# CS 6210 Spring 2013 Final Solution

Name:_____Kishore___GT Number: _____

**Friday May 3, 2011 (8:00 AM to 10:50 AM)**
**Expected time to finish the exam: 100 mins**

**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. Where appropriate **use figures** to convey your points (a figure is worth a thousand words!)
6. **Illegible answers are wrong answers. WE MEAN IT!!**
7. Please look through the whole test before starting so that you can manage your time better.

Good luck!

| Question number | Points earned | Running total |
|---|---|---|
| 1 (Max: 1 pts) | | |
| 2 (Max: 19 pts) | | |
| 3 (Max: 20 pts) | | |
| 4 (Max: 10 pts) | | |
| 5 (Max: 20 pts) | | |
| 6 (Max: 20 pts) | | |
| 7 (Max: 10 pts) | | |
| Total(Max: 100 pts) | | |

**1.** (1 min, 1 point) The academic **great grandfather** of the instructor for this course (Don't worry you get a point irrespective of your answer!)

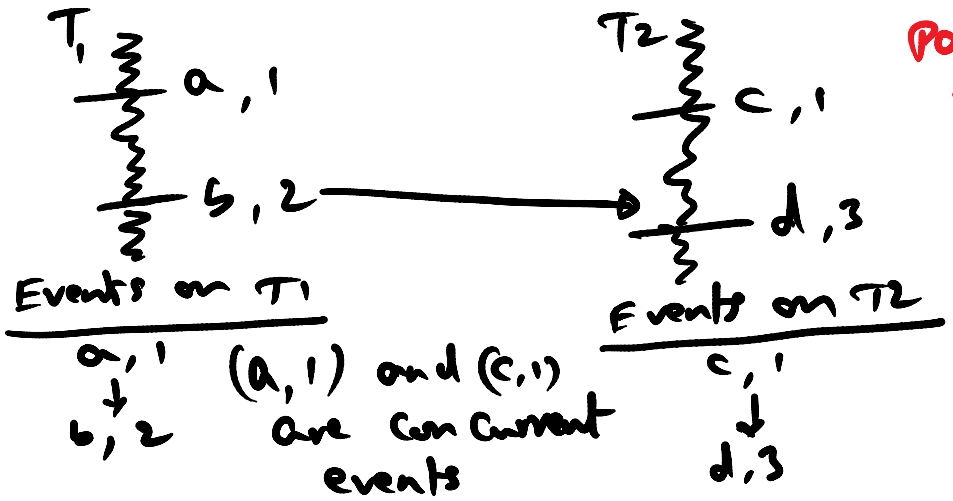
a. Marvin Solomon
b. Art Bernstein
c. Jeff Ullman
d. Alan Demers
e. Prasun Dewan
f. (your guess if you don't think any of the above is right)

# CS 6210 Spring 2013 Final Solution

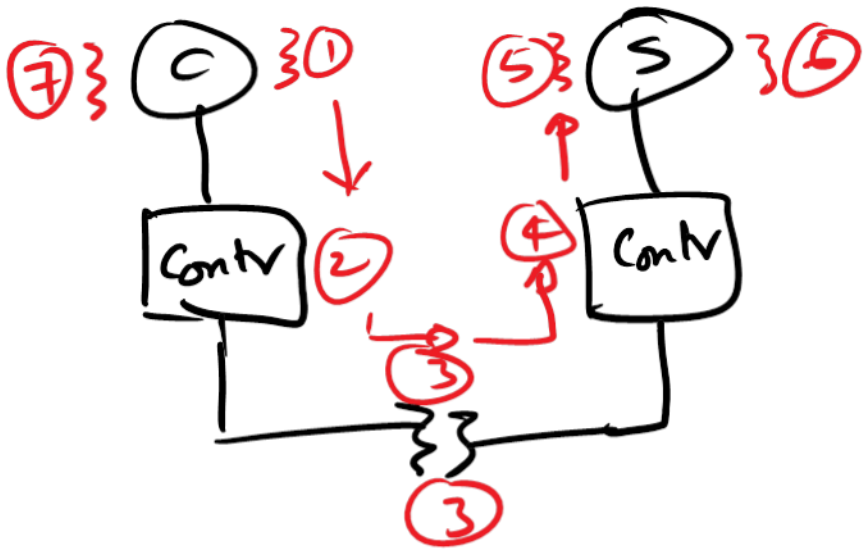
Name:_____Kishore___GT Number: _____

**2. Comm. and Synch. in Distributed Systems (19 min, 19 points)**

(a) (5 points) Explain (with a simple example) why Lamport's "happened before" relationship by itself is insufficient to derive a total order of events in a distributed system.




(b) (7 points)This is with reference to the paper by Thekkath and Levy on limits to low latency communication. For the RPC call shown between client (C) and server (S) in the picture below, give a short one/two sentence description for each of the component times:




Use next page for writing your answer

# CS 6210 Spring 2013 Final Solution

Name:_____Kishore___GT Number: _____

1. Client call:

Args set up; kernel call; validation; marshaling; controller set up

**(+1 if general idea)**

2. Controller latency:

At each of client and server:
Time taken by hardware to move bits from host memory to controller buffer (if necessary); followed by placing the bits from buffer onto the wire

**(+1 if general idea)**

3. Time on the wire:

Actual time taken by the bits to travel from client machine to server machine and vice versa

**(+1 if general idea)**

4. Interrupt handling:

Time taken by the kernel to dispatch the interrupt handler after saving context of currently running process and the time taken by the controller to receive to receive the bits from the wire into its buffer

**(+1 if general idea)**

5. Server call receipt:

Time for kernel to locate server procedure; dispatch server procedure; unmarshall args

**(+1 if general idea)**

6. Server reply:

Exec of server procedure; setup reply similar to client call

**(+1 if general idea)**

7. Client reply receipt:

Dispatch client (currently waiting for RPC to complete)

**(+1 if general idea)**

# CS 6210 Spring 2013 Final Solution

Name:_____Kishore___GT Number: _____

(c) (4 points) The idea of Active Networks is to "customize" network service to packets flowing through routers. Mention two pitfalls with this idea.

- Protection: can one service damage another (either intentionally or unintentionally)
- Resource management: can one service consume arbitrary amount of resources (e.g., network bandwidth) to the detriment of others

**(+2 for each of the above points;**
**If other points raised then points based on validity of the points)**

(d) (3 points) This is in the context of Ensemble/NuPrl approach to combining theory and practice to synthesize communication protocol stacks. Mention two points that capture the goals of this approach.

- Decouple specification, verification, and implementation from one another
- Design, test, optimize individual components (this is the practice part achieved via Ensemble)
- Use theoretical framework (achieved via IOA) to go from specification to synthesizing an un-optimized stack consisting of the components provided by Ensemble
- Use theoretical framework (NuPrl) to optimize the stack

**(+3 if any two of the above points mentioned;**
**If other points raised then points based on the validity of the points)**

**3. DSM and FS (20 mins, 20 points)**

(a) (5 points) (**GMS**)
Node P has a page fault.

- The oldest page in P's local cache is **X with a timestamp of 40.**
- The oldest page in P's global cache is **Y with a timestamp of 30.**

Which page does P evict and why?

P will evict Y (the oldest page from the global cache). **(+2)**
Reason: page fault on P is indicative of the working set of P growing, so expand the local cache and shrink the global cache. **(+3)**

# CS 6210 Spring 2013 Final Solution

## Name:_____Kishore___GT Number: _____

(b) (7 points) (**Treadmarks**)
Consider a shared memory multiprocessor. On each processor, **read/write to memory is atomic.** But there is no guarantee on the interleaving of read/writes across processors.

A parallel program is running on processors P1 and P2.

Intent of the programmer:

| P1 | P2 |
|---|---|
| Modify Struct(A) | Wait for modification<br>Use Struct(A) |

The pseudo code that the programmer has written to achieve this intent:

flag = 0; initialization

| P1 | P2 |
|---|---|
| mod(A);<br>flag = 1; | while (flag == 0); //spinwait<br>use (A);<br>flag = 0; |

(i) (4 points) Will the above code work? If not why not?

The above code is not guaranteed to work. **(+1 if they get this right)**

Reason: read/writes from P1 happen in textual order. But since there is no guarantee on the order in which writes from a given processor become visible to the other, P1's write to flag may become visible to P2 before the modifications to A by P1 have become visible to P2. This will result in violating the intent of the programmer.

**(+3 if they get the reason right; partial credit commensurate with the reasons)**

(ii) (3 points) What is the guarantee needed from the memory system to make this code achieve the programmer's intent?

Sequential consistency:

- Read/writes on each processor respect the program order
- Reads/writes from the different processors interleaved consistent with the program order on individual processors

**(+3 if they say SC; or +1.5 for each of the above two bullets)**

# CS 6210 Spring 2013 Final Solution

Name:_____Kishore___GT Number: _____

(c) (8 points) (**xFS**)
xFS builds on three background technologies. What problem does each of these background technologies set out to solve?
(i)

1. Hardware RAID

- Increase the bandwidth for file access by striping file to several parallel disks

**(+2 if they say this)**

2. Log structured file system

- Solves the small file write problem

**(+2 if they say this)**

3. Zebra file system

- Hardware RAID expensive => use software RAID
- Combines the advantages of both LFS and RAID

**(+2 if they say the second point)**

(ii) Mention two new contribution made by xFS beyond these background technologies.

- Scalable implementation of metadata management by decoupling the location of the file from the metadata management
- Client-to-client cooperative caching
- Subsetting storage servers used for striping
- Distributed log cleaning

**(+2 if they say any two of the above; other points can be considered if valid)**

# CS 6210 Spring 2013 Final Solution

Name:_____Kishore___GT Number: _____

**4. RT and MM (10 min, 10 points)**

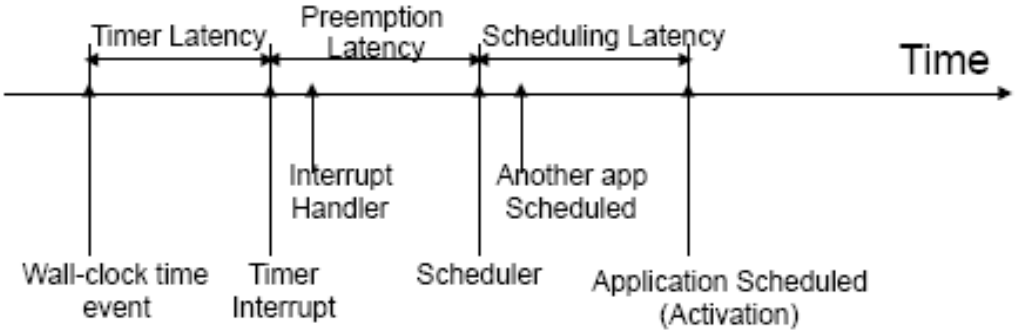
(a) (6 points) (**Time Sensitive Linux**)

Define the following terms(use figures to help get your point across)

(i) "timer latency"

(ii) "preemption latency"

(iii) "scheduling latency"




- Timer latency: distance between event occurrence and timer interrupt due to the granularity of the timing mechanism
- Preemption latency: distance between timer interrupt and opportunity to schedule the event due to activity on the CPU being non pre-emptible (e.g., kernel in the middle of an interrupt handler)
- Scheduling latency: distance between when the event becomes schedulable and when it actually gets scheduled due to higher priority applications already in the CPU scheduling queue

**+2 for each highlighted phrase or equivalent**

(b) (4 points) (**PTS**)

(i) Mention two similarities of the PTS channel abstraction to Unix socket.

- PTS channels globally unique
- PTS channels can be located anywhere
- PTS channels can be accessed from anywhere

**(+2 if any two of the above points; other points can be considered if valid)**

(ii) Mention two differences of the PTS channel abstraction to Unix socket.

- Items in channels temporally indexed
- Many to many connection from threads to channels
- Automatic garbage collection of items from channels

**(+2 if any two of the above points; other points can be considered if valid)**

# CS 6210 Spring 2013 Final Solution

Name:_____Kishore___GT Number: _____

**5. Failures, consistency, and recovery (20 mins, 20 points)**

(a) (10 points) (**LRVM**)

(i) LRVM uses "No-undo/redo value logging". Explain what this means.

- Undo record only in memory, not persisted on the disk **(+2)**
- Redo logs are persisted on the disk **(+2)**

The undo records are in memory only for the duration of the transaction. At the end of the transaction if the transaction commits, they are discarded. If the transaction is aborted, then the original content of the memory is restored from the undo records. **(+1)**

The redo log is committed to disk at the end of the transaction, i.e., a redo log record is written to the disk followed by a commit record if the transaction actually commits. If the transaction aborts, then the redo log is not written to the disk. **(+1)**

(ii) A subsystem using LRVM crashes. Explain the steps by which the subsystem recovers to a stable state upon restart.

- Read the log from the disk into memory **(+1)**
- Starting from the end of the log, scan backwards till you get to a commit record. This represents the last commit before the system crashed. **(+1)**
- Now roll forward from the beginning of the log applying the changes from the log to the data segments. **(+1)**
- Stop processing the log when the last commit record is encountered in the log. **(+1)**

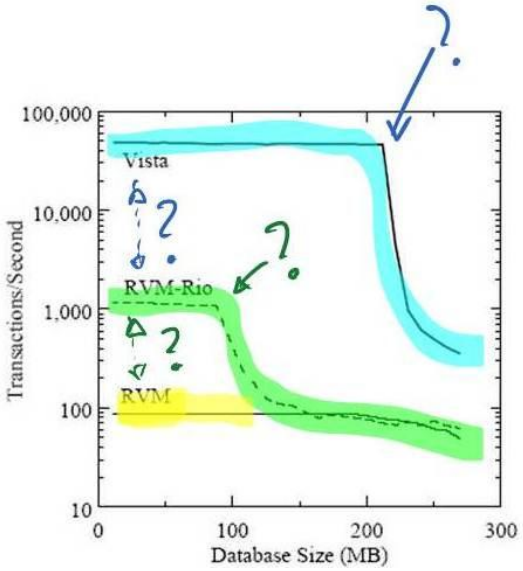
**(2 points for a generic description without all the details)**

# CS 6210 Spring 2013 Final Solution

Name:_____Kishore___GT Number: _____

(b) (4 points) (**Rio/Vista**)
The following figure gives comparative performance of reliable virtual memory on a machine that has 200 MB of memory. The x-axis gives the size of the DB, and the y-axis gives the throughput (higher the better).

- RVM – Same as original LRVM by Satya
- RVM-Rio – Faithful implementation of Satya's LRVM system using the Rio file cache
- Vista – Native implementation of the LRVM API using the Rio file cache




(i) Explain the disparity in performance between RVM-Rio and RVM

RVM does disk I/O at commit point to write the redo logs to disk; RVM-Rio does not.
**(+1)**

(ii) Explain the disparity in performance between RVM-Rio and Vista

Vista does not write redo logs at all; it modifies the DB in place since there are undo log is in the file cache if in case there is a crash before commit the old image of DB can be restored from the undo log.
**(+1)**

(iii) Explain the knee in the curve for RVM-Rio

The machine has 200 MB, and since RVM-Rio does double buffering (redo logs + DB), thrashing starts when size of DB exceeds 100 MB.
**(+1)**

(iv) Explain the knee in the curve for Vista

Since Vista does not write redo logs, the thrashing effect starts when the size of the DB exceeds total memory size (200 MB).
**(+1)**

# CS 6210 Spring 2013 Final Solution

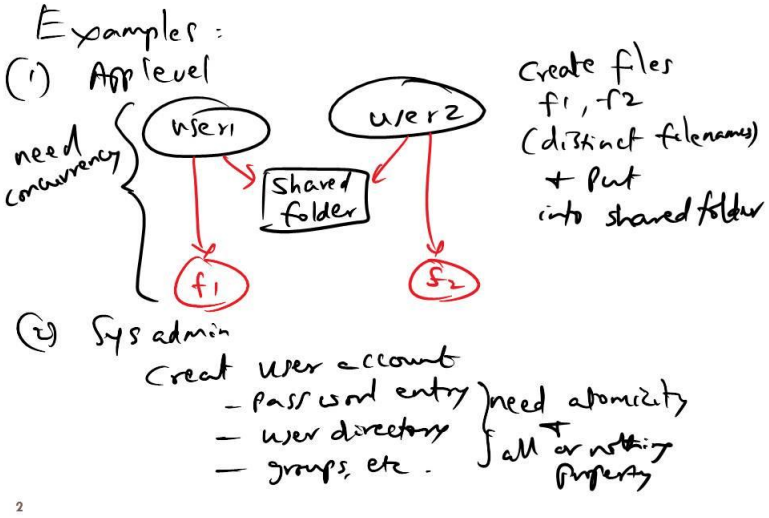
Name:_____Kishore___GT Number: _____

(c) (2 points) (**Quicksilver**)
Give the fundamental reason for including transaction as a first class citizen in the Quicksilver OS.

Since almost all subsystems of an OS (window manager, file system, protocol stack, etc.) need to recover gracefully from failures (power and software crash), Quicksilver decided to make transaction a first class entity.

**(+2 if the above stated; partial credit depending on answer)**

(d) (4 points) (**TxOS**)
Give one user level example and one system level example where transactional semantics will help in increasing concurrency while ensuring atomicity and isolation for the actions.




- User level:
  - User1 and User 2 can create f1, and f2, respectively, concurrently.
  - They need atomicity and isolation to update the shared folder into which they want to place the files.
- System level
  - Need all or nothing property to all the actions shown pertaining to an account creation

**(+2 for each example)**

# CS 6210 Spring 2013 Final Solution

Name:_____Kishore___GT Number: _____

**6. Internet Scale Computing (20 mins, 20 points)**

(a) (4 points) (**Map/Reduce**)
Mention at least 4 important "heavy lifting" work done by the map-reduce infrastructure unbeknownst to the user of the programming model (use a figure if it makes easier to get your points across).

- Automatic splitting of input into key-value pairs
- Spawning worker threads to execute parallel maps
- Spawning worker threads to execute parallel reduce
- Managing the scheduling of the worker threads
- Plumbing intermediate results from the map to reduce worker threads
- Managing machine failures and re-execution of failed map/reduce functions

**(+1 for each of any of the above points upto 4 max; other points also considered if valid)**

(b) (6 points) (**Giant-scale services**)
Explain the pros and cons of the three approaches to online evolution of giant-scale services:

(i) Fast reboot

Pro: whole datacenter up in a short amount of time; especially works well in exploiting diurnal server property for geographical datacenters across the globe **(+1)**
Con: No service (DQ = 0) for the duration of the reboot time **(+1)**

(ii) Rolling upgrade

Pro: graceful degradation of service (DQ never goes to zero) **(+1)**
Con: takes a long time for the datacenter to be fully functional again **(+1)**

(iii) Big flip

Pro: better than fast reboot in terms of availability (DQ is exactly 50% for the duration of total reboot time) **(+1)**
Con: worse than fast reboot in the time it takes for the datacenter to be fully operational **(+1)**

(c) (10 points) (**Coral**)
(i) (2 points) Explain "full" and "loaded" attributes of nodes participating in Coral

Full: the node already has a predefined number of "values" associated with a given key (basically helps in statically dispersing the meta data management for popular content)
**(+1)**
Loaded: the node is already handling a predefines number of queries per unit time for a given key (basically helps in dynamically dispersing the meta data management for popular content)
**(+1)**

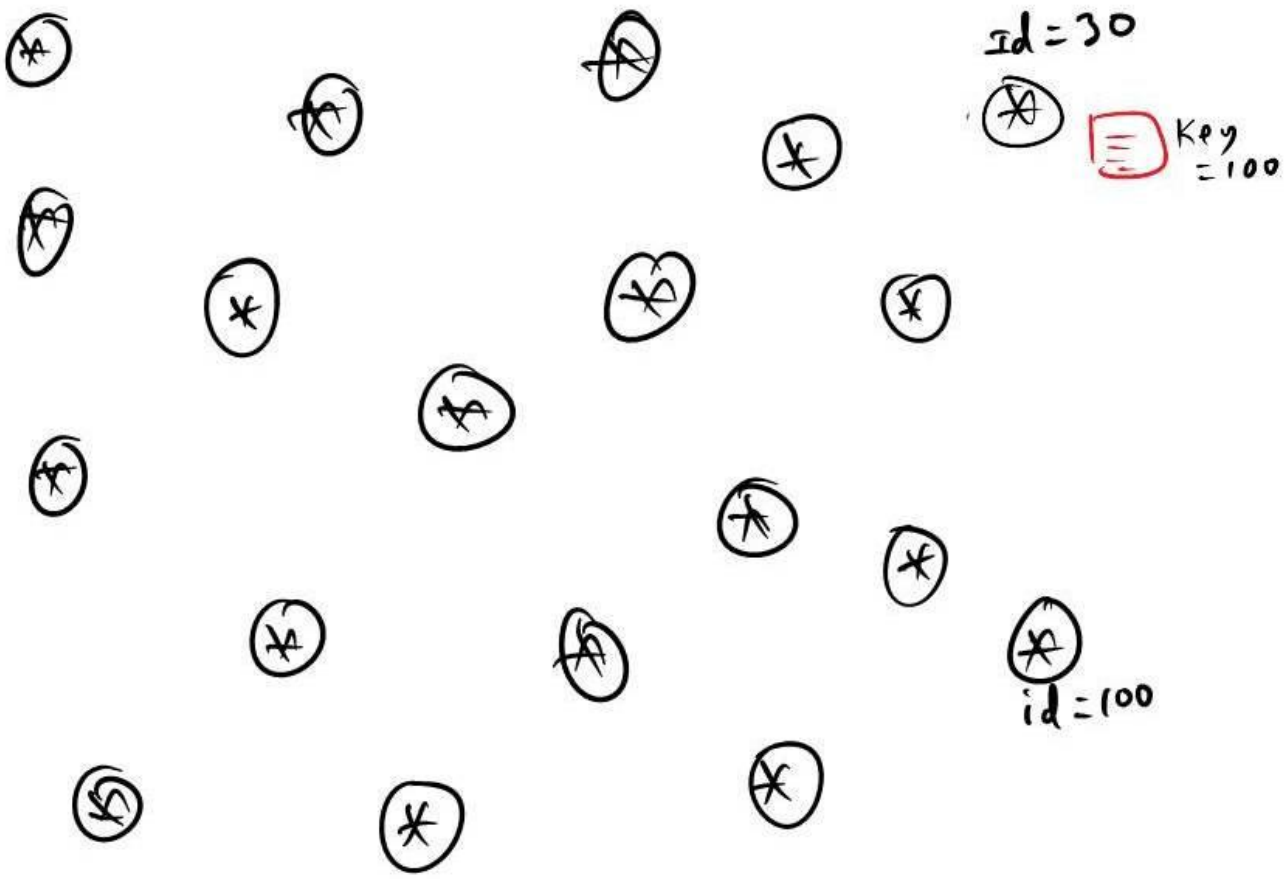
# CS 6210 Spring 2013 Final Solution

Name:_____Kishore___GT Number: _____

(ii) (8 points) Given the following:

- Node id = 30 generates content with a hash value of 100
- A node can hold at most **1 value for a key,**

The initial state of the coral network is shown below:




Show with a series of web accesses (resulting in get/put for <key, value> and content accesses) for the content given by the key 100, how Coral avoids origin server and meta server overload.

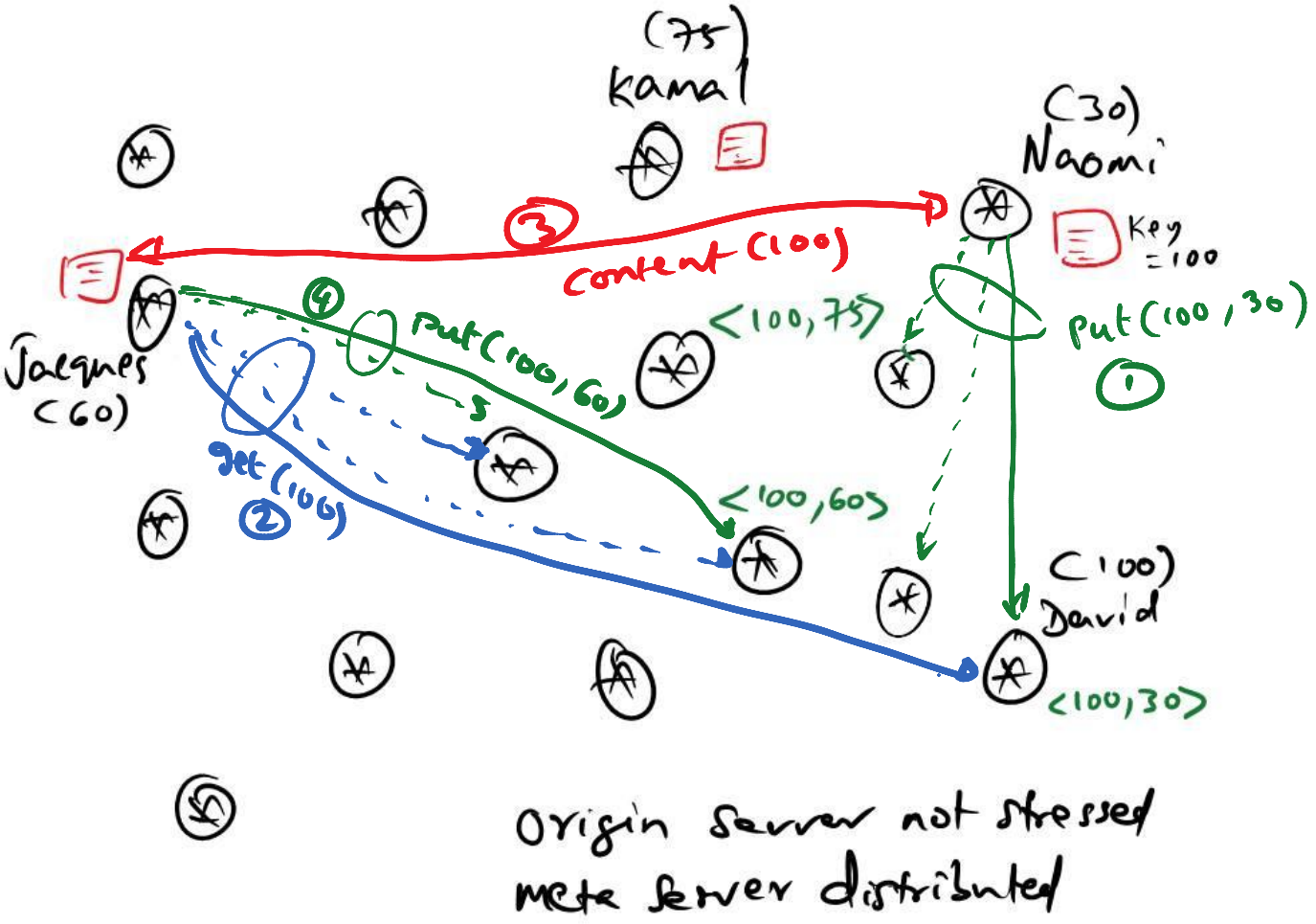
- Feel free to assign IDs to the nodes shown in the above picture
- Show the evolution of the figure by labeling the changes due to the web accesses with numerical values corresponding to the numbered web accesses you generate.

**(2 points if general idea but not specific to the question)**
**(3 points if description is unclear as to what is going on)**
**(4 points if halfway there)**
**(7 points if put is not preceded by get)**
**(8 points if answer complete – shows content and metadata dispersion)**

# CS 6210 Spring 2013 Final Solution

Name:_____Kishore___GT Number: _____

(Extra space for Q6 (c)(ii))




Series of get/put requests that results in the above evolution:

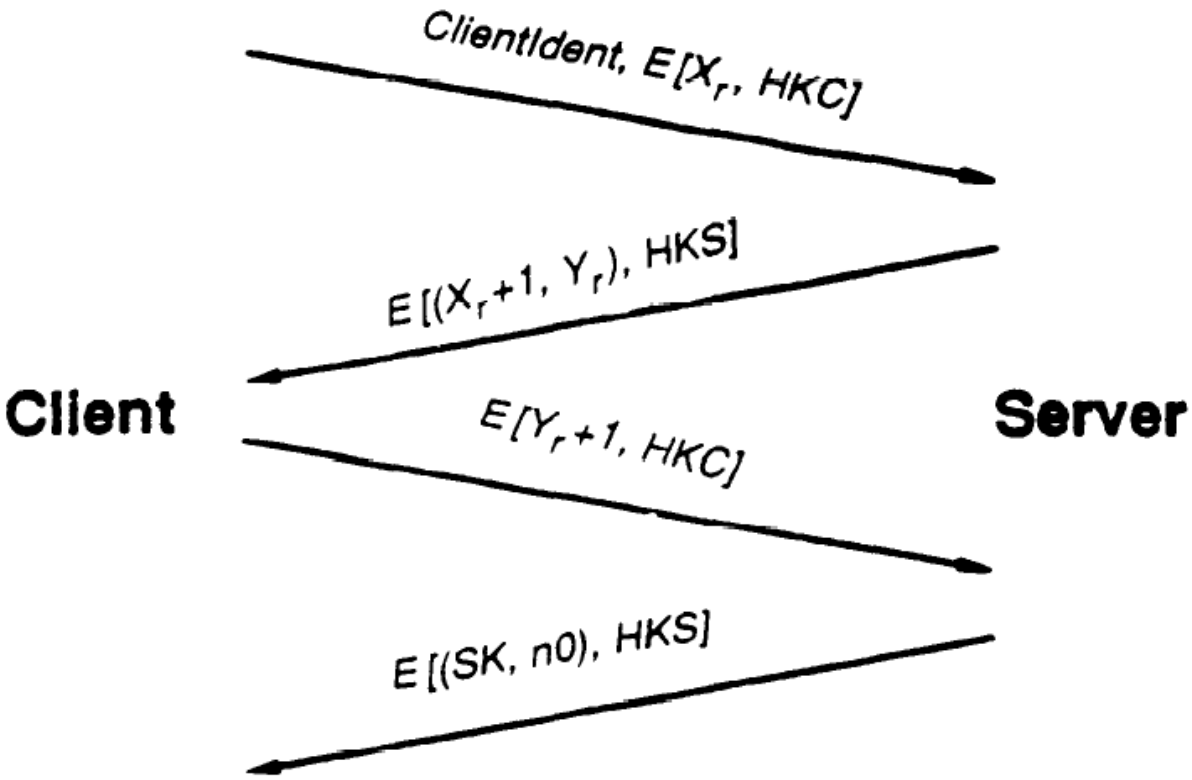
- Naomi: put(100, 30): stored in node-id = 100 (1)
- Jacques (node id = 60):
  - get(100): returned node-id 30 (2)
  - download content from node-id 30 (3)
  - put (100, 60): this is stored into a new meta server en route (4)
- Kamal (node id = 75):
  - get(100): returned node-id 60 (5)
  - download content from node-id 60 (6)
  - put (100, 75): this is stored into a new meta server en route (7)

**(partial credit commensurate with the answer)**

# CS 6210 Spring 2013 Final Solution

Name:_____Kishore___GT Number: _____

**7. Security (10 min, 10 points)(AFS)**




With respect to the above client-server interaction in the Andrew File system, answer the following questions:

(a) (2 points) At what point does the client know that it is talking to the genuine server?

When the client receives the first message from the server E[(Xr+1, Yr), HKS], it will check the field Xr+1. Note that Xr was generated by the client and only if the server was able to decipher the message, Xr+1 field will have the right value. A replay attack will not have the right value for this field.

**(+2 if they say this; partial credit as appropriate)**

(b) (2 points) At what point does the server know that it is talking to a genuine client?

When the server receives the second message from the client E[Yr+1,HKC],it will check the field Yr+1. Note that Yr was generated by the server and only if the client was able to decipher the message, Yr+1 field will have the right value. A replay attack will not have the right value for this field.

**(+2 if they say this; partial credit as appropriate)**

# CS 6210 Spring 2013 Final Solution

Name:_____Kishore___GT Number: _____

(c) (2 points) What does a user who walks up to a workstation use as "ClientIdent" to start her interaction with the server, and what is the key used to encrypt the initial startup message?

ClientIdent will be the user's login "username" and the key for encryption will be the "password" associated with this username (this is known to the user and the system every authorized user of the system).

**(+2 if they say this; partial credit as appropriate)**

(d) (2 points) Is "ClientIdent" sent in plaintext or encrypted? Why in either case?

- "ClientIdent" is sent as plaintext. **(+1)**
- Since the system uses private key encryption, the server needs to know the identity of the requestor to choose the right key for decryption. **(+1)**

(e) (2 points) What is SK? What is its purpose?

- SK is new "session key" generated by the server for the new RPC session that the client has requested. The new RPC session will use SK as the encryption key.
  **(+1)**
- Generating a new SK for each RPC session ensures that the handshake key (HKC) contained in the cleartoken is not over-exposed on the wire during the login session.
  **(+1)**