Name : Rishikesh Khire

cs557 Home work 1

rkhire1@binghamton.edu

1.) In a structured overlay network, messages are routed according to the topology of the overlay. What is an important disadvantage of this approach?

Overlay network is built on physical layer as a virtual network. In overlay network the message is routed on the shortest path between source and destination. This path is shortest in term of distance ,which is the physical best distance ,but the main concern is the shortest path may not be the path with the least delay .So the other better path in terms of minimum delay may be available. For example node in the peer to peer network on the shortest path may be busy (for example a node can router or server) which cause unnecessary delay . While if the message is passed from alternative path which may be longer but the node are free and forward the message quickly without delay .So the alternative path would be efficient as the delay or latency is minimum.

2.) Consider a BitTorrent system in which each node has an outgoing link with a bandwidth capacity Bout and an incoming link with bandwidth capacity Bin. Some of these nodes (called seeds) voluntarily offer files to be downloaded by others. What is the maximum download capacity of a BitTorrent client if we assume that it can contact at most one seed at a time ?

BitTorrent system is a tit for tat system. consider T seeds server and N clients .

Then downloading capacity = T *Bout/ N

if each one of the clients is uploading also i.e helping others then the Downloading capacity = Bout   (for each).

3.) One way to handle parameter conversion in RPC systems is to have each machine send parameters in its native representation, with the other one doing the translation, if need be. The native system could be indicated by a code in the first byte. However, since locating the first byte in the first word is precisely the problem, can this actually work ?

This can work because when we send the message the first byte i.e the 0th byte gives information of the representation .Thus this can easily handle by using the byte function to directly access the

0th byte of the buffer.So we can access it by byte function to read 0th byte whatever the representation may be.   Still if there is problem that can be handled by placing the 1st byte of every word i.e 0th byte of every word with the representation byte . This will solve the problem as every time we read the 1st byte of word it will give representation byte irrespective of the word accessed.

4.) Explain why transient synchronous communication has inherent scalability problems, and how these could be solved?

Transient Synchronous communication has inherent scalability problem because the sender of the request or message is blocked till it receive the reply or acknowledgement from the receiver .This way the sender will always be in blocked or inactive mode if the geographically the node are very far away from each other as the latency for getting acknowledgement will be very very high .This way the throughput or utilization of sender node resources will be very low. This can be handled if we allow the sender node to remain in unblocked or active state i.e it can perform other function till it gets acknowledgement from receiver node . This mode is also called as Asynchronous communication. There are many variants of asynchronous communication which can be used as per requirements .

5.) Consider a client-server application where the client makes remote procedure calls to the server. The link latency between the client and the server is 20 ms each way. Marshalling and un-marshalling of the parameters and return value take approximately 1 ms, respectively. The server takes 150 ms to process each request and can only process one request at a time.

(a) How long would it take to complete two back-to-back requests in a synchronous RPC system?

(b) How long would it take in an asynchronous RPC system? Assume the client is interested in the response and waits for the acceptance of the request before continuing. (Figure 4-10 (b) in Tbook.)

a) time taken for 2 back to back request time = 2*(marshalling at client + link latency + unmarshalling at server + server time +

marshalling at server + link latency +unmarshalling at client)

= 2*(1+20+1+150+1+20+1) = 2*194 = 388 ms

b) time taken in Asynchronous communication

when considering all messages are marshaled and unmarshaled.

time taken for completion = (marshalling at client + link latency + unmarshalling at server +

+ server time + marshalling at server + unmarshalling at server + server time

+ marshalling at server + link latency + unmarshalling at client)

time for completion = (1+20+1+150 +1+1+150+1+20+1) = 346 ms

6.) Traditional RPC cannot handle pointers. What is the problem and how it can be addressed?

Tradition RPC cannot handle the pointers because the pointer refers to the local address of the caller node which is irrelevant at the called node .This memory location may mean something else at the called node which can create problem or faulty solution. This can be handled by using Copy/Restore method .In this we create a copy of array or data item pointed by the pointer and send it to the called node which receive it in its buffer and then called node stub can use pointer to this data element and send it for processing. This way COPY/RESTORE can handle call by reference , but this is only upto arrays copy/restore cannot handle complex data structure pointers such as graphs etc.

7.) Consider the following scenarios for RPC operations in an asynchronous system, which of the three

semantics {at least once, at most once, any number of times} best describes the outcome in that scenario?

(a) The client keeps retransmitting the request until its receives a reply. Server caches all past requests it has received and replies it has sent. Upon receiving a request, it first checks to see if it has seen this request before and if the reply to this request is in the cache – if yes, it returns that reply otherwise it re-executes procedure.

- AT MOST ONCE

(b) Same as (a) above, except that the server only caches requests and replies processed within the past 1 hour.

-AT LEAST ONCE

(c) The server does not keep a cache. It simply executes every request it receives. The client keeps retransmitting request until it receives a reply. In addition, the network might duplicate messages.

-AT LEAST ONCE

(d) Network is dropping all messages.

-ANY NUMBER OF TIMES

(e) Server has crashed, and will not recover.

-ANY NUMBER OF TIMES.