King's College London

This paper is part of an examination of the College counting towards the award of a degree. Examinations are governed by the College Regulations under the authority of the Academic Board.

Degree Programmes	MSci, MSc
Module Code	7CCSMSEN – online exam
Module Title	Security Engineering
Examination Period	January 2022 (Period 1)
Time Allowed	ONE AND A HALF HOURS.
Rubric	Answer ALL THREE questions.
Instructions	You are permitted to access any material you wish, but this is not mandated and is not expected. You may use a calculator if you find this helpful.
Start Time	*TBD*
Submission Deadline	*TBD*
Submission Process	Work must be submitted to the Level 7 Informatics Assessments KEATS page.

Your work must be submitted as a PDF file. If you have prepared some answers on computer and some on paper (which have then been digitised), you may upload at most two PDF files – one for computer-prepared answers and for digitised answers. Do not duplicate answers across the two PDF files. If you do this, the computer-prepared answer will be taken. You should check that your complete work has been uploaded successfully. Ensure you upload the correct file to the submission folder.

ACADEMIC HONESTY AND INTEGRITY

Students at King's are part of an academic community that values trust, fairness and respect and actively encourages students to act with honesty and integrity. It is a College policy that students take responsibility for their work and comply with the university's standards and requirements.

By submitting this assignment, I confirm that this work is entirely my own, or is the work of an assigned or permitted group, of which I am a member, with exception to any content where the works of others have been acknowledged with appropriate referencing.

I also confirm that I have read and understood the College's Academic Honesty & Integrity Policy: https://www.kcl.ac.uk/governancezone/assessment/academic-honesty-integrity
Misconduct regulations remain in place during this period and students can familiarise themselves with the procedures on the College website at https://www.kcl.ac.uk/campuslife/acservices/academic-regulations/assets-20-21/g27.pdf

Question 1. Consider the C programme below. [25 marks]

```
// programme.c
void MyFunc(char* str) {
        char* name[3];
        name[0]= “security”;
        name[1]= “engineering”;
        name[2]= NULL;
        }

void foo(char* str) {
        char buffer[100];
        char arr[300] = “/sh/sh//”;
        strcat(arr, str);
        MyFunc(str);
}
int main(int argc, char** argv) {
        if(!argv[1])
                return -1;
        foo(argv[1]);
        exit(0);}
```

a. **Draw the stack of this programme, including stack frames, local parameters, values, etc. [8 marks]**

b. **What is the value of the register %esp and the value of %eip when foo() finishes execution (after function epilogue)? [4 marks]?**

c. **Given the shellcode below, explain what it does. [5 marks]**

```
jmp string_addr

back:

        popl %ebx

        movl %ebx, 0x8(%ebx)

        movl $0x0, 0xc(%ebx)

        movb $0x0, 0x7(%ebx)

        leal 0x8(%ebx), %ecx

        movl $0x0, %edx

        movl $0xb, %eax

        int $0x80

string_addr:

        call back:

        .string "/bin/sh"
```

d. **Write an injection vector that will overwrite the return address of exit(0) using the shellcode above. Describe the size of this vector. The size of the shellcode in part c is 40 bytes. [4 marks]**

e. **When the attacker provides the injection vector above as input, will the attacker be able to successfully exploit a stack-based buffer overflow vulnerability? If the attacker can successfully exploit a stack-based buffer overflow vulnerability using the injection vector in part d, explain why.**

   If the attacker cannot successfully exploit the vulnerability, explain why not and describe what the attacker needs to do for successful exploitation. [4 marks]

Question 2. A secure software development lifecycle requires security engineering to fit into all the phases of the software development process. [25 marks]

a. **In which way does security engineering fit into requirements, design, implementation, and testing/assurance? Motivate your answer. [7]**

b. **What is a threat model and why is it important? [4]**

c. **Consider Leslie Lamport's Gold Standard.**
   i. **What is it? [2]**

   ii. **Explain each mechanism of Lamport's Gold Standard. [12]**

Question 3. Static analysis is a program analysis technique to analyze a program's code without running it. [25 marks]

a. **List and explain two benefits of static analysis. [4]**

b. **List and explain three drawbacks of static analysis. [3]**

c. **What impact on the software development process does static analysis have? [4]**

d. **What does it mean for static analysis to be complete and how does this differ from being sound? [4]**

e. **Consider a static taint flow analysis as we have discussed in class. Consider the code snippet below: [10]**

```
1  int printf(untainted char* fmt, ...);
2  void read(int , tainted character* input , int);
3
4  char buffer[10];
5
6  read(0, buffer, sizeof(buffer));
7  char *x = buffer;
8  printf(x);
```

Let us assume we are interested in an analysis that identifies no tainted data flows (where untainted < tainted in a lattice). Given the initial taint source and untainted sink:

i. **Create a name for each missing type qualifier, assuming a flow-/path- /context-insensitive analysis. [5]**

ii. **For each statement in the program, generate constraints on possible solutions, assuming a flow-/path-/context-insensitive analysis. [3]**

iii. **Solve the constraints to produce solutions for the type qualifiers identified earlier (1 mark) and state whether the resulting flow is legal or illegal (1 mark). As above, assume a flow-/path-/context-insensitive analysis. [2]**