

CS 194 – The Design and Implementation of Network Services

Instructor: George Porter UC Berkeley

Spring 2005

Your Name: _____

This exam will last 50 minutes

Problem	Point Value	Points
1	35	
2	30	
3	35	

Problem 1 – “Communication APIs”

Consider the following code fragment:

```
int average(int a, int b)
{
    float retval = sum(a,b) / 2;
    return retval;
}
```

Now let's say that we make **sum(a,b)** a remote procedure call. The implementation of **sum** resides on a remote server.

1.1) What is a “stub” call? Why is it needed? (5pt)

1.2) Describe argument marshalling, and explain how it would work in this example. (10pt)

1.3) How does making **sum** an RPC change the semantics of the average function? (10pt)

1.4) Put the following in the right order: (10pt)

- _____ Client procedure calls client stub in normal way
- _____ Client's OS gives message to client stub
- _____ Server does work, returns result to the stub
- _____ Stub unpacks result, returns to client
- _____ Client stub builds message, calls local OS
- _____ Remote OS gives message to server stub
- _____ Client's OS sends message to remote OS
- _____ Server's OS sends message to client's OS
- _____ Server stub unpacks parameters, calls server
- _____ Server stub packs result in message, calls local OS

Problem 2 – “Threading and Concurrency”

Unix allows users to write very simple network services through the **inet.d** mechanism. To do this, a user writes a standard executable program that reads and writes to stdin/stdout, and registers this program with a specific server port. When a new network connection comes in, the O/S sets the network socket up to read and write to stdin/stdout. Each time a new connection comes in, the O/S starts a new process to handle it.

2.1) Why would this approach lend itself to ease of debugging/developing? (10pt)

2.2) What major disadvantage does this approach have over multi-threaded implementations of network services? (10pt)

2.3) Would using all non-blocking system calls benefit the inet.d method, or the multithreaded method, more? Why? (10pt)

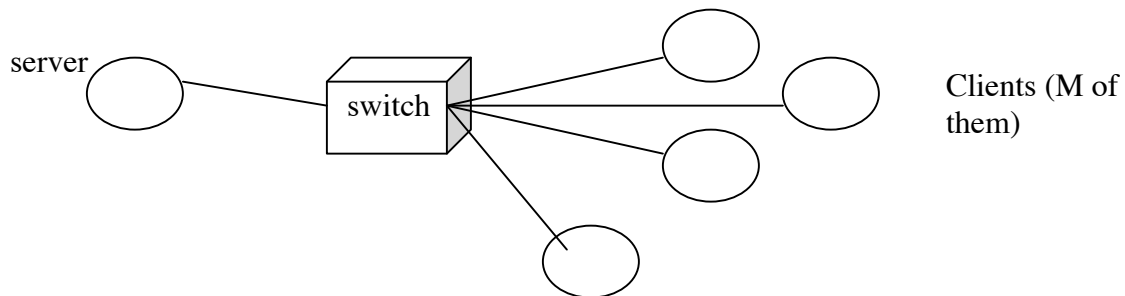
Problem 3 – Inter-server Communication

Recall from the case study that we did in class that we can ensure consistency across servers using a multicast-based cache-invalidation protocol. We're now going to examine some of the tradeoffs such an approach makes.

Attached to this exam is the code and protocol structure of that protocol.

Assume that:

- 1) we have a “server” which stores the authoritative values for our data
- 2) data consists of a set of N records of the form: $\langle \text{id\#}, \text{price} \rangle$
- 3) we have M “clients”, each of which keep cached copies of the data
- 4) the topology of our network is shown below. All links are of infinite bandwidth, and the delay of each link is $100\mu\text{s}$. The switch does not introduce any delay. Writing packets to, and reading packets from the network, does not take any time.
- 5) Whenever entries in the server are updated or deleted, the server sends an “invalidate” message to each client. When the client receives that message, it deletes any cached price.
- 6) When people ask the clients for an item's price, the client will respond with a cached value, if it has one. If it doesn't have a cached value, then it will ask the server for the price, cache it, and give that price to the user.



3.1) Let's say that the server communicated with each client via unicast TCP (through a new connection). Draw the messages between the server and one of the clients below (annotate with times). How many messages would it take for N clients? (15pts)

3.2) Now assume multicast communications. Would the protocol still utilize TCP? Why or why not? (10pts)

3.3) Remote Procedure Call (RPC) is a standard inter-server communication mechanism. In this cache-invalidation example, assume that we have a remote procedure called "update(itemId, newPrice)" which updates the price for the specified item. What assumptions or features does RPC provide that will benefit our application? That will cause performance problems for our application? (10pts)