# CS144
# Intro to Computer Networks
# Practice Final

**Your Name:** ______________________________

**SUNet ID:** ____________________ @stanford.edu

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

**In accordance with both the letter and the spirit of the Stanford Honor Code, I will not distribute this practice exam to future classes.**

**Signature:** ______________________________

| | |
|---|---|
| **1** | /4 |
| **2** | /6 |
| **3** | /5 |
| **4** | /10 |
| **5** | /5 |
| **6** | /5 |
| **7** | /5 |
| **8** | /5 |
| **9** | /10 |
| **10** | /5 |
| **11** | /10 |
| **12** | /15 |
| **13** | /20 |
| **14** | /5 |
| **15** | /5 |
| **16** | /5 |
| **17** | /5 |
| **Total** | /125 |

- Some questions may be much harder than others.
- All questions require you to justify your answer to receive full credit, even multiple choice questions for which you circle the correct answer(s).
- Keep your answers concise. We will deduct points for a correct answer that also includes incorrect or irrelevant information.

# I The Link Less Travelled

**1. [4 points]:**

Circle each of the following statements that is true. Zero, one, or many may be true.

**A** The egress link of an output-queued packet switch never sits idle when it has a packet in its output queue.

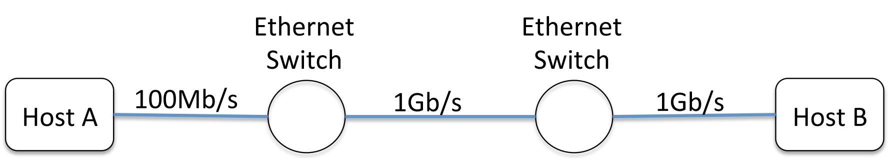
**B** The egress link of an input-queued packet switch can sit idle when it has a packet in its input queue.

**C** An input queued packet switch has lower average packet delay than an output-queued switch.

**D** In an input-queued switch with virtual output queues, a packet is often held up by packets ahead of it in its queue destined to a different output.

**2. [6 points]:**

The Ethernet network below consisting of two end hosts interconnected by Ethernet switches and links running at either 100Mb/s or 1Gb/s. All the links are 200m long. All the switches are store-and-forward devices.




If Host A sends a 1000-bit packet to Host B, how long does it take from when the first bit leaves Host A until the last bit reaches Host B if there are no other packets in the network? The speed of propagation is $2 \times 10^8 m/s$.

**Time:** ___________ seconds.

**3. [5 points]:**

Host A sends two packets (p1 and p2) to Host B, with p2 starting $10\mu s$ after p1 starts. The first bit of p2 arrives at Host B $1\mu s$ after the first bit of p1. Which of the following statements are true?

**A** There are no other packets in the network apart from p1 and p2.

**B** The only possible explanation is that p1 was delayed by $9\mu s$.

**C** One possible explanation is that p1 was delayed in a switch buffer by other packets destined to Host B, but p2 was not delayed at all.

**D** One possible explanation is that both p1 and p2 were delayed by other packets along the way.

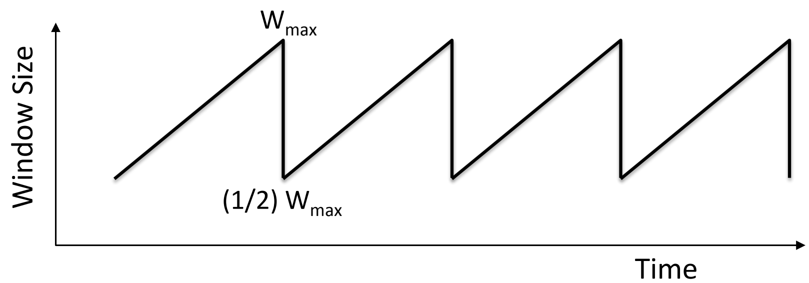
**E** None of the above.

# II Hold to Your Principles

**4. [10 points]:**

IP routers strip away the link layer header and verify that the IP header checksum is valid before encapsulating an IP packet in a new link layer header and sending it along the next link to the next router (or the end host). This is *not* a violation of either the strong or weak end–to–end principle. Why not? Explain in 3-4 short, concise sentences.

# III The AIMD Saw Cuts Congestion




In this question we will assume a network carrying only a *single* AIMD flow. By now you are very familiar with the AIMD "sawtooth" graph that plots a sender's window size as a function of time. For these questions, aassume congestion is only detected by packet loss (not duplicate ACKs) and router buffers are at least as big as $C \cdot RTT$.

**5. [5 points]:**

Does the AIMD sawtooth have a fixed or variable frequency? I.e., are the peaks of the sawtooth are a fixed duration apart?

**A** The sawtooth has a variable frequency.

**B** The sawtooth has a fixed frequency.

Briefly explain your answer:

**6. [5 points]:**

Is the additive increase line (the hypotenuse of the sawtooth) straight or curved?

**A** The hypotenuse is straight.

**B** The hypotenuse is not straight.

Briefly explain your answer:

**7. [5 points]:**

Write down an expression for the throughput of the flow as a function of the line-rate, $C$, of the bottleneck link.

8. **[5 points]:**

Suppose that the network has many flows between different pairs of nodes, such that there many different bottleneck links and the flows are not synchronized. Will the AIMD sawtooth for each sender still have a fixed frequency (i.e. the peaks of the sawtooth would be a fixed duration apart).

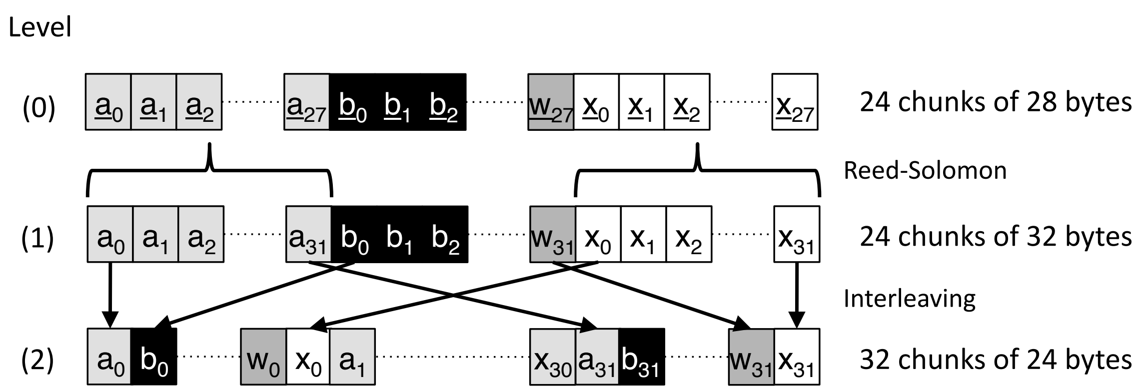
**A** Every flow's sawtooth will have a fixed frequency.

**B** Some flows' sawtooth will not have a fixed frequency.

Explain your answer.

# IV Scrambled Eggs and Bits

Suppose you have a physical layer that operates on $24 \cdot 28 = 672$-byte blocks. The physical layer requires that two separate encodings be applied. First, data bits (Level 0) are transformed into an intermediary "Level 1" representation. Then a second encoding transforms the Level 1 representation into chips (Level 2):




Without these two encodings—if chips were just equal to bits—any single-chip error would cause a block to be received incorrectly. The next two questions explore what happens when you apply one or both of the above encodings.

**9. [10 points]:**

Level 1 encoding breaks the original 672 bytes into 24 chunks ($\underline{a}$, $\underline{b}$, $\underline{c}$ , . . . , $\underline{x}$) of 28 bytes each ($\underline{a}_0$, $\underline{a}_1$, . . . , $\underline{a}_{27}$). It applies a (32,28) Reed-Solomon code to each of the 28-byte chunks. Recall that an $(n, k)$ Reed-Solomon code can recover from $n - k$ erasures and $(n - k)/2$ errors. In the figure, $\underline{a}_5$ means the fifth unencoded data byte of chunk a while $a_5$ means the fifth byte of the Level 1 Reed-Solomon coding of chunk $a$.

Suppose we stopped after this first encoding and defined the physical-layer chips to be the Level 1 encoding of the bits. With this definition, each 28-byte (228-bit) chunk would be transmitted as 256 (= 32 · 8) chips. Thus, an encoded block would represent 672 data bytes (5,376 bits) with $8 \cdot 768 = 6,144$ chips.

What is the shortest string of consecutive chip errors that can corrupt a block if only Level 1 encoding is used? Hint: consider how many encoded bytes must be corrupted, then the shortest string of consecutive chip errors that can corrupt that many encoded bytes.

**Shortest string:** ______ consecutive chips

**Briefly justify your answer:**

**10. [5 points]:**

*Interleaving* is an error correction technique that spreads errors in order to be more robust to strings of consecutive chip errors. Level 2 encoding uses interleaving to transform the 24 chunks of 32 bytes into 32 chunks of 24 bytes, as shown in the above figure. Where Level 1 chunk e is bytes $e_0$ through $e_{31}$, in Level 2 encoding chunk 5 is bytes $a_5$ through $x_5$.

What is the shortest string of consecutive chip errors that can corrupt a level 2 encoded block?

**Shortest string:** ______ consecutive chips

**Briefly justify your answer:**

# V With a Side of Hashes

Recall from that DNS has TXT records. TXT records allow a DNS server to embed a basic text string in a reply. Originally intended for human-readable information, TXT records have also been used as a way to introduce new services without adding new record types.

**11. [10 points]:**

A friend of yours (who hasn't taken CS144) tells you that Comcast forced its customers to use Comcast's DNS servers in 2009. It did this by intercepting all DNS requests on port 53 and diverting them to its own servers, which spoofed responses. If the request was for a non-existent DNS name, rather than report no such host, Comcast's servers provided a (non-authoritative) A record for a web server selling domain name registrations.

Comcast has since stopped doing this, but your friend is very protective of his Internet access. He sets up his own DNS server at Stanford (where he trusts the network operators more) and configures his laptop to use it. To make sure that nobody is spoofing DNS, he programs his DNS server to add a TXT record to every DNS reply. The record reads `mac=xxxx`, where `xxxx` is a hexadecimal representation of a message authentication code (MAC) computed with HMAC-SHA-256 and a private key that exists only on his DNS server and laptop. His DNS server computes the MAC over the DNS payload and UDP pseudo-header, assuming that `xxxx` is all zeroes. He then runs a small utility on his laptop that makes DNS requests and checks if the MAC is correct.

Having just finished coding his protection scheme and testing it successfully in Gates, he heads home to his off-campus apartment to watch a movie. He calls you, angrily telling you that he thinks his ISP is spoofing DNS in a very cunning but imperfect way: the DNS responses he sees have the `mac=xxxx` text record, but the MAC is incorrect. Do you agree with his conclusion, or is there a more likely explanation?

**Circle the best answer.**

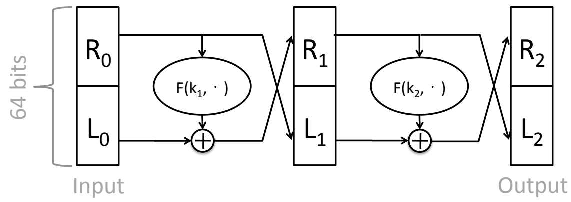
**A** Yes, his ISP is spoofing DNS

**B** No, something else is happening

**If yes, briefly explain why; if no, explain what you think is happening and why the MAC is incorrect:**

**12. [15 points]:**

Suppose that instead of SSL, your bank decides to roll its own protocol for customers to request on-line payments. It hires a very cheap security engineer, who designs a system that uses a block cipher based on something called a two-level Feistel network. The Feistel network takes 64-bits in a block, breaks it into two 32-bit chunks and encrypts them using a function F that takes a key $k$ and a 32-bit input:




The bank then hires you to audit this encryption. They ask you whether it is possible that known plaintexts can leak information. For example, if you are given the output of the input $0^{64}$ (64 consecutive zeroes) and know it is from that input, can you use this information to determine the output of some other inputs without knowing the keys used?

**A** No, it is not possible to determine any outputs.

**B** Yes, it is possible to determine some outputs.

In 2-3 sentences, explain why:

# VI A Stack of Protocols

**13. [20 points]:**

You plug a new laptop into a wired Ethernet jack for the first time. You have already told the network administrators your MAC address, and can join the network with no further action on your part.

Assuming that

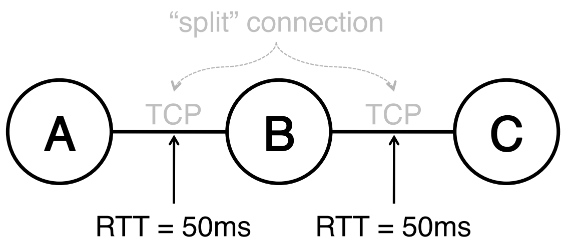
- your DHCP server is 171.64.7.77,
- your Ethernet address is 00:11:22:33:44:55
- the IP address you'll be given is 171.64.7.22
- the gateway IP address is 171.64.7.1
- the gateway Ethernet address is 00:66:77:88:99:00
- the netmask is 255.255.255.0

write down the series of packet exchanges that will occur on the link for your laptop to send a single IP packet to 128.30.2.1. You do not need to describe packets after this IP packet has left the link. Include ARP and DHCP packets, and when possible state the IP and Ethernet addresses of the packets. You do not need to write down message formats: simple descriptions such as "X opens a connection to Y on TCP port 23 and sends login information" are sufficient. If values are unspecified (e.g., DHCP server Ethernet address) you do not need to mention them. There is additional page of whitespace if you need it.

(extra space)

# VII Are Two TCPs Better Than One?

There are devices and services in the Internet, such as proxy servers, that "split" TCP connections. Suppose a host A wants to open a connection to a host C. A device somewhere along the path, B, can terminate A's connection at itself, and open a connection to C. So in this case there are now two TCP connections, A to B and B to C. A thinks it's sending data to C, but B is processing the TCP segments itself and sending acknowledgments back to A, spoofed from B's IP address. Simultaneously, B opens a TCP connection to C, pretending to be A.




Suppose you have the network above, where the RTT from A to B is 50ms, the RTT from B to C is 50ms, and there is no packetization, queueing, or processing delay, such that the RTT from A to C is 100ms. The maximum segment size is 1400 bytes. A is sending an infinite stream of bytes, such that every segment is the maximum segment size. Recall that a TCP flow's throughput can be approximated as

$$\text{MSS} \cdot \sqrt{\frac{3}{2}} \cdot \frac{1}{\text{RTT}\sqrt{p}}$$

where $p$ is the packet drop rate.

Please write out answers numerically and do not leave radicals or variables in your solutions. You may leave fractions. If you do not have a calculator, you may approximate with the following values:

$$MSS \cdot \sqrt{\tfrac{3}{2}} = 13,717 \text{ bits}$$

$$\sqrt{0.1} = 0.32$$

$$\sqrt{0.19} = 0.44$$

$$\sqrt{0.2} = 0.45$$

$$\sqrt{0.21} = 0.46$$

**14. [5 points]:**

Suppose that B does not split the TCP connection, such that packets flow directly from A to C, through B. The route between A and B drops 10% of data segments and does not drop acknowledgments, while the route between B and C does not drop any packet. What will the TCP throughput from A to C be?

____________________ kbps

**15. [5 points]:**

Suppose that B does split the connection, such that packets flow from A to B, terminate at B, then are forwarded in separate flow from B to C. The route between A and B drops 10% of data segements and drops no acknowledgments, while the route between B and C does not drop any packet. What will the throughput from A to C be?

____________________ kbps

**16. [5 points]:**

Suppose that B does split the connection, such that packets flow from A to B, terminate at B, then are forwarded in separate flow from B to C. The route between A and B drops 10% of packets, and the route between B and C also drops 10% of packets. What will the throughput from A to C be?

____________________ kbps

**17. [5 points]:**

Finally, suppose that B does not split the connection, such that packets flow from A to B, passing through but not terminating at B. The route between A and B drops 10% of data segments, and the route between B and C also drops 10% of data segments. What will the throughput from A to C be?

____________________ kbps

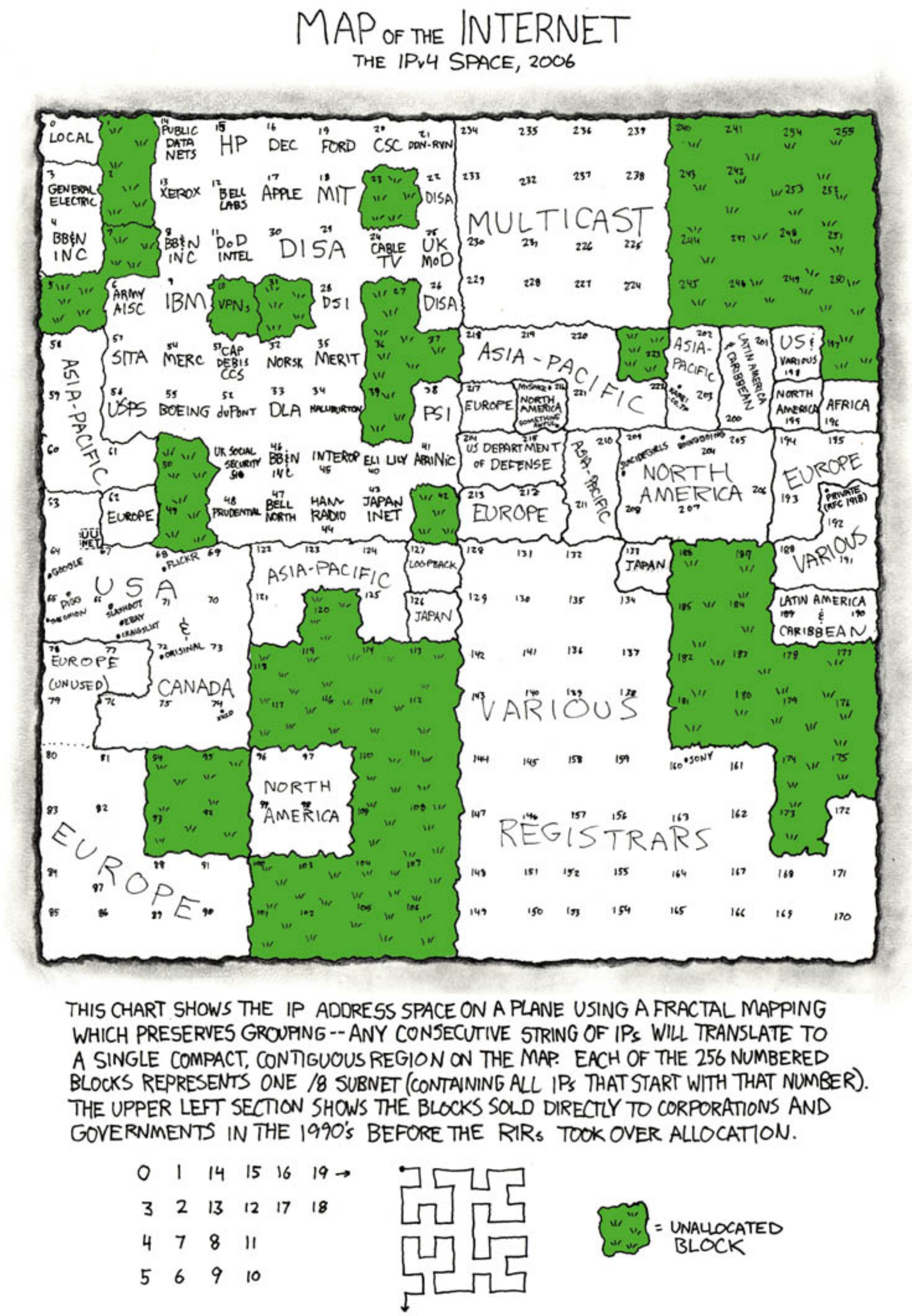



Figure 1: xkcd #194: Map of the Internet