Wagner Spring 2014 — CS 161 Computer Security — Midterm 2

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

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.

SIGN your name: _____

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

Your TA's name: _____

Your section time: _____

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

You may consult one sheet of paper (double-sided) of notes. 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 50 minutes. There are 8 questions, of varying credit (100 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.

Question:	1	2	3	4	5	6	7	8	Total
Points:	16	16	10	12	17	7	12	10	100
Score:									

Problem 1 ***True or False*** **(16 points)**

Circle True or False. Do not justify your answer.

(a) TRUE or FALSE: Firewalls are commonly deployed because they never affect functionality.

(b) TRUE or FALSE: No intrusion detection system is capable of detecting novel attacks.

(c) TRUE or FALSE: Full TCP reassembly is sufficient to implement a network intrusion detection systems (NIDS) that stops all evasion attacks.

(d) TRUE or FALSE: A host-based intrusion detection system (HIDS) is harder to evade than a network intrusion detection system (NIDS) because a HIDS has access to application-layer semantics.

(e) TRUE or FALSE: The false positive and false negative rates of a given intrusion detection system provide enough info to classify it as "good" or "bad" at detecting threats.

(f) TRUE or FALSE: The Kaminsky attack is an on-path attack; off-path attackers cannot mount a Kaminsky attack.

(g) TRUE or FALSE: On-path attackers can successfully eavesdrop on the data sent over a TCP connection.

(h) TRUE or FALSE: Off-path attackers can successfully eavesdrop on the data sent over a TCP connection.

Problem 2 ***More True or False*** **(16 points)**

Circle True or False. Do not justify your answer.

(a) True or False: On-path attackers can successfully tamper with (i.e., modify) the data sent from Alice's web browser to `http://www.cnn.com/`.

(b) True or False: Off-path attackers can successfully tamper with (i.e., modify) the data sent from Alice's web browser to `http://www.cnn.com/`.

(c) True or False: DHCP spoofing only affects the integrity of DNS lookups and has no practical effect on web security.

(d) True or False: Diffie-Hellman is secure against passive eavesdroppers who cannot modify packets or send forged packets.

(e) True or False: Diffie-Hellman is secure against man-in-the-middle attacks.

(f) True or False: Cryptographic hash functions are required to be one-way and collision-resistant.

(g) True or False: The IV in CTR mode (counter mode) must be kept secret.

(h) Alice and Bob share a symmetric key k. Alice sends Bob a message stating, "I owe you \$100", and also sends a message authentication code (MAC) on this message computed using the key k.

True or False: Assuming the MAC algorithm is secure, Bob can now go to his bank and prove to the bank teller that Alice does indeed owe him \$100 by giving his key k to the teller.

Problem 3 ***Vulnerability mitigation*** **(10 points)**

Recall that the version of VSFTPd we used in Project 2 had a widely known vulnerability ("smile at login"). You're the sysadmin for a large company, and you've learned that there is an internal server inside your company's network running this version of VSFTPd. Unfortunately, you can't modify that server.

(a) Describe one way you could use a stateful packet filter to prevent an outside attacker from exploiting the VSFTPd vulnerability.

(Hint: FTP uses TCP and is unencrypted.)

(b) Suppose that we have deployed a stateful packet filter (as in part (a)) at the border of our network, where it connects to the rest of the Internet. Now imagine that an employee's laptop has been infected with malware. Could the malware exploit the VSFTPd vulnerability? Why or why not?

Problem 4 ***Password hashing*** **(12 points)**

Joe runs a large website that allows users to log in and share images. When a new user sets up their account, the website hashes their password with SHA256 and stores the hash in a database. When a user logs in, the website hashes the supplied password with SHA256 and compares it to the stored hash.

Joe figures that with this scheme, if anyone hacks into your database they will only see hashes and won't learn your users' passwords. Out of curiosity, Joe does a Google search on several hashes in the database and is alarmed to find that, for a few of them, the Google search results reveal the corresponding password. He comes to you for help.

(a) What mistake did Joe make in how he stored passwords?

(b) What is the consequence of this mistake? In other words, what is the risk that it introduces and how many of Joe's users could be affected? Does it affect only users whose password hashes are available in Google search, or does it go beyond that?

(c) How should Joe store passwords? More specifically, if a user's password is w, what should Joe store in the database record for that user?

Problem 5 ***Deja vu ... all over again*** **(17 points)**

The C code below should look very similar to something you've seen in Project 2. You have the same goal—you want to write an exploit that spawns a shell. For your reference:

shellcode (as seen in gdb): `0x895e1feb, 0xc0310876, 0x89074688, 0x0bb00c46, ...`
ASCII: A-Z: `0x41-0x90`, a-z: `0x61-0x7a`, newline: `0x0a`, escape: `0x1b`, tab: `0x09`

```
1:  void deja_vu()
2:  {
3:    char door[8];
4:    gets(door);
5:  }
6:  int main()
7:  {
8:    deja_vu();
9:    return 0;
10: }
```

(a) In gdb, what line should you set a breakpoint at to check if your exploit succeeded, without any additional `step` calls?

Line number:

(b) The relevant gdb output below is from **before** you feed in your malicious input.

```
(gdb) x/8x door
0xbffff9b8: 0xbffffa7c  0xb7e5f225  0xb7fed270  0x00000000
0xbffff9c8: 0xbffff9d8  0x0804842a  0x08048440  0x00000000
(gdb) i f 0
Stack level 0, frame at 0xbffff9d0:
 eip = 0x8048412 in deja_vu (dejavu.c:7); saved eip 0x804842a
 Locals at 0xbffff9c8, Previous frame's sp is 0xbffff9d0
 Saved registers:
  ebp at 0xbffff9c8, eip at 0xbffff9cc
```

What should you see if you run `x/8x door` **after** the malicious input has been fed in but before the shellcode executes? Assume that the exploit was successful, and the bare minimum was changed in memory. Your solution should have at least one 4-byte chunk of shellcode.

```
0xbffff9b8:
0xbffff9c8:
```

(c) Recall that in Project 2, your absolute memory addresses were randomized depending on your login. Explain how this kind of randomization could make exploiting the above vulnerability more complicated, if you got an unlucky choice of memory addresses. Provide an example of a specific address for `door` which would cause this problem. Your example address must be between `0xbfff0000` and `0xbffffffc`.

Address of door:

Explanation:

Problem 6 ***A special encryption algorithm*** **(7 points)**

A security company has determined that, even using a bunch of fast computers, brute-force attacks can break at most a 50-bit key in a reasonable amount of time. So, they decide to design an encryption algorithm that will encrypt 64-bit messages under a 64-bit key. They already have a good block cipher E that can encrypt a 32-bit message under a 32-bit key and that is resilient against all attacks except brute-force attacks. So, for the 64-bit encryption algorithm, they simply split the message M in two parts M_1 and M_2 and the key K in two parts K_1 and K_2. The ciphertext C is then computed as $E_{K_1}(M_1)||E_{K_2}(M_2)$.

(a) What is the decryption algorithm? In particular, suppose we have a 64-bit ciphertext C, split into two parts C_1 and C_2, and suppose we know the 64-bit key K (whose two parts are K_1 and K_2). How do we compute M?

(b) Show that this scheme is vulnerable to a known-plaintext attack. In particular, assume we know that C is the encryption of the 64-bit message M (using the 64-bit encryption algorithm, under the unknown key K). Show how to recover the 64-bit key K in a reasonable amount of time.

Problem 7 ***Authentication*** **(12 points)**

Alice would like to send a message M to Bob with confidentiality and integrity. Alice and Bob share symmetric keys k_1, k_2. Bob's public key is K_B; we assume that Alice knows K_B. Below, F is AES-CMAC (a secure message authentication code) and H is SHA256 (a secure cryptographic hash).

Consider the following two schemes:

S1: Alice sends $E_{k_1}(M), F_{k_2}(M)$ to Bob
S2: Alice sends $E_{k_1}(M), H(M)$ to Bob

Here $E_{k_1}(\cdot)$ is AES-CTR mode (a secure symmetric-key encryption scheme).

(a) Which scheme is better for confidentiality, S1 or S2? Why?

(b) Which scheme is better for integrity, S1 or S2? Why?

Next, consider the following two schemes:

S3: Alice sends $E_{K_B}(M), F_{k_2}(M)$ to Bob
S4: Alice sends $E_{K_B}(M), H(M)$ to Bob

Here $E_{K_B}(\cdot)$ represents El Gamal encryption.

(c) Which scheme is better for integrity, S3 or S4? Why?

Problem 8 ***Chosen-ciphertext security of El Gamal*** **(10 points)**

Alice is using El Gamal encryption to send messages to Bob.

Recall that the El Gamal encryption of message M is the ciphertext $C = (C_1, C_2)$, where $C_1 = g^r \bmod p$, $C_2 = M \times B^r \bmod p$, r is a random number chosen separately for each encryption, and B is Bob's public key. Bob's private key is b, and $B = g^b \bmod p$. p and g are fixed and public. To decrypt a ciphertext $C = (C_1, C_2)$, Bob computes $M = C_1^{-b} \times C_2 \bmod p$.

Suppose that, after Bob decrypts a ciphertext, if it does not look like English, Bob will send back an unencrypted email saying "What happened? Your message was corrupted" and attach the decrypted message as an attachment. Thus, Mallory can learn M, the decryption of C, for any C such that M does not look like English.

Suppose that Mallory has intercepted a ciphertext $C^* = (C_1^*, C_2^*)$ sent by Alice to Bob. Mallory knows C^* is the encryption of some English message, but Mallory doesn't know what the message says. Describe a chosen-ciphertext attack that would let Mallory learn the decryption of C^*.