CIS 6373: Foundations of Software Security [Fall 2019]
Test III

NAME: _____

Instructions:

1) This test is 10 pages in length.

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

3) For short-answer and essay problems, respond in complete English sentences. Responses will be graded as described on the syllabus. Avoid bullet points in your responses.

4) This test is closed books, notes, papers, 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 is serialization? [1 sentence]

3. [3 points]
Explain TOCTOUs. Provide an example. [2-3 sentences]

4. [4 points]
Explain the “Thompson Hack”. [1 paragraph]

5. [6 points, essay]
Describe how injection attacks are defined by (a) SQLCheck, (b) the CIAOs paper, and (c) the BroNIEs paper. Use one or more simple examples to illustrate the ideas.

6. [5 points]
Explain and contrast “strong” and “weak” versions of CFI enforcement. [1 paragraph]

7. [5 points]
Describe the CIA classification of policies and its limitations. [1 paragraph]

8. [2 points]
Compare and contrast strongly and weakly typed programming languages. [1-2 sentences]

9. [5 points]
Describe the different kinds of security automata we discussed in class, including their features, operational semantics, and benefits. [1 paragraph]

10. [8 points]
Draw a chart to show formal definitions of policy, property, safety, and liveness in qualitative and quantitative models.

11. [5 points]
Explain how prepared statements mitigate injection attacks. [1 paragraph]

12. [7 points]
Categorize the following policy, i.e., whether it is a property, safety, and/or liveness. Formally prove the correctness of your classification at the level of detail discussed in class.

$$P = \{\{t_1, t_2, \ldots\} \mid \forall i, j, n, m: (input(n) \in t_i \wedge output(m) \in t_i \wedge input(n) \in t_j) \Rightarrow (output(m) \in t_j)\}$$

13. [4 points]
How are firewall policies normally specified? [2-3 sentences]

14. [5 points]
Compare and contrast computer security and medicine, hitting all the main points discussed in class. [1 paragraph]

15. [5 points]
Describe how coauthentication works, at a high level. [1 paragraph]

16. [3 points]
What are SQL-IDIAs? How prevalent are they? [1-2 sentences]

17. [10 points]
a) Explain the steps in a Diffie-Hellman key exchange. [1 paragraph]

b) Under what attack model does the D-H key exchange operate? [1-3 sentences]

c) How could an active attacker mount two different kinds of attacks on D-H? Explain in enough detail to convince a reader that the attacks work. [1 paragraph]

18. [19 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.