Name:

**CSE 291 (Distributed Systems)**

**Winter 2017 (Kesden)**

**Homework #1**

1. **Networks**
2. Given a 100Mbps network with an optimum sliding-window size of 1024 bytes, what would be the optimal window size if the bitrate was increased to 5Gbps and the maximum allowable run of the cable was cut in half? Why?

Opt Window Size = RTT \* throughput, RTT = 1024 bytes / 100 Mbps

new Opt Window Size = RTT \* new throughput = 1024 bytes / 100 Mbps \* 5 Gbps / 2 = 25600 bytes, because the cable length does not changed, the speed of propagation does not change, RTT does not change.

1. When transmitting at the link layer, we normally “frame” data. Why? For example, why do we not just stream bits?

First, a transmitter needs to let its neighbor receiver know where is the start of the data rather than noise or just electricity. Second, the receiver needs to ensure that the information received are correct, which means checksum is needed. Third, some information (header) are needed such as destination ip address to let the receiver know how to deal with the data. Fourth, in case of large data transfer, the error could be detected earlier to improve efficiency.

1. Which is often a bigger concern within a data center? Bandwidth or latency? What about long-haul? Why?

Bandwidth is often a bigger concern within a data center. If a data center does not have enough bandwidth to broadcast all its systems simultaneously, the update at one place will be noticed by others with different delays. If a request happens during this period, it will leads to consistency problem.

In a long haul, latency is a bigger concern. In long haul, we care more about the speed of transmission of the message, there is no different parties sharing some information which need to be informed upon update simultaneously.

1. **Middleware/RPC/RMI**
2. The RMI we’re implementing for class uses a different proxy instance for each remote object instance. Could this be reduced to one proxy instance per client JVM? Or per remote object type? If so, please sketch a new design. If not, why not?

Yes, we can have a data structure in the stub proxy to record the remote object we have called. If in a new call, the parameter matches that record, the stub can invoke the remote methods of that object.

1. Consider the implementation of an RPC system in a heterogeneous environment (different processor architecture, different OS, different programming language, different, different, different). How might the heterogeneity complicate the model? Please consider each of the following:
   1. Simple primitives (consider endian-ness, width, etc)
   2. Complex data types, including structs and strings,
   3. Higher-order language and library data structures, such as linked lists, maps, etc.
   4. Programming paradigms (function pointers, jump table, functors, etc)
2. Some system may use big endian, some may use small. Different systems have different ways to represent value. The network may also use a different representation of data with system.
3. Different language and systems may use different conventions for data representations, such as null termination of strings, memory allocation for data type. When it comes to OOP, the problem becomes more complicated.
4. The standard library may be different for different languages. The implementation of data structures may also be different.
5. Function pointers in different languages are different and hard to convert to each other.
6. Consider Java’s RMI facility, which generates stubs at compile time. Could it, instead, generate the stubs at runtime? For example, could it disassemble a class file, or inspect an object’s properties at runtime, rather than at compile time? If not, why not. If so, what would be the advantages and disadvantages of this model?

Yes. The advantage is that it is clean, prevents version problem and can be dynamic. The disadvantage is that it will make the runtime longer and reduce the efficiency.

1. Consider the design of an RMI facility which supported the migration of clients. Please sketch out a design.

When a client wants to migrate, it will send a request to the server with an identification number. When the server receives the request, it will save the status of the client. When the client is moved to another host, it will send a request to the server to continue with a paired identification number. When the server receives the request and the identification number checks, it will continue the process.

1. **Distributed Concurrency Control**
2. Please describe a situation in a distributed system in which is does not make sense to use a resource queue located at a remote resource to enforce mutual exclusive access to that resource well as explain why it doesn’t make sense.

This kind of centralized approach is not scalable. Also, if a thread dies in critical section, the whole system will be blocked.

1. Under heavy contention, which requires more messages per request to enforce mutual exclusion: a central queue or token ring? Why?

Token ring. Token ring passes the mutual exclusion lock around. When the contention is heavy, there is nearly no wasted message. Every mutual exclusion call will have exactly one message pass. A central queue needs at least three messages per call.

1. The *voting protocol*, and the *voting district* protocol, are based upon participants reaching an agreement as to who can enter the critical section. What happens in the event of a tie that could otherwise risk deadlock?

Once a node notices that it has vote the wrong candidate, it will send an inquiry message to withdraw its vote. If the candidate is in the critical section, there is no deadlock. If the candidate is still waiting, it will give back the vote with a relinquish message.

1. Many coordinator election protocols have analogous mutual exclusion protocols and vice-versa. Please identify one mutual exclusion mechanism for which this is not the case, and explain why it is not the case.

The centralized approach algorithm for mutual exclusion has no analogous coordinator election algorithm. Because coordinator election is a distributed process, it cannot be done in a centralized way.

1. In the event of a partitioning, techniques such as token ring can result in two or more distinct groups, each with its own coordinator. Please give one example illustrating a situation in which this could be a bad thing and another illustrating a situation in which it makes sense.

If two groups only read from the same resources or they do not share any resources across group, it is okay for them to have their own coordinator. If they want to write to some shared file, multiple coordinators will cause a problem.

1. **Logical time**
   * 1. One form of logical time is per-host sequence numbers, e.g. “5.1” is time 5 on host 1. Another form is Lamport’s logical time. What advantage does Lamport time offer? At what cost?

Lamport time can share logical time across hosts and provides a “happens before” relationship to order events. The hosts need to send the timestamp along with messages to synchronize their logical time.

* + 1. Please define both *causality* and *causality violation.*

Causality: An event e1 “happens before” another event e2 if and only if the occurrence of e1 may potentially affect the occurrence of e2. This relationship is called causality.

Causality Violation: A message ordering problem results in one host taking an action based on information that another host has not yet but should have received.

1. **Replication and Quorums**
2. Please describe a locking protocol for use upon write via write-one/read-all quorum with version-numbered objects.

To write to a file, the process must first require a read quorum and then require one write quorum to write the file and advance the version number.

1. How would the protocol you described in 5(a) change if the objects were not version numbered, if at all? Why?

The write process needs to ensure that it writes at least half of the reading file to ensure that the reading process knows which version is the latest. Because the reading process read all files, the writing process needs to write more than half of the replicas.

1. Coda Version Vectors (CVVs) are a form of logical time stamp used to aid in the management of replication. What capability do they enable that would not be available with simple, per server, scalar version numbers? Why can’t simple scalar version numbers support this capability?

They are able to detect conflict. Under normal circumstances, the CVV should be less than or larger than each other or exactly the same. When there are two CVVs forming a conflict, some process writes to the wrong version. For example, <1,1,2,2> and <2,1,1,2> forms a conflict.

Simple version number cannot infer a conflict, it will always satisfy less than, larger than or equal relationship.

1. **Agreement, and the Impossibility Of**
2. Consider content servers located at three distant points across the globe. Can we devise a system to ensure that, at any point in time, *exactly* one of them is visible to each client? Please assume that no more than two are unavailable at any point in time, and that it is possible for more than one to be available to provide server at any point in time. Why or why not? *Hint: Consider the network, all its pieces, and all its failure modes.*

Yes. Say the three servers are S1, S2 and S3. A simple algorithm would be, if S1 is on, only S1 is visible. Once S1 is down, S2 will be visible. Only when S1 and S2 both are down, S3 will be visible.

1. Assume that we have two magical networks. The first, a *Worm Hole*, truly has zero latency. But, unfortunately, each time a message is sent, it is lost with some probability *p*. The second, *Perfect Fiber*, has a latency directly proportional to its length, but never loses or corrupts a message, and always delivers messages in order. Reliable protocols, such as TCP, are available for use on either network. Which network has a faster *guaranteed* delivery time? Why?

Depends on the file size (S), bandwidth (B) and the fiber length (L) and the value of p. The speed of light is c.

time(Worm Hole) = S / B / p

time(Perfect Fiber) = S / B + L / c

When p > 1 – (L\* B) / (c \* S), perfect fiber is faster. Otherwise, worm hole is faster.