CSC175 Data Communications & Networking Spring 2026 Final Exam ANS

Student Name:_____ ID：_____

Total pts	

Note: For Q2-Q9, **it is recommended that you provide a short explanation for each answer key.** If your answer is correct, you will get the full point without the explanations. But in case your answer is incorrect, the explanations may earn you some partial credit.

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9
/10	/15	/10	/16	/10	/14	/10	/8	/7

Q1 (10 pts) Multiple-choice questions: enter your answer keys here:

1	2	3	4	5	6	7	8	9	10
C	C	B	C	C	C	B	B	B	B

For the following multiple-choice questions, each question has exactly one correct answer key. If multiple choices are correct, choose the option “All of the above”. Fill in the answer keys in the table above. **(Answer keys written in the question area will not be counted.)**

1. What is the main purpose of TTL in IPv4?
A. To indicate the packet priority.
B. To identify all fragments of one packet.
C. To prevent indefinite forwarding loops by limiting hop count.
D. To measure propagation delay.

ANS: C. TTL is decremented at each hop so looping packets eventually expire.

2. How does the sender respond when a packet or its acknowledgment is dropped?
A. It waits forever.
B. It sends a negative acknowledgment.
C. It uses a timer and retransmits when the timer expires.
D. It changes the port number.

ANS: C. The sender starts a timer and retransmits if the ack does not arrive before timeout.

3. What is the congestion window (cwnd)?
A. A receiver-side field that chooses the port number.
B. A sender-side output of the congestion-control algorithm that limits sending rate.

C. A checksum used only on duplicate acks.
D. A fixed constant equal to the MSS.

ANS: B. The congestion window is computed by the sender's congestion-control logic to avoid overloading the network.

4. Suppose a TCP sender transmits a segment with sequence number j carrying B bytes. If all prior data has been received in order, what ACK value will the receiver send after correctly receiving this segment?
A. j
B. j + B − 1
C. j + B
D. j + 2B

ANS: C – Cumulative ACKs report the next expected byte, so after receiving bytes j through j+B−1, the receiver ACKs j+B.

5. In slow start, what rate-adjustment behavior causes the window to grow exponentially?
A. Adding a fixed number of packets to CWND each RTT
B. Halving CWND on every ACK
C. Doubling CWND every RTT by adding one packet per ACK in that RTT
D. Keeping CWND constant and only adjusting the timeout

ANS: C – Each ACK increases CWND by one packet; over one RTT with CWND packets, this leads to roughly doubling CWND each RTT.

6. What role does SSTHRESH play in TCP congestion control?
A. It stores the maximum allowed RTT
B. It records the last observed loss rate
C. It remembers a "safe" window size where slow start should stop and AIMD should begin
D. It indicates the number of duplicate ACKs seen so far

ANS: C – SSTHRESH is set to about half the window at which loss occurred; slow start stops when CWND exceeds SSTHRESH, switching to additive increase.

7. What key idea underlies fast recovery in TCP Reno/New Reno?
A. Use ECN bits to avoid all packet loss
B. Grant temporary "credit" for each duplicate ACK to keep packets in flight
C. Reset CWND to 1 packet after every loss
D. Move congestion control into routers instead of hosts

ANS: B – Fast recovery interprets duplicate ACKs as evidence that packets have left the network, so it temporarily inflates CWND by one per duplicate ACK to keep sending.

8. What is a DNS name server?
A. A host that only caches HTTP responses
B. A server responsible for answering DNS requests for some set of domains
C. Any router participating in BGP

D. A server that assigns IP addresses using DHCP

ANS: B – A DNS name server is responsible for answering DNS queries for the domains under its authority.

9. In an active attack, where an attacker captures a valid message and later retransmits it to produce an unauthorized effect, the attack is called:
A. Masquerade
B. Replay
C. Eavesdropping
D. Traffic analysis

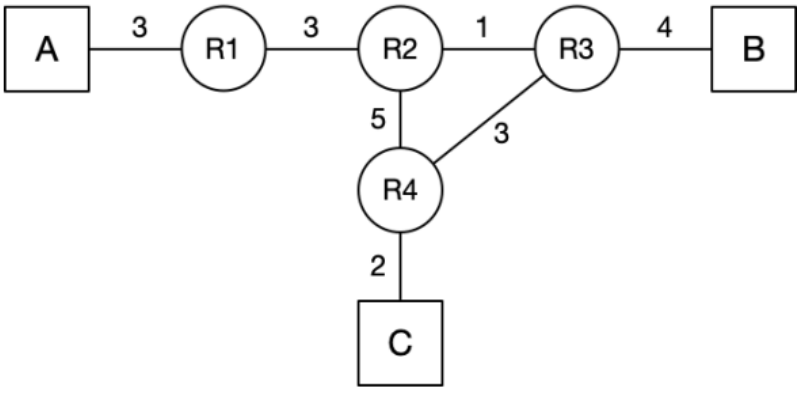
ANS: B – Replay attacks involve capturing and retransmitting valid messages to gain unauthorized effects.

10. Which protocol is specifically used for establishing a shared secret key over an insecure channel, rather than directly encrypting data?
A. RSA
B. Diffie–Hellman key exchange
C. AES
D. DES

ANS: B – Diffie–Hellman is a key exchange protocol that allows two parties to agree on a shared secret used later for symmetric encryption.

Q2. (15 pts) Distance Vector (count-to-infinity)

Routers run a **distance-vector** algorithm with no poisoned reverse or route poisoning, though **split horizon** is enabled. The routing algorithm uses incremental and triggered updates: when a router's table changes, it immediately advertises only the routes that changed. Routers send periodic routing updates **every 2 seconds** starting at t=0 s. Routing table entries expire after **TTL = 6** seconds of receiving no routing updates. Link costs equal propagation delay in seconds.



(a) (2 pts) Fill in the routing table of **R4** after all routes converge.

To	Next-hop Router	Cost
A		
B		
C	Direct	2

(b) (3 pts) Assume that all router-to-host direct links are initialized at time t = 0. How long will it take for R4's routing table to reach its converged state, starting from time t = 0?

(c) (4 pts) Assume that at time t: All routers have converged. All routers send their periodic advertisements at time t-4, t-2, t…. Assume at time t+1, R1 crashes. How many seconds after t will R2's routing table reflect the failure of R1?

(d) (6 pts) Regardless of your answer to the previous subpart, assume R2's routing table reflected the failure of R1 at time t'. Assume t' is the time at which all routers send periodic advertisements as well. Fill out the routing table at R2 and R3 at time t'+3. All rows at the table should be filled. If a route entry is expired, mark it under the "Expired" column with "X". Ignore all triggered updates that might have happened prior to t', but do consider any periodic updates that have happened (for example at time t'-2). (Recall that **split horizon** is enabled.)

Routing table of **R2:**

To	Next-hop Router	Cost

A		
B		
C		

Routing table of **R3:**

To	Next-hop Router	Cost
A		
B	Direct	4
C		

ANS:

(a) (2 pts) Fill in the routing table of **R4** after all routes converge.

To	Next-hop Router	Cost
A	R3	10
B	R3	7
C	Direct	2

(b) (3 pts) 7 seconds

Explanation: Host A: R1 learns about the static route at time t=0, sends an advertisement to R2. at t=3 R2 receives this advertisement and send it to R4 and R3. At time, t=4 R3 receives the route to A and at t=7 R4 receives it. Host B: R3 sends an advertisement to R4 at t=0. At t=3 R4 receives this. So it takes 7 seconds for R4's routing table to reach its converged state.

(c) (4 pts) 9 seconds.

Explanation: At time t, R1 sends a periodic advertisement to R2 and then crashes at time t + 1. This advertisement reaches R2 at t + 3 and R2 updates its table accordingly. This entry will expire at t + 3 + 6 = t + 9 since R2 will not hear any more updates from R1 after t + 1.

(d)

Routing table of **R2:**

To	Next-hop Router	Cost

A	R4	15
B	R3	5
C	R3	6

Routing table of **R3:**

To	Next-hop Router	Cost
A	R2	16
B	Direct	4
C	R4	5

Explanation:
At time t', **R2** has just reflected the failure of **R1**, meaning its route to **A** is gone, and t' is also a periodic-advertisement time. The question says to **ignore triggered updates before** t' but still consider periodic advertisements sent at earlier times such as $t' - 4$ and $t' - 2$. That is why stale information about **A** is still moving through the network after **R2** has already expired its own entry. Timeline:

1. **At** $t' - 4$: all routers send periodic advertisements, and at that moment **R2 still advertises A** because it has not yet reflected the failure.
2. **At** $t' - 2$: all routers send another periodic round, and again **R2 still advertises A**.
3. **At** t': **R2's route to A expires**, so R2 now has **no route to A**, and all routers send a periodic advertisement at this instant.
4. **At** $t' + 1$: an older advertisement sent at $t' - 4$ from **R4** reaches **R2**, advertising **A with cost 10**; because R2's own route has expired, it now accepts this route and installs **A via R4 with cost 15**.
5. **At** $t' + 2$: R2's triggered update reaches **R3**, and since **R3 still believes A is reached via R2**, it accepts the new value and updates to **A via R2 with cost 16**.
6. **At** $t' + 3$: another older advertisement sent at $t' - 2$ from **R4** reaches **R2**, but it still contains the old information and does not improve R2's current route, so no further change happens by this time.

Summary: **R2 loses A, then relearns stale A from R4, then R3 relearns even worse A from R2**.

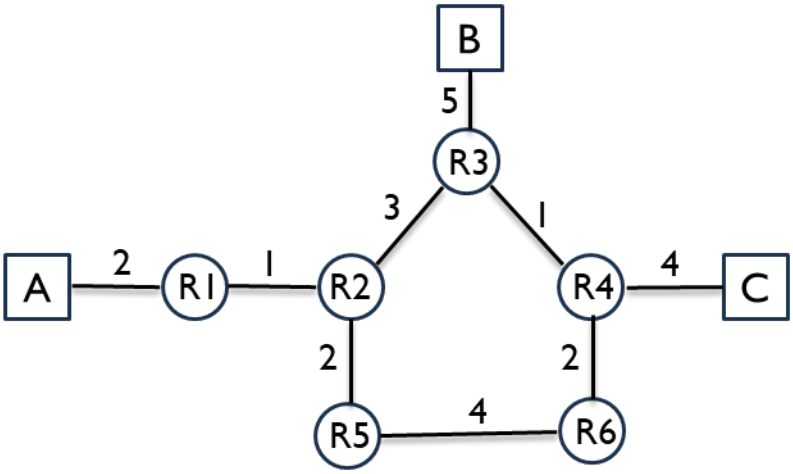
Why split horizon does not stop it. Split horizon prevents a router from advertising a route **back on the interface from which it learned that route**. So **R3 cannot tell R2 about A** when R3's next hop for A is R2. But **R4 can tell R2 about A**, because **R4 learned A via R3**, not via R2, so split horizon does not block the **R4** → **R2** advertisement. **State snapshot:**

Right before t': **R2** has **A via R1**. **R3** has **A via R2**. **R4** has **A via R3**, cost **10**.
At $t' + 1$: **R2** changes to **A via R4**, cost **15**.

At $t' + 2$: **R3** changes to **A via R2**, cost **16**.
This is the start of the **count-to-infinity** pattern.

Q3. (10 pts) Link-State (link failure)

Consider a network with three hosts (A, B, C) and routers R1–R6. Routers are running **link state** algorithm. Link costs equal propagation delay in seconds.



(a) (2 pts) The link R3–R4 fails. R3 and R4 have recomputed their routes but have not yet sent updates. Suppose R3 and R4 never send any updates. What route does a packet from A to C take?

(b) (3 pts) The link R5–R6 fails instead of R3–R4. R5 and R6 have recomputed their routes but have not yet sent updates. Suppose R5 and R6 never send any updates. What route does a packet from A to C take?

(c) (5 pts) At t = 0 a packet from A arrives at R1, destined for C. At t = 3.5 s the link R3–R4 goes down. R3 and R4 recognize and recompute instantly; routing updates propagate at the link prop delay. List the packet path and complete a timing table for every packet arrival and LSA arrival until the packet reaches C (or is detected as undeliverable).

t (s)	Event
0	Packet arrives at R1

ANS:
Original shortest path A→C: A–R1–R2–R3–R4–C with cost 2+1+3+1+4 = 11. Alternative A–R1–R2–R5–R6–R4–C = 2+1+2+4+2+4 = 15. So A→C uses R3.

(a) R3–R4 fails. Pre-update, only R3 and R4 know. The packet from A leaves R1 normally. R2 still believes shortest path is via R3, sends to R3. R3 has recomputed; its best route to C is R3–R2–R5–R6–R4–C = 3+2+4+2+4 = 15. So R3 sends back to R2. R2 will re-forward to R3 again — routing loop between R2 and R3. A–R1–R2–R3–R2 … Packet does not reach C.

(Note: if you wrote A–R1–R2–R3–R2–R5–R6–R4–C it is also given full credit, since the original problem specification did not have the line “Suppose R3 and R4 never send any updates”.)

(b) R5–R6 fails. R5 and R6 recomputed but rest stale. The original best A→C route does not use R5 or R6 (it goes via R3–R4), so the path is unchanged: A–R1–R2–R3–R4–C.

(c) Path: A–R1–R2–R3–R2–R5–R6–R4–C

t (s)	Event
0	Packet arrives at R1
1	Packet arrives at R2 (R1–R2 prop=1)
3.5	R3-R4 link breaks. R3 reconfigures its route to C as R3–R2–R5–R6–R4–C
4	Packet arrives at R3 (R2–R3 prop=3)
6.5	LSA from R3 arrives at R2. R2 reconfigures its route to C as R2–R5–R6–R4–C
7	Packet arrives at R2
9	Packet at R5 (R2–R5 prop=2)
13	Packet at R6 (R5–R6 prop=4)
15	Packet at R4 (R6–R4 prop=2)
19	Packet at C (R4–C prop=4)

Q4. (16 pts) Longest Prefix Matching

A router has 4 ports, with routing table shown below. The routes are written using canonical CIDR notation. Use these rules in order: first longest prefix match, then lowest cost, then smallest port number. If no prefix matches, use the **default route on port 1**. Given the routing table below, for each destination IP address, determine **the output port** to forward it, and **the routing table entry** used to make that routing decision, and with a brief explanation.

Routing table

Port	Destination	Cost
1	1.0.0.0/8	10
1	2.1.0.0/16	15
1	2.2.192.0/20	12
1	4.0.0.0/8	10
1	2.2.0.0/17	15
2	1.1.0.0/16	8
2	2.2.128.0/17	14
2	4.0.0.0/8	8
3	3.0.0.0/8	10
3	2.2.192.0/20	13
3	1.0.10.0/24	8
4	3.4.0.0/16	11
4	1.1.0.0/16	8
4	2.2.0.0/17	14

You can find some useful binary conversions in the table below.

Decimal	Binary

192	11000000
128	10000000
96	01100000
208	11010000
64	01000000
32	00100000

(a) (2 pts) A packet with destination 3.4.0.1

(b) (2 pts) A packet with destination 4.0.0.1

(c) (2 pts) A packet with destination 2.2.208.1

(d) (2 pts) A packet with destination 2.3.0.10

(e) (2 pts) A packet with destination 2.2.204.13

(f) (2 pts) A packet with destination 1.1.21.7

(g) (2 pts) A packet with destination 2.2.96.22

(h) (2 pts) A packet with destination 1.0.10.5

ANS:

(a) 3.4.0.1 goes to Port 4
Explanation: 3.4.0.1 matches both 3.0.0.0/8 and 3.4.0.0/16, and the /16 route on port 4 is the longest-prefix match.

(b) 4.0.0.1 goes to Port 2
Explanation: 4.0.0.1 matches 4.0.0.0/8 on both ports 1 and 2, so the router picks the lower-cost route on port 2.

(c) 2.2.208.1 goes to Port 2
Explanation: 2.2.208.1 matches 2.2.128.0/17 on port 2.

(d) 2.3.0.10 goes to Port 1
Explanation: No listed prefix matches 2.3.0.10, so the packet uses the default route to port 1.

(e) 2.2.204.13 goes to Port 1

Explanation: 2.2.204.13 matches 2.2.192.0/20 on both ports 1 and 3, and port 1 wins because it has the lower cost.

(f) 1.1.21.7 goes to Port 2

Explanation: 1.1.21.7 matches 1.1.0.0/16 on both ports 2 and 4, and the costs tie, so the lower-numbered port 2 is chosen.

(g) 2.2.96.22 goes to Port 4

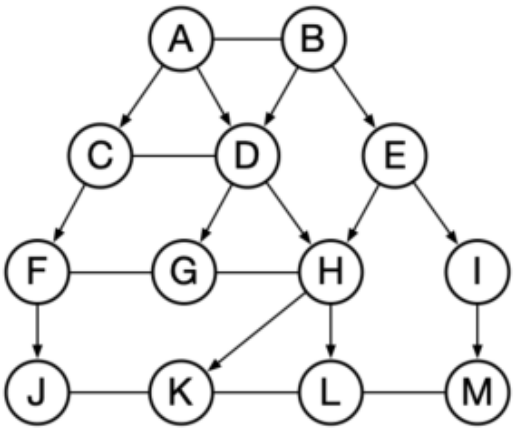
Explanation: 2.2.96.22 matches 2.2.0.0/17 on both ports 1 and 4, and port 4 has the lower cost.

(h) (2 pts) 1.0.10.5 goes to Port 3.

Explanation: 1.0.10.5 matches both 1.0.0.0/8 on port 1 and 1.0.10.0/24 on port 3, and the /24 route is the longer prefix, so the packet is forwarded to port 3.

Q5. (10 pts) Inter-Domain Routing under Gao-Rexford

Consider the AS graph below, where each AS follows the Gao-Rexford import and export policies. Provider to Customer relationship is denoted by arrows; peer to peer relationship is denoted by horizontal lines with dots. (Hint: Every intermediate AS on a legal path must have at least one customer neighbor along that path. Valid AS paths should be valley-free: traffic goes up provider links zero or more times, may cross at most one peer link, and then goes down customer links zero or more times.)



(a) (6 pts) Determine the path (if any) between the following ASes. If there is no path possible, write "None".

(1) J to M

(2) L to F

(3) E to F

(b) (2 pts) When AS D sends BGP advertisements to AS C, which destination ASes can appear in those advertisements?

(c) (2 pts) The link between CD fails, what is the path from J to L?

ANS:

(a)

(1) J to M? J → F → C → A → B →E →I→M
(2) L to F? L →H → D → C → F
(3) E to F? E → B → A → C → F

(b) D, G, H, K, L

Explanation: The reason is pure export policy under Gao-Rexford: When an AS exports routes to a peer, it can export its own routes and the routes of its customers, but not routes learned from other providers or peers. G and H are customers of D, and K and L are downstream under those customers. That means D can advertise reachability to: D itself, G, H, K, L.

(c) J → F → C → A → D → H → L

Q6. (14 pts) TCP Sequence Diagram (fill in seq number)

Consider the TCP sequence diagram with sequence numbers in Bytes. Assume each message segment size is 10 bytes; send window (cwnd) size = 20 bytes. (Assume there is no congestion control, so the send window (cwnd) size is constant.) Upon detecting a packet loss, TCP retransmits the leftmost unacknowledged segment. Give the sequence numbers for each message and ACK.

(a) Timeout scenario

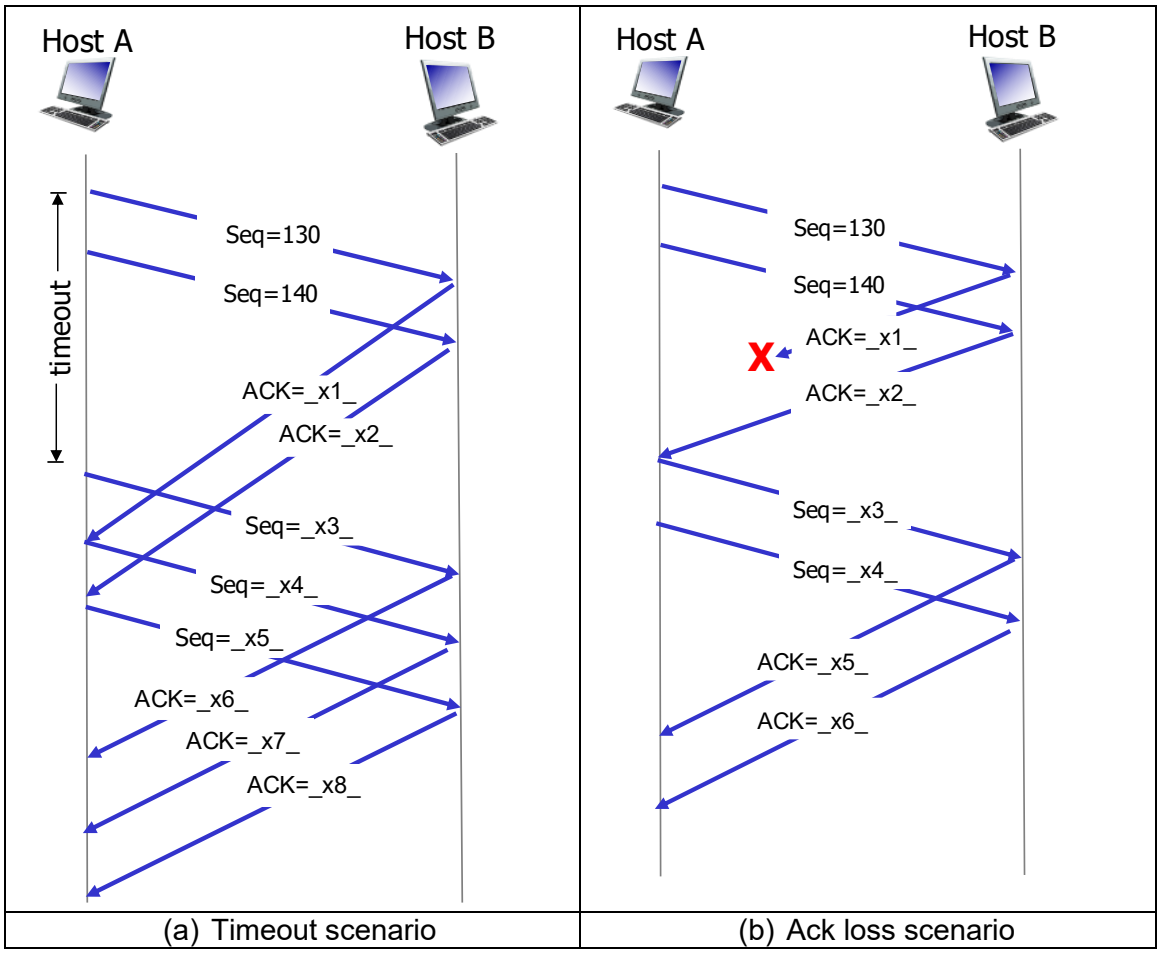
x1 = _____, x2 = _____, x3 = _____, x4 = _____,

x5 = _____, x6 = _____, x7 = _____, x8 = _____.

(b) Ack loss scenario

x1 = _____, x2 = _____, x3 = _____, x4 = _____,

x5 = _____, x6 = _____.



(a) Timeout scenario | (b) Ack loss scenario

ANS:

(a) Timeout scenario

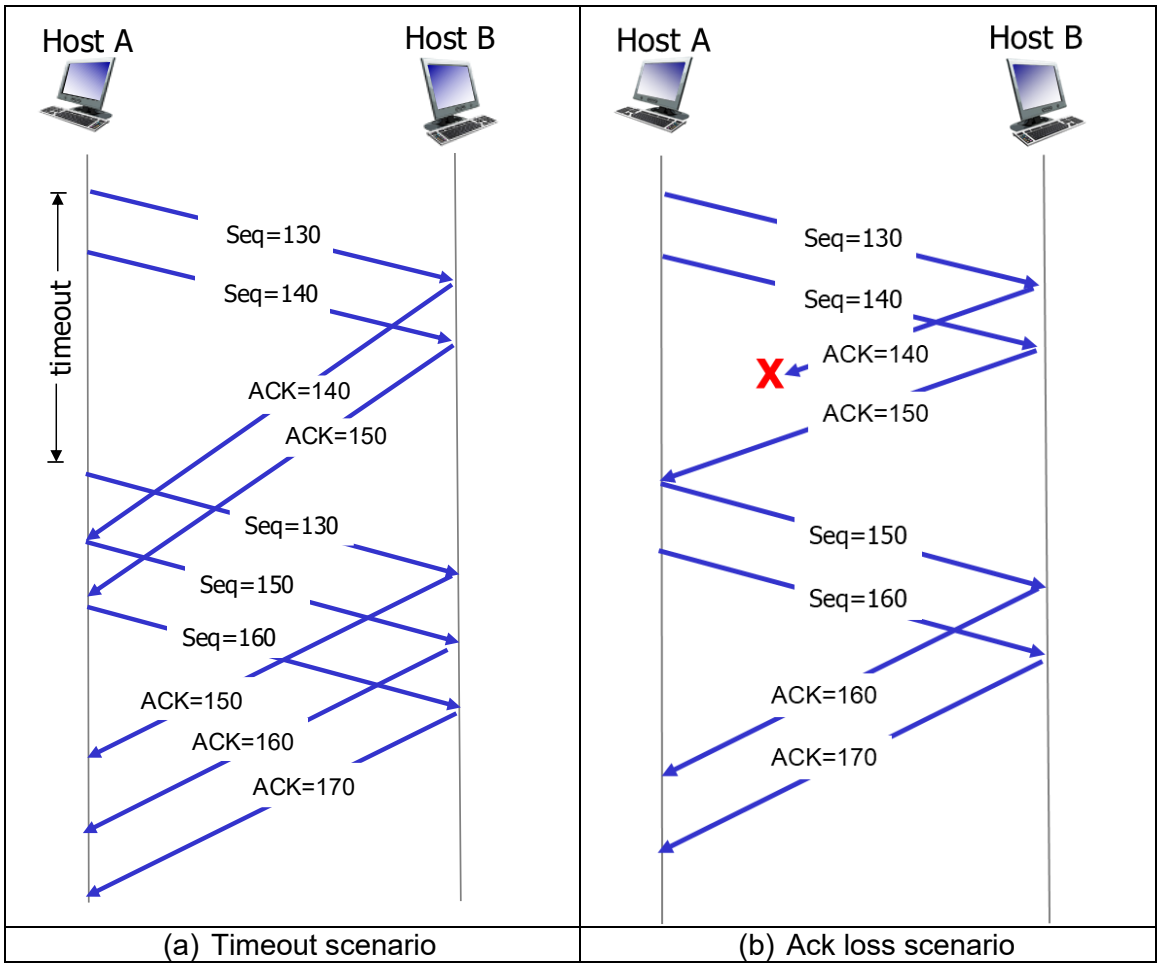
x1 = 140, x2 = 150, x3 = 130, x4 = 150,

x5 = 160, x6 = 150, x7 = 160, x8 = 170

(b) Ack loss scenario

x1 = 140, x2 =150, x3 = 150, x4 = 160,

x5 =160, x6 = 170.



(a) Timeout scenario | (b) Ack loss scenario

Q7. (10 pts)TCP Congestion Control (CWND vs time)

Consider the figure that plots the evolution of TCP's congestion window at the beginning of each time unit (where the unit of time is equal to number of RTTs). The (x, y) coordinates of key points are marked in the figure. TCP sends a "flight" of packets of size cwnd at the beginning of each time unit. The initial value of cwnd is 1. Assume: **TCP new Reno** (slow start + congestion avoidance + fast recovery. Lecture slide attached below for your reference). Loss detected via triple duplicate ACK or timeout. Assume no gap between detecting a packet loss and sending out the next packet.

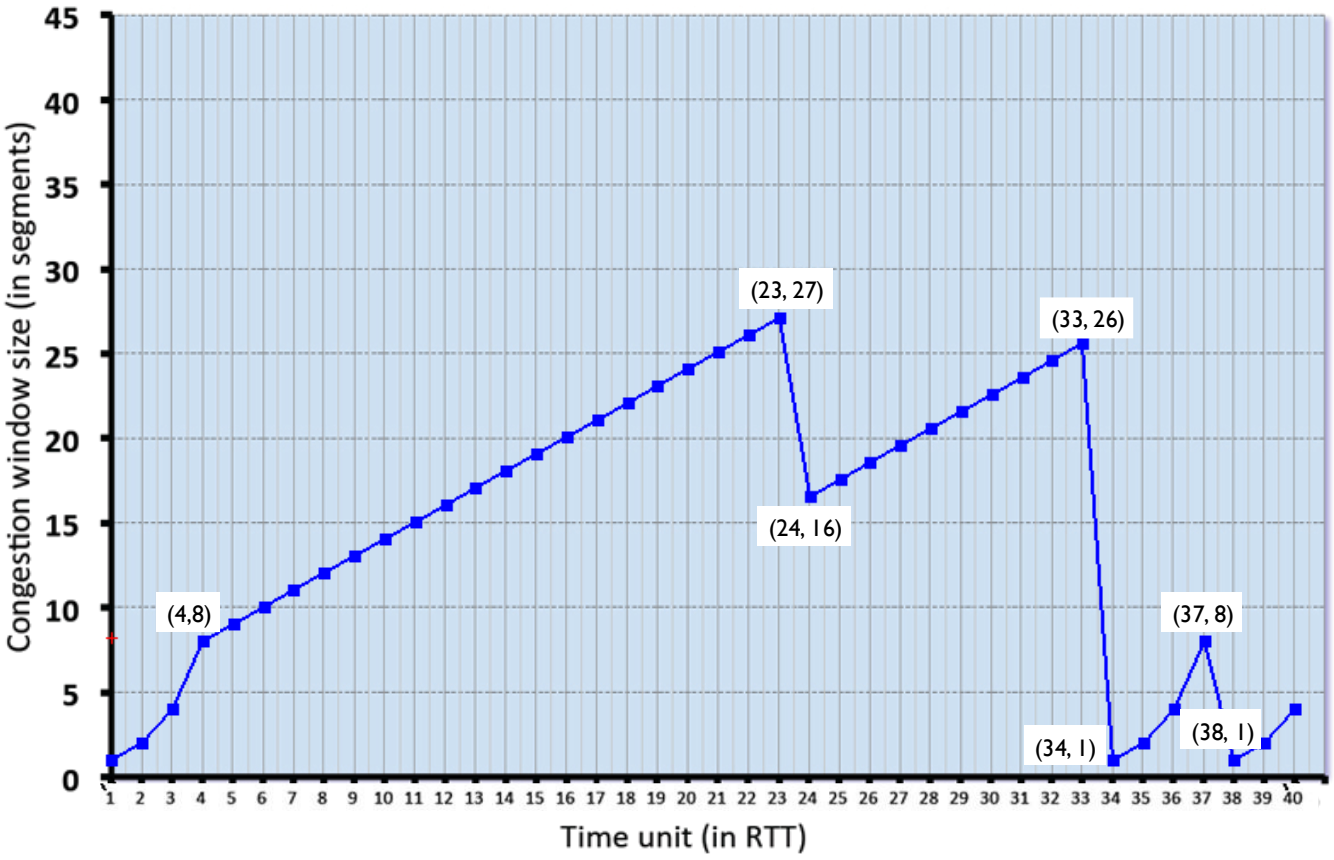
Fast Recovery: Implementation

Conceptually: When a duplicate ack arrives, *artificially extend* the window to let the sender send one more packet.

Implementation:

- When we receive 3 duplicate acks:
  - *SSTHRESH* ← *CWND*/2
  - *CWND* = *CWND*/2 + 3 *(artificially extend for the 3 duplicate acks)*
- While in fast recovery mode, when we receive a duplicate ack:
  - *CWND* = *CWND* + 1 *(artificially extend for each duplicate ack)*
- Exit fast recovery when we receive a new, non-duplicate ack:
  - *CWND* = *SSTHRESH* *(back to 0.5 × rate when the loss happened)*

Note: if cwnd is an odd number, then take the floor $\lfloor \frac{cwnd}{2} \rfloor$.



Q7.1 Identify the RTT intervals where slow start is operating.

Q7.2 Identify the RTT intervals where congestion avoidance is operating.

Q7.3 At which RTT does the first loss event occur? What type is it?

Q7.4 At which RTT does the second loss event occur? What type is it?

Q7.5 What is the initial ssthresh (before time 1)?

Q7.6 What are the values of ssthresh and cwnd immediately after the first loss?

Q7.7 What are the values of ssthresh and cwnd immediately after the second loss?

Q7.8 What is the maximum cwnd reached before each loss event?

Q7.9 Based on the graph, does TCP return to slow start after the first loss event?

Q7.10 How many segments are sent during RTTs 1–6?

ANS:

Q7.1 Slow start intervals

RTT 1–4, RTT 34–37, RTT 38–40 (restart after timeout)

Q7.2 Congestion avoidance intervals

RTT 4–23, RTT 24–33 (After fast recovery, TCP continues in congestion avoidance)

Q7.3 First loss

Occurs at RTT = 23, cwnd before loss = 27. ssthresh = [27 / 2] = 13, cwnd after = ssthresh + 3 = 16

Type: Triple duplicate ACK (fast retransmit/recovery)

Q7.4 Second loss

Occurs at RTT = 33, cwnd drops from 26 → 1

Type: Timeout

(After the first loss, TCP remains in congestion avoidance; after the second loss, TCP resets to $cwnd = 1$ and re-enters slow start phase.)

Q7.5 Initial ssthresh

Slow start stops at cwnd = 8,so ssthresh = 8

Q7.6 After first loss (Reno behavior)

cwnd before loss = 27: ssthresh = ⌊27 / 2⌋ = 13, cwnd after = ssthresh + 3 = 16

Q7.7 After second loss (timeout)

cwnd before loss = 26, ssthresh = ⌊26 / 2⌋ = 13, cwnd after = 1

Q7.8 Maximum cwnd before losses

First loss → 27, Second loss → 26, Third loss → 8

Q7.9 After the first loss at RTT = 23: cwnd drops from 27 to 16, this indicates triple duplicate ACK, not timeout. After this drop: cwnd increases linearly, not exponentially. This shows TCP enters congestion avoidance, not slow start

Q7.10 Segments sent (RTT 1–6)
cwnd values:
RTT 1 → 1
RTT 2 → 2
RTT 3 → 4
RTT 4 → 8
RTT 5 → 9
RTT 6 → 10
Total:
1 + 2 + 4 + 8 + 9 + 10 = 34 segments

Q8. (8 pts) CBC Block Cipher

Consider the 3-bit block cipher in the Table below

Plain	000	001	010	011	100	101	110	111
Cipher	111	110	101	100	011	010	000	001

Suppose the plaintext is 010111010.

(a) (2 pts) Without Cipher Block Chaining (CBC), what is the ciphertext? Show the calculation process.

(b) (6 pts) With CBC and IV = 111, what is the ciphertext? Show the calculation process.

ANS:

(a) Without CBC.

Encrypt each block directly:

- 010 -> 101.
- 111 -> 001
- 010 -> 101

So the ciphertext is: **101001101**

(b) With CBC.

CBC encryption rule:

- C1 = E(P1 ⊕ IV)
- C2 = E(P2 ⊕ C1)
- C3 = E(P3 ⊕ C2)

Block 1

- P1 = 010
- 010 ⊕ 111 = 101
- E(101) = 010

So: **C1 = 010**

Block 2

- P2 = 111
- 111 ⊕ 010 = 101
- E(101) = 010

So: **C2 = 010**

Block 3

- P3 = 010
- 010 ⊕ 010 = 000
- E(000) = 111

So: **C3 = 111**

Therefore, the CBC ciphertext is:

010010111

Q9. (7 pts) Diffie-Hellman

Suppose Alice and Bob wish to do Diffie-Hellman key exchange. Alice and Bob agree on a prime $p = 11$ and a generator $g = 2$.

- Alice chooses her secret number (private exponent) as $a = 3$.
- Bob chooses his secret number (private exponent) as $b = 4$.

Show the step-by-step calculation process:

(a) (2 pts) The value Alice sends to Bob.

(b) (2 pts) The value Bob sends to Alice.

(c) (3 pts) The final shared secret computed by both sides.

ANS:

Alice sends: **8,** Bob sends: **5,** Shared secret: **4**

Alice computes:

$$A = g^a \bmod p = 2^3 \bmod 11 = 8$$

So Alice sends **8** to Bob.
Bob computes:

$$B = g^b \bmod p = 2^4 \bmod 11 = 16 \bmod 11 = 5$$

So Bob sends **5** to Alice.
Bob computes the shared secret:

$$S = A^b \bmod p = 8^4 \bmod 11$$

Now:

- $8^2 = 64 \bmod 11 = 9$
- $8^4 = 9^2 = 81 \bmod 11 = 4$

So Bob gets:

$$S = 4$$

Alice computes the shared secret:

$$S = B^a \bmod p = 5^3 \bmod 11$$

Now:

- $5^2 = 25 \bmod 11 = 3$
- $5^3 = 5 \cdot 3 = 15 \bmod 11 = 4$

So Alice gets:

$$S = 4$$