SUNetID: ________

CS144
Introduction to Computer Networking
Final Exam — Wednesday, June 14th, 2023

Rules: 4 note pages, closed book, no Internet, computers off

Your Name: Answers

SUNet ID: root @stanford.edu

On my honor, and in accordance with the letter and spirit of the Stanford Honor Code, I neither received nor provided any assistance on this exam.

Signature: ____________________

Check if you would like exam routed back via SCPD: ☐

- The exam has 10 questions totaling 129 points.
- You have 180 minutes to complete them.
- Please keep your answers concise. You may lose points for a correct answer that also includes incorrect or irrelevant information.
- If you would like to make any additional commentary on a multiple-choice answer, please write it below the answer section, but nothing additional is necessary to receive full credit.
- Please box your final answers.

1	/16
2	/8
3	/12
4	/6
5	/6
6	/20
7	/15
8	/9
9	/12
10	/25
Total	/129

I Applications

1. [16 points]:

Please fill in the blanks:

- You turn on your computer and connect it to a Wi-Fi network for the first time. Your computer performs a(n) [ ] request to learn its IP address and the IP addresses of its default router and DNS server.
- You open your Web browser and navigate to `https://cs144.stanford.edu`. Your computer sends a(n) [ ] request to learn the Ethernet (MAC) address of the default router, and uses the [ ] protocol to transform the "cs144.stanford.edu" hostname into an IP address.
- Your computer sends a TCP segment, with the [ ] flag set, to initiate a TCP connection with the `cs144.stanford.edu` computer.
- The two computers use the [ ] protocol to establish an encrypted and integrity-protected byte stream inside the TCP byte stream.
- As part of this protocol, `cs144.stanford.edu` presents a certificate attesting that its [ ] key rightfully belongs to "cs144.stanford.edu." Your Web browser verifies that this certificate is signed by a [ ] that it trusts.
- Using the encrypted byte stream, your browser sends a(n) [ ] request for the "/" path and receives a response whose body contains an HTML document.

Answer:

(a) *DHCP*
(b) *ARP*
(c) *DNS*
(d) *SYN*
(e) *TLS or SSL*
(f) *public key*
(g) *certification authority, certificate authority, or CA*
(h) *HTTP*
(i) *HTML*

2. [**8 points**]:

Why can't the IP, UDP, and TCP checksums detect or correct attempts by malicious adversaries to modify datagrams and segments in transmit? (Please write 1–2 concise sentences.)

Answer:

A malicious adversary can modify the data and then also modify the checksum to match—and in many cases, a malicious party can modify the data in a way that leaves the correct checksum unchanged.

How does a MAC, or AEAD cipher, preserve integrity in the face of intentional modification by malicious adversaries? (Please write 1–2 concise sentences.)

Answer:

MACs and AEAD ciphers rely on a shared secret session key that is only known by the sender and receiver. An attacker that doesn't know this key cannot compute the correct MAC value for a modified payload. (Incorrect answers will refer to "public keys" or "certificates.")

II Service Abstractions

3. [12 points]:

Each of the following statements is false. For each one, please use 1–3 concise sentences to explain why (and which part of) the statement is false.

(a) "IP provides an unreliable datagram service, so applications that use IP are not reliable."

Answer:

IP does provide a best-effort datagram service that is not reliable, but applications can use higher-level protocols on top of IP (e.g. TCP, or simply retransmitting until acknowledgement, using a checksum to protect against corruption) to achieve reliability.

(b) "UDP provides an unreliable datagram service, so applications that use UDP are not reliable."

Answer:

Same answer as IP—UDP does provide a best-effort datagram service that is not reliable, but applications can use higher-level protocols on top of UDP (e.g. TCP, or simply retransmitting until acknowledgement) to achieve reliability.

(c) "UDP traffic is not congestion-controlled."

Answer:

This is up to the application or library that is sending UDP. Most applications that use UDP do end up using some congestion-control scheme (e.g. QUIC uses the same schemes that TCP uses, BitTorrent uses UDP with a congestion-control scheme that tries to be "nicer" than most TCP schemes, Skype and FaceTime use UDP with congestion-control schemes intended for real-time video, etc.).

(d) "NATs translate the address and port of a TCP connection between the 'internal' IP/port to the 'external' IP/port. UDP is connectionless, so it doesn't work with NATs."

Answer:

It's true that UDP does not require any per-connection state in the middle of the network, but neither does TCP. It's also true that unlike in TCP, each UDP datagram can *stand alone (an application can send one UDP datagram and then shut down—without expecting a reply or ever sending another datagram). In that sense UDP* can be *connectionless. But even in this case, the datagram has a source IP and port number and a destination IP and port number, and if there is a reply from the other side, it will have the reverse source and destination. A NAT can treat this five-tuple (the "UDP" protocol and the two IPs and two port numbers) as a "connection," just as it does with TCP.*

4. [6 points]:

TCP traditionally runs on top of IP. Suppose you decide to tunnel many TCP flows inside of a single UDP flow, placing full TCP segments, including headers, inside the payloads of UDP datagrams.

(a) Would this affect application throughput by more than 5%? Please circle your answer.

A Yes, tunneling TCP inside UDP will affect application TCP throughput by more than 5%.

B No, tunneling TCP inside UDP will not affect application TCP throughput by more than 5%.

Please justify your answer with 1-3 concise sentences.

Answer:

There would be a small reduction in throughput. The UDP header would add an additional 12 bytes of overhead on each segment, which for typical 1500 byte segments would be an overhead of less than 1%. **Exceptions:** *if the network uses per-flow fair queueing you may see a performance drop because multiple TCP flows are treated as a single flow by the network. In addition, roughly 5% of networks on the real-world Internet appear to block or rate-limit UDP traffic.*

(b) Would this affect the delivery behavior of TCP segments? Please circle your answer.

A Yes, tunneling TCP inside UDP will affect the delivery behavior of TCP segments.

B No, tunneling TCP inside UDP will not affect the delivery behavior of TCP segments.

Please justify your answer with 1-3 concise sentences.

Answer:

TCP would continue to provide a reliable, in-order byte stream. UDP and IP both provide a datagram abstraction, and TCP can be encapsulated inside either of them.

5. [6 points]:

UDP traditionally runs on top of IP. Suppose you decide to tunnel many UDP flows inside a single TCP flow, placing full UDP segments, including headers, in the TCP stream.

(a) Could this affect application throughput by more than 5%? Please circle your answer.

A Yes, tunneling UDP inside TCP could affect application UDP throughput by more than 5%.

B No, tunneling UDP inside TCP could not affect application UDP throughput by more than 5%.

Please justify your answer with 1-3 concise sentences.

Answer:

You may see a significant reduction in throughput in the presence of packet drops, because TCP will rate-limit the UDP flows by capping the number of outstanding segments in the network with its congestion window. The UDP flows will be forced to be TCP-friendly and share capacity with other TCP flows, when normally they don't have to. **Advanced:** *if the network uses per-flow fair queueing you may see a performance drop because multiple UDP flows are treated as a single flow by the network.*

(b) Would this affect the delivery behavior of UDP segments? Please circle your answer.

A Yes, tunneling UDP inside TCP will affect the delivery behavior of UDP segments.

B No, tunneling UDP inside TCP will not affect the delivery behavior of UDP segments.

Please justify your answer with 1-3 concise sentences.

Answer:

Yes. The latency of the first UDP segment in a flow would go up significantly, because it would require a three-way TCP handshake first. The behavior of UDP segments would change significantly: they would become reliable and always be delivered in order.

III Real-World Networking

6. [20 points]:

Consider two users/computers behind different NAPTs, trying to send large amounts of data to each other over the Internet as part of e.g. a video chat or file-sharing application. Recall the following four levels of peer-to-peer (P2P) networking discussed in class:

- Level 9a: P2P networking via a public file server (e.g. Google Drive, GitHub)
- Level 9b: P2P networking via a public proxy/relay/TURN server, e.g. the relay server you used in lab checkpoint 6
- Level 9c: P2P networking via user-installed explicit NAPT rules ("port forwarding")
- Level 9d: P2P networking via NAPT traversal (aka "hole punching")

You are an application developer considering the costs and benefits of different styles of P2P networking.

(a) For which styles of P2P networking would the application developer typically need to operate (or depend on) a server or servers on the public Internet? Please check all that apply:

☐ Level 9a ☐ Level 9b ☐ Level 9c ☐ Level 9d

(b) For each level where you checked a box above, what would be the purpose of the additional server(s)?

- Level 9a: ☐
- Level 9b: ☐
- Level 9c: ☐
- Level 9d: ☐

(c) For which styles would a developer-operated server need to store significant amounts of *state* (proportional to the amount of data being transferred, or worse) between connections?

☐ Level 9a ☐ Level 9b ☐ Level 9c ☐ Level 9d

(d) For which styles would a developer-operated server need to use significant amounts of *CPU or network resources* (proportional to the amount of data being transferred, or worse) during a connection?

☐ Level 9a ☐ Level 9b ☐ Level 9c ☐ Level 9d

(e) For which styles would the application require at least one user to be behind a NAPT that creates non-restrictive rules when it sees a new outgoing connection (aka a "cone" NAPT)?

☐ Level 9a ☐ Level 9b ☐ Level 9c ☐ Level 9d

Answer:

(a) 9a,9b,9d. For 9a, the file server. For 9b, the public relay server. For 9d, the STUN server and rendezvous server.

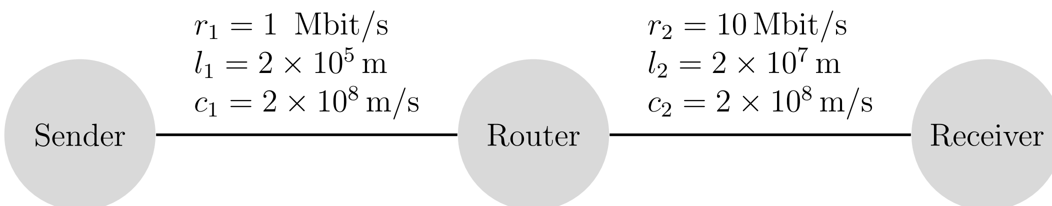
(b) 9a. The public server stores the file for later download.

(c) 9a,9b. 9a serves the file to the requesting computer. 9b reconstructs the bytestream.

(d) 9d.

IV Packet Switching

In this problem, we'll consider a single flow over a top-hop network path. The router is "normal" and processes each packet as a unit (the entire packet must arrive before it can begin being sent on an outgoing link).



7. [15 points]:

(a) Assume there is no other traffic in the network. If the sender sends a single packet of size 20 kbit to the receiver, how long does it take until the packet fully arrives? (In other words, what is its total end-to-end delay?) Please express your answer in milliseconds.

Answer:

$p/r1 + p/r2 + l1/c1 + l2/c2 = 20ms + 2ms + 1ms + 100ms = 123$ *ms*

(b) If the sender sends two packets back to back, both of size 10 kbit, what is the queueing delay that packet p_2 experiences at the router (in milliseconds)?

Answer:

0 milliseconds (no queueing delay)! The first packet has departed the Router before the second packet finishes arriving.

(c) If the sender sends two packets back to back, where the first packet is of size 10 kbit and the second packet is of size p_2, what is the minumum size of p_2 such that the queuing delay is 0?

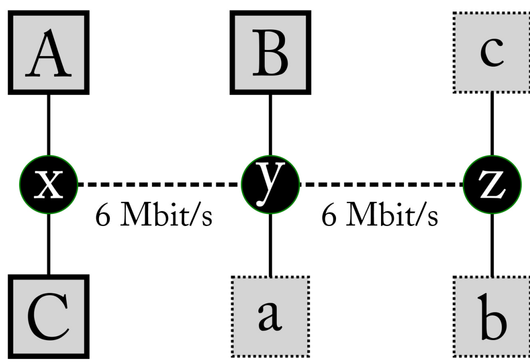
Answer:

$\frac{p_1}{r_2} - \frac{p_2}{r_1} \geq 0$
$\frac{p_1}{r_2} \geq \frac{p_2}{r_1}$
$\frac{p_1}{10} \geq \frac{p_2}{1}$
$p_2 \leq \frac{p_1}{10} = 1kbit$

V Fairness

8. [9 points]:

In the network below, sender A sends a flow to receiver a, sender B sends a flow to receiver b, and sender C sends a flow to receiver c. The link rate between x and y is 6 megabits per second, as is the link rate between y and z.



a. What is the *max-min fair* allocation of throughputs to the three flows?

	$A \to a$	$B \to b$	$C \to c$
throughput (Mbit/s)			

Answer:

throughput (Mbit/s)	*3*	*3*	*3*

b. What is the *max-utilization* allocation of throughputs to the three flows? (This allocation maximizes the **total** of the throughputs of the three flows.)

	$A \to a$	$B \to b$	$C \to c$
throughput (Mbit/s)			

Answer:

throughput (Mbit/s)	*6*	*6*	*0*

c. What is the *proportionally fair* allocation of throughputs to the three flows? (This allocation maximizes the **sum of the log** of each flow's throughput.)

	$A \to a$	$B \to b$	$C \to c$
throughput (Mbit/s)			

Answer:

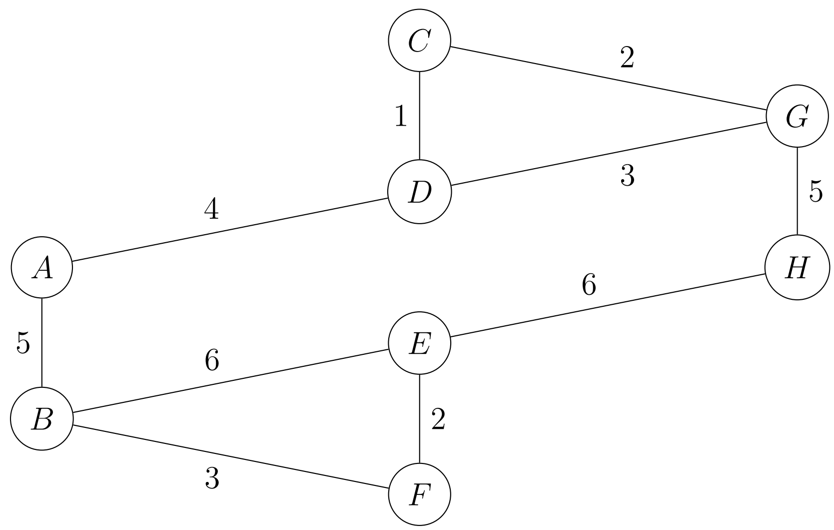
throughput (Mbit/s)	*4*	*4*	*2*

SUNetID: __________

VI Routing

9. [12 points]:

Please see the network diagram below. The nodes represent routers, and the numbers next to each link represent the link's cost. Throughout the following questions, you can interpret "shortest" and "longest" to mean "lowest-cost" and "highest-cost."



(a) For which pair of routers is the lowest-cost path between them the largest? Please give your answer as a pair of router names; for example (A, B).

Answer:

(D, E)

(b) What is the cost of this path (the lowest-cost path between the two routers you identified in part (a))? Please give your answer as a single positive integer.

Answer:

14

(c) What is the minimum number of links that would need to fail at the same time to prevent a pair of routers from being able to communicate with each other?

Answer:

2: example AB, EH

(d) Imagine this network uses the simplified form of the Bellman-Ford algorithm that we saw in lecture to build its routing tables, in which all routers exchange information with their neighbors in lock-step. If all routers start from scratch (i.e., distance vectors initialized to infinity), how many steps will it take for the network to reach its final routing table configuration? Please give your answer as a single positive integer.

Answer:

4

VII Elasticity Buffers

In class, we discussed how an **elasticity buffer** can be used to mitigate a difference in clock rates between the receiver and the sender. Assume you are given:

- a maximum transmission unit (frame size) of MTU
- an inter-packet gap time of p,
- and an elasticity-buffer size B

Further assume that:

- The sender sends packets of size exactly equal to MTU
- The sender always waits for p after it finishes serializing one packet before starting to serialize the next packet
- The reader starts reading from the elasticity buffer when the buffer is half-full, continues reading until it reads the end of a packet, and again waits for the buffer to be half-full before it starts reading again.
- Propagation delay is negligible

Let r_{sender} be the frequency of the sender's clock and $r_{receiver}$ the frequency of the receiver's clock. Assume that $r_{sender} > r_{receiver}$ (we're worried about the *overflow* case). In your answers below, use the "min" function to represent the *smaller* of two numbers. Use the "max" function to represent the *larger* of two numbers.

10. [25 points]:

(a) When the sender *starts* sending a packet, the buffer contains x bytes, $x \leq \frac{B}{2}$). Write an algebraic expressing for the number of bytes that will be in the buffer when the sender *finishes* serializing the packet:

Answer:

If $x \times \frac{r_{sender}}{r_{receiver}} \geq \frac{B}{2}$, $min(x + MTU - r_{receiver} \times \frac{MTU}{r_{sender}}, B)$. *Else*, $min(x + MTU - r_{receiver} \times \frac{MTU - (\frac{B}{2} - x \times \frac{r_{sender}}{r_{receiver}})}{r_{sender}}, B)$

Full credit also given to $min(x + MTU - r_{receiver} \times \frac{MTU - (\frac{B}{2} - x)}{r_{sender}}, B)$.

(b) Call the above value "y". How many bytes can the receiver drain during the inter-packet gap?

Answer: $min(y, \boxed{})$

Answer:

$min(y, p \times r_{receiver})$

(c) Assume that the inter-packet gap p is long enough for the receiver to drain the **whole buffer**.

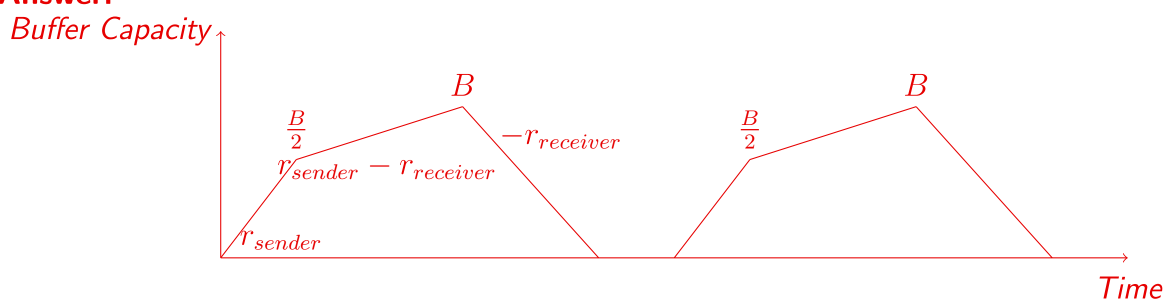
- How small can B be such that there is no overflow?

Answer:

$MTU - r_{receiver} \times \frac{MTU - \frac{B}{2}}{r_{sender}} < B \implies (1 - \frac{r_{receiver}}{r_{sender}}) \times MTU < (1 + \frac{1}{2}\frac{r_{receiver}}{r_{sender}}) \times B$

- Draw a graph of the the buffer occupancy as a function of time, showing **two** packets be sent and received. Clearly mark (1) the slopes of the lines, and (2) the buffer occupancy at any change of slope.

Answer:



(d) Assume, instead, that the combination of B and p is just barely sufficient. In other words, p and B are both as small as possible while still preventing overflow. Draw the graph of the buffer occupancy as a function of time, showing exactly **two** packets be sent and received. Clearly mark (1) the slopes of the lines, and (2) the buffer occupancy at any change of slope.

Answer: