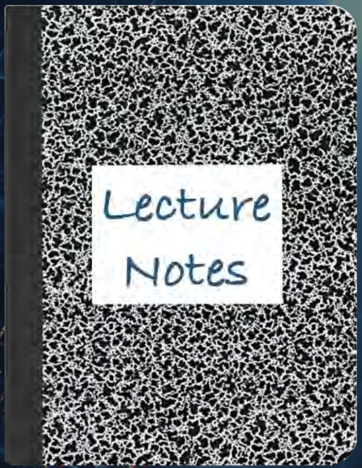
CS 417 – DISTRIBUTED SYSTEMS



Week 6: Exam 1 Review

Paul Krzyzanowski

Question 1

A *single system image* in distributed systems refers to:

(a) All systems running identical copies of the operating system.

(b) A group of computers that present themselves as one system to the user.

(c) A deployment where all servers have the same hardware.

(d) A collection of peer-to-peer systems running identical software.

Single system image: Collection of independent computers that appears as a single system to the user(s)

Question 2

Unlike Moore's Law, *Metcalfe's Law* deals with:

(a) The increasing number of transistors on an integrated circuit.

(b) The decreasing size of processors.

(c) The increasing performance of processors.

(d) The increasing value of a network.

Metcalfe's Law: the value of a telecommunications network is proportional to the square of the number of connected users of the system.

It's a statement of the Network Effect.

Question 3

Caching differs from replication because:

(a) Caches only need to store recently or frequently used data.
(b) Consistency is important in replicas but not in caches.
(c) Caches are stored at clients and replicas are maintained on servers.
(d) Caches are designed to address fault tolerance.

- Replication:
  multiple copies of data for increased fault tolerance
- Caching:
  temporary copies of frequently accessed data closer to where it's needed

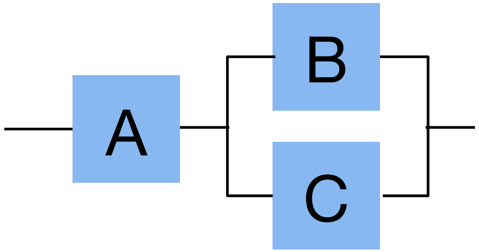
Question 4

In a system of three machines, *A*, *B*, & *C*, *A* and either *B* or *C* must always be running for the system to function. The probability of any computer being down is 10%. What is the availability of the complete system?

(a) 89.1%
(b) 90.0%
(c) 99.0%
(d) 99.96%

No math:

- A & (B | C) are a series system. downtime(A) = 10%, downtime(B|C) > 0
- Therefore, downtime of the system > 10%, so availability < 90%



Math:

- B & C are a parallel system: downtime = 10% × 10% = 1%
- A & (B | C) are a series system:

  downtime = 1 – [(1 – 10%) × (1 – 1%)]
  = 1 – [0.9 × 0.99] = 1 – .891 = 0.109 = 10.9%

  uptime = 1 – downtime = 100% – 10.9% = 89.1%

Question 5

Which of the following is an example of *stale state* in a cache?

(a) Cached data that came from a malicious or untrusted source.

(b) Data that has become corrupted and is therefore not valid.

(c) Data in a cache that is outdated.

(d) Data that has been present in the cache for a long time and is scheduled to get replaced.

- Stale state = information at a node is obsolete
- With caching, it refers to a cached copy of data where the original version has been updated

Question 6

Which of the following is a characteristic of a Byzantine failure in a network?

(a) The network is partitioned in such a way that two or more sets of nodes can no longer communicate.

(b) A node in the network malfunctions to produce unexpected messages.

(c) The failure of one node in the network does not affect the rest of the network.

(d) A node crashes and is therefore unavailable.

- A Byzantine failure is not fail-silent.
  Instead, the component produces faulty data.
- This can be due to bad hardware, software, network problems, or malicious interference

Question 7

Which OSI layer does not exist in the Internet Protocol stack?

(a) Data Link.

(b) Presentation.

(c) Transport.

(d) Network.

The Internet Protocol stack implements the Network (3) & Transport (4) layers and sits on top of the Data Link (2) and Physical (1) layers

Transport Layer (4)	TCP, UDP
Network Layer (3)	Internet Protocol: IPv4, IPv6
Data Link Layer (2)	Ethernet, Wi-Fi, DOCSIS, ATM, Frame Relay, …

Question 8

Which statement is correct?

(a) An advantage of UDP over TCP is that UDP does not have the overhead of setting up a connection.

(b) UDP provides flow control and quality of service management, making it ideal for streaming media.

(c) TCP provides a reliable, fixed-bandwidth data stream between two applications.

(d) TCP is a layer on top of UDP and guarantees that messages arrive in sequence.

- UDP does not require an initial handshake to set up a connection.
- TCP provides flow control – the receiver tells it if it can accept more data
- Neither TCP nor UDP implement quality of service controls
- TCP does provide a reliable data stream, but NOT fixed-bandwidth
- TCP guarantees that messages arrive in order but is NOT a layer above UDP

Question 9

Marshaling in remote procedure calls is:

(a) A mechanism for managing concurrent access to shared resources in a distributed system.
(b) A protocol that handles network connections between a client and server.
(c) The process of converting data into a form suitable for transmission over a network.
(d) A security mechanism that encrypts data transmissions between client and server.

Marshaling serializes the individual parameters as well as other information (function ID, object ID, boot #, transaction #, etc., depending on the implementation

Question 10

Google *protocol buffers* are used in gRPC to:

(a) Provide space to assemble incoming parameters into a network message.

(b) Specify the sequence of messages that need to be passed back and forth between the client and server.

(c) Temporarily store messages at the server that arrive out of order.

(d) Define the interfaces to a set of remote procedures.

- Protocol buffers are a data serialization format
  - Users define data structures and functions, along with their input and output parameters
  - A protocol buffer compiler generates code to enable the serialization and deserialization of these messages

(a) Their goal is not buffer allocation (although that's done at a low level)

(b) They do not specify the sequencing of messages (the protocol)

Question 11

A *cell directory* service in DCE RPC enables:

(a) Clients and servers to obtain an encryption key to communicate securely.
(b) A server to obtain user and group information to authenticate and authorize clients.
(c) A client to map RPC interface identifiers to port numbers at the server.
(d) A client to locate the server that is running a specific service.

- The cell directory service is a server known to a collection of machines in an organization
- It allows a client to query it to locate the server that runs a requested RPC service

Question 12

The use of *leases* for distributed garbage collection is favored over reference counting because:

(a) Clients do not have to contact the server until they delete the object reference.

(b) Objects can be deleted immediately when clients no longer need them.

(c) Multiple clients can have a reference to the same object.

(d) Servers can easily delete objects used by dead clients.

- If a client dies or a network connection is severed, the server will have reference counts > 0 and not clean up the objects.
- With leases, the client will not contact the server to renew the lease, resulting in the server cleaning up the object when its lease expires.
  - This is good and bad – it's bad if we need the object and the client will eventually be able to contact the server.

(a) Clients have to contact the server periodically to renew their lease

Question 13

An advantage of web services over RPC systems such as DCE RPC, Java RMI, and Python RPyC is:

(a) They support site scraping of web pages.
(b) Client-server communication has less overhead.
(c) Their data encoding is more compact because of their use of XML or JSON.
(d) They can leverage existing authentication, encryption, and load-balancing infrastructure.

(b) Web services have a greater overhead because they have to go through the web infrastructure – another protocol layer

(c) XML & JSON are NOT more compact than binary formats

- They can rely on the HTTP/HTTPS to provide authentication & encryption and web servers for load balancing

Question 14

The REST model differs from web services based on XML-RPC or SOAP because:

(a) It provides strong security and encryption mechanisms.

(b) It supports synchronous remote procedure call interactions.

(c) It uses HTML commands and identifies resources via URLs.

(d) It is highly scalable.

- With XML-RPC or SOAP, the entire request was encapsulated in a document that was sent via HTTP
- The idea behind REST was to take advantage of HTTP commands (GET, PUT, POST, …) and URLs to express the interface

Question 15

What is *clock drift*?

(a) The delay between requesting the time from a server and receiving a response.

(b) The rate at which the computer's clock diverges from the actual time.

(c) The difference between the actual time and the time reported by the computer's clock.

(d) The variation in a computer's clock frequency over time.

- The difference in time between two systems is the *offset.*

Question 16

The primary reason that UDP is usually used for clock synchronization is that:

(a) It does not provide reliable messaging.

(b) It is more secure and resistant to tampering.

It's actually easier to spoof UDP data

(c) It operates at a lower layer in the network stack than TCP.

(d) UDP uses a more efficient data encoding format.

TCP or UDP have no bearing on data encoding

UDP has these advantages for clock synchronization:

1. It does not have the overhead of setting up a connection (an extra three network messages).
2. UDP is message rather than byte-stream oriented. It sends packets out immediately. TCP, by default, waits a while to see if more bytes are coming that it can stuff into the same packet (this is Nagle's algorithm).
3. TCP provides reliable, in-order data transmission. If a packet is lost, the receiver will request it again … but this creates a delay, which you don't want in clock synchronization

Question 17

A client sends a time request to the server at 1212 and receives a response containing a timestamp of 1202 at 1232. The best-case round-trip delay is 10. Using Cristian's algorithm, to what value does the client set its clock?

(a) 1212.
(b) 1222.
(c) 1232.
(d) 1242.

- Delay between sending a request and receiving a reply: 1232 – 1212 = 20
- ½ of the delay = 20 ÷ 2 = 10
- Set time to server time + ½ round-trip delay = 1202 + 10 = 1212

Question 18

What does a *stratum* refer to in NTP synchronization?

(a) The number of clients that connect to a specific NTP server.

(b) The distance from a reference clock in terms of server hop count.

(c) The elapsed time since an NTP server synchronized its own clock.

(d) The delay of sending data from a client to a specific NTP server.

- NTP servers are arranged in strata
  - Stratum 0 = reference clock
  - Stratum 1 = NTP server connected to the reference clock
  - Stratum 2 = NTP server that synchronizes from stratum 1
  - …

Question 19

What statement is true about Lamport timestamps $L(a)$ and $L(b)$ for two distinct events *a* and *b*?

(a) If $L(a) < L(b)$ then $a \rightarrow b$.

(b) If $L(a) = L(b)$ then *a* and *b* are concurrent events.

(c) If $L(a) \neq L(b)$ then *a* and *b* are concurrent events.

(d) None of the above.

- With Lamport timestamps, if $a \rightarrow b$ then $L(a) < L(b)$ but the inverse is not necessarily true
  - You can have two concurrent events with different Lamport timestamps
- However, if $L(a) = L(b)$ and *a* and *b* are distinct events, we know that:
  - *a* and *b* cannot occur on the same process since each successive event gets an incremented time stamp
  - If *a* and *b* are on different machines and causally related, they cannot have the same timestamp. Therefore, *a* and *b* must be concurrent events on different processes.

Question 20

An advantage of *vector clocks* over Lamport clocks is:

(a) Both causal and concurrent relationships can be identified by comparing timestamps.
(b) They identify the real-time ordering of concurrent events in addition to causal events.
(c) They provide causally-ordered sequencing to remote as well as local events.
(d) Vector clocks can scale to much larger environments than Lamport clocks.

- With Lamport timestamps, you cannot tell whether events are causal or concurrent by comparing timestamps.
- With vector clocks, you can.

Question 21

A requirement of *atomic multicasts* that reliable multicasts do not have is that:

(a) The multicast sender must first check that all group members are alive and reachable.

(b) The multicast sender sends a multicast message by iterating over all group members.

(c) A multicast message cannot be delivered to only a subset of the group.

(d) A multicast message must be delivered to all receivers at the same time.

- Atomicity is an "all or nothing" property:
  The message is either delivered to *all* group members or none.
- You cannot deliver the message to only some group members

Question 22

Which statement is correct?

(a) Total ordering implies causal ordering.

(b) Causal ordering implies total ordering.

(c) Causal ordering implies FIFO ordering.

(d) FIFO ordering implies causal ordering.

(a) Total ordering means that all processes receive multicast messages in the SAME order

- That order does not have to be causal, just consistent

(b) Causal ordering does NOT imply total ordering since non-causally related messages can be delivered in any order at different processes

(c) Since messages on the same process are causally ordered, all messages from the same process will be delivered in FIFO order. Messages from other processes may be interleaved in any order.

(d) FIFO ordering implies causal ordering ONLY for events on one process.

Question 23

One challenge that PIM-SM (Protocol Independent Multicast – Sparse Mode) does ***not*** have is:

(a) Each router having to send a copy of each multicast message to every connected network interface.

(b) Configuring a rendezvous point and getting all senders and receivers to know about it.

(c) Latency, since traffic may not take the most efficient path from the sender to each receiver.

(d) Scaling, since all multicast traffic must flow through one router.

PIM-SM:

- Requires routers at all endpoints to know about the rendezvous point
- May not route multicast messages in the most direct route since the rendezvous point may be far away (in terms of network hops)
- May incur extra traffic since all multicast packets must be routed to, and flow through, the router that's configured as the rendezvous point

PIM-SM doe NOT send a copy of the packet to every connected router – Dense Mode does that (PIM-DM)

Question 24

What is a *view* in virtual synchrony?

(a) A collection of nodes that can communicate with each other.

(b) A sequence of messages that are guaranteed to be delivered in the same order to all members of a group.

(c) A set of multicast addresses that are used to send messages to various groups in the system.

(d) A list of members in a group that is maintained by a membership service.

- A view is the current definition of group members who are alive.
- No system can tell the global state but the group membership service propagates knowledge of what it believes to be the group membership
  - If a system cannot contact the group membership service, it's out of the group even if it's alive

Question 25

Virtual synchrony implements atomic multicast by:

(a) Having all live group members multicast all their non-delivered messages to all group members if the sender dies.

(b) Electing another group member to take over a multicast if the sender dies during a multicast.

(c) Writing messages into a persistent log so they could be read and sent again even if the system reboots.

(c) There is no persistent logging

(d) Routing all messages through the Group Membership Service (GMS).

(d) Messages are never routed through the GMS

- A sender tries to send a message to all group members
- The received messages are not delivered to the process until the receiver is informed by the sender that *all* group members received it. That makes it **stable**.
- If a sender dies before it sends the message to all members or sends a confirmation, a view change is triggered by the GMS. Each process sends all unstable messages to every other group member.

The End