IERG4130 Introduction to Cyber Security (2017 Fall)

Laboratory Assignment 1

DEADLINE: Oct 18, 2017, HKT 11:59 pm

General Guidelines

In terms of ethical hacking, the techniques you apply here are restricted only to the specified targets in this lab manual. You must not break into any other CUHK and non-CUHK systems.

Students are required to complete every task in this lab assignment. Submit a lab report in pdf format to **Blackboard Learn** on or before the deadline. **No hardcopy or late submission will be accepted**. You are reminded that you are not allowed to copy from any sources without proper citations and acknowledgements. All lab reports must include the following declaration:

Declaration of Academic Honesty

I declare that the assignment here submitted is original except for source material explicitly acknowledged, and that the same or related material has not been previously submitted for another course. I also acknowledge that I am aware of University policy and regulations on honesty in academic work, and of the disciplinary guidelines and procedures applicable to breaches of such policy and regulations, as contained in the website http://www.cuhk.edu.hk/policy/academichonesty/

Name: Student ID:

When doing this lab, please follow the general homework policies

General homework policies:

A student may discuss the problems with others. However, the work a student turns in must be created COMPLETELY by oneself ALONE. A student may not share ANY written work or pictures, nor may one copy answers from any source other than one's own brain.

Each student **MUST LIST** on the homework paper the **name of every person he/she has discussed or worked with**. If the answer includes content from any other source, the student **MUST STATE THE SOURCE**. Failure to do so is cheating and plagiarism and will result in sanctions. Copying answers from someone else is cheating even if one lists their name(s) on the homework.

If there is information you need to solve a problem but the information is not stated in the problem, try to find the data somewhere. If you cannot find it, state what data you need, make a reasonable estimate of its value, and justify any assumptions you make. You will be graded not only on whether your answer is correct, but also on whether you have done an intelligent analysis.

You can discuss with or consult other people but make sure all the experiments are done by yourself. We reserve the rights to interview and decide what portion of marks to give.

PROCEDURES

1. Download the virtual machine (VM) SEEDUbuntu9_August_2010.tar.gz:
    - http://home.ku.edu.tr/~akupcu/security/SEEDUbuntu9_August_2010.tar.gz
2. Download the lab manual SEED_Book_1_2011.pdf from Blackboard Learn.
3. Read Chapter 7 (p. 184) and prepare the VM environment.
4. Create a directory named with your student ID (e.g. /home/seed/Desktop/1155xxxxxx), and finish all tasks in that directory. **Show the directory name in the screen captures.**
5. Complete the following tasks
    - 4. 1 Buffer Overflow Vulnerability Lab Tasks 1-4 (p.10-17).
        - ✧ **Bonus points will be given if you write your own shellcode/attack code with explanation on how it works.**
    - 6.4 Linux Capability Exploration Lab Tasks 1-2 (p.160-166)
        - ✧ **Q2.2 and Q6 are set as bonus questions.**
    - Answer the question about buffer overflow on the next page.
6. Beware of special characters, such as " ' ", when copying source code from the lab manual.
7. Write a detailed lab report to describe what you have done and observed. Describe your steps with the aid of code snippets and/or screen captures.

The tasks of this lab assignment are adopted from the SEED project (http://www.cis.syr.edu/~wedu/seed) maintained by Prof. Wenliang Du of Syracuse University.

Question : Buffer Overflow

(a) Suppose the following C function is running on web server of insecure.com.

```
/* Guaranteed: length == size(username) */
void insecure(uint64_t username[], unsigned int length){
unsigned char buf[18];
int i;
for (i=0; i<length; i++) {
memcpy(&(buf[i]), &(username[i]), sizeof(uint64_t));
}
}
```

where `username[]` is the content provided by user from the following HTML fragment:

```
<form action="submit" method="post">
Username: <input type="text" name="username" maxlength=
    20>
<!-- The rest of the html tags omitted -->
</form>
```

The developers of insecure.com have implemented their server-side program in C very carefully so that whenever `insecure()` is called by the code they implemented, the argument length is equal to the number of elements in `username[]` (i.e., the precondition of `insecure()` always holds).

Explain how an attacker can bypass the length restriction to launch buffer overflow attack.

(b) To fix the problem in (a), the developers decide to add the code below for server side validation:

```
/* Guaranteed: length == size(username) */
void insecure(uint64_t username[], unsigned int length){
unsigned char buf[18];
int i;
for (i=0; i<length && i<18; i++) {
memcpy(&(buf[i]), &(username[i]), sizeof(uint64_t));
}
}
```

Does any form of buffer overflow attack still exist? Justify your answer.

(c) Name and briefly describe two ways to mitigate buffer overflow attack.