

Software and Web-Security

Assignment 4, Monday, March 2, 2015

Handing in your answers: Submission via Blackboard (<http://blackboard.ru.nl>)

Deadline: Monday, March 9, 24:00 (midnight)

1. This exercise is about the size of heap space available to a program.
 - (a) Write a program (in a file called `exercise1.c`), which determines the maximal amount of heap space that can be allocated in one call to `malloc`. The output of the program should be of the following form (where `XXX` is replaced by the correct number):

```
One malloc can allocate at most XXX bytes.
```
 - (b) Is the output of the program always the same? Explain why. Write your answer to a file called `exercise1b`.
2. Consider the following code snippet:

```
void heap_attack(void)
{
    //...
}

int main(void)
{
    char *s1 = malloc(8);
    if(s1 == NULL) return -1;
    char *s2 = malloc(8);
    if(s2 == NULL) return -1;

    //...

    heap_attack();

    printf("student 1: %s\n", s1);
    printf("student 2: %s\n", s2);

    return 0;
}
```

- (a) Copy this snippet to a file called `exercise2.c`.
- (b) Replace the `//...` in the `main` function to set `s1` and `s2` to your student numbers.
- (c) Why does the program allocate 8 bytes of heap space for the 7-character strings `s1` and `s2`? Write your answer to a file called `exercise2c`.
- (d) Replace the `//...` in the function `heap_attack` such that one of the two `printf` function calls in `main` prints both `s1` and `s2` (separated by one or several spaces). The output of the other `printf` call should be unchanged.
Note: You may not change anything in the `main` function!
Hint 1: Note that no address is passed to `heap_attack`, so you will have to search the heap for the two strings. If you allocate memory on the heap, it is most likely going to be close to `s1` and `s2`.
Hint 2: If you access memory that is too far outside the allocated area for your program, it will crash with a *segmentation fault*. If this happens, you are probably searching in the wrong direction (or at the wrong place).
- (e) We intentionally did not `free` the memory that is allocated for `s1` and `s2`. What happens if you free this memory after calling your implementation of `heap_attack`? Explain why. Write your answer to a file called `exercise2e`.

3. Consider the following C code:

```
#include <stdlib.h>
#include <stdio.h>

int main(void)
{
    int i,j;
    unsigned long long **m;
    unsigned long long **mt;

    while(1)
    {
        // allocate matrix m
        m = malloc(1000*sizeof(unsigned long long*));
        if(m == NULL) return -1;
        for(i=0;i<1000;i++)
        {
            m[i] = malloc(1000*sizeof(unsigned long long));
            if(m[i] == NULL) return -1;
        }

        // allocate matrix mt
        mt = malloc(1000*sizeof(unsigned long long*));
        if(mt == NULL) return -1;
        for(i=0;i<1000;i++)
        {
            mt[i] = malloc(1000*sizeof(unsigned long long));
            if(mt[i] == NULL) return -1;
        }

        for(i=0;i<1000;i++)
            for(j=0;j<1000;j++)
                m[i][j] = 1000*i+j;

        // transpose matrix m, write to mt
        for(i=0;i<1000;i++)
            for(j=0;j<1000;j++)
                mt[i][j] = m[j][i];

        // free matrices m and mt
        free(m);
        free(mt);
    }

    return 0;
}
```

- (a) Write the code to a file called `exercise3.c`.
- (b) Compile and run the code; describe what happens and explain why. Write your answer to a file called `exercise3b`.
- (c) Fix the problem in this code.

4. Place the files

- `exercise1.c`,
- `exercise1b`,
- `exercise2.c`,
- `exercise2c`,
- `exercise2e`,
- `exercise3.c` (with the fix from exercise 3c), and
- `exercise3b`

in a directory called `sws1-assignment4-STUDENTNUMBER1-STUDENTNUMBER2` (again, replace `STUDENTNUMBER1` and `STUDENTNUMBER2` by your respective student numbers). Make a `tar.gz` archive of the whole `sws1-assignment4-STUDENTNUMBER1-STUDENTNUMBER2` directory and submit this archive in Blackboard.