Secure programming
Homework 1
Due: Friday, Nov. 11 & 2016

The practice of buffer overflow: stack smashing test.
Fig. 1 is an example C source code. Please

(1) Write down the possible buffer overflow problems in this program (only observe it without using any analysis tools). You should explain *how* and *why* these buffer overflow flaws can be exploited.
(2) Use a static analysis tool called flawfinder (running in Linux system, see http://www.dwheeler.com/flawfinder/) to audit this code and explain the security issues.
(3) Apply some tools such as GDB to do the stack smashing attack under the operation systems of Windows or Linux (Linux is better; you are suggested to install a ***virtual machine*** for Linux environment). Note that you may disable some protections by OS or compilers to make your attack successful. You should write down the detail steps for your attack process.
(4) Modify this program to avoid the buffer overflow attack and other possible threats you found. You should explain your resolving method.

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

int UPtest(char *, char *,    char *);
void myprivatetest(void);

int main(int argc, char**argv){

    if(UPtest(argv[1], argv[2], argv[3])){
       printf("Access granted...\n");
    } else {
       printf("Wrong username and password!!!!\n");
    }
    return 0;
}

int UPtest(char *a1 , char *a2, char *a3){

    char Uid[27], Uname[25], Upass[70];
    strcpy(Uid, a1);
    strcpy(Uname, a2);
    strcpy(Upass, a3);

if(!strcmp(Uname, "Admin") && !strcmp(Upass, "PassAd009"))
        return 1;
else
        return 0;
}

void myprivatetest(){
printf("This is test code to run other system program.\n");
system("/usr/bin/xeyes");
}
```

Fig. 1 The C code to test buffer overflow attack