Question 1: **(10** Marks**)**

(a) Describe and illustrate the client-server architecture of one or more major Internet applications (for example, the Web, email or netnews).

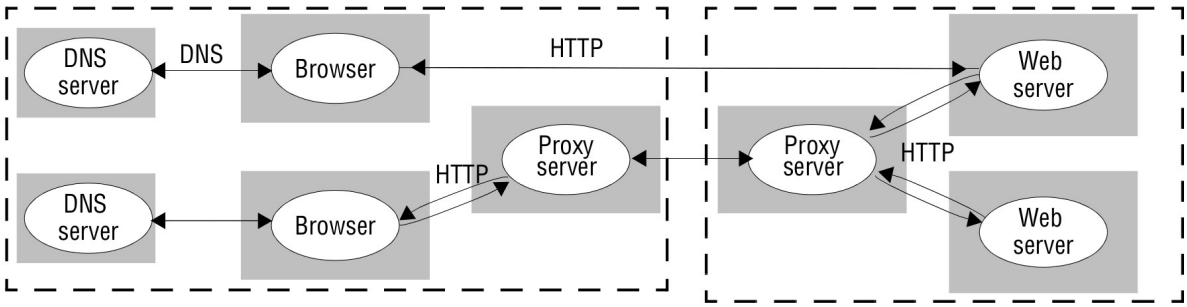
(b) The Network Time Protocol service can be used to synchronize computer clocks. Explain why, even with this service, no guaranteed bound is given for the difference between two clocks.

(c) The Internet is far too large for any router to hold routing information for all destinations. How does the Internet routing scheme deal with this issue?

Solution:

(a)

Web:



Browsers are clients of Domain Name Servers (DNS) and web servers (HTTP). Some intranets are configured to interpose a Proxy server. Proxy servers fulfill several purposes – when they are located at the same site as the client, they reduce network delays and network traffic. When they are at the same site as the server, they form a security checkpoint and they can reduce load on the server. N.B. DNS servers are also involved in all of the application architectures described below, but they ore omitted from the discussion for clarity.

(b) Any client using the ntp service must communicate with it by means of messages passed over a communication channel. If a bound can be set on the time to transmit a message over a communication channel, then the difference between the client's clock and the value supplied by the ntp service would also be bounded. With unbounded message transmission time, clock differences are necessarily unbounded.

(c) If a router does not find the network id portion of a destination address in its routing table, it dispatches the packet to a default address an adjacent gateway or router that is designated as responsible for routing packets for which there is no routing information available. Each router's default address carries such packets towards a router than has more complete routing information, until one is encountered that has a specific entry for the relevant network id.

Question 2: **(10** Marks**)**

(a) How does a newly installed personal computer connected to an Ethernet discover the IP addresses of local servers? How does it translate them to Ethernet addresses?

(b) Sun XDR aligns each primitive value on a 4-byte boundary, whereas CORBA CDR aligns a primitive value of size n on an n-byte boundary. Discuss the trade-offs in choosing the sizes occupied by primitive values.

(c) Why can't binary data be represented directly in XML, for example, by representing it as Unicode byte values? XML elements can carry strings represented as *base64*. Discuss the advantages or disadvantages of using this method to represent binary data.

Solution:

(a) The first part of the question is a little misleading. Neither Ethernet nor the Internet support 'discovery' services as such. A newly-installed computer must be configured with the domain names of any servers that it needs to access. The only exception is the DNS. Services such as BootP and DHCP enable a newly-connected host to acquire its own IP address and to obtain the IP addresses of one ore more local DNS servers. To obtain

the IP addresses of other servers (e.g. SMTP, NFS, etc.) it must use their domain names. In Unix, the nslookup command can be used to examine the database of domain names in the local DNS servers and a user can select appropriate ones for use as servers. The domain names are translated to IP addresses by a simple DNS request. The Address Resolution Protocol (ARP) provides the answer to the second part of the question. Each network type must implement ARP in its own way. The Ethernet and related networks use the combination of broadcasting and caching of the results of previous queries.

(b) Marshalling is simpler when the data matches the alignment boundaries of the computers involved. Four bytes is large enough to support most architectures efficiently, but some space is wasted by smaller primitive values. The hybrid method of CDR is more complex to implement, but saves some space in the marshalled form.

(c) Binary data can't be represented directly in XML, because somewhere the embedded binary data will include the representation of a special character such as '<' or '>'. Even if binary is represented as CDATA, it might include the terminator for CDATA: ']]'. Binary data can be encoded in base64 and represented in XML as a string. Base64 encoding takes three bytes, each consisting of eight bits, and represents them as four printable characters in the ASCII standard (padding when necessary).

Thus a disadvantage in using base64 is that the quantity of data is increased by a factor of 4/3 and the translation process at both ends can take time. In addition, the encoded value isn't really a string and in some cases may be passed on to an application as a string. The only advantage of base64 is that it can be used when necessary. It is in fact used in XML Security to represent digital signatures and encrypted data. In both cases, the data is enclosed in elements that specify, for example, signature or encrypted data.

Question 3: **(10** Marks**)**

(a) Describe a scenario in which a client could receive a reply from an earlier call.

(b) Discuss whether the following operations are *idempotent*:

i) pressing a lift (elevator) request button;
ii) writing data to a file;
iii) appending data to a file.

Is it a necessary condition for idempotence that the operation should not be associated with any state?

(c) The *Election* service must ensure that a vote is recorded whenever any user thinks they have cast a vote. Discuss the effect of *maybe* call semantics on the *Election* service. Would *at-least-once* call semantics be acceptable for the Election service or would you recommend *at-most-once* call semantics?

Solution:

(a) Client sends request message, times out and then retransmits the request message, expecting only one reply. The server which is operating under a heavy load, eventually receives both request messages and sends two replies. When the client sends a subsequent request it will receive the reply from the earlier call as a result. If request identifiers are copied from request to reply messages, the client can reject the reply to the earlier message.

(b) (i) Pressing the lift request button is an idempotent operation. (ii) The operation to write data to a file can be defined as in Unix where each write is applied at the read-write pointer, in which case the operation is not idempotent; or as in several file servers where the write operation is applied to a specified sequence of locations, in which case, the operation is idempotent because it can be repeated any number of times with the same effect. (iii) The operation to append data to a file is not idempotent, because the file is extended each time this operation is performed.

The question of the relationship between idempotence and server state requires some careful clarification. It is a necessary condition of idempotence that the effect of an operation is independent of previous operations. Effects can be conveyed from one operation to the next by means of a server state such as a read-write pointer or a bank balance. Therefore it is a necessary condition of idempotence that the effects of an operation should not depend on server state. Note however, that the idempotent file write operation does change the state of a file.

(c) Maybe call semantics is obviously inadequate for vote! That the voter's number can be used to ensure that the user only votes once. This means that the server keeps a record of who has voted. Therefore at-least-once semantics is alright, because any repeated attempts to vote are foiled by the server.

Good Luck
Dr. Islam Elshaarawy