LiTH, Linköpings tekniska högskola
IDA, Institutionen för datavetenskap
Ulf Kargén

Distance exam

TDDC90 Software Security

2021-08-25

Teacher on duty

Ulf Kargén, ulf.kargen@liu.se, 013-285876

Instructions and grading

There are 7 questions on the exam. Your grade will depend on the total points you score. The maximum number of points is 36. The following grading scale is preliminary and might be adjusted during grading. You may answer in Swedish or English.

Grade	3	4	5
Points required	19	27	32

Answers should be submitted through Lisam as a single PDF or ASCII text (.txt) document before the end of the exam time. If you don't have access to a good software for making technical drawings that you are already familiar with, it is recommended to draw figures by hand and scan/photograph them. Figures could either be integrated into the PDF, or submitted as separate files in JPEG or PNG format. If you chose the latter approach, file names should be of the format Figure1, Figure2, and so on. All attached figures should be clearly referred to in the text by their file name. To facilitate easier grading, it is preferred that you express formulas, program code, etc. as text, and not as handwritten figures.

You are allowed to use any aid, but **all kinds of collaboration with others is strictly forbidden**. Also, **copying any part of an answer from another source will be considered plagiarism**. The questions will be checked for plagiarism and sharing of answers between students using Urkund, and any suspected cheating will be reported to the university disciplinary board. **By taking the exam, you solemnly promise to abide by the above rules.**

In the event that you experience technical difficulties with Lisam that prevent you from submitting, it is allowable to submit answers via email. However, this should only be used as a last resort if Lisam for whatever reason would stop functioning.

The distance exam will not be anonymous due to the exceptional COVID-19 situation.

Ulf Kargén will be available to answer questions during the exam via email and phone.

Question 1: Secure software development (4 points)

a) The two activities below are either mandatory or optional steps of SDL. For each of the two, state in which of the SDL phases the activity belongs, and motivate why it makes most sense to have the activity in that particular phase. (Just stating the phase without motivation gives no points.)

   i. Static analysis
   ii. Penetration testing

b) In which phase of SDL would it be most suitable to use *attack trees*? Clearly motivate your answer.

Question 2: Exploits and mitigations (5 points)

a) In your own *words* (i.e., *not* using figures or code), give a high-level explanation of how a use-after-free bug could be used to achieve arbitrary code execution. (Your answer should address the general case, not a specific example.)

b) Which of the two following mitigations would provide the biggest obstacle for an attacker trying to exploit a use-after-free bug? Briefly explain why.

   i. ASLR
   ii. Stack Cookies.

Question 3: Design patterns (2 points)

One of the design patterns mentioned in the course literature can reduce the risk of use-after-free bugs. Explain the main idea of this design pattern, and why it reduces the risk of use-after-free bugs.

Question 4: Web security (6 points)

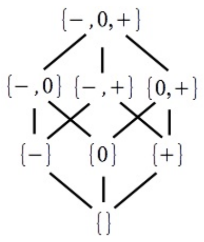
a) Using pseudo-code, write server-side code that contains a vulnerability that allows for *reflected* XSS. Your code should be detailed enough that it is clear how XSS attacks can be made. Explain your code in English (or Swedish). Give a (realistic) example of how an attacker could use the bug to attack a user of the affected site. Finally, show how the code should be altered in order to mitigate the attack.

b) Give *two* examples of web attack types discussed in the course that can lead to arbitrary code execution on a web server, and briefly explain how they work.

Question 5: Static analysis (7 points)

Assume `int` denotes integers with an absolute value that can be arbitrarily large (i.e., no integer overflows). Consider the following procedure.

```
1  void foo(int x){
2    if(x < 0){
3      x = -x;
4    }
5    int i = x;
6    int y = 0;
7    int z = 0;
8    while(i > 0){
9      y = y + 3;
10     z = z + 2;
11     i = i - 1;
12     assert(0 <= i);
13   }
14   assert(y - z == x);
15 }
```



We aim to check the assertions `(0 <= i)` and `(y - z == x)` respectively at lines 12 and 14 in procedure `foo`. Questions:

1. Symbolic execution:
   (a) Give the SSA formula for the path condition that corresponds to the path starting at line 1 of procedure `foo` and performing one iteration of the loop before violating the assertion at line 14. (1 pt)
   (b) Given value of `x`, how many paths perform loop iterations and violate the assertion at line 14? Is any of them satisfiable? Explain. (2 pt)
2. Abstract interpretation: Annotate, after convergence of the analysis, the end of each line in `foo` with the abstract element associated to each one of the defined variables. (1pt)
3. The following part has 3 questions. The three answers result in a minimum of 0 pt and a maximum of 3 pts. Each wrong answer counts negative. E.g., one correct answer (1 * 1 pt), one wrong (1 * -1 pt) and not answering one (1 * 0 pt) results in a score of 0 pt out of the 3 possible points. Giving two wrong answers (2 * -1 pt) and one correct (1 * 1 pt) gives 0 points. State, without justification, whether each of the following Hoare-triples is true or false.
   (a) $\{y == 0\}$ `while(x > 0){x--; y++;}` $\{x <= 0\}$
   (b) $\{y + x > 0\}$ `x = 0;` $\{y > 0\}$
   (c) $\{x = 2 \text{ and } y = 2\}$ `while(x > 0){x--; y++;}` $\{y > 3\}$

Question 6: Security testing (6 points)

a) Rank the following security-testing techniques from fastest to slowest in terms of test-generation speed (i.e., how long it takes to generate a single test case): *Black-box mutational fuzzing*, *Concolic testing*, *Greybox fuzzing*. Also, clearly explain the reasons for the differences in speed.

b) Mutation-based fuzzers typically do not work so well for fuzzing implementations of stateful network protocols. Explain why, name a fuzzing technique more suited for this use case, and explain why it is a better choice.

Question 7: Vulnerabilities in C/C++ programs (6 points)

The code on the next page shows a function that takes a simple greyscale bitmap image as input, and renders it as ASCII art. The code has a serious security bug.

It can be assumed that the input buffer (`data`) always contains exactly `width*height` bytes.

a) Identify the bug in the code, and state what kind of bug it is (out of the security bug types discussed in the course).

b) Clearly explain, using an example, what an input to the function should look like in order to trigger the bug, and what the consequences of a successful exploit could be (i.e., what the bug allows the attacker to do).

c) Show using code/pseudocode (and a clear motivation) how the bug should be fixed.

```c
// Returns a malloc-allocated string with the ASCII-art render of an 8-bit
// grayscale bitmap image. Returns a NULL-pointer in case of error.
char* render_ascii(unsigned int width,
                   unsigned int height,
                   const char* data)
{
   if( (width+1) > UINT_MAX / height)
      return NULL;

   char* output = malloc( (width+1) * height );
   size_t out_pos = 0;

   for(unsigned int y = 0; y < height; y++) {
      for(unsigned int x = 0; x < width; x++) {
         unsigned char pixel = data[y*width + x];
         char character;

         if(pixel < 64)
            character = '#';
         else if (pixel < 128)
            character = '*';
         else if (pixel < 192)
            character = '-';
         else
            character = ' ';

         output[out_pos++] = character;
      }
      output[out_pos++] = '\n'; // Add line-break
   }

   // Replace last line-break with NULL-terminator to finish output
   output[out_pos - 1] = 0;

   return output;
}
```