Department of Computer Science

Distributed Systems

Exam 1

March 2, 2026

Discussion

Prepared with tender loving care by Paul Krzyzanowski

100 POINTS – 25 QUESTIONS – 4 POINTS EACH – For each statement, select the *most* appropriate answer.

1. *Partial failure* in a distributed system refers to:
   a. Some components failing and then restarting with stale state.
   b. Some components failing while others continue to operate.
   c. Components that occasionally produce incorrect output.
   d. Network delays severe enough to make the system appear unresponsive.

Correct answer: (b)
Partial failure is the defining characteristic that distinguishes distributed systems from centralized ones. A single machine typically fails completely or not at all. In a distributed system, some nodes can crash or become unreachable while the rest of the system continues running, forcing designers to handle this mixed state explicitly.

Incorrect answers:
a. Crash-and-restart behavior is a related failure mode called fail-restart, but it is a specific type, not the general definition of partial failure.
c. Producing incorrect output describes Byzantine failure, not partial failure.
d. Extreme delays can make a system appear failed, but "partial failure" refers specifically to some components stopping while others continue, not to timing behavior.

2. In terms of availability, a *series* system differs from a *parallel* system because:
   a. Series systems fail if any component fails; parallel systems can tolerate some component failures.
   b. Series systems require TCP; parallel systems require UDP.
   c. Series systems have lower latency; parallel systems have higher throughput.
   d. Series systems are asynchronous; parallel systems are synchronous.

Correct answer: (a)
Series composition creates a single point of failure at every component; parallel adds redundancy.

Incorrect answers:
b. Transport choice is unrelated to series vs parallel availability structure.
c. Latency and throughput are not the defining distinction.
d. Timing models are orthogonal to how components are arranged.

3. The *end-to-end principle* of the Internet implies that:
   a. Routers should maintain per-connection state for scalability.
   b. IP should try to provide ordered, exactly-once packet delivery.
   c. Encryption must be implemented at the network layer.
   d. Reliability and correctness features primarily belong at the endpoints, not in the network.

Correct answer: (d)
The network stays simple (best effort); endpoints implement recovery, ordering, and security.

Incorrect answers:
a. Fate sharing pushes the connection state to endpoints, not routers.
b. IP is best-effort datagram delivery with no ordering or delivery guarantees.
c. Encryption can be done at endpoints (for example, TLS) and is not limited to the network layer.

4. What is a significant downside of achieving *full transparency* in a distributed system?
   a. It can hide failures and mask degraded behavior from applications.
   b. It is prohibitively expensive to implement in practice.
   c. It prevents the system from scaling horizontally.
   d. It requires custom hardware at every node.

Correct answer: (a)
Transparency is the goal of making distribution invisible to users and applications. But hiding failures can prevent applications from responding to them appropriately. If a node is slow or unreachable, a fully transparent system may silently retry or delay rather than surfacing the problem, which can be worse than explicitly exposing the failure. This is one reason full transparency is rarely achievable or even desirable.

Incorrect answers:

b. Cost is not the primary theoretical objection to transparency. The concern is the ability to reason about and respond to failures, not implementation expense.
c. Transparency does not inherently prevent horizontal scaling. Many transparent systems scale well.
d. Transparency is a software design goal. It does not require specialized hardware

5. Why does NTP use UDP rather than TCP?
   a. UDP guarantees that packets arrive in the order they were sent.
   b. UDP reduces per-client state on busy servers.
   c. UDP provides built-in protection against spoofed packets.
   d. TCP retransmissions introduce asymmetric delays.

Correct answer: (d)
Why Correct answer: TCP retransmission and recovery behavior can make the effective delay depend on earlier loss, which can break the symmetry assumptions used for offset estimation.

Incorrect answers:
a. UDP does not guarantee ordered delivery (or delivery at all).
b. UDP can reduce per-client transport state, but that is not the main reason NTP avoids TCP for time estimation.
c. UDP has no built-in protection against spoofing; protection requires authentication mechanisms above the transport..

6. What does "*best-effort delivery*" mean in the context of the Internet?
   a. The network will retransmit lost packets but will not reorder them.
   b. Routers will find the most efficient path to the destination.
   c. Packets may be lost, delayed, duplicated, or delivered out of order.
   d. The network ensures in-order delivery but permits some packet loss.

Correct answer: (c)
The Internet makes no delivery guarantees. Routers forward packets when they can and drop them when they cannot. A packet may be lost due to congestion, delayed by routing changes, duplicated by retry mechanisms, or arrive out of order because packets may take different paths. Handling all of these cases is the responsibility of the software at the endpoints.

Incorrect answers:
a. The network does not retransmit. Retransmission is implemented at the endpoints by protocols like TCP, not by routers.
b. Routers do attempt to find good paths, but this describes routing policy, not best-effort delivery semantics.
d. The Internet does not guarantee in-order delivery. Packets travel independently and can arrive in any order.

7. What is the role of a server stub (skeleton) in an RPC system?
   a. It packages the client's arguments and sends them across the network to the server.
   b. It receives incoming requests, extracts the parameters, and calls the actual local procedure.
   c. It reads an interface definition and generates stub code for both client and server.
   d. It maintains a registry of available services so clients can discover and connect to them.

Correct answer: (b)
The server stub handles everything the server-side framework needs to do transparently: waiting for requests, unpacking the marshalled parameters, calling the actual procedure, packaging the result, and sending the response. The actual procedure does not know it is being called remotely.

Incorrect answers:
a. Packaging arguments and sending them to the server is the job of the client stub, not the server stub. .
c. Reading an interface definition and generating stubs is the job of the IDL compiler, such as rpcgen in ONC RPC. .
d. Maintaining a registry of services is the job of a name server or service registry, not the stub.

8. *At-most-once* RPC execution is typically achieved by:
   a. Relying on TCP to retransmit lost replies.
   b. Using a single global sequencer for all calls.
   c. Tagging requests with unique IDs and suppressing duplicates at the server.
   d. Requiring the client to wait for an explicit acknowledgment before issuing any subsequent call.

Correct answer: (c)
Unique request IDs let the server detect and drop or replay duplicate requests.

Incorrect answers:
a. TCP does not prevent the server from executing a request multiple times.
b. A global sequencer is not required and harms scalability.
d. Serializing calls per client does not stop a single call from being executed twice if the request or reply is lost and the client retries, and it does not provide server-side duplicate suppression.

9. The *receiver-makes-right* strategy for handling data representation differences between machines is designed to
   a. Ensure all machines convert data to a standard format before transmitting it.
   b. Allow receivers to recover from corrupted data without requesting retransmission.
   c. Provide version numbers so receivers can support multiple versions of an interface.
   d. Avoid unnecessary data conversion when the sender and receiver share the same architecture.

Correct answer: (d)
Receiver-makes-right allows the sender to transmit data in its own native format. The receiver converts only if its architecture differs from the sender's. When two machines share the same architecture – the common case in a homogeneous cluster – no conversion happens at all, saving work on every message. DCE RPC used this approach with NDR.

Incorrect answers:
a. Converting to a standard format before transmitting describes the canonical format strategy, used by ONC RPC with XDR. It is the opposite approach.
b. Recovering from corrupted data describes error correction, which is unrelated to byte-order conversion.
c. Version numbers have nothing to do with the data representation; they're part of a marshaled message that enables a server to identify which version of an interface is being used.

10. *Reference counting* is a poor strategy for garbage collecting remote objects in a distributed system primarily because:
    a. Clients may crash without decrementing the count, causing objects to be permanently retained.
    b. Reference counts become inaccurate when multiple clients hold references to the same object simultaneously.
    c. Sending increment and decrement messages across the network introduces unacceptable latency.
    d. Reference counts cannot be atomically updated when the two machines are running different operating systems.

Correct answer: (a)
If a client crashes, it cannot send the decrement message that would signal it has released the object. The server's reference count never reaches zero, and the object is never freed, creating a memory leak. This is the fundamental problem that leases and heartbeats were designed to solve.

Incorrect answers:
b. Multiple clients referencing the same object is not a problem: each client increments the count when it acquires a reference and decrements when it releases it. The counts add up correctly as long as all decrement messages arrive.
c. Network overhead is a real cost, but it is not the fundamental correctness problem. If needed, a system can be configured to use batching or lazy updates. Permanent leaks from client crashes cannot be corrected after the fact.
d. The operating systems of the two machines are irrelevant to reference counting. The problem is the unreliable delivery of decrement messages, not platform heterogeneity.

11. *gRPC* introduced a communication capability that traditional RPC frameworks like ONC RPC and DCE RPC did not support. Which of the following is that capability?
    a. Versioning support to allow gradual client migration without breaking existing clients.
    b. Bidirectional streaming, allowing both client and server to send sequences of messages over a single connection
    c. Compiling an interface definition to automatically generate client and server stub code.
    d. Binary encoding of parameters for compact and efficient message transmission.

Correct answer: (b)
gRPC supports four communication patterns: unary request-response, server streaming, client streaming, and bidirectional streaming. Traditional RPC systems were strictly request-response. Streaming support makes gRPC suitable for use cases like live data feeds, long-running computations that produce incremental results, and real-time telemetry.

Incorrect answers:
a. ONC RPC included versioning support from its early design, allowing multiple versions of a service to coexist and clients to migrate gradually. .
c. ONC RPC's rpcgen compiler and DCE RPC's IDL compiler both generated stubs from interface definitions. This was a standard feature of traditional RPC systems. .
d. ONC RPC used XDR and DCE RPC used NDR, both binary encoding formats. Binary encoding was not new with gRPC.

12. Which of the following is the greatest advantage of *Protocol Buffers* over JSON for microservice communication?
    a. Protocol Buffers messages are human-readable, making it easier to debug communication between services.
    b. Protocol Buffers does not require a schema, giving teams more flexibility to evolve their APIs.
    c. Protocol Buffers is natively supported by web browsers, reducing the need for client-side libraries.
    d. Protocol Buffers fields are identified by numeric tags, so new fields can be added without breaking clients that do not know about them.

Correct answer: (d)
Each field in a Protocol Buffers message carries a numeric tag defined in the schema. Code that was compiled against an older schema simply ignores fields with tags it does not recognize, and missing fields take default values. This makes it straightforward to add new fields to a message type without breaking any existing clients or servers that have not yet been updated.

Incorrect answers:
a. Protocol Buffers messages are binary and are not human-readable. JSON's human readability is one of the reasons it remains popular for public APIs and debugging.
b. Protocol Buffers requires a schema defined in a .proto file. The schema is central to how stub code is generated and how fields are identified.
c. Web browsers have no native support for Protocol Buffers. JSON has native browser support through the JavaScript runtime, which is one reason REST over JSON remains the standard for public-facing APIs.

13. A client sends a time request at local time *t=100* and receives the response at *t=180*. The response contains a server timestamp of *155*. Using Cristian's algorithm, what time does the client set its clock to?
    a. 155
    b. 180
    c. 195
    d. 235

Correct answer: (c)
Cristian's algorithm estimates the current time as Ts + (t1 - t0)/2, accounting for the time elapsed since the server generated its timestamp.

Incorrect answers:
a. Using the server timestamp directly ignores the propagation delay that elapsed before the response arrived.
b.The client's local receive time reflects only when the packet arrived, not what time the server would report at that moment.
d. Adding the full round-trip time to the server timestamp overcorrects by counting the one-way delay twice.

14. Using the scenario from the previous question, the minimum round-trip time is known to be 50. What is the maximum *synchronization error*?
    a. $e \leq 15$
    b. $e \leq 25$
    c. $e \leq 40$
    d. $e \leq 50$

Correct answer: (a)
The error bound is (RTT - min_RTT) / 2. The actual round-trip time from question 2 is 80, so (80 - 50) / 2 = 15.

Incorrect answers:
b. 25 is half the minimum RTT, not the error bound; dividing the minimum RTT by 2 confuses it with the correction term.
c. 40 is half the actual RTT, which ignores the tightening effect of the minimum RTT constraint.
d. 50 is the minimum RTT itself; it narrows the error bound, but is not the bound.

15. PTP achieves sub-microsecond clock synchronization accuracy while NTP typically achieves only millisecond accuracy. The primary reason for this difference is:
    a. PTP captures timestamps in the network interface card.
    b. PTP measures one-way delays in each direction separately to correct for asymmetric network paths.
    c. PTP applies more sophisticated statistical filtering to eliminate outlier time samples.
    d. PTP uses multicast delivery, which provides lower and more consistent latency than NTP's unicast requests.

Correct answer: (a)
Hardware timestamping at the NIC captures the exact moment a packet crosses the physical layer, removing the OS scheduling jitter and network stack processing delay that dominate NTP error.

Incorrect answers:
b. PTP does use bidirectional message exchange, but NTP does too; asymmetric delay compensation is not the primary driver of the accuracy gap.
c. NTP also applies statistical filtering and outlier rejection; the filtering approach is not what separates the two protocols' accuracy.
d. PTP's use of multicast is a scalability choice, not the source of its accuracy; the accuracy gain comes from hardware timestamping.

16. A developer observes that event *a* has Lamport timestamp 10 and event *b* has Lamport timestamp 15, and concludes that *a causally preceded b*. This reasoning is:
    a. Correct answer: if $L(a) < L(b)$, then *a* happened before *b* by definition of the Lamport clock rules.
    b. InCorrect answer: Lamport timestamps only provide ordering within a single process, not across processes.
    c. InCorrect answer: Lamport timestamps only guarantee that if *a* happened before *b*, then $L(a) < L(b)$, not the converse.
    d. Correct, provided the difference is large enough to exceed clock drift.

Correct answer: (c)
The happened-before relation implies a lower timestamp, but the implication does not reverse. Two events on different processes with no causal link can still have timestamps 10 and 15; the lower timestamp does not prove causality.

Incorrect answers:
a. This reverses the direction of the implication; LC(a) < LC(b) is necessary but not sufficient to conclude a -> b.
b. Lamport timestamps do provide a consistent ordering across processes; the problem is not scope but the inability to detect concurrency from timestamps alone.
d. Lamport timestamps are logical, not physical; clock drift is irrelevant to their interpretation

17. Two vector timestamps V_a and V_b indicate ($a \rightarrow b$) when:
    a. $V_a[i] < V_b[i]$ for all i
    b. $V_a[i] = V_b[i]$ for all i
    c. $V_a[i] \leq V_b[i]$ for all i and $V_a[i] < V_b[i]$ for at least one i
    d. There exist i, j with $V_a[i] < V_b[i]$ and $V_a[j] > V_b[j]$

Correct answer: (c)
One value must be ≤ the corresponding value in the other vector for each element, with at least one value being smaller than the corresponding value in the second vector to show causality: that the first element happened before the second and they are causally related.

Incorrect answers:
a. Strictly less in all components is stronger than needed and often false.
b. Equality means the same causal history, not "happened-before."
d.Mixed greater/less implies concurrency.

18. A key motivation for *hybrid logical clocks* (*HLC*) is to:
    a. Stay close to physical time while preserving happened-before relationships.
    b. Provide total ordering without synchronized clocks.
    c. Be able to detect concurrency by comparing timestamps.
    d. Eliminate the need for physical clocks by combining sequence counters with process IDs.

Correct answer: (a)
Correct answer: HLC combines an L close to physical time with a counter to maintain causality.

Incorrect answers:
b. HLC still relies on physical clocks and cannot provide perfect total order.
c. HLC preserves happened-before but does not generally detect concurrency like vectors.
d. Sequence counters combined with process IDs are used to adapt Lamport timestamps to enable total ordering.

19. A distributed video conferencing system multicasts audio frames to all participants. The sender crashes after sending a frame to three of the five participants. Under *best-effort reliable multicast* (without sender recovery), what happens?
    a. All five participants deliver the frame because other recipients retransmit it to the remaining two.
    b. The three recipients deliver the frame, and the other two do not.
    c. No participant delivers the frame because the protocol rolls back delivery when the sender crashes.
    d. The frame is discarded by all participants because it has not been acknowledged by the full group

Correct answer: (b)
Best-effort reliable multicast only guarantees delivery if the sender completes without crashing. If the sender crashes mid-transmission, some recipients may have the message and others may not, and the protocol makes no consistency promise in that case.

Incorrect answers:
a. Best-effort reliable multicast does not include a retransmission protocol among recipients; only full reliable multicast provides the agreement property that ensures all correct processes deliver the message.
c. Best-effort reliable multicast has no rollback mechanism; delivered messages stay delivered.
d. The acknowledgment mechanism ensures stability under full reliable multicast, not best-effort multicast

20. Which statement correctly describes the relationship between *causal ordering* and *single-source FIFO* ordering?
    a. They are equivalent: any system providing one automatically provides the other.
    b. Causal ordering implies single-source FIFO ordering, but not vice versa.
    c. Single-source FIFO ordering implies causal ordering, but not vice versa.
    d. They are independent: a system can provide either without providing the other.

Correct answer: (b)
Messages from the same sender are causally related by definition, so causal ordering subsumes FIFO ordering. However, a system can deliver messages in FIFO order per sender while delivering concurrent messages from different senders in inconsistent order across processes.

Incorrect answers:
a. FIFO ordering makes no claim about messages from different senders that are causally related through intermediate steps; a causally ordered system would handle those, but a FIFO system would not.
c. This reverses the implication; FIFO ordering only constrains per-sender sequence and is strictly weaker than causal ordering.
d. The two are not independent because messages from the same sender are causally related; any causal ordering protocol must also satisfy FIFO ordering as a consequence

21. In a virtual synchrony *view change*, what is the main purpose of the *flush* phase?
    a. Choose the new membership.
    b. Deliver all buffered messages immediately.
    c. Elect a coordinator for the next view.
    d. Propagate in-flight (buffered) messages to survivors.

Correct answer: (d)

Flush exchanges buffered, not-yet-delivered messages so survivors converge on the same set.

Incorrect answers:
a. Membership agreement is the Commit phase outcome.
b. delivery waits until messages are determined stable in the transition.
c. Leader election is a separate mechanism, not required by view change.

22. Why does PIM Sparse Mode use a *Rendezvous Point* (RP)?
    a. To flood traffic quickly to all destinations before pruning.
    b. To give sources one destination to send to and receivers a point to join.
    c. To eliminate the need to consult unicast routing tables.
    d. To ensure physical-time ordering of multicast packets.

Correct answer: (b)
Sparse mode assumes few receivers and builds distribution via explicit joins toward an RP.

Incorrect answers:
a. Flood-and-prune is dense mode behavior.
c. PIM is "protocol independent" because it uses unicast routing tables.
d. Real-time ordering is not achievable and not an RP function

23. In an asynchronous system, a perfect *failure detector* is impossible primarily because:
    a. Heartbeats can be forged by other systems.
    b. Processes cannot send periodic messages reliably.
    c. A slow process is indistinguishable from a crashed process.
    d. Clock drift makes it impossible to measure timeouts accurately.

Correct answer: (c)
Without timing bounds, you cannot tell delay from failure.

Incorrect answers:
a. Encryption is orthogonal to detectability.
b. They can send heartbeats, but interpretation is the issue.
d. even with perfect clocks, asynchronous message delays are unbounded .

24. In *Lamport's mutual exclusion* algorithm, a process may enter the critical section when:
    a. It receives a *GRANT* message from the process that previously held the resource.
    b. It holds a token for that resource.
    c. It has replied to all other requests.
    d. Its request is first in its local queue, and it received acknowledgments from all other processes.

Correct answer: (d)
Entry requires head-of-queue plus acks from all peers.

Incorrect answers:
a. Lamport's algorithm does not have a concept of GRANT messages – the centralized and ring algorithms do.
b. That is the token ring mutual exclusion.
c. Replying to others is not the entry condition.

25. In the *ring election algorithm*, when a process receives an ELECTION message containing its own ID, it should:
    a. Declare itself the winner and send an *ELECTED* message around the ring.
    b. Treat it as a forgery, do not forward the message, and start a new election.
    c. Send an *OK* message to the previous process and stop.
    d. Replace the *ID* with the highest process *ID* it has previously seen and continue the election.

Correct answer: (a)
Seeing your own ID return implies it is the largest among participants in that election.

Incorrect answers:
b. The election terminates when the largest ID returns to its originator.
c. OK is bully-algorithm messaging, not ring-election messaging.
d. Resetting IDs is not part of the protocol and breaks correctness.

The end.