CS 161 Summer 2026 — Introduction to Computer Security — Final

Name: _____

Student ID: _____

This exam is 170 minutes long. There are 9 questions of varying credit. (100 points total)

Question:	1	2	3	4	5	6	7	8	9	Total
Points:	0	8	15	14	8	16	16	11	12	100

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

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

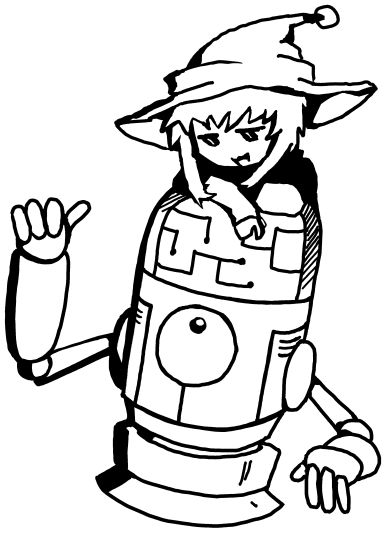
For questions with **square checkboxes**, you may select one or more choices.

- ■ You can select
- ■ multiple squares (completely filled).
- ☑ (Don't do this)

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

Pre-Exam Activity (0 points)

Crossover Episode:
CS 161 and CS 188 are finally teaming up!



Art by: rinklyrat

Who's the real brains of this operation?

- ○ EvanBot
- ○ CS 188 Bot
- ○ Other: _____

Q1 *Honor Code* 📜 **(0 points)**

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.

Read the honor code above and sign your name: _____

Q2 *Potpourri* 🍲 **(8 points)**

Each subpart is worth 0.5 points.

Q2.1 True or False: In the code to the right, the bounds check on Line 3 is sufficient to prevent an overflow of `buf`.

◯ True ◯ False

```
1 void copy(int len, char *src) {
2     char buf[16];
3     if (len <= 16) {
4         memcpy(buf, src, len);
5     }
6 }
```

Q2.2 A server builds a SQL query by concatenating a base query string with user input.

True or False: Escaping every quote character in the user input will always prevent SQL injection.

◯ True ◯ False

Q2.3 True or False: A script injected into a page served by `bank.com` runs with the origin of `bank.com`, not with the origin of the attacker's site.

◯ True ◯ False

Q2.4 True or False: Two pages share the same origin only if their protocol, domain, and port all match exactly.

◯ True ◯ False

Q2.5 True or False: If two different sites both agree, they can opt to share data despite the same-origin policy.

◯ True ◯ False

Q2.6 An attacker steals a database of hashed, salted passwords.

True or False: Salting stops the attacker from recovering any user's password by brute force.

◯ True ◯ False

Q2.7 How many operations are needed to find a collision in a hash function with a 256-bit output?

◯ 2^{64} ◯ 2^{128} ◯ 2^{256} ◯ 2^{512}

Q2.8 True or False: MACs prevent replay attacks.

◯ True ◯ False

Q2.9 A browser receives a certificate chain: a certificate for `evanbot.com`, signed by an intermediate CA, whose certificate is signed by a root CA.

What must be true for the browser to accept the certificate for `evanbot.com`?

- ◯ The browser must contact the root CA online to confirm the chain.
- ◯ The browser must already hold the root CA's public key.
- ◯ The root CA's certificate must be signed by a higher-level CA.
- ◯ The certificate must be signed directly by the root CA.

Q2.10 True or False: A PRNG seeded only with the current time in seconds is secure, as long as the PRNG algorithm itself is cryptographically strong.

◯ True ◯ False

Q2.11 True or False: An off-path attacker who wants to inject data into an existing TCP connection must guess the sequence numbers.

◯ True ◯ False

Q2.12 What does verifying a TLS certificate establish?

- ◯ The party the client is talking to holds the matching private key.
- ◯ The rest of the connection will have forward secrecy.
- ◯ The server has not been compromised since the certificate was issued.
- ◯ According to a trusted CA, the public key in the certificate belongs to the site in the certificate.

Q2.13 True or False: SYN cookies encode state into the sequence number, so that the server does not allocate memory until the client sends an ACK.

◯ True ◯ False

Q2.14 True or False: A prime and probe attack relies on detecting which cache lines are accessed quickly in the victim process (not the attacker process).

◯ True ◯ False

Q2.15 True or False: A valid defense against Spectre is to ensure that the results of all mispredicted branches are not written to disk, memory, or registers.

◯ True ◯ False

Q2.16 True or False: LLM prompt injection works because there is no reliable way to distinguish trusted instructions from untrusted data.

◯ True ◯ False

Q3 *Memory Safety: A Disastrous Format* 🖨️ **(15 points)**

Consider the following vulnerable C code:

```
1 void vuln() {
2     char buf[64];
3     fgets(buf, 64, stdin);
4     printf(buf);
5 }
```

Stack at Line 4

RIP of `vuln`
SFP of `vuln`
`buf`

Assumptions:

- All memory safety defenses are disabled.
- The RIP of `vuln` is located at address `0xffffd60c`.
- Your goal is to execute shellcode, which is already placed in memory at address `0x001f000d`.

Hint: Putting `N$` directly after the `%` causes a format specifier to operate on the `N`th argument to `printf`. Examples:

- `printf("%2$x", 10, 20)` prints `20`.
- `printf("%1$s", "hi", "hello")` prints `hi`.
- `printf("AAAA%3$hn", &one, &two, &three)` writes 4 (the number of bytes printed) to `&three`.
- `printf("%1$x")` in the vulnerable code above prints the first four bytes of `buf`.
- Note that `printf("%10c")` (without the `$`) still prints 10 characters.

Q3.1 (2 points) Which of these is true about the vulnerability on Line 4? Select all that apply.

- ☐ The `%x` and `%s` specifiers allow the attacker to read from memory.
- ☐ The `%n` specifier allows the attacker to write to memory.
- ◯ None of the above

Q3.2 (6 points) Provide an input to `buf` that executes shellcode.

```
'\x      \x      \x      \x      ' +
'\x      \x      \x      \x      ' +
'%      c' + '%      hn' +
'%      c' + '%      hn'
```

For Q3.3–Q3.4, assume stack canaries are enabled. All other defenses are still disabled.

Q3.3 (2 points) Which single format specifier prints the stack canary off the stack?

Note: This subpart is independent of the exploit in Q3.2.

Q3.4 (2 points) Can the exploit from Q3.2 be modified to work with stack canaries enabled?

- ◯ Yes, by writing around the stack canary.
- ◯ Yes, by overwriting the stack canary with itself.
- ◯ No, because the exploit overwrites the canary with garbage.
- ◯ No, because the exploit writes above the canary, and all memory above the canary is read-only.

Q3.5 (3 points) Which defenses would cause the correct exploit in Q3.2 (without modifications) to fail? Assume shellcode lives in an executable part of memory. Consider each choice independently.

- ☐ Change Line 4 to `printf("%s", buf)`.
- ☐ Enable NX (non-executable pages).
- ☐ Enable ASLR.
- ◯ None of the above

Q4 *Memory Safety: Call Me Maybe* ☎ (14 points)

Consider the following vulnerable C code:

```
1  void log_event() {
2      /* Implementation not shown. */
3  }
4
5  void process(char *input) {
6      void (*cb)(void) = log_event;
7      char buf[32];
8      strcpy(buf, input);
9      cb();
10 }
```

Stack at Line 7

(1)
RIP of `process`
SFP of `process`
(2)
(3)

Assumptions:

- Unless otherwise specified, all memory-safety defenses are disabled.
- `buf` is located at address `0xffffcf40`.
- `log_event` is located at address `0x08049180`.
- Your goal is to place and execute a 28-byte `SHELLCODE`.
- If part of the input can be any non-zero value, use `'A' * n` for `n` bytes of garbage.

(1 point each) What goes in blanks (1) – (3) in the stack diagram above?

Q4.1 Blank (1): ◯ `input` ◯ `cb` ◯ `buf`

Q4.2 Blank (2): ◯ `input` ◯ `cb` ◯ `buf`

Q4.3 Blank (3): ◯ `input` ◯ `cb` ◯ `buf`

Q4.4 (4 points) Provide an `input` to `strcpy` that causes `process` to execute `SHELLCODE`.
Your exploit must execute shellcode immediately after `process` returns (not at any other time).

For Q4.5–Q4.6, assume stack canaries are enabled.

Q4.5 (4 points) Provide an `input` to `strcpy` that still causes `SHELLCODE` to execute (at any time).

Q4.6 (3 points) Which of the following defenses, if enabled alongside stack canaries, would cause the exploit from Q4.5 (without modification) to fail?

Consider each defense independently. Select all that apply.

☐ NX / non-executable stack ☐ Add a second canary directly above the first

☐ ASLR ◯ None of the above

Q5 *Web Security: You've Got Mail* 📬 **(8 points)**

EvanMail is an email website that serves content from two domains:

- The user's sensitive emails are served from `https://mail.evanbot.com`.
- Untrusted attachments are served from `https://usercontent.evanbot.com`.

Q5.1 (2 points) Select all URLs with the same origin as `https://mail.evanbot.com/inbox`.

- ☐ `https://mail.evanbot.com/attachments/3`
- ☐ `https://mail.evanbot.com/`
- ☐ `http://mail.evanbot.com/inbox`
- ☐ `https://mail.evanbot.com:8080/inbox`
- ☐ `https://usercontent.evanbot.com/inbox`
- ◯ None of the above

Q5.2 (2 points) Why does using a separate domain for the attachments protect the emails?

- ◯ Because same-origin policy stops JavaScript in one domain from running in another domain.
- ◯ Because data from a separate domain is automatically scanned for malware by the browser.
- ◯ Because the browser encrypts all traffic to each domain separately.
- ◯ Because cookies set by one domain are automatically sent to the other.

Q5.3 (2 points) Consider two methods for displaying an attachment in an email. Which method(s) allow the attachment to access sensitive data in the email? Select all that apply.

- ☐ `<iframe src="https://usercontent.evanbot.com/attach/3"></iframe>`
- ☐ `<script src="https://usercontent.evanbot.com/attach/3"></script>`
- ◯ None of the above

Q5.4 (2 points) Alice opens an attachment in a separate browser tab. Select all cookies that the browser attaches to the request.

- ☐ Cookies with `Domain=evanbot.com`.
- ☐ Cookies with `Domain=mail.evanbot.com`.
- ☐ Cookies with `Domain=usercontent.evanbot.com`.
- ☐ Cookies with `Domain=more.usercontent.evanbot.com`.
- ◯ None of the above

Q6 *Cryptography: Galois in the Middle* 🎩 **(16 points)**

In Q6.1–Q6.4, Alice sends a three-block message to a server, encrypted with AES-CTR.

Mallory is an MITM attacker who can XOR the ciphertext with any value of her choosing.

Notation:
- Alice's message is: $M = M_1 \parallel M_2 \parallel M_3$.
- Mallory XORs the message with $\Delta = \Delta_1 \parallel \Delta_2 \parallel \Delta_3$.
- The server receives and decrypts $C = C_1 \parallel C_2 \parallel C_3$.

Q6.1 (2 points) Mallory knows M and wants the server to decrypt a different message M' of her choosing.

What value of Δ should Mallory use?
You may use M, M', and any mathematical operators and base-10 constants.

$\Delta =$

Q6.2 (1 point) AES-CTR is IND-CPA secure. Why does this not stop the attack in Q6.1?

- ◯ IND-CPA guarantees confidentiality, but not integrity.
- ◯ IND-CPA is theoretical and does not apply to real-world schemes.
- ◯ Mallory's attack only succeeds when Alice reuses the nonce.
- ◯ AES-CTR is deterministic.

Q6.3 (1 point) Mallory does not know M, and wants to flip the fourth bit of the last block (where the right-most bit is the 0th bit).

What values of Δ_1, Δ_2, and Δ_3 should Mallory use?
You may use any mathematical operators and base-10 constants.

$\Delta_1 =$	$\Delta_2 =$	$\Delta_3 =$

Q6.4 (2 points) Alice and the server switch to using Encrypt-then-MAC (encrypt the message, and attach a MAC on the ciphertext). Can Mallory still carry out the attack from Q6.3?

- ◯ Yes, because flipping a bit in the tag causes the corresponding plaintext bit to flip.
- ◯ Yes, because the server only checks the MAC on the plaintext, not the ciphertext.
- ◯ No, because changing C causes the server to compute an incorrect tag.
- ◯ No, because Mallory no longer knows where one block ends and the next block starts.

Q6.5 (2 points) What properties are necessary for a secure Authenticated Encryption (AE) scheme?

- ◯ Both IND-CPA and Ciphertext Integrity
- ◯ Only Ciphertext Integrity
- ◯ Only IND-CPA
- ◯ Neither IND-CPA nor Ciphertext Integrity

Q6.6 (2 points) Which of the following are true about AES-GCM? Select all that apply.

- ☐ The GMAC tag is computed over the ciphertext rather than the plaintext (Encrypt-then-MAC).
- ☐ AES-GCM supports additional data that is unencrypted but authenticated.
- ☐ Reusing the same (K, IV) pair to encrypt two different messages is safe in AES-GCM.
- ◯ None of the above

For Q6.7–Q6.8, use the Polynomial MAC scheme from lecture. For an ℓ-block message M, the tag is

$$t = \left(M_\ell \cdot k_0^\ell\right) + \left(M_{\ell-1} \cdot k_0^{\ell-1}\right) + \ldots + \left(M_1 \cdot k_0\right) + \left(k_1\right) \mod N.$$

Q6.7 (4 points) Alice forgets that Polynomial MACs are one-time secure, and reuses the same (k_0, k_1) for every message.

Alice sends two one-block messages. Mallory observes these two message–tag pairs:

$$M = 2 \to t = 13 \qquad M = 5 \to t = 28$$

Provide a forged tag for a new message $M^* = 4$.

For simplicity, ignore mod N in your calculations. Your answer should be a single integer.

Hint: Try solving for k_0 and k_1.

$t^* =$

Q6.8 (2 points) Is Polynomial MAC vulnerable to a length extension attack? In other words, given only a single t (but not the corresponding M), can the attacker compute the tag of $M \parallel M'$ without knowing the key?

- ◯ Yes, because Polynomial MAC uses the Merkle-Damgård construction.
- ◯ Yes, because the attacker can multiply the tag by M and add k_1.
- ◯ No, because k_1 is added at the very end, which hides the internal state.
- ◯ No, because length extension attacks only apply to hashes, not MAC schemes.

Q7 *Cryptography: No Take-Backs* 🔒 **(16 points)**

Consider a Diffie-Hellman key exchange where Alice sends $A = g^a \bmod p$, and Bob sends $B = g^b \bmod p$.

Q7.1 (2 points) For this subpart only, consider a modified exchange where the shared key is $g^{a+b} \bmod p$.

Write an expression that an eavesdropper can evaluate to recover the shared key.
You may use g, p, A, B, and any mathematical operators and constants.

Q7.2 (2 points) In the normal exchange, why can't someone who sees g, p, A, and B compute the key?

- ◯ Computing $g^{ab} \bmod p$ from $g^a \bmod p$ and $g^b \bmod p$ is hard.
- ◯ Factoring p is hard.
- ◯ Finding two values that hash to $g^{ab} \bmod p$ is hard.
- ◯ Running AES decryption without the secret key is hard.

For Q7.3–Q7.4, Mallory is a MITM attacker. She replaces $B = g^b \bmod p$ with $B' = B^c \bmod p$, where c is a value of her choosing.

Q7.3 (2 points) What are the shared keys K'_A and K'_B that Alice and Bob compute?
You may use g, p, a, b, c, and any mathematical operators and constants.

Alice computes:

$K'_A =$

Bob computes:

$K'_B =$

Q7.4 (2 points) Using only what Mallory observes (g, p, A, B, and c), can she compute K'_A from Q7.3?

- ◯ Yes, she evaluates $K'_A = A^{bc} \bmod p$.
- ◯ Yes, she evaluates $K'_A = A^c \cdot B^c \bmod p$.
- ◯ No, because computing $g^{abc} \bmod p$ from $g^a \bmod p$, $g^b \bmod p$, and c is hard.
- ◯ No, because factoring p is hard.

Q7.5 (2 points) Mallory runs a MITM attack, completing two separate exchanges with Alice and Bob.

After the attack, can Mallory read and modify all traffic between Alice and Bob?

- ◯ Yes, by decrypting and re-encrypting each message.
- ◯ Yes, by solving the discrete log problem.
- ◯ No, because Diffie-Hellman authenticates both parties.
- ◯ No, because she cannot solve the discrete log problem.

Q7.6 (2 points) Which defense prevents the MITM attack from Q7.5?

- ◯ Use a larger p, so that the discrete log problem becomes harder.
- ◯ Use signatures and certificates to verify $g^a \bmod p$ and $g^b \bmod p$.
- ◯ Use the same a and b values in all exchanges.
- ◯ Use AES to encrypt $g^a \bmod p$ and $g^b \bmod p$ before sharing them.

Q7.7 (2 points) Alice and Bob exchange $\text{Sign}(\text{SK}_{\text{Alice}}, g^a \bmod p)$ and $\text{Sign}\big(\text{SK}_{\text{Bob}}, g^b \bmod p\big)$ to derive $K = g^{ab} \bmod p$.

If Mallory records an exchange and steals SK_{Bob}, can she decrypt messages encrypted with K?

- ◯ Yes, because SK_{Bob} is used to derive the encryption key.
- ◯ Yes, because she can forge signatures as Bob.
- ◯ No, because Mallory must also know SK_{Alice}.
- ◯ No, because SK_{Bob} is not used to derive any encryption keys.

Q7.8 (2 points) Alice picks a random K and sends $\text{RSA-Enc}(\text{PK}_{\text{Bob}}, K)$ to Bob.

If Mallory records an exchange and steals SK_{Bob}, can she decrypt messages encrypted with K?

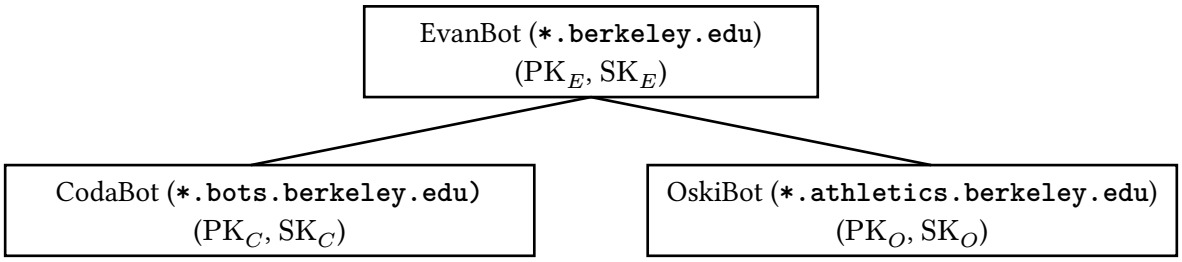
- ◯ Yes, because SK_{Bob} decrypts the messages.
- ◯ Yes, because SK_{Bob} decrypts the encryption key.
- ◯ No, because Mallory must also know SK_{Alice}.
- ◯ No, because SK_{Bob} is not used to derive any encryption keys.

Q8 *Cryptography and Web: EvanBot's Certificate Authority* 🤖 **(11 points)**

Each scenario in this question is independent.

Scenario 1: EvanBot (PK_E, SK_E) runs a certificate authority, and delegates signing power to two intermediate CAs:

- CodaBot (PK_C, SK_C) can issue certificates for `*.bots.berkeley.edu`,
- OskiBot (PK_O, SK_O) can issue certificates for `*.athletics.berkeley.edu`.



CodaBot has issued a certificate, saying "`cs161.bots.berkeley.edu` has public key PK_L."

Mallory steals SK_C. Alice's browser trusts only PK_E (not PK_C or PK_O) as a root.

Q8.1 (3 points) Which of these actions can Mallory perform? Select all that apply.

$(\text{PK}_M, \text{SK}_M)$ is a key pair that Mallory knows.

- ☐ Issue a valid certificate saying "`cs161.bots.berkeley.edu` has public key PK_M."
- ☐ Issue a valid certificate saying "`eecs.bots.berkeley.edu` has public key PK_M."
- ☐ Issue a valid certificate for `stadium.athletics.berkeley.edu`.
- ☐ Issue a valid certificate for `bank.com`.
- ☐ Forge EvanBot's signature on a message of Mallory's choosing.
- ☐ Decrypt previously recorded TLS traffic encrypted with PK_L.
- ◯ None of the above

Q8.2 (1 point) EvanBot notices that SK_C has been stolen and wants to publish a Certificate Revocation List (CRL). What certificate should EvanBot add to the list?

- ◯ CodaBot's certificate.
- ◯ Mallory's certificate.
- ◯ OskiBot's certificate.
- ◯ EvanBot's certificate.

Q8.3 (1 point) Why should the CRL be signed by EvanBot?

- ◯ To prevent Alice from reading the list of revoked certificates.
- ◯ To prevent an attacker from modifying the list.
- ◯ To compress the list so it transfers faster.
- ◯ To allow EvanBot to use TLS to share the CRL.

Scenario 2: Two CAs use different revocation strategies, as follows:

- **CA 1**: Certificates are valid for 60 days. No CRL is published.
- **CA 2**: Certificates are valid for 10 years. A CRL is updated every hour.

Q8.4 (1 point) The day a server's certificate is issued, an attacker steals the server's private key. If the client never fetches the CRL, which CA's revocation strategy will mitigate this attack first?

◯ CA 1 ◯ CA 2

Q8.5 (1 point) Which CA requires more work from website operators?

◯ CA 1 ◯ CA 2

Q8.6 (2 points) If a CRL temporarily goes down, what is the security vs. usability trade-off that clients face when trying to access a website? Write at most 10 words in each box.

The more secure choice:

The more usable choice:

Scenario 3: SSH uses trust-on-first-use.

Alice tries to connect to a server, but Mallory is a MITM attacker who might try to impersonate the server.

Q8.7 (1 point) Alice tries to connect to the server for the first time, and sees this warning: "Are you sure you want to save `server.example.com`'s public key?"

Which public key(s) might cause this error message to appear? Select all that apply.

☐ The server's public key ☐ Mallory's public key ◯ None of the above

Q8.8 (1 point) After saving the key from Q8.7, Alice tries to connect again and sees this warning: "`server.example.com`'s public key has changed."

Which public key(s) might cause this error message to appear? Select all that apply.

☐ The server's public key ☐ Mallory's public key ◯ None of the above

Q9 *Isolation: EvanBot's Epic Containerized Computer* (12 points)

EvanBot wants to design an Awesome Web Service—the Epic Containerized Computer™ (EC2).

EC2 should allow users to run any code of their choosing, while enforcing isolation between users.

Assumptions:
- No memory safety defenses are used.
- The code runs sequentially. No parallelism or multithreading is used.
- The system has finite compute and memory.

Version 1: Each user's code is copied into the same C program. Then, EC2 runs the single C program.

Q9.1 (1 point) Can an attacker break isolation in Version 1?

- ◯ Yes, because one user's code can access other users' variables.
- ◯ Yes, because memory is not split evenly between users.
- ◯ No, because the code is isolated by the operating system.
- ◯ No, because C is memory safe.

Version 1 example:

```
void ec2_v1() {
  // CodaBot's code
  ...

  // Mallory's code
  ...
}
```

Version 2: Each user's code is copied into the same C program, in separate functions. Then, EC2 runs the single C program.

Q9.2 (3 points) Write two attacks that break isolation in Version 2.

There are many correct answers. Use at most 10 words per box; the staff solutions are 3 words each.

Attack 1:

Version 2 example:

```
void ec2_v2() {
  void codabot() {
    // CodaBot's code
  }

  void mallory() {
    // Mallory's code
  }
}
```

Attack 2:

Version 3: EC2 runs an operating system (OS) and runs each user's code in a separate process.

For Q9.3–Q9.5, select the technique that the OS uses to isolate each resource.

Q9.3 (1 point) CPU Registers: ◯ Time multiplexing ◯ Translation ◯ Emulation

Q9.4 (1 point) Main Memory: ◯ Time multiplexing ◯ Translation ◯ Emulation

Q9.5 (1 point) I/O devices: ◯ Time multiplexing ◯ Translation ◯ Emulation

Q9.6 (3 points) The OS translates a virtual address by looking up its page number in the process's page table and copying the offset (the bottom 12 bits) unchanged.

CodaBot's page table contains:

Virtual Page Number	Physical Page Number
`0x00004`	`0x60C25E6`
`0x00005`	`0x1F96EC7`
`0x00006`	`0xEC70DB7`
`0x00007`	`0x6997724`

CodaBot reads from virtual address `0x00006A3C`. What physical address does the OS access?

Q9.7 (2 points) Write an attack that breaks isolation in Version 3.

There are many correct answers. Use at most 10 words; the staff solution is 3 words.

(Question 9 continued...)

Post-Exam Activity: Spirit Animal

Bot is learning to summon spirit animals. What's yours?

Comment Box

Congratulations for making it to the end of the exam! Feel free to leave any final thoughts, comments, feedback, or doodles here: