CS177: Computer Security
Prof. Stefano Tessaro

UC Santa Barbara
Spring 2016

Homework 4

Posted: Thursday, May 5, 2016 – 11:59pm
Due: Monday, May 16, 2016 – 5pm – online submission only! Read instructions!

Instructions. Read the Piazza post at `https://piazza.com/class/im8xip1tcj4ab?cid=115`, and follow the instructions to create and access your personal account. Your account will give you a safe environment to experiment with buffer overflows. To submit your solutions, create a sub-directory `hw4` in your home directory, containing:

(1) Your exploits, named `exploit1.x`, `exploit2a.x` and `exploit2b.x`, where `x = c` or `x = py`,

(2) A `README` file documenting what you have done.

Once logged in, you are not allowed to use your account for any other purposes than solving this homework. Also, if you gain user privileges for `mr177`, you are only allowed to confirm you have gained access as explained below.

Exploits can take the form of a program (in C or python) calling `auth` or `auth2` with a carefully crafted argument, and possibly setting other system variables.

Task 1 – Simple Control-Flow Hijacking (15 points)

The executable `/home/mr177/bin/auth` is vulnerable to a buffer overflow. The C code of `auth` is available on the Piazza page. The program is owned by `mr177` and has the setuid flag, i.e., it runs with `mr177`'s privileges. No stack protections are activated. The program has syntax

```
/home/mr177/bin/auth password
```

and running it with the right password will result in a secret word being printed to screen.

The goal of this task is to write an exploit (in Python or in C) resulting in auth outputting the secret word *without* guessing the password. While many solutions are possible, it may help you to learn to use `gdb` properly, and answer the following questions:

a) What is the address of the first instruction inside `main` executed if the output of `authenticate` is indeed 0.

b) Where are the relevant accessible portions of the stack within `authenticate`, in particular with respect to the position of the stored return pointer?

Task 2 – Shellcode (15 + 10 points)

Another vulnerable executable is `/home/mr177/bin/auth2`. It has the same identical behavior as `auth`, and the only small difference is to be seen in the corresponding source code, also available on Piazza.

a) Write an exploit for `auth2` resulting in opening up a shell with `mr177`'s privileges.

b) (Bonus 10 points) Do the same for `auth`.

Create an empty file named `FirstnameLastname` in `/home/mr177/visitors/` once you have obtained shell access. Creating this file will only be possible by obtaining `mr177`'s access privileges.

Hint: While you may use other ones, it is recommended that you use the (right) shellcode from AlephOne's tutorial.