Paxson Spring 2017 — 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 be reported to the Center for Student Conduct and may lead to a "F"-grade for the course.

SIGN your name: _____

PRINT your class account login: `cs161-`_____ and SID: _____

Your TA's name: _____

Number of exam of person to your left: _____

Number of exam of person to your right: _____

You may consult two 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.

You have 180 minutes. There are 9 questions, of varying credit (600 points total). The questions are of varying difficulty, so avoid spending too long on any one question. Parts of the exam will be graded automatically by scanning the **bubbles you fill in**, so please do your best to fill them in somewhat completely. Don't worry—if something goes wrong with the scanning, you'll have a chance to correct it during the regrade period.

If you have a question, raise your hand, and when an instructor motions to you, come to them to ask the question.

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

Question:	1	2	3	4	5	6	7	8	9	Total
Points:	80	74	48	72	64	54	96	56	56	600
Score:										

Problem 1 ***True/False*** **(80 points)**

For each of the following, FILL IN THE BUBBLE next to **True** if the statement is correct, or next to **False** if it is not. Each correct answer is worth 4 points. Incorrect answers are worth 0 points. Answers left blank are worth 1 point.

(a) Thanks to strong cryptography, a TLS connection to your bank is secure even if their web server's TCP/IP implementation has a buffer overflow vulnerability.

True **False**

(b) Thanks to strong cryptography, a TLS connection to your bank is secure even if your home router's TCP/IP implementation has a buffer overflow vulnerability.

True **False**

(c) To protect against Kaminsky blind spoofing attacks requires servers to implement a new version of the DNS protocol.

True **False**

(d) Using DNSSEC to resolve `example.com` guarantees authenticity and integrity on subsequent HTTP connections to `example.com`, but not confidentiality.

True **False**

(e) A properly configured firewall can prevent any DDoS attack from disrupting the ability of remote users to access your network.

True **False**

(f) Using a prepared statement to feed user input to an SQL query ensures that nothing the user enters will be treated as an SQL command.

True **False**

(g) VPN can enable you to safely connect to your company when using an untrusted public WiFi network.

True **False**

(h) When configuring a firewall, it's safer to use a whitelisting approach than it is to use a blacklisting approach.

True **False**

(i) A malicious website can execute a successful clickjacking attack even if the victim website uses HTTPS and the user's browser correctly implements the same origin policy.

True **False**

(j) A secure hash function will not produce any collisions.

True **False**

(k) Recall that secure-cookies are cookies which the browser will only transmit over HTTPS connection. Using HTTPS and secure-cookies is one way to prevent click-jacking attacks.

O True **O False**

(l) Suppose Alice has signed up for text-message two factor authentication on `bank.com`. If `bank.com` randomly generates a long number (e.g., a 16-digit number) for its 2FA codes and an attacker doesn't hijack Alice's phone number, then Alice's `bank.com` account is secure against phishing attacks.

O True **O False**

(m) For AES-CBC encryption, the IV does not need to be kept secret.

O True **O False**

(n) For AES-CTR encryption, the IV does not need to be kept secret.

O True **O False**

(o) If all messages are the same length and a message is never repeated, then it is secure to re-use the same one-time-pad for encryption.

O True **O False**

(p) To securely store user passwords, a server should use AES to encrypt each user's password and only store the ciphertexts in its database.

O True **O False**

(q) If *Website A* loads a website from another domain (*Website B*) inside of an iframe, the same origin policy prevents Javascript from *Website A* from accessing any of the other website's content in the iframe.

O True **O False**

(r) A certificate authority that issues a TLS certificate for `example.com` can also passively decrypt TLS traffic to `example.com`.

O True **O False**

(s) Consider a worm that spreads by each infected instance uniform randomly selecting a 32-bit IP address. We would expect the worm to initially spread exponentially fast, but then slow down its spread during the later part of its propagation.

O True **O False**

(t) The Slammer worm spread extra-fast because each infected instance of the worm kept increasing its scanning speed.

O True **O False**

Problem 2 ***Multiple Choice*** **(74 points)**

(a) (6 points) Suppose an attacker steals the private key of a website that uses TLS, and remains undetected. What can the attacker do using the private key? **Mark ALL that apply.**

- ○ Decrypt recorded past TLS sessions that used RSA key exchange.
- ○ Decrypt recorded past TLS sessions that used Diffie–Hellman key exchange.
- ○ Successfully perform a MITM attack on future TLS sessions.
- ○ None of these.

(b) (6 points) DNSSEC provides which of the following security properties for DNS responses? **Mark ALL that apply.**

- ○ Confidentiality
- ○ Authentication
- ○ Integrity
- ○ Availability
- ○ None of these

(c) (8 points) "Mixing program control and user data" is a class of vulnerabilities where a program/application accidentally treats user input as code and executes it. Which of the following attacks exploit this class of vulnerabilities? **Mark ALL that apply.**

- ○ Buffer overflows
- ○ Stored XSS
- ○ CSRF
- ○ Reflected XSS
- ○ SQL Injection
- ○ Clickjacking
- ○ None of these

(d) (6 points) To verify that she is visiting the correct website, Alice is told to make sure to check that the URL in the browser's address bar is the URL she actually wants to visit. Which of the following statements are true? **Mark ALL of the following statements that apply.**

- ○ Of relevance for this situation is the principle of Least Privilege
- ○ This will help Alice defend herself against some DNS spoofing attacks
- ○ Of relevance for this situation is the principle of Consider Human Factors
- ○ This will help Alice defend herself against some phishing attacks
- ○ This will help Alice defend herself against CSRF attacks
- ○ None of these

(e) (6 points) Alice is trying to visit `maps.google.com` and neither her machine nor her local resolver have any entries in their DNS caches. In the following, assume that `google.com` subdomains use HTTPS and are on the predefined HSTS (HTTP Strict Transport Security) list in Alice's browser. You do not need to worry about attacks on availability, nor attacks based on stealing private keys, malware infections, or obtaining a fraudulent `google.com` certificate. **Mark ALL that apply.**

- O For DNSSEC to work securely, the root and .com zones will need to sign their NS and glue/additional records.
- O For DNSSEC to work securely, the root and .com zones will need to encrypt their NS and glue/additional records.
- O Because google.com subdomains are on the predefined HSTS list, Alice's visit to `maps.google.com` is secure against DNS spoofing attacks.
- O Because google.com subdomains are on the predefined HSTS list, Alice's visit to `maps.google.com` is secure against MITM attacks.
- O Because google.com subdomains are on the predefined HSTS list, Alice's visit to `maps.google.com` is secure against *ssl-strip* attacks.
- O None of these apply.

(f) (8 points) Gandalf is surfing the web and visits the URL `http://gondor.berkeley.edu`. Assume that neither his machine nor his local resolver have any entries in their DNS caches, and that `berkeley.edu` is the authoritative name server for all `berkeley.edu` subdomains. Assuming global deployment and use of DNSSEC, and that DNS zones use Key Signing Keys (KSKs) and Zone Signing Keys (ZSKs), which of the following are True? **Mark ALL that apply.**

- O Gandalf's machine can use the `berkeley.edu` KSK to encrypt the query it sends to the `berkeley.edu` DNS server.
- O `berkeley.edu`'s ZSK will be signed by the root's KSK.
- O `berkeley.edu`'s ZSK will be signed by `berkeley.edu`'s KSK.
- O Gandalf's machine will receive a final A record for `gondor.berkeley.edu` that is signed with `berkeley.edu`'s ZSK.
- O Gandalf's machine will receive a final A record for `gondor.berkeley.edu` that is encrypted with a public key that Gandalf provides in his DNS query.
- O The final A record for `gondor.berkeley.edu` will have object security.
- O If zones correctly implement DNSSEC, then Gandalf is secure against a MITM attacker who attempts to modify content retrieved from the `gondor.berkeley.edu` web site.

(g) (6 points) A border firewall's primary purpose is (**Mark ONE**):

- ○ Block incoming VPN connections.
- ○ Prevent CSRF attacks.
- ○ Detect buffer overflows.
- ○ Prevent XSS attacks.
- ○ Prevent a network intruder inside the network from spreading internally.
- ○ None of these.

(h) (8 points) Which of the following attacks might allow an attacker to steal one of your browser cookies (**Mark ALL that apply**):

- ○ Reflected XSS
- ○ Stored XSS
- ○ Clickjacking
- ○ None of these
- ○ Buffer overflow
- ○ TLS downgrade
- ○ DDoS

(i) (6 points) Alice and Bob want to communicate over an insecure channel using one of the following schemes, where M is the message in plaintext. Which scheme should they use in order to avoid padding oracle attacks? Assume that (1) all of the algorithms are secure, and (2) MAC and Sign do not leak anything about M. **Mark ALL that apply.**

- ○ $\text{Enc}(M), \text{MAC}(M)$
- ○ $\text{Enc}(M \mid\mid \text{MAC}(M))$
- ○ None of these
- ○ $\text{Enc}(M), \text{MAC}(\text{Enc}(M))$
- ○ $\text{Enc}(M), \text{Sign}(M)$

(j) (6 points) Let S be a publicly available trusted service that knows the public keys of all users. Alice communicates with S to obtain Bob's public key using the following protocol:

1. $A \longrightarrow S \ : \ A, B$
2. $S \longrightarrow A \ : \ [K_B, B]_{K_S^{-1}}$

In step 1, Alice sends along her identity A and asks S for Bob's public key. In step 2, S responds by returning Bob's public key K_B along with his identity B, and signs the message.

Which of the following attacks is this protocol vulnerable to? **Mark ALL that apply.**

- O Mallory can tamper with S's response so as to substitute her own public key K_M instead of K_B.
- O Since S's response is not encrypted, Mallory can use K_B to decrypt any messages Alice sends to Bob in the future.
- O Mallory can tamper with S's response so as to substitute an older key K'_B that Bob might have revoked.
- O None of these.

(k) (8 points) For the same situation as in the previous question, which of the following modifications to step 2 would defend against the attacks that the protocol in that question is vulnerable to? **Mark ALL that apply.**

- O $S \longrightarrow A : [K_B, A, B]_{K_S^{-1}}$
- O $S \longrightarrow A : [K_B, B, T]_{K_S^{-1}}$ where T is a timestamp
- O $S \longrightarrow A : [K_B, T]_{K_S^{-1}}$ where T is a timestamp
- O $S \longrightarrow A : [K_B, B, A, N]_{K_S^{-1}}$, where N is a nonce randomly selected by S
- O $S \longrightarrow A : [K_B, B, N]_{K_S^{-1}}$, where N is a nonce randomly selected by S
- O None of these

Problem 3 ***Bypassing ASLR*** **(48 points)**

Mallory is trying to perform a return-to-libc attack on a simple stack buffer overflow vulnerability. She wants to overwrite the return address of the vulnerable function with the address of the `system` function, and pass it an arbitrary command argument. But the system she wants to attack has ASLR enabled, so `&system` (the address of `system`) is different every time.

Wanting to explore this further, Mallory writes the simple program:

```
#include <stdio.h>
void main() {
    printf("system is at 0x%x\n", &system);
}
```

She runs this five times, with ASLR enabled, and gets the following output:

```
system is at 0xbf9d7f14
system is at 0xbf9d7f99
system is at 0xbf9d7f88
system is at 0xbf9d7f36
system is at 0xbf9d7f08
```

(a) (16 points) She shouts "Eureka! It won't work every time, but I can easily break this now!". What did Mallory learn? How can she use it to successfully exploit the buffer overflow with a return-to-libc attack?

(b) (8 points) What is the probability that Mallory will succeed if she has 1 chance to perform her return-to-libc attack?

(c) (24 points) Suppose Mallory is able to control the input (i.e. `argv[1]`) to the following silly backup program, written by programmers from Junior University (assume headers necessary for this code to compile have been included):

```
// Protect our data by making 2 copies!
void double_copy(char *data) {
    char buf1[16];
    char buf2[16];

    strcpy(buf2, data);
    strcpy(buf1, data);
}

int main(int argc, char *argv[]) {
    // recall: argv[0] is the name of the program
    if (argc != 2)
        return -1;

    double_copy(argv[1]);
}
```

Give an input that will cause "`sudo rm -rf /`" to be run on the victim machine with probability equal to what you answered in the previous part.

Use the following assumptions about the victim system:

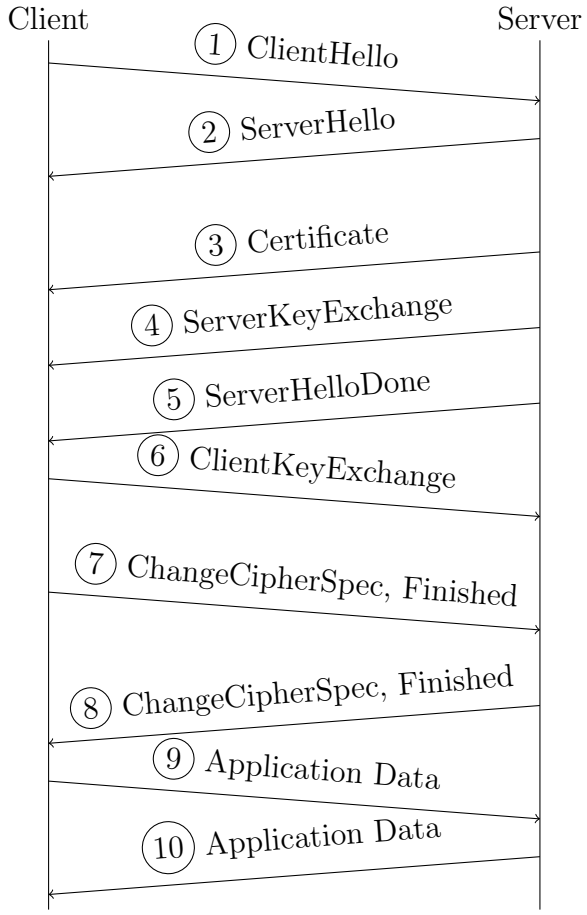
1. It is an IA-32 platform with 4-byte words (recall it's also little endian).
2. The stack is aligned at word granularity.
3. Local variables of each function are placed on the stack in the order they appear in the source code.
4. ASLR is enabled for the stack segment.
5. `argv[1] == 0x07070707` will always evaluate to true.

Hint: `#` is the shell comment character.

You can use `\x**` (where the `*`s are replaced by hex digits) to represent a character in hexadecimal form. **Fill in** the answer below:

Problem 4 ***Attacks on TLS*** **(72 points)**

Recall the TLS protocol, depicted in the figure below. We use the following notation: $\{M\}_K$ denotes a message M encrypted using the key K. $[M]_K^{-1}$ denotes a message M along with a signature over M using the key K^{-1}.



1. Client sends a 256-bit random number R_b and supported cipher suites C
2. Server sends a 256-bit random number R_s and chosen cipher suite C_{ser}
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 $\text{MAC}(dialog, I_b)$
8. Server sends $\text{MAC}(dialog, I_s)$
9. Client data takes the form $\{M_1, \text{MAC}(M_1, I_b)\}_{C_b}$
10. Server data takes the form $\{M_2, \text{MAC}(M_2, I_s)\}_{C_s}$

Figure 1: TLS 1.2 Key Exchange

(a) (24 points) Suppose the client and server use RSA to exchange the premaster secret. Mallory intercepts the ClientKeyExchange message and replaces PS with a fake value PS'. Assume that Mallory can modify the messages *after* ClientKeyExchange as well, if required. Which of the following are true? **Mark ALL that apply.**

- O Mallory will be able to decrypt the application data sent by the client to the server.
- O Mallory will be able to decrypt the application data sent by the server to the client.
- O The server will detect the tampering when it receives ClientKeyExchange.
- O Mallory can avoid detection until the server receives Finished from the client, at which point she'll be detected.
- O Mallory can avoid detection until the client receives Finished from the server, at which point she'll be detected.
- O None of these

(b) Now suppose the client and server use Diffie-Hellman for exchanging the premaster secret. Mallory wants to decrypt the data sent by the server to the client by *downgrading* the cipher suites. She doesn't care about the data sent by the client to the server. If the server always picks the strongest cipher suite and parameters available, specify whether Mallory's attack will succeed in the following scenarios **(Yes/No)**.
If yes, then list the handshake messages Mallory will need to *necessarily* modify. If not, explain why.
Assume that unless specified, all cryptographic algorithms supported by the client and server are secure.

i. (12 points) Suppose the client and server support 3DES in addition to AES. Mallory is aware of an attack on 3DES that allows her to learn any message encrypted using it. She therefore wishes to force the client and server to use 3DES instead of AES as the encryption algorithm.

ii. (12 points) Suppose the client and server support a weak variant of Diffie-Hellman (DH_{weak}). Mallory is aware of an attack on DH_{weak} that allows her to learn the exchanged secret. She therefore wishes to force the client and server to use DH_{weak} instead of standard Diffie-Hellman.

(c) (24 points) Recall that ClientHello contains a nonce R_b, along with C, the cipher suites supported by the client. ServerHello contains a nonce R_s along with C_{ser}, the cipher suite chosen by the server. Which of the following modifications to the TLS protocol would prevent Mallory from conducting *any* downgrade attacks on the cipher suites? **Mark ALL that apply.**

- ○ ServerKeyExchange includes $[R_b]_{K^{-1}_{\text{server}}}$, $[C]_{K^{-1}_{\text{server}}}$
- ○ ServerKeyExchange includes $[C]_{K^{-1}_{\text{server}}}$, $[C_{\text{ser}}]_{K^{-1}_{\text{server}}}$
- ○ ServerKeyExchange includes $[R_b, C]_{K^{-1}_{\text{server}}}$
- ○ ServerKeyExchange includes $[C \,||\, C_{\text{ser}}]_{K^{-1}_{\text{server}}}$
- ○ ServerKeyExchange includes $[C]_{K^{-1}_{\text{server}}}$
- ○ ServerKeyExchange includes $[R_b \,||\, C \,||\, C_{\text{ser}}]_{K^{-1}_{\text{server}}}$
- ○ ServerKeyExchange includes $[C_{\text{ser}}]_{K^{-1}_{\text{server}}}$
- ○ None of these

Problem 5 ***Software vulnerability*** **(64 points)**

Here is a fragment of Python source code for a fictitious email-based spell-checker service:

```
def process_incoming_email(msg):
    return_addr = msg.get("From")
    search_term = msg.get("Subject")
    status = os.system("fgrep " + search_term + " /usr/share/dict/words")
    if status == 0: # exit code 0 means success
        response = "The word " + search_term + " is spelled right!"
    else:
        response = "Sorry, " + search_term + " is not a word."
    send_response(return_addr, response)
```

In this service, users submit a word to check as the `Subject:` header field of an email. For example:

```
To: spellcheck@example.com
From: user@berkeley.edu
Subject: phenommenon
```

The `process_incoming_email` function is responsible for checking the spelling of the word, generating an appropriate response message, and sending the response back to the original sender via email. The function works by extracting a search term from the `Subject` header field, then using the `fgrep` command to search for the term in `/usr/share/dict/words`, a file containing a list of English words. The `fgrep` command searches a file for a fixed text pattern; its syntax is '`fgrep` *`pattern filename`*'.

`os.system` is a Python function that accepts a single string and executes the string using the command shell. It works the same as the C `system()` library routine that we discussed in lecture.

(a) (16 points) The `process_incoming_email` function has a vulnerability. What email `Subject:` could you send that would cause the server to pause for 10 seconds before replying?

`Subject:`

(b) (16 points) What email `Subject:` could you send that would tell you whether or not there is a user called `dbadmin` on the spellcheck server? The list of users is stored in the file `/etc/passwd`.

`Subject:`

(c) (32 points) State **one** way that you could fix the vulnerability? (If you name more than one, we will only grade the first.)

Problem 6 ***Coffee Shop Worries*** **(54 points)**

Alice and Bob just arrived at Brewed Awakening, the local coffee shop. Eve is already there, enjoying a cup of tea.

(a) (6 points) Alice wants to connect to Brewed Awakening's WiFi network. Under which protocols would her connections be safe from sniffing attacks by other coffee shop visitors, such as Eve? **Mark all that apply.**

- O WEP
- O WPA2 - Enterprise mode
- O WPA2 - Personal mode
- O None of these

(b) (24 points) Turns out that Brewed Awakening's network has no encryption. Alice warns Bob that its not safe to use this connection, but Bob disagrees. Bob connects to the WiFi, and tests that he has Internet connectivity by going to `https://kewlsocialnet.com`. It loads without issues. Bob says the Alice: "See, no problem! That access was totally safe!"

If Bob is correct and the access to `kewlsocialnet.com` was safe, explain why he is correct. If he is not correct, provide a network attack against Bob.

Answer:

(c) (24 points) Now that he has tested his WiFi access, Bob then tells Alice: "I want to buy that last muffin at the counter. Let me check if I have enough money in my bank account." Eve hears this and panics! She wants the last muffin too but is waiting for her friend Mallory to bring enough cash to buy it. She is now determined to somehow stop Bob from buying that last muffin by preventing him from checking his bank account. Through the corner of her eye, Eve sees Bob start to type `https://bank.com` in his browser URL bar ...

Describe two network attacks Eve can do to prevent Bob from checking his bank account. For each attack, describe clearly in one or two sentences how Eve performs the attack.

Attack #1:

Attack #2:

Problem 7 ***The Great Cannon*** **(96 points)**

In 2015, Github experienced a DoS attack orchestrated by China using the so-called "Great Cannon" (GC). It worked as follows. (Some details of the attack have been simplified or modified for this problem.)

Many websites include a fetch for a script for analytics from Baidu, a large Internet service in China somewhat similar to Google. The script would be retrieved via `http://hm.baidu.com/h.js`. The GC operated in-path at the border between China and the rest of the Internet. Upon seeing a request for this script, the GC would prevent the original HTTP request from being forwarded, and would instead return a different script, which instructed clients to repeatedly load `http://github.com/cn-nytimes`.

You can assume that Baidu served its traffic using servers in China; Github did so from servers in the USA; and websites using the analytics script were hosted all over the world.

(a) (6 points) For which of the following layers would the GC need to **guess** or **infer** header values it could not directly determine in order to carry out the attack? **Mark ALL that apply.**

- O Physical
- O Transport
- O Link
- O Application
- O Network
- O None of these

(b) (6 points) Which layer was this attack meant to particularly stress regarding Github's servers? **Mark the BEST choice.**

- O Physical
- O Transport
- O Link
- O Application
- O Network
- O None of these

(c) (4 points) Whose traffic contributed to the DDOS attack? **Mark the BEST choice.**

- O Web browsers inside China
- O Both of these
- O Web browsers outside China
- O Neither of these

(d) (4 points) Which packets would the implementers of this attack need to inspect? **Mark the BEST choice.**

- O Packets going into China
- O Both of these
- O Packets going out of China
- O Neither of these

(e) (12 points) Why doesn't the Same Origin Policy prevent this attack? (Limit your answer to no more than 2 sentences.)

(f) (12 points) For this and the next question, suppose that after the attack began, Github installed a NIPS to deal with this particular attack. Assume the NIPS is deployed on the Ethernet link connecting the `github.com` server to the public Internet. What kind of detection is MOST LIKELY to be effective under the circumstances? **Mark the BEST choice** and provide a **short explanation**.

- O Signature-based
- O Behavioral
- O Anomaly-based
- O Honeypots
- O Specification-based
- O Vulnerability scanning

Explanation:

(g) (12 points) Suppose that the attack caused Github to receive 50 times as many bogus requests as legitimate requests, and that Github will consider a defense successful if it reduces the volume of flooding requests by at least a factor of 50, so the flooding is no larger than the volume of legitimate requests. Suppose further that Github found that their NIPS had a precision of 0.999 and a recall of 0.99 when detecting this attack. To what degree would this represent a successful defense? **Mark ONE of the following** and **BRIEFLY explain** ($\leq$ 2 sentences) your answer.

- O Yes, the NIPS provided a successful defense.
- O No, the NIPS did not provide a successful defense.
- O Additional information is needed to tell whether the NIPS provided a successful defense.
- O Such a combination of precision and recall values is not possible under these circumstances.

Explanation:

(h) (8 points) This attack occurred for sets of HTTP requests. Which of the following changes would have prevented the attack? Consider each choice in isolation (i.e., assess whether it prevents the attack assuming none of the other choices are in effect). **Mark ALL that apply.** For each choice, assume that the content that the site serves remains the same.

- O Every website that uses Baidu's analytics switches to serve its content using HTTPS instead of HTTP.
- O Baidu's analytics server redirects any incoming HTTP connection to a corresponding HTTPS URL.
- O None of these.
- O Baidu switches its analytics server over to only be accessible using an HTTPS URL.
- O Github's server redirects any incoming HTTP connection to a corresponding HTTPS URL.
- O Github switches its server over to only be accessible using HTTPS.

(i) (8 points) Which of the following techniques could Github have used to make the attack ineffective? **Mark ALL that apply.**

- O Blacklist any packets from Chinese IP addresses
- O Use SYN cookies for all new connections
- O None of these
- O Move the affected Github server to a new IP address
- O Remove all use of Baidu analytics from Github web pages

(j) The remainder of this problem concerns a Web security feature called Subresource Integrity (SRI). It works by adding an attribute to the `script` tag for externally loaded scripts:

`<script src="http://example.com/script.js" integrity="[`*CRYPTOGOOP*`]">`

Browsers then validate the integrity of the script retrieved from the given `src=` location.

i. (8 points) What should *CRYPTOGOOP* contain for it to achieve its goal of assuring integrity, while minimizing the effort required by web developers to adopt it? **Mark the BEST answer.**

- O An encryption of the script being loaded
- O A digital signature of the script being loaded
- O A MAC of the script being loaded
- O A hash of the URL of the script
- O A hash of the script being loaded

ii. (8 points) Suppose every website with Baidu's analytics starts using SRI. Given GC's capabilities, could it still redirect some Baidu analytics traffic to Github?

- O Yes
- O No

Explanation (1 sentence):

iii. (8 points) Name **ONE** drawback to a website's owner from deploying SRI. (If you name more than one, we will only grade the first.)

Drawback:

Problem 8 ***Computing on encrypted data*** **(56 points)**

Recall the El Gamal scheme: The El Gamal public key is (p, g, h), x is the private key, and $h = g^x \bmod p$. The encryption of a message M is $\mathsf{Enc}(M) = (g^r \bmod p, M \times h^r \bmod p)$, for a random r.

(a) (24 points) Say function F can be computed over El Gamal ciphertexts. This means given only $C_1 = \mathsf{Enc}(M_1) = (s_1, t_1)$, $C_2 = \mathsf{Enc}(M_2) = (s_2, t_2)$, and the El Gamal public key, anyone can compute a ciphertext $C_3 = \mathsf{Enc}(F(M_1, M_2))$

Which of the following arithmetic operations can be computed over El Gamal ciphertexts? **Mark ALL that apply.**

- O Modular Addition: $F(M_1, M_2) = M_1 + M_2 \bmod p$
- O Modular Exponentiation: $F(M_1, M_2) = {M_1}^{M_2} \bmod p$
- O Modular Multiplication: $F(M_1, M_2) = M_1 \times M_2 \bmod p$
- O None of the above

For each arithmetic operation you select, write down the equation that someone can use to compute C_3 using the components of C_1, C_2 (i.e., s_1, t_1, s_2, t_2), and the public key. Or if none of the computations is possible, explain why not.

Equation(s) or Explanation:

(b) (24 points) Suppose Alice sends Bob a message M_0 after encrypting it with Bob's El Gamal public key. Let $C_0 = (s_0, t_0)$ be the corresponding ciphertext. Mallory wants to learn the message M_0. Bob agrees to decrypt a single ciphertext $C_1 = (s_1, t_1)$ of Mallory's choice, as long as $C_1 \neq C_0$. Explain how Mallory can take advantage of Bob's offer in order to learn M_0.
Hint: Mallory observes that she can manipulate C_0 in a way that allows her to obtain another valid ciphertext that also decrypts to M_0.

(c) (8 points) Which of the following best describes the attack in the previous question?

- O Ciphertext-only attack
- O Chosen plaintext attack
- O Known plaintext attack
- O Chosen ciphertext attack

Problem 9 ***Password Cracking*** **(56 points)**

Mallory has an account on `www.lamesec.com`, a hot new social networking site. So does her rival, Alice. Mallory desperately wants to break into Alice's account (username "alice") to read Alice's private messages. `www.lamesec.com` specifies that account passwords must:

1. Be no longer than 7 characters.
2. These characters must be either lowercase letters or one of the following symbols: `+`, `-`, `_`, `$`, `*`, `!`.
3. Should be randomly chosen given these constraints.

One of Alice's many traits that Mallory finds annoying is that Alice will always comply with rules like these.

Mallory has observed that if she tries to use her browser to guess a possible password for Alice's account, she receives a reply from the `www.lamesec.com` web server that looks like:

No Dice

Your attempt to authenticate as **alice** *failed*.

Apache/2.4.25 (Fedora) Server at www.lamesec.com Port 80

Job status: started 10:39:32 May 12 2017, completed 10:39:32 May 12 2017
Total # instructions executed 21333427
Total memory required 1522667 bytes
Total disk storage 0 bytes
Have a nice day

Mallory in addition notices that the site uses a framework for which the below Python code at the server validates authentication attempts:

```
def CheckPassword(account, submitted_password):
  if len(submitted_password) != len(account.password):
    return False
  for i in range(len(submitted_password)):
    if submitted_password[i] != account.password[i]:
      return False
  return True
```

Assume that the code is compiled without any optimization, and that all comparison operators take a single instruction to execute. Also assume that `len(x)` always takes

the same number of instructions to execute regardless of how long `x` is, and access to `account.password` likewise takes constant time.

Knowing that this is the specific code that is used, Mallory analyzes the information returned for a number of failed authentication attempts she makes to *her own* account. In doing so, she is free to repeatedly change her password to new values if she wishes. After analyzing this information, Mallory feels ready to try to attempt to infer information about Alice's password.

(a) (12 points) How many authentication attempts will suffice for Mallory to determine the *length* of Alice's password? Choose the **MINIMUM** such number of attempts that *guarantees* success for Mallory:

- ○ 1 attempt
- ○ 10^3 attempts
- ○ 10^6 attempts
- ○ 10^9 attempts
- ○ 10^{12} attempts
- ○ Mallory can do this but will need more than 10^{12} attempts
- ○ Mallory cannot do this

(b) (16 points) How many authentication attempts will suffice for Mallory to determine the *exact value* of Alice's password? Choose the **MINIMUM** such number of attempts that *guarantees* success for Mallory:

- ○ 1 attempt
- ○ 10^3 attempts
- ○ 10^6 attempts
- ○ 10^9 attempts
- ○ 10^{12} attempts
- ○ Mallory can do this but will need more than 10^{12} attempts
- ○ Mallory cannot do this

(c) (12 points) Suppose `www.lamesec.com` instead uses the following code to validate authentication attempts:

```
def CheckPassword(account, submitted_password):
  if len(submitted_password) != len(account.password):
    return False
  num_correct = 0
  num_incorrect = 0
  for i in range(len(submitted_password)):
    if submitted_password[i] == account.password[i]:
      num_correct = num_correct + 1
    if submitted_password[i] != account.password[i]:
      num_incorrect = num_incorrect + 1
  return num_incorrect == 0
```

Given this change, now how many authentication attempts will suffice for Mallory to determine the *length* of Alice's password? Choose the **MINIMUM** such number of attempts that *guarantees* success for Mallory:

- O 1 attempt
- O 10^3 attempts
- O 10^6 attempts
- O 10^9 attempts
- O 10^{12} attempts
- O Mallory can do this but will need more than 10^{12} attempts
- O Mallory cannot do this

(d) (16 points) Continuing with the new version of `CheckPassword`, now how many authentication attempts will suffice for Mallory to determine the *exact value* of Alice's password? Choose the **MINIMUM** such number of attempts that *guarantees* success for Mallory:

- O 1 attempt
- O 10^3 attempts
- O 10^6 attempts
- O 10^9 attempts
- O 10^{12} attempts
- O Mallory can do this but will need more than 10^{12} attempts
- O Mallory cannot do this