo log generated on the disk has the new value for m1.

b. (2 points) Assume T1 is the only thread that is currently running. The application crashes when T1 is at line 6. What is the value of m2 upon crash recovery? Justify your answer. (No points without justification)

   Answer: 11. Redo log generated on the disk at the end of T1 only contains the value of m1. The address and size provided in the set_range call at T1 does not cover m2.

c. (4 points) Assume T1 and T2 are both running concurrently. The application crashes when T1 is at line "6" and T2 is at line "5", respectively. What is the value of m2 upon crash recovery? Justify your answer. (No points without justification)

   Answer: Indeterminate since there is a data race (+1). If line 4 in T1 executes after line 4 in T2, the value of m2 will be 16.(+1). If line 3 in T2 executes after line 4 in T1, the value of m2 will be 16 (+1); if line 4 in T1 executes after line 3 in T2 and before line 4 in T2 the value of m2 will be 13 (+1).

d. (2 points) Assume T1 and T2 are both running concurrently. The application crashes when T1 is at line "4" and T2 is at line "5", respectively. What is the value of m1 upon crash recovery? Justify your answer. (No points without justification)

   Answer: 10. Since T1 has not executed the end transaction yet, the value of m1 will be the initial value (i.e., before the begin transaction).

RioVista [7 points]

2. [7 points] A server is built on top of RioVista that uses multiple data segments mapped to different regions of its virtual address space. The server wishes to make sure that changes to the multiple data segments should be atomic.

(a) [3 points] What steps should it take to ensure the desired atomicity?

Answer:

- Server should enclose ALL changes to virtual memory that affect the mapped data segments within a transaction (+1)
- Server should execute a distinct "set-range" call for the data structures it wishes to modify in each mapped region corresponding to the data segments at the beginning of the transaction. (+2)

(b) [4 points] Does RioVista need to do anything special to ensure the desired atomicity during crash recovery? Justify your answer.

Answer:

- Nothing special needs to be done by RioVista. (+1)
- Check for UNDO logs. (+1)
- Apply UNDO logs to all the affected data segments. (+1)
- Restart the server. (+1)

QuickSilver [10 points]

3. (10 points) A file system is implemented on top of Quicksilver which uses the built-in recovery management. The file system consists of components that execute at each client workstation (denoted by C), a file server (denoted by F) that keeps meta-data for the files, and data servers (denoted by D1 and D2). Client machines are prone to failures while the server machines (F, D1, and D2) are robust and seldom fail. A client workstation C makes a file system call to the file server F. F in turn calls data servers D1 and D2 to satisfy the client request. The above actions result in the creation of breadcrumbs B1 at C; meta-data M1 at F, and intermediate data I1, and I2, at D1 and D2, respectively.

a) (2 points) what would be the structure of the transaction tree for the above call from C?
Answer: C->F->{D1, D2}

b) (2 points) What can be done to ensure the robustness of the file system to ensure proper recovery management?
Answer: Client machine C should designate (one of the robust servers F, D1, or D2) to be the coordinator for a transaction that originates at C.

c) (2 points) What would be logged at each node?
Answer: B1 at C; M1 at F; I1 at D1; I2 at D2

d) (4 points) C designates F as the coordinator for the transaction tree. what should happen if Workstation C crashes and then reboots again?
Answer: The transaction will result in an abort. When the workstation C is rebooted the TM at C will re-join the transaction tree for which F is the coordinator (+1). TM at F will initiate the abort using the transaction tree (+1). The recovery management code in the file system should use the logs (B1, M1, I1, and I2) to restore the file system to the state prior to the crash (+2).

Security [15 points]

Security Principles, AFS [15 points]

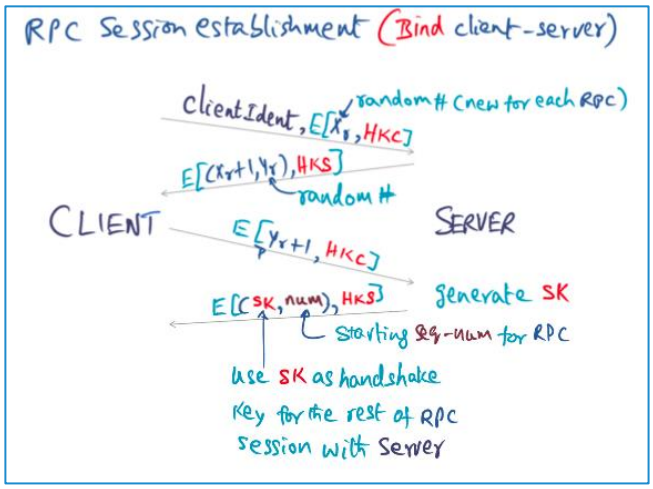
1. [6 points] Assume that you are a student at CMU in the 80s. You walk up to your workstation implemented via the Andrew File System (AFS). Assume that you have been granted a username "usr6210", and you have set your password as "gatech" to login to the system. List the steps taken by Venus during the login process (i.e., what is exchanged between Venus and Vice in plaintext and cyphertext) to set you up for using AFS for RPC sessions for file transfers. For this question, you can use the "bind" protocol for mutual authentication as a primitive abstraction and not describe the message exchanges pertaining to that protocol in the login process.

Answer:

- Venus establishes mutual authentication with Vice using the bind protocol with "usr6210" as the clientident in cleartext, and "gatech" as the key for encryption. (+1)
- Upon completion of the bind protocol, Vice generates a pair of tokens {secrettoken, and cleartoken} using the authentication server. (+1)
- Vice encrypts the tokens {secrettoken, and cleartoken} with the password ("gatech") and sends the cyphertext to Venus. (+1)
- Venus decrypts using the password ("gatech")and extracts the tokens. (+1)
- Venus extracts the handshake key (HKC) from the cleartoken which is a data structure known to Venus and Vice. (+1)

- Venus will use the bitstring "secrettoken" as the clientident, and HKC as the key for the duration of the login session. (+1)

2. [9 points] Shown below is the bind protocol which is at the core of mutual authentication.



(a) [3 points] Explain the significance of Xr, Yr, HKC, HKS presented in diagram with respect to the client and server authenticity.

The client sends a challenge to the server in the form of a random number Xr in the encrypted cipher using the HKC. Only the genuine server will be able to decrypt the message and extract the random number. It shall then increment the number, encrypt it using HKS and send it back to the client. This helps the client verify the authenticity of the server. (+1)

Similarly, the server sends a challenge to the client in the form of random number Yr encrypted with HKS. Only the genuine client would be able to decrypt the number, increment it and send it back to server by encrypting it with HKC. This establishes the genuineness of the client to the server. (+1)

By design HKC = HKS. (+1)

(b) [2 points] Alex is in the middle of an RPC session with AFS. During this RPC session Alex sends a request to read a file. What key will be used Venus on behalf of this request?

Answer: Session key SK. To reduce over-exposure of HKC, Vice provides SK for use in this RPC session.

(c) [2 points] Mallory sniffs Alex's read request and replays it AFS setting the source IP address to her machine. Her message arrives at the server ahead of Alex's message. Will the file system security be breached? If not, explain what exactly will happen that ensures the intended security?

Answer: AFS will process the request as a legitimate request and return the result to Mallory's machine. However, she cannot read the file contents since she does not know the session key SK to decrypt the message.

(d) [2 points] Alex's message arrives subsequently at AFS. What will AFS do?

Answer: AFS will recognize this is duplicate request since the sequence number in the message will be the same. It will assume its response was lost. It will retransmit the response to Alex's machine and all is well.

Potpourri [18 points]

1. [2 points] (Answer True/False with justification) Your friend is submitting a map reduce job. Her input data is split into 100 shards. Ignoring set up time, the actual time for the map function to execute on an input shard is T. To speed up the map phase, she asks for 200 nodes. With this allocation, the map phase on the entire input will complete in T/2 time units.

ANS: 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.

2. (6 points) (Answer True/False with justification)

   a. (2 points) By favoring scheduling latency for time-sensitive tasks, TS-Linux could starve throughput-oriented tasks from making forward progress.

   Answer: False.

   TS-Linux uses proportional period scheduling for admission control and this allows it to reserve a portion of the time for throughput oriented tasks.

   b. (2 points) The primary overhead associated with APIC-one-shot timers is timer reprogramming

   Answer:

   False.

   With modern hardware, reprogramming a timer takes only a few

cycles. The primary overhead for the one-shot timing mechanism in firm timers lies in fielding interrupts

c. (2 points) Tuning the overshoot parameter associated with firm timers to zero is equivalent to using soft timers. (No credit without justification)

Answer:
False.
By tuning the overshoot parameter to 0 we obtain one-shot timers and tuning it to a large value gives us soft timers.

3. [2 points] Answer True/False with justification

RioVista's use of battery-backed file cache enables it to resume transactions from the exact point of failure after a crash, without any data loss or state inconsistency.

- Answer: False. While RioVista's battery-backed file cache does ensure the persistence of UNDO logs through crashes, thus preventing data loss and maintaining state consistency, it does not enable the resumption of transactions from the exact point of failure. After a crash, RioVista does not resume the transactions but starts the server afresh, initiating a new instance of the server. This ensures that all systems are initialized correctly, and that the data integrity is maintained without complications from the interrupted transaction state.
- Rubric: +1 for correct answer (False), +1 for justification: Despite the resilience of the UNDO logs due to the battery-backed file cache, RioVista does not resume transactions from the point of interruption but restarts them to ensure the consistency of data segments.

4. [2 points] (Answer True/False with justification) An application using the built-in recovery management of Quicksilver would incur additional communication overhead.

Answer: False, no additional communication overhead is incurred since setting up the transaction graph for recovery management is piggybacked on top of the normal IPC communication between the clients and the servers.

5. [6 points] Assume that AFS is implemented using a public-key encryption system.

a. (2 points) (Answer True/False with justification) AFS will use the same public key for all the users in the system.

Answer: False. AFS will generate a unique {public-key, private-key} pair for each user.

b. (2 points) (Answer True/False with justification) Venus never has to send anything in cleartext to Vice.

Answer: False. Venus will use the public-key of the user as the client identity in cleartext for all communication with Vice.

c. (2 points) (Answer True/False with justification) Person-in-the-middle attack that breaches the security of the system is easy with the public-key encryption system since the public key is exposed.

Answer: Although public-key is used as the client identity for message exchange between Vice and Venus (and thus over-exposed), person-in-the-middle cannot decrypt any message since they don't have the associated private-key which is needed for decryption thanks to the one-way functions.