Weaver Spring 2021 — CS 161 Computer Security — Final

For questions with **circular bubbles**, you may select exactly *one* choice on the Exam Tool.

◯ Unselected option

● Only one selected option

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

■ You can select

■ multiple squares

For questions with a **large box**, you need to write your answer in the text box on the Exam Tool.

There is an appendix at the end of this exam, containing descriptions of all C functions used on this exam.

You have 170 minutes, plus a 10-minute buffer for distractions or technical difficulties, for a total of 180 minutes. There are 10 questions of varying credit (200 points total).

The exam is open note. You can use an unlimited number of handwritten cheat sheets, but you must work alone.

Clarifications will be posted on the Exam Tool.

Q1 *MANDATORY – Honor Code* (5 points)

Read the following honor code and type your name on Gradescope.

> 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 and a corresponding notch on Nick's Stanley Fubar demolition tool.

Solution: Everyone gets 5 free points for embracing the suck this semester.

We won't take any points off if you entered something for a subpart that doesn't exist, or if you filled in a text box on a multiple-choice question, or vice-versa. To be consistent, we will not consider any unnecessary writing/bubbling on your exam during grading (pretend it's scratch work).

Q2 *True/false* **(38 points)**

Each true/false is worth 2 points.

Q2.1 TRUE or FALSE: WPA2 is a protocol that translates IP addresses to MAC addresses.

○ TRUE ● FALSE

Solution: False. ARP translates IP addresses to MAC addresses. WPA2 creates a secure connection at the link layer in a local wireless network and does not use IP addresses at all.

Q2.2 TRUE or FALSE: Specification-based detection uses a blacklist.

○ TRUE ● FALSE

Solution: False. Specification-based detection uses a whitelist. Signature-based detection uses a blacklist.

Q2.3 TRUE or FALSE: If a pseudorandom number generator (pRNG) is secure, then an attacker who only sees the output of the pRNG is unable to learn its internal state.

● TRUE ○ FALSE

Solution: True. If the internal state of the PRNG is known, then all future outputs can be predicted.

Q2.4 TRUE or FALSE: Argon2 and PBKDF2 are appropriate algorithms to use when hashing and storing passwords in a database.

● TRUE ○ FALSE

Solution: True. Argon2 is a slow hashing algorithm specifically designed for hashing passwords.

Q2.5 TRUE or FALSE: All forms of two-factor authentication (2FA) are resistant to phishing attacks.

○ TRUE ● FALSE

Solution: False. The only method of 2FA that is resistant to phishing attacks by design is security keys/WebAuthn/U2F. All other methods are vulnerable to phishing attacks.

Q2.6 TRUE or FALSE: Logging is a method of intrusion detection in which server log files are preserved so they can be asynchronously scanned to detect malicious activity.

● True ○ False

Solution: True. This is the definition of logging.

Q2.7 True or False: Cryptographically secure MACs can be constructed using secure cryptographic hash functions.

● True ○ False

Solution: True. HMAC, a secure MAC construction, is constructed using secure hash functions.

Q2.8 True or False: When analyzing a cryptographic hashing scheme, preimage resistance (one-way) implies collision resistance.

○ True ● False

Solution: False. Preimage resistance says that given y, it is computationally hard to find any x such that $H(x) = y$. Collision resistance says that it is computationally hard to find $a \neq b$ such that $H(a) = H(b)$. While these properties are related, one does not imply the other.

Q2.9 True or False: Sending all DNS requests and responses over HTTPS can be used as an effective defense against censorship by preventing censors from knowing what websites you are visiting.

○ True ● False

Solution: False. The censor may be unable to see the contents of the DNS response (because it's encrypted in TLS), but the censor can still see what name servers you're talking to and deduce what websites you're visiting. The censor can also see what websites you're visiting when you make a connection to the actual website.

Q2.10 True or False: One-time pads, as long as they are used correctly, are secure against an adversary with infinite computational power.

● True ○ False

Solution: True. As long as your key is randomly generated, XORing the key with the plaintext will result in a truly random ciphertext.

Formally, given a ciphertext, every single plaintext is equally likely, because each plaintext corresponds to a different key, and each key is equally likely. Even with infinite computational power, an attacker cannot determine what plaintext was sent.

Q2.11 True or False: TLS is able to prevent on-path attackers from learning metadata about your communications (e.g. request and response times, message length) by encrypting communications from a client to a server.

○ True ● False

Solution: False. TLS does not hide message length. Recall that TLS uses symmetric encryption to encrypt messages after the plaintext, and symmetric encryption does not hide message length.

Q2.12 True or False: Publicly accessible stairs, walkways, and elevators can be considered part of the physical equivalent of a trusted computing base for airport security.

○ True ● False

Solution: False. None of these items need to be relied upon for the correct functioning of airport security. Things like metal detectors, X-ray scanners, and TSA agents would be considered part of the TCB, since their malfunctioning would affect the effectiveness of airport security.

Q2.13 True or False: Clickjacking refers to a class of attacks where the attacker manipulates the user interface of a website to convince the user to click something that they did not intend to click on.

● True ○ False

Solution: True. Clickjacking attacks are often accomplished through the use of JavaScript, CSS, and iframes to create a web page that entices users to click on a dialog they would not otherwise click on.

Q2.14 Consider two different detectors with the same false positive rate and false negative rate. Assume that false negatives and false positives are equally costly.

True or False: A website with a high volume of users but a low volume of attacks would benefit more from placing the detectors in series rather than in parallel.

● True ○ False

Solution: True. Detectors in series produce a lower false positive rate but higher false negative rate, which results in lower errors overall due to the high volume of users.

Q2.15 True or False: Signature-based intrusion detection systems are good at identifying novel network attacks that have not been previously seen.

○ True ● False

Solution: False. Novel attacks that have not been seen before will not have existing signatures. Signature-based IDS does not identifying novel attacks.

Q2.16 True or False: A primary advantage of a host-based intrusion detection system (HIDS) over a network-based intrusion detection system (NIDS) is that traffic can be analyzed in plaintext, since the host can access decrypted TLS traffic.

● True ○ False

Solution: True. It is hard for a NIDS to analyze encrypted TLS traffic, because TLS is designed to be end-to-end secure between the client and server. However, a HIDS is installed directly on the server, so it can access decrypted TLS traffic.

Q2.17 True or False: For organizations with a large number of network devices, network-based intrusion detection systems (NIDS) are easier to deploy and manage than host-based intrusion detection systems (HIDS).

● True ○ False

Solution: True. NIDS can be deployed at a single on-path location, while HIDS must be deployed on every single host and kept in sync with the latest rules/signatures/configuration.

Q2.18 True or False: The UDP protocol guarantees that packets are delivered to the destination server by detecting dropped packets and retransmitting them until they are acknowledged.

○ True ● False

Solution: False. UDP is best-effort and provides no delivery guarantees.

Q2.19 True or False: Alice decides to use Tor to protect herself from tracking and surveillance online. The Tor circuit contains three Tor nodes: an entry node, a relay node, and an exit node. Assume the nodes do not collude. The exit node knows Alice's IP address but not the domain of the website she is visiting.

○ True ● False

Solution: False. Using three nodes in the circuit, the Tor exit node will know the domain of the destination website, but does not know which IP address initiated the request. The exit node only knows the IP address of the middle relay node from which it received the request.

Q2.20 True or False: `EvanBot is a real bot.` (0 points)

● True ○ False

Solution: True. How dare you doubt our trusty AI.

Q3 *Full Stack Security* (17 points)

Examtool is a test-taking website located at `https://exam.cs161.org/`. Assume that all network connections are made over HTTPS, unless otherwise specified.

Examtool uses session tokens for user authentication. Session tokens are stored as cookies with `Domain=exam.cs161.org` and no other cookie attributes (no `Secure` flag, no `HttpOnly` flag, `Path=/`).

When a student fills out or changes an answer, their browser makes a POST request to `https://exam.cs161.org/submit_question` with the student's updated answers.

Q3.1 (5 points) Which of the following attacks could allow an adversary to read the session token cookie? Select all that apply.

■ (A) Reflected XSS attack at `https://exam.cs161.org/`

■ (B) Stored XSS attack at `https://exam.cs161.org/`

■ (C) Exploitable buffer overflow vulnerability in the student's browser

☐ (D) Root access to another device on the same Wi-Fi network that the student is using

☐ (E) Root access to the Wi-Fi access point that the student is using

☐ (F) None of the above

Solution: An XSS attack (either stored or reflected) on `https://exam.cs161.org` would allow the attacker to execute arbitrary JavaScript on `https://exam.cs161.org`. JavaScript on `https://exam.cs161.org` can access a cookie with `Domain=exam.cs161.org` and no `HttpOnly` flag. For example, the attacker could use the XSS vulnerability to inject JavaScript that reads the session token cookie and sends its value to the attacker.

An exploitable buffer overflow vulnerability could allow an attacker to execute arbitrary code as the browser. If the attacker controls the entire browser application, they could read the cookies stored in the browser.

Root access to another device on the same Wi-Fi network makes the attacker an on-path network attacker. Root access to the Wi-Fi access point makes the attacker a MITM network attacker. However, since the requests are made with HTTPS/TLS, and TLS is end-to-end secure, network attackers will not be able to read cookie values (which are encrypted in TLS).

Q3.2 (4 points) For a question on an exam, Alice first submits "A" and then later changes her answer and submits "B". What could a MITM attacker between Alice's computer and the `exam.cs161.org` server do? Select all that apply.

■ (G) Perform a DoS attack to prevent Alice from submitting an answer choice

☐ (H) Perform a replay attack to restore Alice's saved answer to "A"

☐ (I) Modify Alice's submitted answer choice to "C"

☐ (J) Run JavaScript in Alice's browser

☐ (K) None of the above

☐ (L) ——

Solution: HTTPS/TLS is end-to-end secure, so a MITM attacker cannot modify the content of Alice's message. This prevents an attacker from modifying Alice's answer choice (the contents of the message). The attacker also cannot modify the response from the server to send some JavaScript to Alice's browser.

However, TLS does not guarantee availability, so a MITM could prevent Alice's request from reaching the server. For example, the MITM could use TCP RST injection to end the TLS connection (remember that TLS runs on top of TCP).

Q3.3 (4 points) Suppose the MITM attacker has identified a vulnerability in HTTPS that allows them to arbitrarily read and modify data in transit without detection. Alice submits another answer. What could a MITM attacker between Alice's computer and the `exam.cs161.org` server do? Select all that apply.

■ (A) Set cookie values for the page at `https://exam.cs161.org/`

■ (B) Redirect Alice's browser to `https://evil.com/`

☐ (C) Access any file on Alice's computer

■ (D) Change Alice's answer choice without detection

☐ (E) None of the above

☐ (F) ——

Solution: The MITM attacker can now modify the contents of Alice's message and change Alice's answer choice.

The MITM can also modify the contents of the server's response to send some Javascript to Alice's browser. This Javascript looks like it came from `https://exam.cs161.org`, so it could set a cookie for `https://exam.cs161.org`. Javascript can also redirect Alice to another website.

Remember that browsers are sandboxed, so JavaScript from a website cannot access files on Alice's computer.

The following subparts are independent of the previous subparts.

An instructor uploads an exam to Examtool by applying some cryptography to the exam and sending it over an insecure channel.

Assumptions:

- m is the message to encrypt (i.e. the exam).

- $\|$ is concatenation.
- k_1 and k_2 are two different secret keys known only to the Examtool server and the instructor.
- $E(k, m)$ is the encryption function of an IND-CPA secure symmetric encryption scheme.
- $\text{MAC}(k, m)$ is a secure MAC function.

For each pair of cryptographic schemes, select the scheme with fewer potential vulnerabilities.

Q3.4 (2 points) Select the more secure scheme:

◯ (G) $C = C_1 \| C_2$, where $C_1 = E(k_1, m)$ and $C_2 = \text{MAC}(k_1, C_1)$

● (H) $C = C_1 \| C_2$, where $C_1 = E(k_1, m)$ and $C_2 = \text{MAC}(k_2, C_1)$

◯ (I) ——

◯ (J) ——

◯ (K) ——

◯ (L) ——

Solution: The first option reuses a key k_1 in two different algorithms. The second option uses different keys k_1 and k_2 for two different algorithms. Key reuse is insecure, so the second option is more secure.

Q3.5 (2 points) Select the more secure scheme:

● (A) $C = C_1 \| C_2$, where $C_1 = E(k_1, m)$ and $C_2 = \text{MAC}(k_2, C_1)$

◯ (B) $C = E(k_1, m || \text{MAC}(k_2, m))$

◯ (C) ——

◯ (D) ——

◯ (E) ——

◯ (F) ——

Solution: As shown in lecture, encrypt-then-MAC is more secure than MAC-then-encrypt.

The first option is encrypt-then-MAC, because the message is encrypted first, and then the MAC is applied on the ciphertext. The second option is MAC-then-encrypt, because the plaintext is MAC'd first, and then the message and MAC are encrypted.

Q4 *"Bank-Grade" Security* **(28 points)**

Bear Bank is using a third-party analytics service called ABtesters. To use it, the bank website includes a tag to load the ABtesters JavaScript library.

Bear Bank's website is located at `https://bearbank.com` and contains the following HTML:

```
1 <script src="https://cdn.abtesters.com/lib.js"></script>
2 <form name="login" action="/login" method="POST">
3   <input type="text" name="username" />
4   <input type="password" name="password" />
5 </form>
```

Q4.1 (5 points) In the same-origin policy, which of the following are used in determining the origin of an HTTP webpage? Select all that apply.

☐ (A) IP ■ (C) Protocol ☐ (E) Request path

■ (B) Port ■ (D) Domain name ☐ (F) None of the above

Solution: Web content's origin is defined by the scheme (protocol), host (domain), and port of the URL used to access it. Two objects have the same origin only when the scheme, host, and port all match.

Q4.2 (3 points) Bear Bank is concerned that the ABtesters JavaScript library could steal customer passwords from the login form if the JavaScript library were compromised. Is this a valid concern?

◯ (G) Yes, because the ABtesters JavaScript library executes with the origin of ABtester's webpage.

● (H) Yes, because the ABtesters JavaScript library executes with the origin of Bear Bank's webpage.

◯ (I) No, because `https://cdn.abtesters.com` uses a certificate that is signed for different domain name.

◯ (J) No, because the ABtesters JavaScript library can only execute specific JavaScript functions required for its basic functionality.

◯ (K) —

◯ (L) —

Solution: JavaScript inherits the origin of the page that loads it, so the ABtesters library will have the same origin as `bearbank.com`. This lets the ABtesters JavaScript access elements on `bearbank.com`, such as the login form.

Q4.3 (3 points) Bear Bank decides to move the login form to `https://auth.bearbank.com` and embed it on the homepage (`https://bearbank.com/`) in an iframe.

Can the ABtesters JavaScript library running on Bear Bank's homepage steal customer passwords from the login form in the iframe?

◯ (A) Yes, because the ABtesters JavaScript library is running on the same page as the iframe.

◯ (B) Yes, because the ABtesters JavaScript library can execute any JavaScript it wants on the Bear Bank's homepage.

◯ (C) No, because the ABtesters JavaScript library is not developed by Bear Bank itself.

● (D) No, because the ABtesters JavaScript library has a different origin than the login form.

◯ (E) ——

◯ (F) ——

Solution: No. iframes have the origin of the website that's being loaded, so the login form (domain `auth.bearbank.com`) inside the iframe has a different origin from the JavaScript library (domain `bearbank.com`) outside the iframe. The ABtesters JavaScript library outside the iframe will not be able to directly interact with the login form inside the iframe.

After a user successfully logs into their account, Bear Bank's website sets a `session_token` cookie to track the user's logged in status and allows users to transfer funds by making a GET request to `https://bearbank.com/transfer`.

Q4.4 (3 points) Which of the following cookie attributes would cause the `session_token` cookie to be sent in a request to `https://bearbank.com/transfer`? Select all that apply.

☐ (G) Domain=bearbank.com; Path=/transactions

■ (H) Domain=bearbank.com; Path=/transfer; Secure

☐ (I) Domain=auth.bearbank.com; Path=/login; HttpOnly; Secure

☐ (J) None of the above

☐ (K) ——

☐ (L) ——

Solution: The browser sends a cookie to a given URL if the cookie's Domain attribute is a domain-suffix of the URL domain, and the cookie's Path attribute is a prefix of the URL path. In other words, the URL domain should end in the cookie's Domain attribute, and the URL path should begin with the cookie's Path attribute.

The first option will not be sent because the cookie path `/transactions` does not match the website path `/transfer`.

The second option will be sent because the domain and path both match, and the request is over HTTPS (so the Secure flag does not stop the cookie from being sent).

The third option will not be sent because the cookie path `/login` does not match the website path `/transfer`.

Note that the HttpOnly flag prevents cookies from being loaded by JavaScript, so it is irrelevant here.

Q4.5 (3 points) Bear Bank realizes that there are no CSRF protections on the transfer form, which means attackers can steal money from users' accounts.

Which of the following are reliable defenses against CSRF attacks? Select all that apply.

Clarification during exam: Everyone will receive credit for this question because we did not specify what it means for a defense to be "reliable."

■ (A) Add a random CSRF token to the transfer form each time the page loads

■ (B) Check the referrer header on the server when processing the transfer form submission

☐ (C) Move the transfer form to an iframe hosted at `https://transfer.bearbank.com`

☐ (D) None of the above

☐ (E) ——

☐ (F) ——

Solution: The definition of "reliable" was not clear, so we gave everyone points for this question.

The intended solution was:

As discussed in lecture, CSRF tokens and checking the referer headers are valid CSRF defenses.

Submitting the transform form with a POST request is not a valid CSRF defense, because CSRF attacks are still possible on POST requests.

Moving the transfer form to a different origin does not prevent CSRF attacks.

The following subparts are independent of the previous subparts.

Tree Bank is different bank considering alternative security methods. Once a user is logged in, they can send HTTP requests to Tree Bank to make transactions. Each request contains a session token set by the server when the user first logged in. The requests do not contain any counters or timestamps. The requests are sent over HTTP (not HTTPS).

Eve is an on-path attacker.

Q4.6 (4 points) Eve observes a single request from EvanBot to Tree Bank, which contains a transaction. What can Eve do? Select all that apply.

■ (G) Learn EvanBot's session token

■ (H) Learn the contents of EvanBot's transaction

☐ (I) Learn EvanBot's password

■ (J) Repeat EvanBot's transaction

☐ (K) None of the above

☐ (L) ——

Solution: Eve sees the request and the session token. The token lets Eve hijack EvanBot's session. Eve can replay the transaction (or log in and re-execute the transaction manually). Eve never sees EvanBot's password in the request.

Q4.7 (4 points) Assume that the user knows Tree Bank's public key, and Tree Bank's corresponding private has not been compromised. Suppose that Tree Bank requires that the user encrypt the entire HTTP request (including the transaction and token) with the ElGamal scheme from lecture before sending it to the bank.

Eve observes a single encrypted request from EvanBot to Tree Bank, which contains a transaction. What can Eve do? Select all that apply.

☐ (A) Learn EvanBot's session token

☐ (B) Learn the contents of EvanBot's transaction

☐ (C) Learn EvanBot's password

■ (D) Repeat EvanBot's transaction

☐ (E) None of the above

☐ (F) ——

Solution: El Gamal is vulnerable to replay attacks, so Eve can replay EvanBot's transaction.

However, El Gamal does not leak plaintext if Eve only sees a single encryption.

Q4.8 (3 points) What is the best way for the bank to defend against Eve's attacks, and what concept best describes the design flaw that allowed Eve to compromise EvanBot's requests?

◯ (G) Use DNSSEC. Don't build your own crypto.

● (H) Use TLS. Don't build your own crypto.

◯ (I) Use DNSSEC. Security is economics.

◯ (J) Use TLS. Security is economics.

◯ (K) Use DNSSEC. Least privilege.

◯ (L) Use TLS. Least privilege.

Solution: TLS gives us end-to-end security for communication across an insecure channel. The bank and user are trying to build their own crypto which is a bad idea.

Q5 *TC sPeedy* **(15 points)**

To improve the speed of TCP, Alice suggests modifying the TCP protocol to allow data to be sent in the SYN and SYN-ACK packets during the 3-way handshake. The data in the SYN packet is immediately accepted by the server during the initial handshake (before the 3-way handshake finishes).[1]

Clarification during exam: In sub-parts 4 and 5, in subsequent connections, the token is sent only in the SYN packet.

Q5.1 (3 points) Which of the following attacks are possible on this modified scheme? Select all that apply.

Clarification during exam: "Reliably"means that the attacker doesn't have to guess any values.

☐ (A) An off-path attacker can reliably inject packets after a connection has been established.

☐ (B) An off-path attacker can reliably execute a RST injection attack.

■ (C) An off-path attacker can fool the server into accepting some spoofed data.

☐ (D) None of the above

☐ (E) ——

☐ (F) ——

Solution: After a connection has been established, an off-path attacker needs to guess sequence numbers to inject a packet, so they cannot reliably inject packets into the connection or execute a RST injection attack.

However, the off-path attacker can spoof a SYN packet with some data. Since the SYN packet only contains a random initial sequence number, the off-path attacker doesn't need to guess any sequence numbers. Since the server immediately accepts data in the SYN packet, the server will accept this spoofed data.

Q5.2 (2 points) Alice notices that her modified scheme may be vulnerable to a DoS attack where the attacker sends a large data payload in the SYN packet without completing the TCP handshake. She proposes including SYN cookies as part of her modification.

TRUE or FALSE: SYN cookies provide a valid defense against the proposed DoS attack.

◯ (G) True ● (H) False ◯ (I) —— ◯ (J) —— ◯ (K) —— ◯ (L) ——

Solution: No, SYN cookies do not protect against an attacker who conducts a DoS attack by sending a large data payload in the SYN packet. SYN cookies merely protect against the accumulation of connection metadata due to incomplete 3-way handshakes. However,

[1] TCP Fast Open (TFO) started out in 2011 in an effort to reduce the the overhead of TCP's 3-way handshake. However, the TFO project was scrapped and is currently disabled on all browsers. In this question, you will explore the TFO protocol and its potential vulnerabilities.

regardless of whether SYN cookies are enabled or not, the server will still have to consume resources to receive data before the 3-way handshake is complete, making it vulnerable to a DoS attack.

The intent of this question is for students to analyze the ability of SYN cookies to mitigate the DoS vulnerability presented by the inclusion of data as part of the SYN packet, prior to the completion of the 3-way handshake.

Q5.3 (4 points) Alice uses her modified 3-way handshake to form a TCP connection with a server. Assume that source port randomization is not in use.

What fields would an **on-path attacker** have to guess in order to inject some data from Alice's client to the server?

- ☐ (A) Client IP address and port
- ☐ (B) Server IP address and port
- ☐ (C) Server sequence number
- ☐ (D) Client sequence number
- ■ (E) None of the above
- ☐ (F) ——

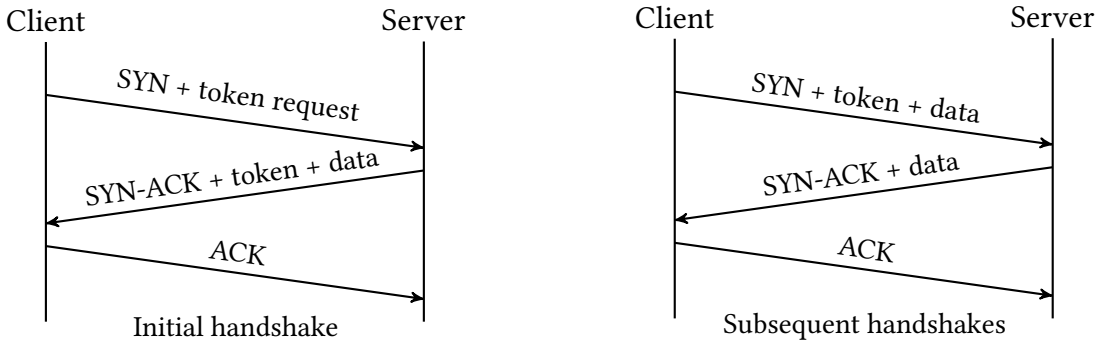
Solution: The on-path attacker can see all the fields in the unencrypted TCP packet, including both IP addresses, both ports, and both sequence numbers, so they don't need to guess anything.

Alice modifies her protocol to use a cryptographic token. When a client and server connect for the first time:

1. The client sends a SYN packet with a token request.
2. The server generates a token using a MAC function with a key known only to the server and responds with a SYN-ACK packet to the client containing the token. The client and server both store the token.
3. The client responds with an ACK packet, as in normal TCP.

In subsequent connections, the client skips the 3-way handshake by sending the SYN packet with both the token and data (similar to Alice's modification from previous parts). The server verifies the value of the token and acknowledges both the SYN and the data. The server may begin sending data to the client before receiving the client's ACK as part of the handshake. The server rejects the SYN and data if the token is invalid.

Here are diagrams detailing the protocol:



Q5.4 (3 points) Which of the following attacks on TCP becomes more difficult with the addition of the token? Select all that apply.

☐ (G) RST injection

☐ (H) Blind hijacking

☐ (I) MITM hijacking

■ (J) None of the above

☐ (K) ——

☐ (L) ——

Solution: The token is merely intended to speed up the establishment of new connections in lieu of redundant 3-way handshakes. It makes no security provisions.

Q5.5 (3 points) A major issue with this protocol is that it is vulnerable to replay attacks, as an adversary can spoof a connection by replaying the token. A potential workaround is to modify the TTL (time to live) of the token. Name **one** benefit and **one** drawback of using a shorter TTL rather than a longer TTL.

Enter your answer in the text box on Exam Tool.

Solution: If one wants to mitigate replay attacks, it is intuitive to use a **short** TTL, since the window in which replay attacks can be conducted will be minimized. If one wants to maximize speed, it is intuitive to use a **long** TTL.

Q6 *UnicornBox v2* **(17 points)**

UnicornBox decides to implement 2-factor authentication (2FA).

The server stores a table of active codes with the following schema:

```
CREATE TABLE IF NOT EXISTS users (
    username TEXT,
    code TEXT,
    -- Additional fields not shown.
);
```

When a user wants to log in:

1. The user logs in by making a POST request with their username and password.
2. The server randomly generates a 10-digit numerical code and stores it in the `users` table.
3. The server sets a cookie with name = `auth_user` and value = the user's username in the user's browser. The server also sends a text to the user's phone with the code.
4. The user makes a GET request to `https://unicornbox.com/confirm?code=$code`, where `$code` is the code that was entered.
5. The server runs the SQL query `SELECT username FROM users WHERE code = '$code'`, where `$code` is the value submitted by the user.
6. The server checks that the value returned by the SQL query matches the username sent in the `auth_user` cookie in the request submitted by the user.

Clarification during exam: For all sub-parts, the user has an entry in the table.

Clarification during exam: "CalCentral" should be "UnicornBox" in the question text.

Clarification during exam: In step 1, the server verifies the password and will not proceed if the password is wrong.

Q6.1 (5 points) Assume that `evan` is the name of an account in UnicornBox with an entry in the `users` table.

Construct an input for `$code` that would cause the SQL query in step 5 to return `evan`.

Enter your answer in the text box on Exam Tool.

Solution: One possible answer:

```
' OR username = 'evan' --
```

The first quote closes the opening quote in the query. The OR statement forces the query to return the username evan. The comment at the end forces the query to ignore the closing quote. The full query becomes

```
SELECT username FROM users WHERE code = '' OR username = 'evan' --'
```

Q6.2 (4 points) How can you log in as `evan` without knowing their password? You may use `PAYLOAD` to reference your answer from the previous part.

Hint: You will need 2 steps. List both.

Enter your answer in the text box on Exam Tool.

Solution: First, we should figure out where to put `PAYLOAD` from the previous question. Note that `$code` in the SQL query comes from the argument supplied to `https://unicornbox.com/confirm?code=$code` in Step 4, so we probably want to make a GET request to this URL with the payload. Next, note that in Step 6, the result of the SQL query is compared to the username in the cookie, so we want to set a cookie with the name evan. In summary:

First: Set a cookie `auth_user=evan`.

Second: Make a GET request to `https://unicornbox.com/confirm?code=PAYLOAD`.

Q6.3 (4 points) Which of these defenses would stop your exploit from above? Select all that apply.

■ (A) Using SQL prepared statements

☐ (B) Rate limiting requests to the UnicornBox server

☐ (C) Putting the hash of the username in the cookie instead of the username

☐ (D) Using a 20-digit code instead of a 10-digit code

☐ (E) None of the above

☐ (F) ——

Solution: Prepared statements will defend against the SQL injection attack, so the exploit will no longer work.

Rate limiting requests will not stop the exploit, since we only needed to send one GET request.

Putting the hash of the username in the cookie will not stop the exploit, since we can just modify the first step to put the hash of the victim's username in the cookie. (Remember that everyone can calculate a hash.)

A firewall would probably not be able to detect the contents of the exploit, because it's being sent over HTTPS.

Q6.4 (4 points) Consider a modification to Steps 5 and 6. If there are any rows returned by the SQL query, then the verification succeeds without checking the value of the returned username. However, the server returns an error without executing the query if the format of the code is not exactly 10 numerical digits.

TRUE or FALSE: The modified scheme is no longer exploitable using SQL injection. Briefly justify (1 sentence) your answer.

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

Enter your answer in the text box on Exam Tool.

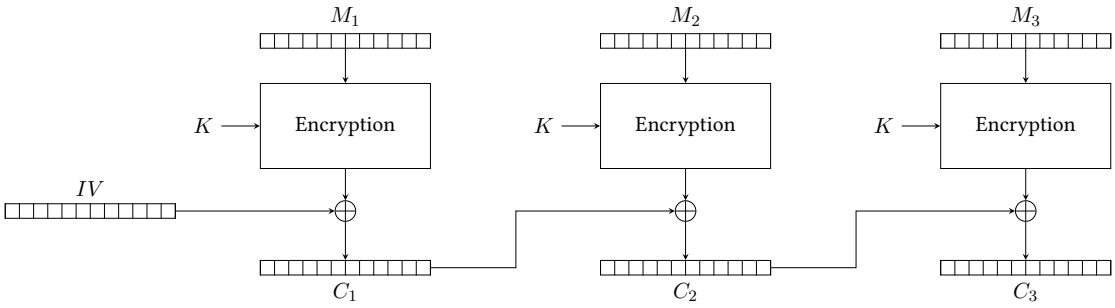
Solution: It is impossible to bypass the single quote if the input contains only numerical digits, so the scheme is safe from SQL injection.

Q7 *Plaintext Feedback* **(15 points)**

Consider the "plaintext feedback" (PFB) mode where the encryption formula for ciphertext block C_i is given as follows:

$$C_0 = IV$$
$$C_i = E(K, P_i) \oplus C_{i-1}$$

E is AES encryption and D is AES decryption.



Clarification during exam: IVs are always randomly generated and never reused in this question.

Clarification during exam: M_i in the encryption diagram refers to plaintext block P_i.

Q7.1 (3 points) Which of these is the corresponding decryption equation?

● (A) $P_i = D(K, C_i \oplus C_{i-1})$

○ (B) $P_i = D(K, C_i \oplus P_{i-1})$

○ (C) $P_i = D(K, P_i \oplus P_{i-1})$

○ (D) $P_i = E(K, C_i \oplus C_{i-1})$

○ (E) $P_i = E(K, C_i \oplus P_{i-1})$

○ (F) $P_i = E(K, P_i \oplus P_{i-1})$

Solution: $P_i = D(K, C_i \oplus C_{i-1})$

Q7.2 (3 points) Alice and Bob are communicating using PFB mode. Alice encrypts and sends a 10-block message encrypted using PFB. Bob receives the message, but the 6th ciphertext block C_6 is lost in transmission. Which blocks of plaintext can Bob recover? Assume Bob is aware that C_6 was lost in transmission.

○ (G) Bob can recover all blocks of the message.

○ (H) Bob can recover all blocks up to and including P_6, but no block after that.

○ (I) Bob can recover all blocks up to and including P_5, but no block after that.

● (J) Bob can recover all blocks except for P_6 and P_7.

○ (K) Bob can recover all blocks except for P_6.

◯ (L) Bob cannot recover any block of the message.

Solution: According to the decryption equation, to recover plaintext block P_j, we need C_j and C_{j-1}.

If we're missing C_6, we won't be able to decrypt P_6, because decrypting P_6 requires C_6 and C_5. We also won't be able to decrypt P_7, because decrypting P_7 requires C_7 and C_6. No other plaintext blocks depend on C_6 for decryption.

Q7.3 (3 points) PFB mode is not IND-CPA secure. To prove this, the adversary will win the IND-CPA game against the challenger as follows:

First, the adversary sends two messages, P and P'. The first message P is 3 unique, randomly generated blocks, $P = P_1 \| P_2 \| P_3$. Which of the following values of P' would allow the adversary to win the IND-CPA game?

● (A) $P' = P'_1 \| P'_1 \| P'_1$, where P'_1 is a randomly generated block

◯ (B) $P' = P'_1 \| P'_2 \| P'_3$, where P'_1, P'_2, and P'_3 are unique, randomly generated blocks

◯ (C) $P' = P_1 \| P_2 \| P_3$

◯ (D) $P' = P'_1 \| P'_2 \| P'_3$, where P'_i is the same as P_i, but with the last bit flipped

◯ (E) $P' = P'_1 \| P'_2 \| P'_3$, where P'_i is the same as P_i, but every bit flipped

◯ (F) ——

Solution: Intuitively, we want to pick a plaintext such that the ciphertext leaks some information about what message was encrypted.

First, we can eliminate (C), because if we sent two identical plaintext messages, the problem of deciding which message was encrypted wouldn't be well-defined.

We can also eliminate (D) and (E). If we flip some bits in the plaintext block, after it gets passed through the block cipher encryption, the output will be indistinguishable from random, so this won't help us distinguish what message was encrypted.

Similarly, if we send a completely randomly different message, the output of the block ciphers will still be indistinguishable from random.

However, if we choose the plaintext to be the same block repeated 3 times, the block cipher will produce the same output 3 times, because block ciphers are deterministic. We can leverage this to win the IND-CPA game (see the next part for how).

Q7.4 (3 points) The challenger sends back a ciphertext $C = C_0 \| C_1 \| C_2 \| C_3$, which is an encryption of either P or P'. Describe a strategy that the adversary should use to deduce whether P or P' was encrypted that would allow them to win the IND-CPA game with probability greater than $\frac{1}{2}$.

Enter your answer in the text box on Exam Tool.

Solution: Let's try running the encryption algorithm with a message that consists of one plaintext block repeated three times.

Note that we're denoting every P_i as P_1 because the plaintext blocks are repeated.

$$C_1 = E(K, P_1) \oplus C_0 = E(K, P_1) \oplus IV$$
$$C_2 = E(K, P_1) \oplus C_1 = E(K, P_1) \oplus E(K, P_1) \oplus IV = IV$$
$$C_3 = E(K, P_1) \oplus C_2 = E(K, P_1) \oplus IV$$

There are a few discernible patterns here to help you win the IND-CPA game. You could note that $C_0 = C_2 = IV$, or $C_1 = C_3$. More generally, if we pass in repeated plaintext blocks, the ciphertext will be two repeated blocks.

Also note that this pattern does not occur if we pass in random, different blocks, because the output of the block cipher will be different for each plaintext block.

Thus a winning strategy would be: if you see repeated ciphertext blocks, guess P. Otherwise, guess P'.

Another strategy is to XOR each ciphertext block with the previous ciphertext block. If the plaintext blocks are the same, then the result will be the same values:

$$C_1 \oplus C_0 = (E(K, P_1) \oplus IV) \oplus IV = E(K, P_1)$$
$$C_2 \oplus C_1 = (E(K, P_1) \oplus C_1) \oplus C_1 = E(K, P_1)$$
$$C_3 \oplus C_2 = (E(K, P_1) \oplus C_2) \oplus C_2 = E(K, P_1)$$

If the plaintext blocks are different, the results $E(K, P_1), E(K, P_2), E(K, P_3)$ would likely be different.

Q7.5 (3 points) Which of the following are true about PFB mode? Select all that apply.

■ (A) Decryption is parallelizable

☐ (B) PFB provides integrity

■ (C) The plaintext must be padded to a multiple of the block length

☐ (D) None of the above

☐ (E) ——

☐ (F) ——

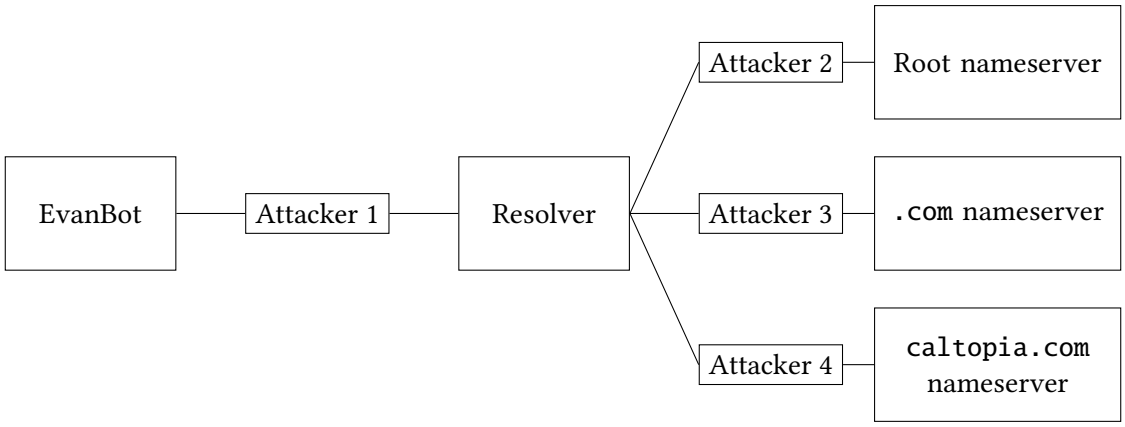
Solution: Decryption is parallelizable. Looking at the equation, we see that we need only ciphertext blocks C_i and C_{i-1} to generate the plaintext block P_i. We already have all the ciphertext blocks when we're decrypting, and we don't need to wait for any other plaintext blocks to be calculated before we can run the decryption algorithm on a block.

Block ciphers do not provide integrity.

Because the message blocks are being directly used as an input to the cipher, the blocks must be exactly the right length (128 B for AES), requiring padding.

Q8 *Caltopia DNS* (21 points)

EvanBot is trying to determine the IP address of `caltopia.com` with DNS. However, some attackers on the network want to provide EvanBot with the wrong answer.



Assumptions:

- Each attacker is a man-in-the-middle (MITM) attacker between their two neighbors on the diagram above.
- No attackers can perform a Kaminsky attack.
- Standard DNS (not DNSSEC) is used unless otherwise stated.
- No private keys have been compromised unless otherwise stated.
- In each subpart, both EvanBot's cache and the local resolver's cache start empty.
- Each subpart is independent.

Clarification during exam: Assume that bailiwick checking is in use for this entire question.

In each subpart, EvanBot performs a DNS query for the address of `caltopia.com`.

Q8.1 (4 points) In this subpart only, assume the attackers only passively observe messages.

Which of the attackers would observe an `A` record with the IP address of `caltopia.com` as a result of EvanBot's query? Select all that apply.

- [x] (A) Attacker 1
- [ ] (B) Attacker 2
- [ ] (C) Attacker 3
- [x] (D) Attacker 4
- [ ] (E) None of the above
- [ ] (F) ——

Solution: The `A` type record is sent from the `caltopia.com` name server to the resolver, and then from the resolver to EvanBot.

Q8.2 (3 points) Which of the attackers can poison the local resolver's cached record for `cs161.org` by injecting a record into the additional section of the DNS response? Select all that apply.

Note: Attacker 1 has intentionally been left out as an answer choice.

■ (G) Attacker 2 ☐ (I) Attacker 4 ☐ (K) ——

☐ (H) Attacker 3 ☐ (J) None of the above ☐ (L) ——

Solution: `cs161.org` is in bailiwick for root, so Attacker 2 could add a record for `cs161.org` in the response from root.

However, `cs161.org` is not in bailiwick for `.com` or `caltopia.com`, so attackers 3 and 4 cannot add a record for `cs161.org` in the responses from `.com` or `caltopia.com`.

Q8.3 (4 points) Assume that the resolver and the name servers all validate DNSSEC, but EvanBot does not validate DNSSEC. Which of the attackers can poison EvanBot's cached record for `caltopia.com` by modifying the DNS response? Select all that apply.

■ (A) Attacker 1 ☐ (C) Attacker 3 ☐ (E) None of the above

☐ (B) Attacker 2 ☐ (D) Attacker 4 ☐ (F) ——

Solution: Since the resolver and the name servers all validate DNSSEC, any attacker between the resolver and a name server can't do anything to inject malicious records. However, since EvanBot doesn't validate DNSSEC, Attacker 1 can inject a malicious `A` record.

Q8.4 (5 points) In this subpart only, assume the attackers only passively observe messages.

Assume that everyone validates DNSSEC. Which of the following records would Attacker 3 observe as a result of EvanBot's query? Select all that apply.

☐ (G) `DS` record with hash of the `.com` name server's public KSK

■ (H) `DS` record with hash of the `caltopia.com` name server's public KSK

☐ (I) `A` record with the IP address of `caltopia.com`

■ (J) `A` record with the IP address of the `caltopia.com` name server

■ (K) `DNSKEY` record with the `.com` name server's public KSK

☐ (L) None of the above

Solution: The `.com` name server returns:

- A `DNSKEY` record with its public keys (option K)
- An `NS` record with the domain of the next name server (`caltopia.com`)
- An `A` record with the IP of the next name server (`caltopia.com`) (option J)

- A `DS` record with hash of the next name server's public KSK (option H)

Option (G) would be returned by `.com`'s parent (the root), so Attacker 2 would see this record, not Attacker 3.

Option (I) would be returned by the `caltopia.com` name server, so Attacker 4 would see this, not Attacker 3.

Q8.5 (3 points) Assume that everyone validates DNSSEC, and the `caltopia.com` name server's private KSK has been compromised (i.e. all attackers know the `caltopia.com` name server's private KSK). No other private keys have been compromised.

Can EvanBot trust that they received the correct IP address of `caltopia.com`?

◯ (A) Yes, because the ZSK that signs the `A` record has not been compromised

◯ (B) Yes, because the trust anchor (the root's KSK) has not been compromised

◯ (C) No, because the compromised KSK can be used to sign a malicious `A` record

● (D) No, because the compromised KSK can be used to sign a fake ZSK that is used to sign a malicious `A` record

◯ (E) ——

◯ (F) ——

Solution: The chain of trust has been broken, so EvanBot can't trust that they received the correct IP address anymore.

The KSK is only used to sign ZSKs, so the attacker will have to sign a fake ZSK first, and then use the fake ZSK to sign the malicious `A` record.

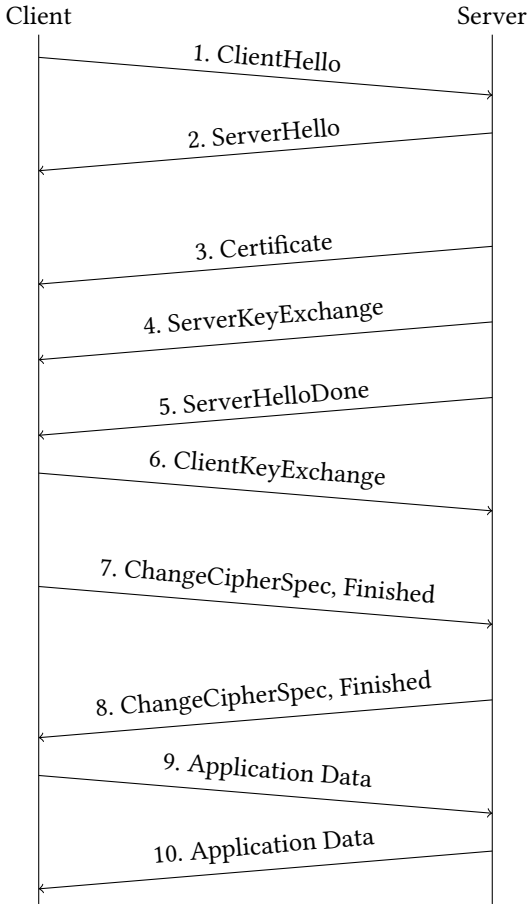
Q8.6 (2 points) TRUE or FALSE: DNSSEC prevents Attacker 4 from learning the IP address of `caltopia.com`.

◯ (G) True ● (H) False ◯ (I) —— ◯ (J) —— ◯ (K) —— ◯ (L) ——

Solution: DNSSEC provides no confidentiality over the DNSSEC records.

Q9 *Mutuality* **(18 points)**

Recall the TLS handshake:



1. Client sends 256-bit random number R_b and supported ciphers
2. Server sends 256-bit random number R_s and chosen cipher
3. Server sends certificate
4. DH: Server sends $\{g, p, g^a \bmod p\}_{K_{\text{server}}^{-1}}$
5. Server signals end of handshake
6. DH: Client sends $g^b \bmod p$
   RSA: Client sends $\{PS\}_{K_{\text{server}}}$
   Client and server derive cipher keys C_b, C_s and integrity keys I_b, I_s from R_b, R_s, PS
7. Client sends MAC(dialog, I_b)
8. Server sends MAC(dialog, I_s)
9. Client data takes the form $\{M_1, \text{MAC}(M_1, I_b)\}_{C_b}$
10. Server data takes the form $\{M_2, \text{MAC}(M_2, I_s)\}_{C_s}$

In TLS, we verify the identity of the server, but not the client. How would we modify TLS to also verify the identity of the client?

Clarification during exam: All parts of this question refer to a modified TLS scheme designed to verify the identity of the client.

Q9.1 (3 points) Which of these additional values should the client send to the server?

- ○ (A) A certificate with the client's public key, signed by the client's private key
- ○ (B) A certificate with the client's public key, signed by the server's private key
- ○ (C) A certificate with the client's private key, signed by a certificate authority's private key
- ● (D) A certificate with the client's public key, signed by a certificate authority's private key
- ○ (E) ——
- ○ (F) ——

Solution: This is analogous to the server sending its certificate, which has the server's public key, signed by the certificate authority's private key.

Q9.2 (3 points) How should the client send the premaster secret in RSA TLS?

● (G) Encrypted with the server's public key, signed by the client's private key

○ (H) Encrypted with the client's public key, signed by the server's private key

○ (I) Encrypted with the server's public key, signed by a certificate authority's private key

○ (J) Encrypted with the client's public key, signed by a certificate authority's private key

○ (K) —

○ (L) —

Solution: The client should encrypt the premaster secret with the server's public key so that the server can decrypt it (just like in regular TLS).

However, the client should additionally sign the premaster secret, so that the server can validate the signature and confirm that the server is talking to the correct client. The client should sign the premaster secret with their own private key. (The client doesn't know the certificate authority's private key, and the CA's private key is only used to sign certificates anyway.)

Q9.3 (3 points) EvanBot argues that the key exchange protocol in Diffie-Hellman TLS doesn't need to be changed to support client validation. Is EvanBot right?

○ (A) Yes, because only the client knows the secret a, so the server can be sure it's talking to the legitimate client

○ (B) Yes, because the server has already received and verified the client's certificate

● (C) No, the client must additionally sign their part of the Diffie-Hellman exchange with the client's private key

○ (D) No, the client must additionally sign their part of the Diffie-Hellman exchange with the certificate authority's private key

○ (E) —

○ (F) —

Solution: Diffie-Hellman on its own doesn't provide any authenticity. We also need the client to sign their Diffie-Hellman message.

Q9.4 (2 points) TRUE or FALSE: The server can be sure that they're talking to the client (and not an attacker impersonating the client) immediately after the client and server exchange certificates.

◯ (G) True ● (H) False ◯ (I) —— ◯ (J) —— ◯ (K) —— ◯ (L) ——

Solution: False. Remember that certificates are public, and attackers can present a certificate for anyone. The ClientHello and ServerHello messages only contain random nonces and an agreement on what algorithms to use, so they also do not give the client and server any guarantees about who they're talking to.

The client and the server need to wait at least until the signatures are exchanged to verify that they're talking to the correct person. If an attacker tampers with the handshake, the client and the server may even have to wait until the MACs are exchanged.

Q9.5 (3 points) At what step in the TLS handshake can both the client and server be sure that they have derived the same symmetric keys?

◯ (A) Immediately after the TCP handshake, before the TLS handshake starts

◯ (B) Immediately after the ClientHello and ServerHello are sent

◯ (C) Immediately after the client and server exchange certificates

◯ (D) Immediately after the client and server verify signatures

● (E) Immediately after the MACs are exchanged and verified

◯ (F) ——

Solution: The reasoning here is the same as in regular TLS. A MITM could tamper with messages, and the client and server will only detect this once they verify the MAC on the entire handshake.

Q9.6 (4 points) Which of these keys, if stolen individually, would allow the attacker to impersonate the client? Select all that apply.

■ (G) Private key of a certificate authority

■ (H) Private key of the client

☐ (I) Private key of the server

☐ (J) Public key of a certificate authority

☐ (K) None of the above

☐ (L) ——

Solution: If the attacker steals the private key of a trusted CA, they can sign a fake certificate claiming that the attacker's public key belongs to the client.

If the attacker steals the private key of the client, they can sign messages as the client.

Stealing the public key of the server doesn't help the attacker impersonate the client.

Q10 *Storefront* (26 points)

Consider the following vulnerable C code:

```
1  void copy_string(char *dst, const char *src, size_t n) {
2      for (size_t i = 0; i < n + 1; i++) {
3          dst[i] = src[i];
4          if (src[i] == '\0') {
5              break;
6          }
7      }
8  }
9
10 void add_to_store(char *lst) {
11     char listing[256];
12
13     copy_string(listing, lst, 256);
14
15     printf("Contacting server to add: %s...\n", listing);
16     contact_server_and_wait(listing); // Implementation not shown.
17 }
18
19 void invoke(char *lst) {
20     add_to_store(lst);
21 }
22
23 int main(void) {
24     char buf[4096];
25     do {
26         fgets(stdin, buf, 4096);
27         invoke(buf);
28     } while (strcmp(buf, "exit") != 0);
29     return 0;
30 }
```

Definitions of relevant C functions may be found on the last page of this exam.

Assume you are on a little-endian 32-bit x86 system. Assume that there is no compiler padding or saved additional registers in all questions. For the first four parts, assume that **no memory safety defenses** are enabled.

Clarification during exam: The strcmp function is identical to strncmp, except that it doesn't take an argument n.

Clarification during exam: There are no vulnerabilities present outside of the provided source code (so there are no vulnerabilities in `contact_server_and_wait`).

Clarification during exam: Line 26 should be `fgets(buf, 4096, stdin)`.

Q10.1 (3 points) Which of the following memory safety vulnerabilities is present in this code?

◯ (A) Format string vulnerability

◯ (B) Signed/unsigned vulnerability

● (C) Off-by-one

◯ (D) None of the above

◯ (E) —

◯ (F) —

Solution: The `copy_string` function contains an off-by-one vulnerability that allows the first byte immediately following the `dst` buffer to be overwritten with a NULL byte.

Q10.2 (3 points) Which of the following values on the stack can be partially or completely overwritten by the call to `copy_string` at line 13? Select all that apply.

Hint: Draw a stack diagram.

■ (G) `listing`

■ (H) SFP of `add_to_store`

☐ (I) RIP of `add_to_store`

☐ (J) None of the above

☐ (K) —

☐ (L) —

Solution: The attacker can overwrite all of `listing` and the least-significant byte of the sfp of `add_to_store` because of the off-by-one vulnerability in `copy_string`.

Q10.3 (6 points) Assume that the address of `listing` is `0xcffb5030`. Construct an input at Line 26 that would allow an attacker to execute malicious shellcode. You may reference the variable `SHELLCODE` as a 28-byte shellcode in your answer. Write your answer in Python 2 syntax (just like in Project 1).

Enter your answer in the text box on Exam Tool.

Solution: Taking advantage of the off-by-one vulnerability, we want to overwrite the least-significant byte of the SFP to point into the `description` buffer. The easiest method of doing this is to overwrite the least-significant byte to `0x28` by placing it as the 257th byte of our input, pointing the SFP 8 bytes before the end of the description buffer. The first 4 bytes are popped off as the SFP upon returning from the `invoke` function, and the last 4 bytes are popped of as the malicious RIP, which we will craft in the next part.

Next, we need to construct the first 256 bytes of our input. We place the shellcode at the beginning of the input. Next, we add dummy bytes for padding until we reach the last 4 bytes for our SFP, or $256 - 28 - 4 = 224$ bytes. Finally, we place the address of the shellcode and then the least-significant byte we decided on from earlier. We arrive at the following answer:

```
SHELLCODE + 'A' * 224 + '\x30\x50\xfb\xcf' + '\x28'
```

Q10.4 (3 points) Your exploit from above may not necessarily work with all possible addresses of `listing`. Provide **one** such address that would prevent your exploit from working. Write your answer in a format like `0xdeadbeef`.

Enter your answer in the text box on Exam Tool.

Solution: This question was broken, so we gave everyone points.

The original intended answer was: The off-by-one attack is not possible if the LSB rolls over to the next significant byte somewhere between the current value of the SFP (which is the address of the previous SFP) and 8 bytes before the end of the `listing` array. This can happen if the LSB of the address of the SFP is below `0x08` (so that there are fewer than 8 bytes in in the range of a modified SFP) or if the LSB of the address of the SFP is above `0xf4`, so that the current value of the SFP points to an address that is too high and rolls over to the next significant byte. The LSB of the address of the SFP is equal to the LSB of the address of `listing`, so any answer that satisfies one of these two criteria is correct.

However, some students noted that if the LSB of the address of the SFP is too low, an attacker could overwrite the LSB to make the address of the SFP *higher*. The attacker controls `buf`, which is located above the SFP, so the exploit would still work with a modified input to `buf`.

Q10.5 (3 points) Which of the following techniques could an attacker use to execute malicious shellcode if W^X and no other defenses are enabled? Select all that apply.

- ■ (A) Return-oriented programming
- ☐ (B) ret2esp
- ☐ (C) Server-side request forgery
- ☐ (D) None of the above
- ☐ (E) ——
- ☐ (F) ——

Solution: Return-oriented programming is the standard way of bypassing W^X, by chaining code snippets present in existing code to replicate the desired malicious behavior. ret2esp is an ASLR bypass technique that is not relevant to W^X, and server-side request forgery is a vulnerability in web security that is not relevant to memory safety.

Q10.6 (4 points) True or False: Stack canaries with no other defenses would prevent an attacker from executing malicious shellcode in this code (not necessarily using your exploit from above). Assume that all 4 bytes of the stack canary are randomized. Justify your answer.

● True ○ False

Enter your answer in the text box on Exam Tool.

Solution: It is actually possible to leak the value of the stack canary in this question. By inputting a string of length at least 256, `printf` will print the value of `listing` followed by the value of the stack canary since there are no NULL bytes on the stack.

However, the stack canary prevents malicious code from being executed since the off-by-one vulnerability no longer overwrites the SFP, preventing the off-by-one attack seen in Project 1.

Q10.7 (4 points) TRUE or FALSE: ASLR with no other defenses would prevent an attacker from executing malicious shellcode in this code (not necessarily using your exploit from above). Justify your answer.

○ TRUE ● FALSE

Enter your answer in the text box on Exam Tool.

Solution: It is possible to leak an address on the stack by printing the arguments of `add_to_store`. By inputting a string of length at least 256, `printf` will print the value of `listing` followed by the SFP, RIP, and `lst`, which is a pointer to the stack. From there, an attacker can calculate the address of `listing` relative to `buf` and use that address to execute the off-by-one attack as before.

Q11 *Cat* **(0 points)**

What is the name of Nick's gray cat?


Enter your answer in the text box on Exam Tool.

Solution: Fuzzbucket, aka Willow, aka Babykitty, aka Fuzzybut, aka Little Pest

C Function Definitions

```
int strncmp(const char *s1, const char *s2, size_t n);
```

> The strncmp() function compares the first (at most) n bytes of two strings s1 and s2. It returns an integer less than, equal to, or greater than zero if s1 is found, respectively, to be less than, to match, or be greater than s2.

```
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