CS 168 — Introduction to the Internet

Summer 2025 — Final Exam

Solutions last updated: Aug 15, 2025

PRINT Your Name: ______________________________

PRINT Your Student ID: ______________________________

You have 170 minutes. There are 8 questions of varying credit. (100 points total)

Question:	1	2	3	4	5	6	7	8	Total
Points:	14	16	10	16	13	13	10	8	100

For questions with **circular bubbles**, you may select only one choice.

- ○ Unselected option (Completely unfilled)
- ◉ Don't do this (it will be graded as incorrect)
- ● Only one selected option (completely filled)

For questions with **square checkboxes**, you may select one or more choices.

- ■ You can select
- ■ multiple squares
- ☑ Don't do this (it will be graded as incorrect)

Anything you write outside the answer boxes or you ~~cross out~~ will not be graded. If you write multiple answers, your answer is ambiguous, or the bubble/checkbox is not entirely filled in, we will grade the worst interpretation.

Honor Code: Read the honor code below and sign your name.

I understand that I may not collaborate with anyone else on this exam, or cheat in any way. I am aware of the Berkeley Campus Code of Student Conduct and acknowledge that academic misconduct will be reported to the Center for Student Conduct and may further result in, at minimum, negative points on the exam.

SIGN your name: ______________________________

Doodle credit: Andrea Lou

Clarifications

- Q4: Top level text "R1, R2, and R3" should be "R1, R3, and R4 are using NAT"

Q1 *Potpourri* **(14 points)**

Q1.1 (1 point) Consider TCP with congestion control, as seen in lecture. The window size should always be set to RTT × bandwidth to take advantage of the network capacity.

- ○ True
- ● False
- ○ Not enough information

> **Solution:** False, because this describes the bandwidth-delay product which is the max window size, but TCP window size should also be adjusted based on flow control (receiver's buffer capacity) and congestion control signals.

Q1.2 (2 points) Select all that apply. An end host participant in TCP, with no congestion control, has to maintain and update:

- ■ Which packets have been sent and not acknowledged.
- ■ How much longer is on the timer before a resend is needed.
- ■ A buffer of received out of order packets.
- □ A buffer of all received packets.
- □ Congestion window size.
- ○ None of the above

> **Solution:** Even without congestion control, TCP needs to track unacked packets for reliability, maintain retransmission timers, and buffer out-of-order packets for in-order delivery. Data is not needed after data is passed on to the application layer therefore a buffer of all received packets should not be kept. Congestion window size is only needed if there is congestion control.

Q1.3 (1 point) Assuming routers have a single FIFO buffer and are running TCP with congestion control, increasing the queue size at routers is always a good way to decrease network delays.

- ○ True
- ● False
- ○ Not enough information

> **Solution:** False, because increasing queues can cause bufferbloat. Larger queues increase queueing delay and can actually hurt TCP's congestion control by delaying loss signals, resulting in full buffers and higher latency.

Q1.4 (1 point) A user only knows about one DNS recursive resolver, which goes down. At this point, the user cannot reach `www.google.com`.

- ○ True
- ○ False
- ● Not enough information

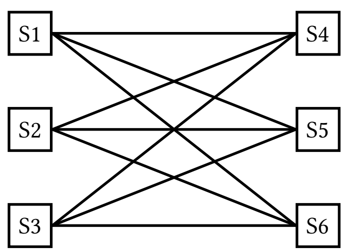
> **Solution:** Not enough information, because this depends on whether `www.google.com` is already contained in the user's stub resolver's cache.

Q1.5 (2 points) Select all of the following that Content Delivery Networks (CDNs) can help with.

- ☐ Loading private (i.e. `Cache-Control` header is set to private) user specific data quickly.
- ■ Serving a small company's (does not own infrastructure) webpage fast and globally.
- ■ Serving a large company's (owns infrastructure) webpage fast and globally.
- ■ Reducing the need for network bandwidth.
- ■ Reducing the load on the origin server.
- ○ None of the above

Solution: Content Delivery Networks (CDNs) cache content closer to users. They do not cache user specific data like passwords, or icons which are only cached in private caches. They can be rented to serve a small companies information or deployed by a large company. They reduce the need for network bandwidth since the backbone does not need to be provisioned for all traffic, instead it can be served closer to users. They reduce load on the origin server by serving requests closer to the user.

Q1.6 (2 points) What is the bisection bandwidth of the network shown below, with links of bandwidth B, and 6 total switches that each have 3 physical ports?



- ○ B
- ● $5B$
- ○ $9B$
- ○ $3B$
- ○ $6B$
- ○ Not enough information

Solution: Bisection bandwidth is the least amount of bandwidth connecting two halfs of the network (3 and 3). Cutting S2-S4, S2-S5, S1-S6, S3-S4, S3-S5 disconnects the two halfs, therefore 5B is the answer. Partial credit for 9B (3 switches $*$ 3 links $*$ bandwidth B).

Q1.7 (1 point) Encapsulation and decapsulation in datacenter networks can help with both multi-tenancy and managing virtualization.

- ● True
- ○ False
- ○ Not enough information

Solution: True, encapsulation enables overlay networks that isolate tenant traffic and allow virtual networks to span physical infrastructure, supporting both multi-tenancy and virtualization.

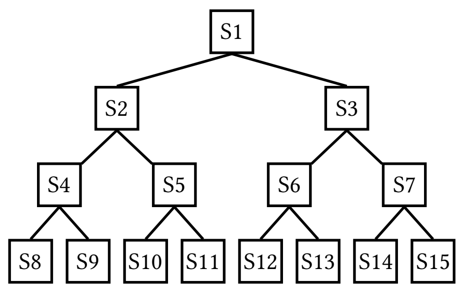
Q1.8 (1 point) In a wide-area network with low-bandwidth links and non-zero queuing delay at routers, Remote Direct Memory Access (RDMA) can increase file transfer speed.

- ○ True
- ● False
- ○ Not enough information

Solution: False, RDMA (remote direct memory access) bypasses the CPU and reduces latency in high-speed networks. In low-bandwidth WANs, the bottleneck is link capacity, not the CPU, so RDMA won't help with end-to-end latency.

Q1.9 (2 points) Consider the physical topology below. Calculate the average stretch (i.e. $\frac{\text{underlay hops}}{\text{overlay hops}}$) of an overlay ring topology designed for collective operations. Assumptions:

- All tree leaf nodes and the root node participate in the ring topology.
- Each physical and virtual link is cost 1.



- ○ 28/9
- ○ 10/3
- ○ 9
- ○ 5/3
- ○ 3
- ● Not enough information

Solution: Not enough information, because the order of the nodes in the overlay ring, which is not specified, is crucial for calculating stretch.

Q1.10 (1 point) Calculate the predicted Signal to Interference and Noise Ratio ($SINR_{dB}$) for a signal with a power of 10 Watts in a room with an average measured background noise of 200 mWatts and interference noise of 800 mWatts.

- ○ 0.01 dB
- ○ 20 dB
- ○ 40 dB
- ● 10 dB
- ○ 30 dB
- ○ Not enough information

Solution:
$10 \times \log_{10}\left(\frac{10}{.8+0.2}\right) = 10 \text{ dB}$

Q2 *TCP Congestion Control: No Fair!* **(16 points)**

Fairness in life (including on the Internet) is hard to define and quantify. In this **entire** question, you can assume that fairness means that all **flows** using a link will eventually converge to equal link bandwidth.

For subparts **Q2.1 to Q2.2**, fill in the blank to form the fairest congestion control adjustment algorithm.

Q2.1 (2 points) ______ Increase

- (●) Additive
- ( ) Multiplicative

Solution: Additive

Explain your selection in approximately 15 words or less.

Solution: Additive increase goes up with a slope of 1, converging to the fairness point

Q2.2 (2 points) ______ Decrease

- ( ) Additive
- (●) Multiplicative

Solution: Multiplicative

Explain your selection in approximately 15 words or less.

Solution: Multiplicative decrease goes down quickly towards the origin, converging to fairness point.

Q2.3 (1 point) In the TCP congestion control algorithm discussed in class, fairness depends on RTT.

- (●) True
- ( ) False
- ( ) Not enough information

Solution: The rate at which a sender can increase its window size is inversely related to the RTT. During congestion avoidance, the window size is increased once every RTT. This means that a connection with a shorter RTT will increase faster, getting more bandwidth.

For subparts **Q2.4 to Q2.5**, imagine you are a greedy TCP developer who does not care about fairness. Which of the following adjustment algorithms gets you the most link bandwidth? Assume everyone else participating in the network is using the TCP congestion control algorithm from class.

Q2.4 (1 point) What algorithm should we use for the increase method?

- ○ $\text{CWND} = \left(\frac{1}{2}\right) \times \text{CWND}$
- ○ $\text{CWND} = 1 + \text{CWND}$
- ● $\text{CWND} = 2 \times \text{CWND}$
- ○ $\text{CWND} = 10 + \text{CWND}$
- ○ $\text{CWND} = 2 + \text{CWND}$
- ○ $\text{CWND} = \text{CWND} - 10$

Solution: Aggressively take bandwidth and decreases as little as possible on loss.

Q2.5 (1 point) What algorithm should we use for the decrease method?

- ○ $\text{CWND} = \left(\frac{1}{2}\right) \times \text{CWND}$
- ○ $\text{CWND} = \text{CWND} - 2$
- ● $\text{CWND} = \text{CWND} - 1$
- ○ $\text{CWND} = \lceil \log(\text{CWND}) \rceil$
- ○ $\text{CWND} = \text{CWND} - 10$
- ○ $\text{CWND} = \left(\frac{1}{4}\right) \times \text{CWND}$

Solution: Aggressively take bandwidth and decreases as little as possible on loss.

You have heard that using delay as an indication of congestion can sometimes result in better performance because adjustments occur before loss occurs. For subparts **Q2.6 to Q2.8**, consider the following delay based congestion control algorithm. Assume the topology does not change and RTT is in seconds.

- The `BaseRTT` is the accurate expected RTT with no queueing delay.
- Flow rate is calculated by $\frac{\text{CWND}}{\text{RTT}}$
- When the (expected flow rate − the measured flow) × `BaseRTT` is greater than 5, decrease CWND by 1
- When the (expected flow rate − the measured flow) × `BaseRTT` is less than 1, increase CWND by 1

Q2.6 (2 points) If all end hosts in the network use this delay based algorithm, bandwidth allocation is fair. Assume the `BaseRTT` is the same for all flows in the network.

● True ○ False ○ Not enough information

Why? Explain your answer in approximately 20 words or less.

Solution: Bandwidth allocation is fair because all end hosts will increase at the same rate until delay occurs, then they will all slowly decrease at the same rate.

Q2.7 (2 points) If the **`BaseRTT`** for different flows is different, and all the end hosts in the network use the above delay based algorithm, bandwidth allocation is fair.

○ True ● False ○ Not enough information

Why? Explain your answer in approximately 20 words or less.

Solution: This is not technically fair since the window of possible queue size between 1 and 5 is large enough that it could converge to non-equal rates. Alt answer: this delay based congestion control algorithm is generally fair since it uses rates, therefore if the queue contribution is equal there will be equal bandwidth usage because it is normalized by RTT. This is because an increase in RTT affects both flows proportionally to their BaseRTT and absolute queuing delay is shared.

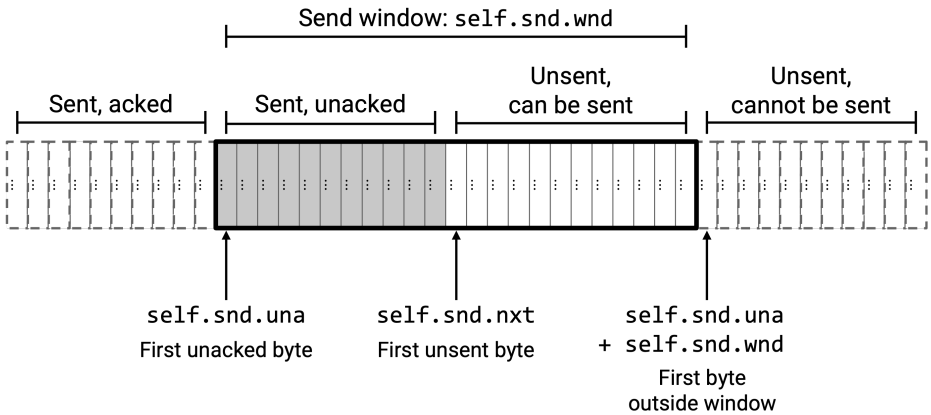
Q2.8 (2 points) If only one flow in the network uses this delay based algorithm, and the rest of the flows use the TCP congestion control algorithm from class, bandwidth allocation is fair.

○ True ● False ○ Not enough information

Why? Explain your answer in approximately 20 words or less.

Solution: This is not fair because the TCP congestion control algorithm from class will not react to delay and continue to take bandwidth until there is loss. Meanwhile, the delay based approach will backoff much earlier. Therefore the delay based flow will get less bandwidth.

Mini Project Quiz: For **Q2.9 to Q2.10**, consider the Transport Project 3 from class. The following project information is copied from the specification and included here for your reference and convenience:



The `TXControlBlock` object has the following relevant instance variables:

- `self.snd.iss`: Your initial sequence number.
- `self.snd.una`: The oldest unacknowledged sequence number that you sent.
- `self.snd.nxt`: The next sequence number you should send.
- `self.snd.wnd`: The current size of your send window, determined by how much buffer space the other side (recipient) has left.

If you receive a TCP segment `seg`, you can extract these fields from its header:

- `seg.seq`: The sequence number.
- `seg.ack`: The ack number.
- `seg.win`: The advertised window (how much buffer space is left on the other side).

Q2.9 (2 points) Calculate the amount of unsent data bytes which can be sent.

- ◯ `remaining = self.snd.iss |PLUS| self.snd.wnd |MINUS| self.snd.nxt`
- ◯ `remaining = self.snd.nxt |PLUS| self.snd.wnd |MINUS| self.snd.una`
- ◯ `remaining = self.snd.una |PLUS| self.snd.wnd |MINUS| seg.seq`
- ● `remaining = self.snd.una |PLUS| self.snd.wnd |MINUS| self.snd.nxt`

Solution: The window section of the diagram can be calculated by adding `self.snd.una` to `self.snd.wnd` then subtract `self.snd.nxt` to get the "unsent, can be sent" section.

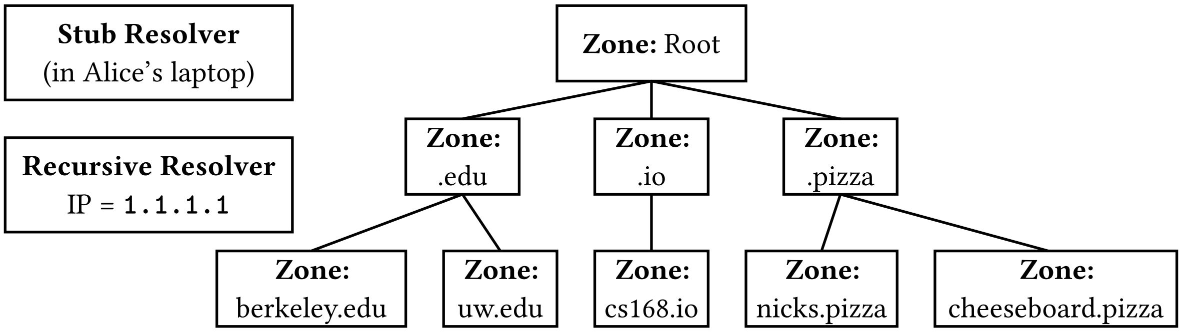
Q2.10 (1 point) In the `check_ack` function, we should only call `self.handle_accepted_ack` if the ACK number in the received segment corresponds to a packet that was sent but has not yet been acknowledged. Which of the following correctly bounds a sent, but unacked packet?

- ● `self.snd.una |LT| seg.ack and seg.ack |LE| self.snd.nxt`
- ○ `self.snd.una |LT| seg.ack and seg.ack |LT| self.snd.nxt`
- ○ `self.snd.una |LE| seg.ack and seg.ack |LE| self.snd.nxt`
- ○ `self.snd.una |LE| seg.ack and seg.ack |LT| self.snd.nxt`

Solution: The ack number indicates the next expected byte (i.e. the first unreceived byte) so an ack corresponds to byte number `seg.ack`−1 therefore we should check if `self.snd.una` is less than `seg.ack` and `seg.ack` is less than or equal to `self.snd.nxt` for the "Sent unacked" region

Q3 *DNS: Resolving... The Best Local Pizza* **(10 points)**

Consider the following DNS name server hierarchy, stub resolver, and recursive resolver. Each box in the server hierarchy represents a zone. Alice, a CS168 student, is typing URLs into her browser.



Q3.1 (1 point) What protocol helps the stub resolver learn the IP address of the recursive resolver?

- ● DHCP
- ○ DNS
- ○ ARP
- ○ BGP

Solution: DHCP provides network configuration to clients including the IP address(es) of DNS recursive resolvers).

Q3.2 (2 points) Starting with empty caches everywhere, Alice searches for **`www.nicks.pizza`**. How many queries must the recursive resolver make to get the IP address for **`www.nicks.pizza`**?

- ○ 1
- ○ 2
- ● 3
- ○ Not enough information

Solution: With empty caches: (1) query root for .pizza NS, (2) query .pizza NS for nicks.pizza NS, (3) query nicks.pizza NS for www.nicks.pizza. The nicks.pizza NS tells the recursive resolver the IP.

Q3.3 (2 points) Continuing from **Q3.2** (i.e. caches not cleared), Alice searches for **`www.cheeseboard.pizza`**. How many queries must the recursive resolver make to get the IP for **`www.cheeseboard.pizza`**?

- ○ 1
- ● 2
- ○ 3
- ○ Not enough information

Solution: The resolver already has the .pizza NS cached from the previous query. It needs to then (1) query .pizza NS for cheeseboard.pizza NS, (2) query cheeseboard.pizza for ww.cheeseboard.pizza. The cheeseboard.pizza NS tells the recursive resolver the IP.

Q3.4 (2 points) Alice orders pizza from the best of the two places and heads to Soda to meet her friends. She attaches her laptop to the WiFi network and searches **`www.cs168.io`** in her browser. How many queries must the recursive resolver make to learn the IP of **`www.cs168.io`**?

○ 1 ○ 2 ○ 3 ● Not enough information

> **Solution:** Not enough information: It isn't known if the new network's recursive resolver (which may be different from Alice's resolver at 1.1.1.1) has anything cached. Recursive resolvers are typically shared among many users, so it's likely to have popular (at least in Soda) servers like cs168.io cached.

Subparts **Q3.5 to Q3.7**: Alice is searching for Connie's webpage in the CS168 site. The stub resolver makes a request to the recursive resolver. Who sends each of the following records to the recursive resolver?

Q3.5 (1 point) **Record:** `connie.cs168.io A 192.160.161.188`

○ Root name server ○ `.io` name server ● `cs168.io` name server

> **Solution:** This is the answer record for Connie's IP. The cs168.io name server owns this zone and has authority over all ∗.cs168.io records, so it sends this one.

Q3.6 (1 point) **Record:** `cs168-dns.io A 192.160.162.189`

○ Root name server ● `.io` name server ○ `cs168.io` name server

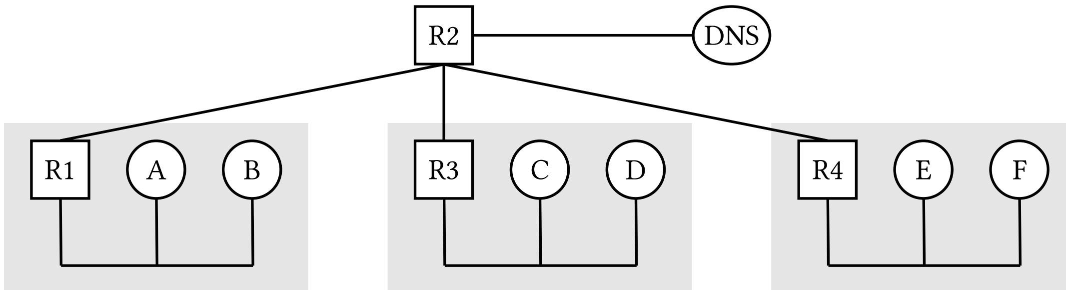
> **Solution:** The .io server tells the recursive resolver that cs168.io is handled by cs168-dns.io, when it does this it also needs to provide the IP for cs168-dns.io, so the .io nameserver sends this A record.

Q3.7 (1 point) **Record:** `cs168.io NS cs168-dns.io`

○ Root name server ● `.io` name server ○ `cs168.io` name server

> **Solution:** The .io nameserver knows who's authoritative for cs168.io and tells the recursive resolver "go ask cs168-dns.io for anything.cs168.io", so the io nameserver sends this NS record.

Q4 *End-to-End: Three's a Crowd* **(16 points)**



In this question, consider this network topology containing three different subnets.

Routers R1, R2, and R3 are using NAT with Port Address Translation.

There is one DNS resolver with IP 8.8.8.8 in the network, labeled DNS. Everyone uses this DNS resolver.

1. R1's private address is 1.1.1.1, and its public address is 192.1.1.1.
   It can allocate the addresses 1.1.1.2, 1.1.1.3, and so on, to the hosts in its subnet.
2. R3's private address is 3.3.3.1, and its public address is 192.3.3.3.
   It can allocate the addresses 3.3.3.2, 3.3.3.3, and so on, to the hosts in its subnet.
3. R4's private address is 4.4.4.1, and its public address is 192.4.4.4.
   It can allocate the addresses 4.4.4.2, 4.4.4.3, and so on, to the hosts in its subnet.

Q4.1 (1 point) Suppose host A joins the network, with all caches empty and no active connections.
In Host A's DHCP Discover request, what is the source IP address?

- ● 0.0.0.0
- ○ 1.1.1.1
- ○ 1.1.1.2
- ○ 255.255.255.255

Solution: Since A does have an IP address yet, the default IP it uses is 0.0.0.0.

Q4.2 (1 point) Which best explains the source port Host A uses when sending a DHCP request?

- ○ Host A uses source port 67 since that's what all hosts use as source ports for DHCP.
- ○ Host A uses the source port that was burned within its hardware.
- ● Host A picks a random source port.
- ○ Host A doesn't need to add a source port when sending a DHCP request.

Solution: When a host uses DHCP to connect to a network, it picks an ephemeral source port at first, which can be randomly selected.

For the remainder of this problem, suppose all six hosts have completed the DHCP handshake, in alphabetical order, one after the other, and have been assigned IP addresses respectively.

Q4.3 (2 points) At this point, which addresses does A know? Assume hosts process all broadcast packets.

- ■ B's private IP
- ☐ R1's public IP
- ■ R1's private IP
- ■ DNS's IP

Solution: A knows B's private IP address because B will broadcast the DHCP configuration it has selected, and B joins the network after A, so A will hear the broadcast from B and note it down.

A knows R1's private IP since R1 is the gateway router for A's subnet, and the gateway router's IP is given to A during the DHCP configuration process.

A knows DNS's IP because the resolver's IP is also given to A during the DHCP configuration process.

Q4.4 (2 points) Suppose A knows B's IP and wants to send a unicast packet to B. What will A do first?

- ○ Make a DNS request, which ultimately returns B's MAC Address.
- ● Make an ARP request, which ultimately returns B's MAC address.
- ○ Make an ARP request, which ultimately returns R1's MAC address.
- ○ Send the packet to B immediately since they're within the same subnet.

Solution: In order for A to send a packet to B, it needs to know B's MAC address (even if A and B are both on the same subnet) in order to send the packet over the wire.

To do this, A makes an ARP request so that B can respond to A with its MAC address.

Q4.5 (2 points) Suppose A knows D's IP and wants to send a unicast packet to D. What will A do first?

- ○ Make a DNS request, which ultimately returns D's MAC Address.
- ○ Make an ARP request, which ultimately returns D's MAC address.
- ● Make an ARP request, which ultimately returns R1's MAC address.
- ○ Send the packet to C immediately since they're within the same subnet.

Solution: Since A and D are both on different subnets, A will be given the gateway router's MAC address (R1's MAC address). This way, if A wants to send a packet to D, it will first send the packet to R1, and eventually from R1 the packet will reach D.

Q4.6 (2 points) Suppose E sends a DNS request to the DNS resolver. Write the **destination IP in the DNS response packet** from this DNS request.

192.4.4.4

Solution: Since we are using NAT, the DNS resolver does not know E's private IP, but instead will think that it's chatting with R1′s public IP, which is 192.4.4.4. R1 does the translation between E's private IP + port to R1′s public IP + port. Therefore, the DNS resolver will send a response to R1′s public IP.

Q4.7 (2 points) Suppose C and D each send a packet to F. Select the true statement.

- ○ R3 forwards both packets to R2 without modifying any of the headers for either packet.
- ○ R3 always uses a randomized source port number for C's and D's packet before sending to R2.
- ○ R3 drops one of the packets due to possible ambiguity when F sends a packet.
- ● R3 forwards a packet with C's source port and a packet with D's (different) source port to R2.

Solution: Since we are using NAT with Port Address Translation, R3 will include both C's and D's source ports in their respective packets before sending them out (we are assuming that the source ports being used for C and D are different). This will help deal with ambiguity further down the road when R3 receives response packets, as both response packets will have destination IP 192.3.3.3, but they will have different destination ports (which will help R3 know where to send these response packets).

Now, suppose E sends a packet to A, and a packet to B.

Q4.8 (1 point) Can E assign the same source port for both of these packets?

● Yes ○ No

Solution: It doesn't matter if E uses the same source port for each of these packets, because E can tell who sent the response packet from the source IP. The actual hosts within the network don't need to consider NAT: outside of the router performing translations, the hosts themselves should have a seamless experience, and not necessarily need to be aware that NAT is being used.

R4 however will have to rewrite at least one of the source ports so it can differentiate the two flows.

Q4.9 (2 points) Can D start a direct TCP connection to E? Explain in approximately 20 words or fewer.

Solution: D cannot start a direct TCP connection to E, because NAT does not support inbound connections (D does not know E's private IP).

Q4.10 (1 point) If F knows E's MAC address, can F send packets to E even if E gets a new IP address?

- (●) Yes
- ( ) No

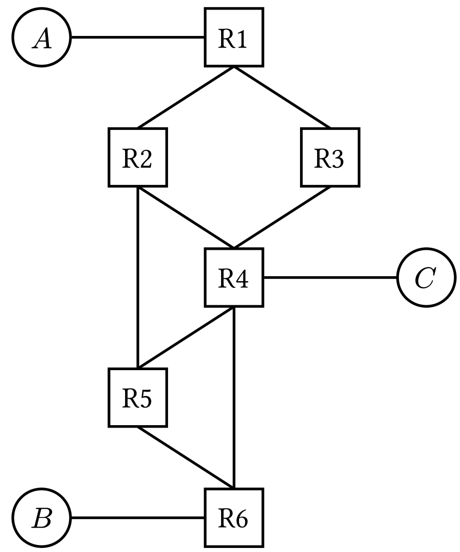
Solution: Since E and F are within the same subnet, only MAC addresses are needed to send a packet from host to another within this subnet.

Q5 *STP: Standard STP* **(13 points)**

Consider running the Spanning Tree Protocol (STP) for the network topology to the right.

Assume the IDs are ordered according to the router labels. For example, R4 has a lower ID than R5.

For the first two subparts, option "R1, to R2" means "the port from R1 going to R2," and likewise for other options. Assume ties are broken by choosing the router with the lowest ID. Assume all the links have a cost of 1.



Q5.1 (2 points) Select all Root Ports.

☐ R1, to R2	☐ R2, to R5	☐ R4, to R3	☐ R5, to R4
☐ R1, to R3	■ R3, to R1	☐ R4, to R5	☐ R5, to R6
■ R2, to R1	☐ R3, to R4	☐ R4, to R6	■ R6, to R4
☐ R2, to R4	■ R4, to R2	■ R5, to R2	☐ R6, to R5

Solution: The root ports will consist of all the ports that lead the router to the shortest path to the root, which in this case is R1.

For R2, the root port will be the one going straight to R1.

For R3, the root port will also be the one going straight to R1.

For R4, since we are assuming ties are broken by lowest ID, it will choose the port that goes directly to R2 rather than to R3.

For R5, the root port will be the one going straight to R2 since that port gets to the root in a number of hops less than if we were to go to the port that leads straight to R4.

For R6, since we are assuming ties are broken by lowest ID, it will choose the port that goes directly to R4 rather than to R5.

Q5.2 (2 points) Select all Blocked Ports.

☐ R1, to R2	☐ R2, to R5	■ R4, to R3	■ R5, to R4
☐ R1, to R3	☐ R3, to R1	☐ R4, to R5	☐ R5, to R6
☐ R2, to R1	☐ R3, to R4	☐ R4, to R6	☐ R6, to R4
☐ R2, to R4	☐ R4, to R2	☐ R5, to R2	■ R6, to R5

Solution: The blocked ports will be the ports that aren't root ports and also aren't designated ports (aka ports that lead you further away from the root node).

Q5.3 (2 points) Select all true statements.

- ■ The Designated Ports in the network should be all the ports not selected in the above subparts.
- ☐ A link can only be disabled when both connected routers agree to disable it.
- ☐ Blocked ports lead to routers that are further away from the root.
- ■ After disabling links with STP, the resulting topology will contain no cycles.
- ○ None of the above

Solution:

Option 1 is true. Every port is either a root port, a blocked port, or a designated port.

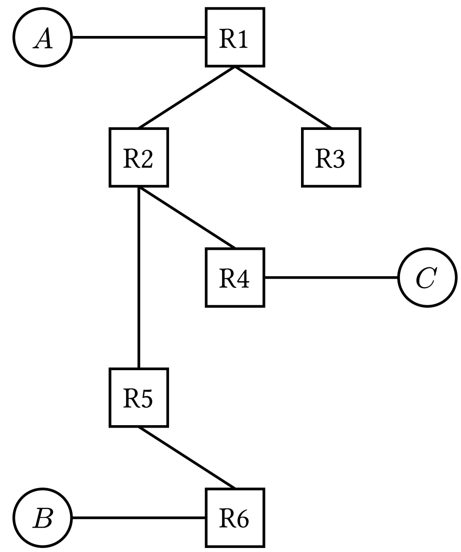
Option 2 is false. A link in STP can only be disabled by the router that is further away from the root.

Option 3 is false. Designated ports lead to routers that are further away from the root. Blocked ports lead to the root but should not be used to prevent a loop.

Option 4 is true. STP is designed such that disabling the right links will result in a tree (no cycles).

Suppose STP has converged. Regardless of your answers to the previous subparts, assume that the following 3 links are disabled (also shown in the diagram to the right):

- R3-to-R4
- R4-to-R5
- R4-to-R6



Switches R1 to R6 are all learning switches.

At the start, the only entry in all forwarding tables is a valid least-cost next-hop to host B.

For each of the remaining subparts, a host will attempt to send a packet to another host.

The packets are sent one after the other. In other words, forwarding table entries created in one subpart carry over to later subparts.

Q5.4 (1 point) A sends a packet to B.

Which of the following will most likely occur from this packet being sent?

- ○ Every switch floods the packet to all neighbors, eventually reaching host B.
- ● The packet goes from A to R1 to R2 to R5 to R6 to B, without reaching any other routers.
- ○ The packet goes from A to R1 to R2 to R4 to C, thus never reaching B.
- ○ A sends the packet to R1 who proceeds to drop it, thus never reaching B.

Solution: Since all forwarding tables have a valid least-cost next-hop to host B, this means the packet will go from A to B in the most efficient route with no flooding.

Q5.5 (2 points) B sends a packet to A.

Select all switches that will receive the packet.

■ R1 ■ R2 □ R3 □ R4 ■ R5 ■ R6

Solution: After 5.4, routers R1, R2, R5, and R6 will have a next-hop to A. Therefore, when B sends a packet to A, no flooding occurs, and the packet will go from B to R6 to R5 to R2 to R1 to A.

Q5.6 (2 points) A sends a packet to C.

Select all switches that will receive the packet.

- [x] R1
- [x] R2
- [x] R3
- [x] R4
- [x] R5
- [x] R6

Solution: Since no routers have a next-hop to C in their forwarding tables, this packet will flood across the network, so all routers will receive this packet.

Q5.7 (2 points) C sends a packet to B.

Select all switches that will receive the packet.

- [ ] R1
- [x] R2
- [ ] R3
- [x] R4
- [x] R5
- [x] R6

Solution: Since all routers at this point have a next-hop to B, the packet will go from C to R4 to R2 to R5 to R6 to B.

Q6 *Datacenters: Making a Mega Datacenter* **(13 points)**

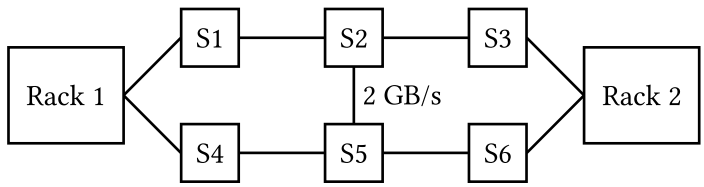
Paola is building a datacenter and is considering a few possible designs.

Q6.1 (3 points) Which topologies supports Rack 1 sending data to Rack 2 at 2 GB/s? Select all that apply.

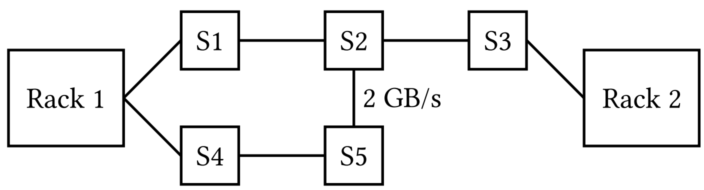
Unmarked edges have a bandwidth of 1 GB/s.

- [x] Topology 1
- [ ] Topology 2
- [x] Topology 3

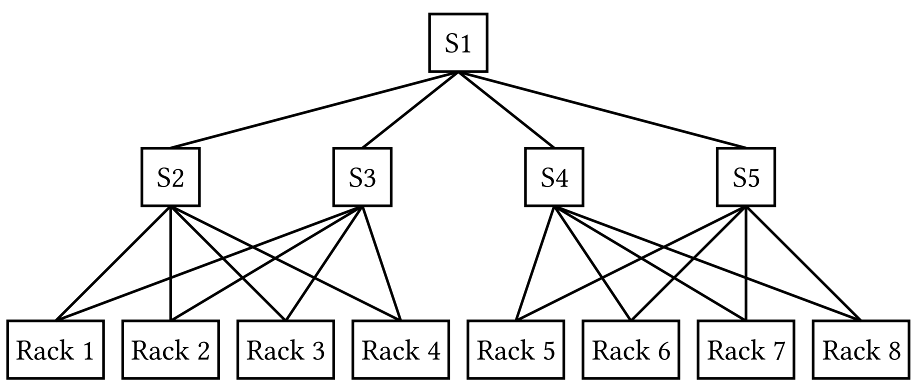
Topology 1:



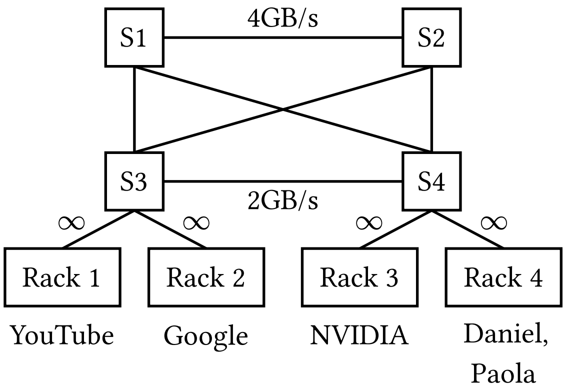
Topology 2:



Topology 3:



Solution: We can split flows larger than 1 GB equally amongst available links using traffic engineering. In Topology 1, the paths S1 - S2 - S3 and S4 - S5 - S6 can each transmit up to 1 GB/s for a total flow of 2 GB/s from Rack 1 to Rack 2 or vice versa. In Topology 2, flows from Rack 2 to anywhere cannot be larger than 1 GB/s because there is only one link connecting it to S3. In Topology 3, if Rack 1 wants to send 2 GBs of data to Rack 5, they can send 1 GB to S2 and 1 GB to S3.



Paola builds a datacenter with an underlay and overlay (as seen in lecture).

Each rack has a virtual switch that interfaces with the virtual machine(s) on that rack.

The links from a rack to a switch have infinite bandwidth.

Q6.2 (2 points) What is the bisection bandwidth?

4GB/s

Solution: Bisection bandwidth is computed as the minimum possible bandwidth of a cut splitting the network into two equal parts. In this topology, we can disconnect S3-to-S1, S3-to-S2, and S3-to-S4.

Q6.3 (2 points) Daniel downloads a large YouTube video (100+ GB).
Select all true statements about using traffic engineering to transfer this large video.

- [x] Sending the traffic across multiple paths could make the transfer faster.
- [x] Sending the traffic across multiple paths could overwhelm the receiver (Daniel).
- [ ] It is possible for Daniel to receive the video at rate 6 GB/s.
- ◯ None of the above

Solution:

(A) is true. Using multiple paths increases the amount of bandwidth available.

(B) is true. Packets could arrive out-of-order, overwhelming the TCP client on Daniel's machine.

(C) is false. S4 can only receive 4 GB/s (sum of its incoming link bandwidths).

Q6.4 (2 points) All of the switches are busy processing Daniel's data transfer. At the same time, Paola wants to download a small 30-second YouTube video.

Select all protocols that could help Paola load the video before Daniel's transfer is done.
Consider each answer independently.

- [ ] TCP
- [x] OpenFlow API Flow Tables
- [x] Packet Priorities
- [x] Encapsulation and Decapsulation
- [x] Equal Cost Multi-Path Routing (ECMP)
- [ ] Constrained Shortest Path First (cSPF)

Solution: TCP doesn't help the response to Paola's request transmit faster as it doesn't affect the path chosen.

Packet Priorities can allow the response for the 30-second Short to have higher priority than the lecture, guaranteeing that Paola's Packets are processed before Daniel's.

Equal Cost Multi-Path Routing under the uniform cost metric (all paths have the same cost) restricts future packets of Daniel's video response to a single path and allows switches to route Paola's response to a different path to S4 than the one taken by Daniel's video, as they have a different 5-tuple. Then, S4 can instantly transmit the Short to Paola.

OpenFlow API Flow Tables are a possible way to implement Traffic Engineering. Paola could construct the tables such that S4 and S3 allow the YouTube Short request and response to travel before the lecture recording request/response.

Encapsulation and Decapsulation, as seen in the Lecture 21 slides, can be used to add headers to each packet so that Paola's packets take a different route than Daniel's, allowing them to be received before Daniel is done transmitting.

Constrained Shortest Path First does not affect the state of the datacenter as clogging all switches with Daniel's packets is already optimal as per the algorithm. Since Paola's packet was received at S3 after Daniel's, it will only be processed once a path opens up.

Q6.5 (2 points) Paola adds a new VM, Meta, to Rack 1. Which devices need to update their forwarding table? Select all that apply.

- [x] VMs sending data to Meta
- [ ] Switches S3 and S4
- [ ] VMs sending data to YouTube
- [x] Rack 1′s virtual switch
- [ ] Switches S1 and S2
- ○ None of the above

Solution: VMs sending data to Meta will need an additional entry in their forwarding tables to know which encapsulation header to add for Meta packets.

VMs sending data to Youtube can continue doing so as the same encapsulation header still works.

As per layering, switches sending data to or receiving data from Rack 1 forward based on the addresses of the Racks, not the end-recipient servers within the racks. Forwarding to the latter is the role of the virtual/edge switches connected to each rack.

Rack 1′s virtual switch needs to correctly decapsulate incoming encapsulation headers and forward packets to the appropriate tenant.

Q6.6 (2 points) Google wants to communicate with NVIDIA with very small latency. How could Paola update the datacenter to achieve this? There may be multiple answers.

Solution: Possible answers:

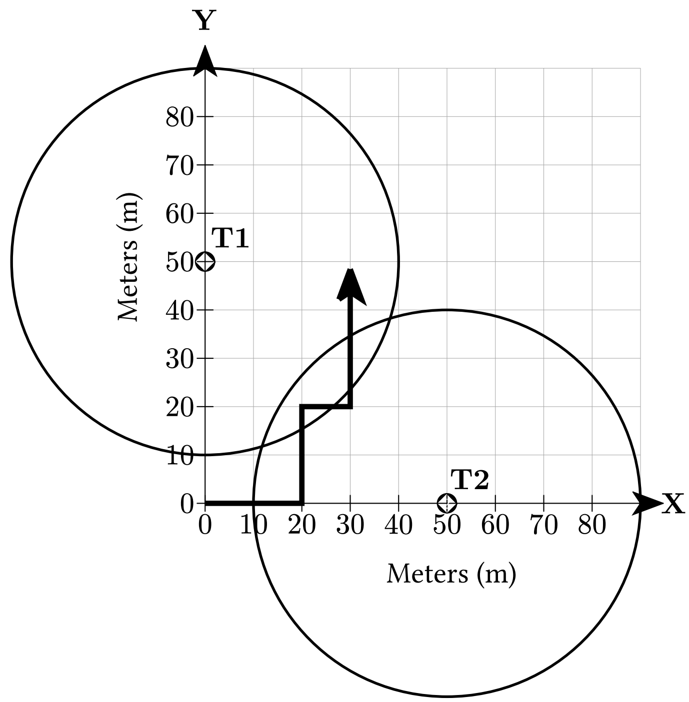
- Move NVIDIA's servers into Google's Rack (R2) or vice versa to allow for faster accesses even without NIC offloading.
- Connect Rack 2 to Rack 3 with very fast (sub-second bandwidth) links and optionally a switch.
- Increase the bandwidth of S3 - S4 to ∞
- Implement one of RDMA, NIC offloading or Kernel Bypassing in the Google and NVIDIA servers.

Q7 *Wireless: Stuck in the Middle* **(10 points)**

There are two wireless terminals (T1 and T2) constantly attempting to send location information on the same frequency to V, a mobile autonomous vehicle. V has a wireless receiver and attempts to receive data from the terminals (T1 and T2). V follows the path starting from $(x, y) = (0, 0)$ and ending at $(30, 50)$.

Assumptions for Subparts Q7.1 to Q7.3:

- The vehicle (V) maintains a velocity of 10 m/s.
- T1 is located at $(0, 50)$. T2 is located at $(50, 0)$.
- Each terminal (T1/T2) has a transmit range of 40 m.
- There is no background noise or interference.
- If there are wireless collisions at T1,T2, or V, they cannot receive data.
- T1,T2, and V use **CSMA**.
- The vehicle starts moving from $(0, 0)$ at $t = 0$.



Q7.1 (2 points) At $t = 2$, which of the following is true?

- ☐ V can receive data from T1
- ■ V can receive data from T2
- ☐ T1 and T2 can exchange data
- ☐ Hidden terminal causes collisions
- ☐ Exposed terminal stops a transmit
- ○ None of the above

> **Solution:** V is only in range of T2, so it can only receive data from T2.
>
> T1 and T2 are not in range of each other.

Q7.2 (2 points) At $t = 4$, which of the following is true?

- ☐ V can receive data from T1
- ☐ V can receive data from T2
- ☐ T1 and T2 can exchange data
- ■ Hidden terminal causes collisions
- ☐ Exposed terminal stops a transmit
- ○ None of the above

> **Solution:** V is in range of T1 and T2, but T1 and T2 are not in range of each other, so both will try and transmit to V resulting in collisions. This is the hidden terminal problem in action.

Q7.3 (2 points) At $t = 7$, which of the following is true?

- [x] V can receive data from T1
- [ ] V can receive data from T2
- [ ] T1 and T2 can exchange data
- [ ] Hidden terminal causes collisions
- [ ] Exposed terminal stops a transmit
- ○ None of the above

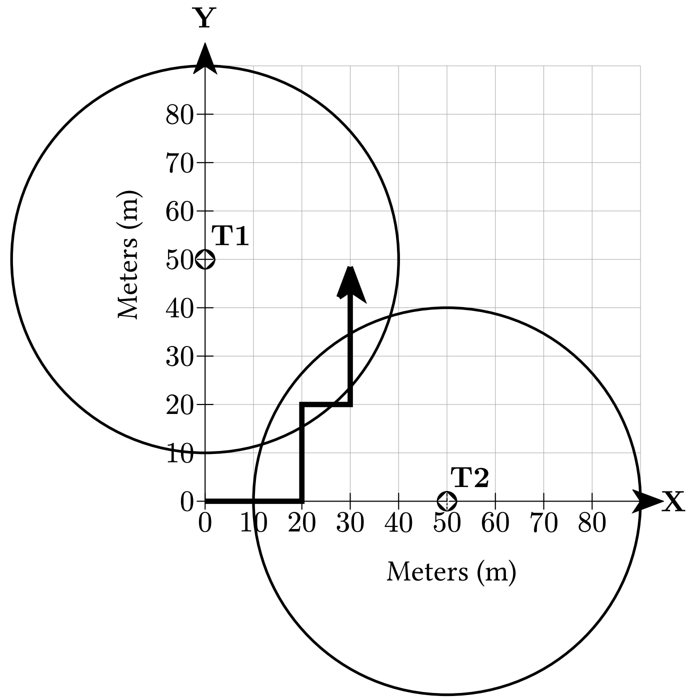
Solution: V is only in range of T1, so it can only receive data from T1.

T1 and T2 are not in range of each other.

The assumptions and diagram are reprinted below. The only change is CSMA $\longrightarrow$ MACA.

Assumptions for Subparts Q7.4 and Q7.5:

- The vehicle (V) maintains a velocity of 10 m/s.
- T1 is located at $(0, 50)$. T2 is located at $(50, 0)$.
- Each terminal (T1/T2) has a transmit range of 40 m.
- There is no background noise or interference.
- If there are wireless collisions at T1,T2, or V, they cannot receive data.
- T1,T2, and V use **MACA**.
- The vehicle starts moving from $(0, 0)$ at $t = 0$.



Q7.4 (2 points) To fix the hidden terminal and exposed terminal problem, MACA is implemented instead of CSMA. What assumption must be made to fix the exposed terminal problem?

- ○ RTS/CTS is sent directly after data transfers finish.
- ● CTS can be heard over data broadcasts.
- ○ The contention window is doubled on failure.
- ○ ACKs are implemented for reliability.
- ○ It is not possible to fix the exposed terminal problem.

Solution: If the CTS is not heard over other data broadcasts, then the transmitter will not know that it can send and will not transmit data. This is the exposed terminal problem in action.

Q7.5 (2 points) With MACA, the first terminal to send a successful RTS/CTS exchange will likely succeed in most future exchanges, and the other terminal will be unable to send data. This is an issue since the vehicle should get data from both terminals. Explain how to fix this. Please answer in approximately 20 words or less.

Solution:

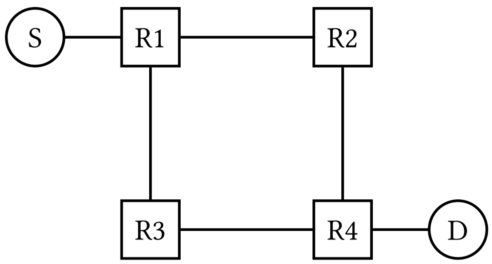
Possible Answers:

- Have everybody backoff randomly like in ALOHA instead of MACA's exponential backoff.
- Use time-based or frequency-based slots to schedule transmissions.

Q8 *Project 1: Traceroute Storm* **(8 points)**

In this question, Host S runs traceroute on the network topology below. All links cost 1.

Unless otherwise specified, you can assume no network errors occur (e.g., no packet drops, no corruption, no additional duplication beyond the behavior described, etc).



- R1 has IP address 1.1.1.1.
- R2 has IP address 2.2.2.2.
- R3 has IP address 3.3.3.3.
- R4 has IP address 4.4.4.4.
- D has IP address 100.0.0.1.

Instead of keeping a routing table, each router **floods** every packet it receives, by forwarding a copy of that packet out of every outgoing port, including the one you received the packet from. If a router creates a new packet to be sent, that packet is also forwarded out of every outgoing port.

End hosts do not flood packets across the network. If a packet is flooded to them, they do not forward it elsewhere.

Host S runs traceroute with destination D, sending 1 probe at each TTL. The result is a list of sublists:

`[[____], [____], [____], [____]]`
(1) (2) (3) (4)

Q8.1 (1 point) Which IP(s) are in the first sublist (1)? Select all that apply.

■ 1.1.1.1 □ 2.2.2.2 □ 3.3.3.3 □ 4.4.4.4 □ 100.0.0.1

Solution:

A probe with TTL 1 reaches R1, then expires.

Q8.2 (1 point) Which IP(s) are in the second sublist (2)? Select all that apply.

□ 1.1.1.1 ■ 2.2.2.2 ■ 3.3.3.3 □ 4.4.4.4 □ 100.0.0.1

Solution:

A probe with TTL 2 is sent to R1, which then floods the probe to R2 and R3. The probe expires at R2 and R3.

Q8.3 (1 point) Which IP(s) are in the third sublist (3)? Select all that apply.

- [x] 1.1.1.1
- [ ] 2.2.2.2
- [ ] 3.3.3.3
- [x] 4.4.4.4
- [ ] 100.0.0.1

Solution:

A probe with TTL 3 is sent to R1, which then floods the probe to R2 and R3.

R2 then floods the packet to R1 and R4, and the probe expires at R1 and R4.

R3 also floods the packet to R1 and R4, and the probe expires at R1 and R4.

Note that even though R1 and R4 get multiple copies of the probe and send back multiple "TTL Exceeded" messages, Host S still adds R1 and R4's IPs once each to this sublist.

Q8.4 (1 point) Which IP(s) are in the fourth sublist (4)? Select all that apply.

Assume that all reply packets received at each TTL are processed and added to the list, i.e. we don't return early after finding the destination.

- [ ] 1.1.1.1
- [x] 2.2.2.2
- [x] 3.3.3.3
- [ ] 4.4.4.4
- [x] 100.0.0.1

Solution:

Continuing from the previous subpart: A packet with TTL 3 reaches R1 and R4.

R1 floods the packet to R2 and R3, and the probe expires at R2 and R3.

R4 floods the packet to R2 and R3, and the probe expires at R2 and R3.

R4 also floods the packet to D, which sends back a "Port Unreachable" error.

Note that D doesn't appear in any earlier sublist because probes with TTL 1, 2, 3 will not reach D.

Subparts Q8.5 to Q8.8: Assume the following:

- When a router creates a new packet, the TTL on that new packet is always set to 3.
- A duplicate is a probe response that reveals an IP that S has already added to the current sublist.
- All probes and responses at a given TTL are processed before moving onto the next TTL, so there are no badly-delayed duplicates.
- If S receives a packet with TTL 1, it still processes that packet.

As a reminder, each router **floods** every packet it receives. End hosts do not flood packets across the network. If a packet is flooded to them, they do not forward it elsewhere.

For instance, if R4 was sending a response to a traceroute request from S, it would flood this packet to all of its neighbors: R2, R3, and D. Each of its neighbors would then flood the packet to each of their neighbors, **including** R4.

Q8.5 (1 point) How many duplicates are discarded by S when building the first sublist (1)?

○ 0 ○ 1 ● 2 ○ 3 ○ 4

Solution:

The probe reaches R1, and a single response is flooded.

The response is flooded to S.

The response is also flooded to R2, which forwards back to R1, which forwards to S again, creating one duplicate.

The response is also flooded to R3, which forwards back to R1, which forwards to S again, creating another duplicate.

Q8.6 (1 point) How many duplicates are discarded by S when building the second sublist (2)?

● 0 ○ 1 ○ 2 ○ 3 ○ 4

Solution:

R2 and R3 each receive a copy of the probe, and they each send back a response.

R2's response is flooded to R1, then to S. With TTL 3, there are no other ways this packet is flooded to S.

R3's response is flooded to R1, then to S. With TTL 3, there are no other ways this packet is flooded to S.

The two responses reveal two different IPs, so there are no duplicates at TTL 2.

Q8.7 (1 point) Suppose that we re-run traceroute (still from S to D), now sending 3 probes at each TTL instead of 1 probe.

Does the resulting list of sublists change?

- ○ Yes, the result is different and unpredictable on each run.
- ○ Yes, the result is different, but the same on every run.
- ● No, the result is still the same on every run.

Solution: Whether we send one probe or multiple probes, each probe still reaches the same routers, so the same result is generated either way.

Q8.8 (1 point) Suppose that we re-run traceroute from S, but now to a destination that is not connected to the network. We still send 1 probe at each TTL.

What does the resulting list of sublists look like at higher TTLs?

- ○ `[..., [], [], [], [], ...]`
- ● `[..., [1, 4], [2, 3], [1, 4], [2, 3], ...]`
- ○ `[..., [1, 2, 3, 4], [1, 2, 3, 4], [1, 2, 3, 4], ...]`
- ○ `[..., [1], [2], [3], [4], [1], [2], [3], [4], ...]`
- ○ `[..., [4], [4], [4], [4], ...]`
- ○ `[..., [1], [1], [1], [1], ...]`

Solution:

The pattern is similar to what we saw in earlier subparts.

When the TTL is even, the packet expires at R2 and R3 (e.g. consider a path like S–R1–R3–R1–R3–R1–R3).

When the TTL is odd, the packet expires at R1 and R4 (e.g. consider a path like S–R1–R2–R4–R2–R4).

Comment Box

Congrats for making it to the end of the exam! Leave any thoughts, comments, feedback, or doodles here.

Nothing in the comment box will affect your grade.