Popa & Wagner
Spring 2016

CS 161 Computer Security

Final Exam

PRINT your name: ____________________________, ____________________________
(last) (first)

I am aware of the Berkeley Campus Code of Student Conduct and acknowledge that any academic misconduct on this exam will lead to a "F"-grade for the course and that the misconduct will be reported to the Center for Student Conduct.

SIGN your name: ____

PRINT your class account login: `cs161-`______ and SID: __________________________

Name of the person sitting to your left: ____________________ Name of the person sitting to your right: ____________________

You may consult three sheets of notes (each double-sided). You may not consult other notes, textbooks, etc. Calculators, computers, and other electronic devices are not permitted. Please write your answers in the spaces provided in the test. We will not grade anything on the back of an exam page unless we are clearly told on the front of the page to look there.

You have 180 minutes. There are 12 questions, of varying credit (200 points total). The questions are of varying difficulty, so avoid spending too long on any one question.

Do not turn this page until your instructor tells you to do so.

Problem 1 ***True or false*** **(21 points)**

Circle True or False. Do not justify your answer.

(a) True or False: DHCP spoofing can be prevented by using Ethernet instead of wireless.

(b) True or False: SYN Cookies help defend against distributed syn flooding attacks.

(c) True or False: Source port randomization is a helpful defense against Kaminsky blind spoofing.

(d) True or False: An off-path attacker is more powerful than an on-path attacker: anything an on-path attacker can do, so can an off-path attacker.

(e) True or False: An on-path attacker is more powerful than an off-path attacker: anything an off-path attacker can do, so can an on-path attacker.

(f) True or False: Internet censorship requires an in-path attacker (i.e., an on-path attacker that can both observe all packets and also drop any packets the censor wishes).

(g) True or False: If you use HTTPS but not DNSSEC, the confidentiality of data you send over HTTPS is protected against on-path attackers (ignoring implementation bugs and/or CA failures).

(h) True or False: If you use HTTP and DNSSEC, the confidentiality of data you send over HTTP is protected against on-path attackers (ignoring implementation bugs and/or CA failures).

(i) True or False: CBC mode encryption provides both confidentiality and integrity.

(j) True or False: CBC mode encryption provides confidentiality against chosen-plaintext attacks (IND-CPA security).

(k) True or False: Using a pseudorandom number generator, seeded by the current time of day (measured to microsecond precision), is a good way to generate an AES key.

(l) True or False: Cryptography is a reasonable defense if an adversary might be able to eavesdrop on network packets.

(m) True or False: Stack canaries are a good defense against heap-based buffer overflows.

(n) True or False: Fuzz-testing requires access to source code to find vulnerabilities.

(o) True or False: Prepared statements are a good defense against SQL injection.

(p) True or False: Setting the “secure” flag on a cookie (so it will only be sent over HTTPS) is a good defense against CSRF.

(q) True or False: TLS does not provide confidentiality if you use a TCP implementation whose TCP initial sequence numbers are predictable.

(r) True or False: Access control ensures that authorized users who have access to sensitive data won’t misuse it.

(s) True or False: Alice wants to communicate with Bob using Tor with 3 intermediaries. If the first 2 intermediaries are dishonest, they will be able to determine that Alice and Bob are communicating.

(t) True or False: When a client sends a message over the Tor network, Tor’s onion routing works because each intermediary encrypts the message they receive with the public key of the following intermediary, so no one else can decrypt the messages.

(u) True or False: The homomorphic properties of encryption allow servers to compute products or sums of encrypted data without having to decrypt it, as long as the data is encrypted with the right encryption algorithm.

Problem 2 ***Identify the vulnerability*** **(12 points)**

You are doing a security test of a website, `ShopSMart.com`. In each part below, based on the symptoms you observe, give the name of the type of vulnerability that is most likely responsible for those symptoms. Write just the name of the type/category of vulnerability (e.g., "buffer overrun"); you don't need to write a detailed description of the specific attack or vulnerability.

(a) You try to sign up for an account and type `Can't stop the signal` into the text field for entering your last name. When you click the button to submit the form, you see an error page that mentions a syntax error in the database query.

What type of vulnerability does this most likely indicate?

(b) You sign up under your real name, then place an order for 65539 bookmarks: you add the bookmark to your cart and enter a quantity of 65539. You place the order. A week later, a giant package arrives at your doorstep containing 65539 bookmarks. You check your credit card bill, and you see you've only been charged for 3 bookmarks.

What type of vulnerability does this most likely indicate?

(c) `ShopSMart.com` has a one-click buy feature: on the web page for each item, there is a button labelled "Buy instantly!". If you click that button while logged in, it instantly places an order for the item and charges your credit card on file, without requiring you to go through any other web pages (there's no separate checkout page, no confirmation page—the order is placed immediately). You view the HTML source for the page for the bookmark, and you see

```
<FORM ACTION="http://shopsmart.com/instantbuy&item=bookmark">
  <BUTTON TYPE="submit" VALUE="Buy instantly!">
</FORM>
```

You know this means that when the button is clicked, the browser will load the page `http://shopsmart.com/instantbuy&item=bookmark`. Looking through the rest of the page source, the page doesn't appear to contain any Javascript.

What type of vulnerability does this most likely indicate?

Problem 3 ***Cryptographic hashes*** **(8 points)**

You have a secure cryptographic hash function $H : \{0,1\}^{2n} \to \{0,1\}^n$ that takes $2n$-bit inputs and outputs a n-bit digest. You want to create a secure cryptographic hash function $F : \{0,1\}^{4n} \to \{0,1\}^n$ that takes $4n$-bit inputs and outputs a n-bit digest. How could you do this? Fill in the definition below, assuming w, x, y, z are each n-bit values:

$F(w, x, y, z) =$

(Don't explain or justify your answer.)

Problem 4 *Software security* (12 points)

(a) A *code injection* attack is one that involves attacker-chosen code being introduced into an application and then executed. Which of the following attacks are a form of code injection attack? Circle all that apply.

1. Buffer overflow
2. Port scanning
3. Kaminsky-style blind spoofing of DNS responses
4. CSRF
5. TCP SYN flooding
6. Clickjacking
7. XSS
8. Browser-in-browser attacks
9. None of the above

(b) In iOS, each app runs with the userid `mobile`. In Android, each app runs with a unique userid specific to that app. Which of these two better follows the principle of least privilege? Why?

Problem 5 ***Web security, XSS, and CSRF*** **(12 points)**

The Chrome browser contains a built-in filter which attempts to block certain attacks. When a user visits `http://website.com/foo.html?name=value`, Chrome looks at the HTML response and checks all Javascript scripts in the page to see if any of them are contained in (that is, a substring of) `value`; if so, Chrome refuses to execute that script.

For each part, circle Yes or No, then explain in a sentence why or why not. Assume the attacker doesn't try to do anything special to bypass the filter.

(a) Will this stop reflected XSS attacks?

Yes No

Explanation:

(b) Will this stop stored XSS attacks? (Stored XSS is also known as persistent XSS.)

Yes No

Explanation:

(c) Will this stop CSRF attacks?

Yes No

Explanation:

Problem 6 *Web security* (15 points)

(a) When users of `bank.com` are logged in, a request to `bank.com/session.js` returns a Javascript file containing

```
var session_id = "0123456789";
```

except that 0123456789 is replaced with the session ID for the user who made the request.

An attacker controls `evil.com` and would like to learn Alice's session ID for `bank.com`. How can the attacker do this? Explain why the same-origin policy doesn't stop this attack. (Assume the attacker can get Alice to visit `evil.com`.)

(b) When `bank.com` learns of this problem, they fix it by beginning all Javascript files with

```
if (!document.location.includes("http://bank.com")) {
    while (1) {}   // infinite loop
}
```

Explain why this doesn't work. How could an attacker defeat this defense?

(c) Propose better Javascript code to put at the start of all Javascript files.

Problem 7 ***TBC mode of operation*** **(18 points)**

Tracy proposes a new mode of operation for AES encryption, which we'll call the Tracy block chaining (TBC) mode. The message M is split into 16-byte blocks, $M = M_1||M_2||\cdots||M_k$. The ciphertext $C = C_0||C_1||\cdots||C_k$ is computed as follows: we pick a random 16-byte IV, let $C_0 =$ this IV, let $M_0 =$ this IV, and define $C_i = E_k(M_{i-1} \oplus M_i)$ for $i = 1, 2, \ldots, k$. Here $E_k(\cdot)$ denotes encryption of a single block using the AES block cipher and $D_k(\cdot)$ denotes decryption of a single block using the AES block cipher.

(a) Define the decryption algorithm. If the ciphertext is $C_0||C_1||\cdots||C_k$, give an equation specifying how the recipient can recover $M_1, \ldots, M_k$ as a function of the C_i's:

$M_i =$

(b) Is TBC mode IND-CPA secure? Why or why not? Circle "yes" or "no", then justify your answer.

YES or NO

Justification:

(c) Alice encrypts the message $M =$ `cs161 is so easy` using TBC. M is 16 bytes, and Alice uses no padding, so here $k = 1$. Let the resulting ciphertext be $C = C_0||C_1$.

Alice sends C to Bob.

Mallory is a man in the middle. Mallory wants to tamper with this ciphertext and replace it with some other ciphertext C', so that when Bob decrypts C', he will receive $M' =$ `cs161 is so hard`. Can Mallory do this? Circle "yes" or "no", then justify your answer: if you circled yes, describe the modified ciphertext C' that Mallory should send; if you circled no, explain why Mallory can't do it.

YES or NO

Justification:

Problem 8 ***TLS*** **(20 points)**

The TLS specification says that, during the handshake, the browser should send a random 256-bit number R_B and the server should send a random 256-bit number R_S. This question asks about the consequences if a browser or server fails to implement this correctly. Each part is independent. Assume the browser and server use RSA-based key exchange (not Diffie-Hellman).

(a) Suppose Alice downloads a buggy version of the Chrome browser that implements TLS incorrectly. Instead of picking R_B randomly, it increments a counter and sends it instead. Which of the following attacks are possible? Circle all that apply.

1. A man-in-the-middle can compromise Alice's confidentiality (e.g., learn the data she sends over TLS).
2. If Alice visits a HTTPS URL, a man-in-the-middle can successfully replay that HTTP request to the server a second time.
3. If Alice visits the same HTTPS URL twice, when Alice visits that URL the second time a man-in-the-middle can successfully replay the HTML page that was returned by the server on Alice's first visit.
4. A man-in-the-middle can learn the symmetric MAC keys that protect data sent over TLS connections initiated by Alice's browser.
5. None of the above

(b) Alice downloads a patch for her buggy version of Chrome, which fixes the previous problem but introduces another bug. Now, her client generates the pre-master secret by incrementing a counter. Which of the following attacks are possible? Circle all that apply.

1. A man-in-the-middle can compromise Alice's confidentiality (e.g., learn the data she sends over TLS).
2. If Alice visits a HTTPS URL, a man-in-the-middle can successfully replay that HTTP request to the server a second time.
3. If Alice visits the same HTTPS URL twice, when Alice visits that URL the second time a man-in-the-middle can successfully replay the HTML page that was returned by the server on Alice's first visit.
4. A man-in-the-middle can learn the symmetric MAC keys that protect data sent over TLS connections initiated by Alice's browser.
5. None of the above

(c) Alice downloads an updated version of Chrome that is finally correct. With her fixed browser, she visits `https://lazycodrz.com/`. That server's TLS implementation has a bug: instead of picking R_S randomly, it always sends all zeros. Which of the following attacks are possible? Circle all that apply.

1. A man-in-the-middle can compromise Alice's confidentiality (e.g., learn the data she sends over TLS).
2. If Alice visits a HTTPS URL, a man-in-the-middle can successfully replay that HTTP request to the server a second time.
3. If Alice visits the same HTTPS URL twice, when Alice visits that URL the second time a man-in-the-middle can successfully replay the HTML page that was returned by the server on Alice's first visit.
4. A man-in-the-middle can learn the symmetric MAC keys that protect data sent over TLS connections initiated by Alice's browser.
5. None of the above

Problem 9 ***Software security*** **(14 points)**

In older versions of iOS, the implementation of `malloc()` looked approximately like this:

```
struct malloc_header {
    size_t len;
}
void * malloc(size_t len) {
    struct malloc_header *hdr;
    size_t n;
    n = sizeof(malloc_header) + len;
                           // (*)
    hdr = system_alloc(n); // allocate n bytes of memory
    hdr->len = n;
    return hdr + 1;
}
```

You can assume the call to `system_alloc()` will never fail; it always succeeds in allocating `n` bytes of memory. You can also assume this runs on a 32-bit architecture and `size_t` is a 32-bit value.

(a) This implementation of `malloc()` could create a security problem for apps. Give an example of an invocation of `malloc()` that could be problematic. Make sure to show a specific value for `malloc()`'s parameter (it's OK to use hex or a mathematical expression). Explain briefly why this causes a problem.

(b) Fix this problem by adding some extra code at the line marked `(*)`. What C code should we add there?

Problem 10 ***Javascript meets crypto*** **(18 points)**

After doing Project 2, Neo decided to make a simple version of the Part 1 client that runs in the browser using only JavaScript. Neo calls his new service CryptoCatalog. It gives each registered user a single encrypted text file. The encrypted file is stored on the CryptoCatalog server (which is untrusted, like the storage server in Project 2). There is no sharing or revocation, and there is only one file per user.

A new user visits CryptoCatalog.com and creates an account (setting a username and password). After logging in, the CryptoCatalog JavaScript code prompts the user for their encryption passphrase (which is *different* from their login password). Using that passphrase, it generates two 128-bit symmetric keys, k_c and k_i, for confidentiality and integrity, by hashing the passphrase 10,000 times with SHA-256: H(H(...H(passphrase)...)) $= k_c \mid\mid k_i$. The JavaScript code gets the encrypted file from the server, and then authenticates and decrypts it using k_i and k_c. Whenever the user saves the file, the new contents are encrypted using AES-CBC under k_c and then the ciphertext is MAC'd using HMAC-SHA256 under k_i.

(a) Can evil.com read the plaintext file of a user of CryptoCatalog, if the user doesn't visit evil.com? Briefly, why or why not?

(b) Can evil.com read the plaintext file of a user of CryptoCatalog, if the user visits evil.com (but doesn't interact with evil.com at all)? Briefly, why or why not?

(c) Evil.com was able to steal a copy of CryptoCatalog's database of encrypted files. How could evil.com decrypt at least some of the files?

(d) Suppose evil.com manages to compromise the CryptoCatalog servers and gains complete control over them. How can evil.com break the confidentiality of all of CryptoCatalog's users' files?

(e) With regard to the previous question, why is the web version, CryptoCatalog, more vulnerable than your Project 2 client?

Problem 11 ***On wifi at a coffee shop*** **(24 points)**

You're sitting in a coffee shop enjoying a mocha latte and some relaxing computer security reading online at `http://awesome-security-stuff.com`. You're connected on the coffee shop's wifi network.

(a) Assuming you are only browsing `http://awesome-security-stuff.com`, who is potentially able to observe what articles you are reading? Circle all that apply.

1. Other coffee shop patrons
2. The manager of the store next door to the coffee shop who occasionally leeches off of the coffee shop's wifi
3. Your friend in a dorm a few miles away
4. The coffee shop's ISP
5. The website `awesome-security-stuff.com`
6. None of the above

(b) Name two technologies that could reduce the number of parties in part (a) that can observe your traffic. Do not give explanations, simply write down their names.

(c) If you use both the technologies you listed in part (b), who will still be able to know a complete list of all the articles you view?

1. Other coffee shop patrons
2. The manager of the store next door to the coffee shop who occasionally leeches off of the coffee shop's wifi
3. Your friend in a dorm a few miles away
4. The coffee shop's ISP
5. The website `awesome-security-stuff.com`
6. None of the above

(d) You notice that each article has a Facebook Like button icon, loaded from `facebook.com`, allowing you to indicate on Facebook that you enjoyed this article. If Facebook wanted to, could it track what articles you are visiting, if you don't click on the Like button? Circle yes or no, then explain why or why not.

YES NO

Explanation:

(e) The coffee shop only allows you to surf online for 30 minutes, but you usually like to spend more than that there.

After 30 minutes, you notice that visiting any URL in the browser results in a blocked page (indicating you can no longer browse the Internet). However, you notice you are able to successfully ping any valid domain name. Describe how you might prepare to surf for longer in the future.

Problem 12 ***DNS and DNSSEC*** **(26 points)**

Sarah is using a laptop on her home network to browse the Internet over HTTP. Consider an adversary with a smartphone and an Internet connection and the following capabilities:

A. No additional capabilities
B. A $10k GPU compute cluster
C. Can compromise Sarah's home router
D. Can compromise the primary nameserver for the zone
E. Can compromise the primary nameserver and offline signing key for the zone

(a) In the following table, **mark with an X** which attacks each adversary could successfully perform against Sarah if everyone uses **DNS without DNSSEC**:

	A	B	C	D
Spoof existence of records that actually don't exist in the zone				
Spoof values of records that exist in the zone				
Spoof non-existence of records that actually do exist in the zone				
Enumerate all names in the zone				

(b) Remember that when using DNSSEC, the zone is signed with a key that is not stored on the primary nameserver. Instead, the key is stored at an "offline" location. Also, NSEC records sign each adjacent pair of names in the zone. In the following table, **mark with an X** which attacks each adversary could successfully perform against **DNSSEC with NSEC records**:

	A	B	C	D	E
Spoof existence of records that actually don't exist in the zone					
Spoof values of records that exist in the zone					
Spoof non-existence of records that actually do exist in the zone					
Enumerate all names in the zone					

(capabilities repeated from the previous page)

A. No additional capabilities
B. A $10k GPU compute cluster
C. Can compromise Sarah's home router
D. Can compromise the primary nameserver for the zone
E. Can compromise the primary nameserver and offline signing key for the zone

(c) NSEC3 records were introduced to fix certain issues with NSEC records: each name in the zone is hashed, the hashes are sorted, and each NSEC3 record signs an adjacent pair of hash digests. In the following table, **mark with an X** which attacks each adversary could successfully perform against **DNSSEC with NSEC3 records**:

	A	B	C	D	E
Spoof existence of records that actually don't exist in the zone					
Spoof values of records that exist in the zone					
Spoof non-existence of records that actually do exist in the zone					
Enumerate all names in the zone					

(capabilities repeated from the previous page)

A. No additional capabilities
B. A \$10k GPU compute cluster
C. Can compromise Sarah's home router
D. Can compromise the primary nameserver for the zone
E. Can compromise the primary nameserver and offline signing key for the zone

(d) NSEC5 records are a further improvement. They work as follows.

When signing a zone, for each name n in the zone, compute $R(n) = H(\text{Sign}_K(n))$ where K is a secondary RSA private key and H is a secure cryptographic hash function. The secondary signing key K is separate from the main key used to sign the zone. Sort all the R values in the zone and sign each consecutive pair (R_i, R_{i+1}) using the primary signing key. The signed zone and the secondary signing key are stored on the primary nameserver.

When answering a query for a non-existent name n, the nameserver responds with $\text{Sign}_K(n)$ (generated on-the-fly) and the NSEC5 record that represents the gap containing $R(n)$.

In the following table, **mark with an X** which attacks each adversary could successfully perform against **DNSSEC with NSEC5 records**:

	A	B	C	D	E
Spoof existence of records that actually don't exist in the zone					
Spoof values of records that exist in the zone					
Spoof non-existence of records that actually do exist in the zone					
Enumerate all names in the zone					