CNT 4419: Secure Coding [Fall 2019]
Final Exam

NAME: _____

Instructions:

1) This test is 10 pages in length.

2) You have 120 minutes to complete and turn in this test.

3) Short-answer and essay questions include guidelines for how much to write. Respond in complete English sentences. Avoid using bullet points and enumerated lists. Responses will be graded as described on the syllabus.

4) This test is closed books, notes, papers, smartphones, laptops, friends, neighbors, etc.

5) Use the backs of pages in this test packet for scratch work. If you write more than a final answer in the area next to a question, circle your final answer.

1. [2 points]
What does it mean for software to be secure? [1 sentence]

2. [2 points]
What does it mean for a programming language to be type safe? [1-2 sentences]

3. [2 points]
What is steganography? [1 sentence]

4. [2 points]
What are password salts and why are they used? [2-3 sentences]

5. [4 points] [Short essay]
Explain information-flow policies, noninterference, and their relationship.

6. [4 points] [Short essay]
Compare and contrast MACs and digital signatures.

7. [17 points]
a) Draw and label the standard three tier architecture for web applications.

b) Draw the TCP 3-way handshake as we did in class.

c) List the layers in the OSI model and their corresponding layer numbers. Circle the OSI layers that are also part of the Internet protocol suite.

d) Illustrate ECB and CBC modes. Then describe and contrast these two modes in 2-4 sentences of text.

e) Draw a diagram illustrating a man-in-the-middle attack against the Diffie-Hellman key exchange. Then briefly explain the attack in 2-3 sentences of text.

8. [18 points]

Categorize the following 4 policies, i.e., whether they are properties, safety, and/or liveness. Formally prove the correctness of your classifications, as we did in class.

(a) $P_1 = \{\{t^1, t^2, ...\} \mid \forall\ i, j: (open(j) \leq t^i) \Rightarrow (write(j) \in t^i)\}$

(b) $P_2 = \{\{t^1, t^2, ...\} \mid \forall\ i, j: (open(j) \leq t^i) \Rightarrow (write(j) \notin t^i)\}$

(c) $P_3 = \{\{t^1, t^2, ...\} \mid \forall\ i: \exists\ t : (t^i = open(\text{‘f’});write(\text{‘f’});t)\ \}$

(d) $P_4 = \{p \mid p \cup p \subseteq p \cap p\}$

(e) Of P_1 to P_4, which are the easiest and second easiest to enforce (precisely) in practice? How would the enforcement mechanisms work? [2-3 sentences]

11. [14 points]
a) What is session hijacking, and what are standard defense mechanisms? [2 sentences]

d) What is session fixation, and what are standard defense mechanisms? [2 sentences]

c) What is CSRF, and what are standard defense mechanisms? [2-3 sentences]

d) What are XXE attacks, and what are standard defense mechanisms? [2 sentences]

e) What is XSS (explain both primary categories), how can XSS violate the same-origin policy, and what are standard defense mechanisms? [Short essay]

10. [15 points]
Assume the following tables are used by a laser-tag arena to display scores on a website.

Players

PlayerID	Name
1	Laser Larry
2	Terrible Tim
3	Missing Mike

Games

Game	Player	Score
1	1	200
2	1	300
1	2	-100

a) Using a RIGHT JOIN, for every player, return the player's name, game number, and score for each game.

b) What is the output of Part a?

c) Return the player ID and number of games played for all players that have played at least one game.

d) Assume a website uses the following code to query the database, where the input variable contains user input.

```
conn.executeQuery("SELECT * FROM Games WHERE ABS(score) > " + input)
```

What might an attacker be able to input, to delete or otherwise access the Players table?

e) Describe the standard methods for preventing this attack (from Part d). For full credit, show code when it is helpful (e.g., for any techniques we saw code for in class).

11. [20 points]
Consider the following code, assuming a 32-bit architecture, that all needed #include's are present, that each int and char is stored in 1 byte, and that getUInt securely inputs an unsigned int from the user. According to its documentation, the 2nd argument to fgets, here an unsigned int, "is the maximum number of characters to be read (including the final null-character)". For each of the following, respond with a brief essay at the level of detail used in class, including showing specific inputs when it's reasonable to do so.

```
0       #define MAX 128
1       void getMessage(char *name, unsigned int size) {
2          char msg[MAX];
3          printf(name);
4          if(size+1 <= MAX) { //add 1 to be sure there's enough space for the null-char
5               fgets(msg, size, stdin);
6          }
7       }
8       int main(int argc, char *argv[]) {
9          char name[MAX];
10         unsigned int size;
11         fgets(name, MAX, stdin);
12         size = getUInt();
13         getMessage(name, size);
14      }
```

a) Describe how an attacker could overflow a buffer in this code. E.g., which buffer could be overflowed, on which line of code, which inputs would the attacker provide, and what would those inputs enable?

b) Using your basic technique from Part a, explain a stack-smashing attack on this code. Assume that the system lacks ASLR, NX, and StackGuard.

c) Describe a format-string attack on this code in detail, assuming the system uses ASLR and StackGuard. Be sure to describe the information an attacker gains from this attack.

d) Referencing Parts a-c as helpful, describe an attack on this code assuming ASLR, NX, and StackGuard are in use.