Case Western Reserve University

Department of Computer and Data Sciences

EECS 349&444: Computer Security

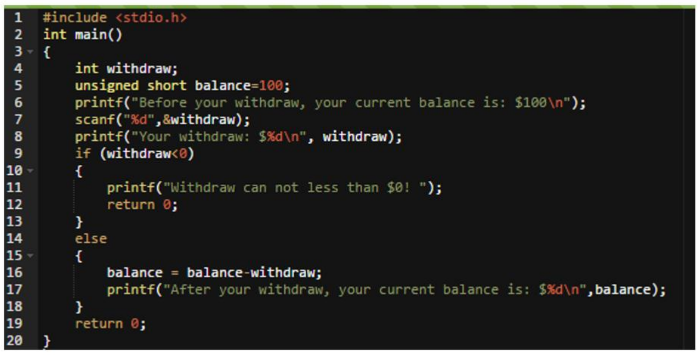
Assignment Date:	09/17/2019
Sumission Date:	09/23/2019
First Name:	
Last Name:	
Google Drive Link:	
Abstract of the feedback:	

* This is the fist half-part of HW1 which contains 40 points. You are encouraged to finish independently. Any submitted work that it copied from any source or too similar to be an independent write-up will not be given credit. Please submit this homework (e-copy) on SIS and cc your primary TA with the email title of "EECS349/444-HW1(1) submission – (your name)" by 23:59pm on 09/23/2019.

Problem 1 (10 pts). Interger Overflow Practice. ***(In-class**, i.e., you won't get credits if you submit after class)*

Practice: Hacking the ATM

- ❑ The current balance in Mr. Smith's banking account is **$100**. Now because of the buffer overflow of the software in the ATM, Mr. Smith exploited the vulnerability and found that when he withdrew $X, his balance didn't decrease but increased to $200?! What should $X be?
- ❑ To maximize the balance in his account, how much $ he should withdraw?



```
#include <stdio.h>
int main()
{
    int withdraw;
    unsigned short balance=100;
    printf("Before your withdraw, your current balance is: $100\n");
    scanf("%d",&withdraw);
    printf("Your withdraw: $%d\n", withdraw);
    if (withdraw<0)
    {
        printf("Withdraw can not less than $0! ");
        return 0;
    }
    else
    {
        balance = balance-withdraw;
        printf("After your withdraw, your current balance is: $%d\n",balance);
    }
    return 0;
}
```

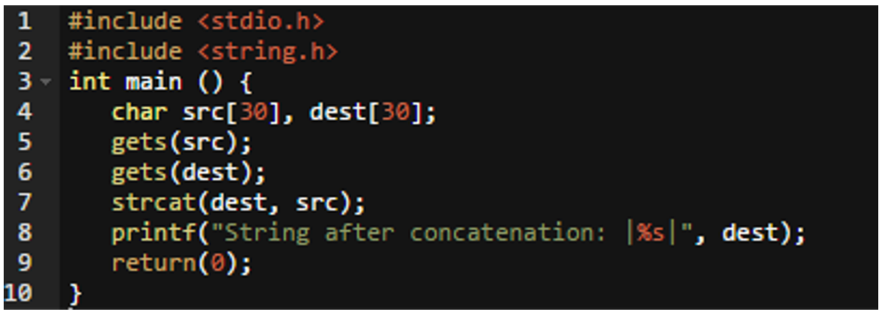
Problem 2 (10 pts). Stack overflow – gets(), strcat()

- ❑ What are the outputs for the corresponding inputs?

  - Input (1):
    src
    dest+
  - Input (2):
    src
    Do you think I can successfully append the src to the dest?

- ❑ Rewrite the program so that it is no longer vulnerable to a buffer overflow.

Note: Please also design the abuse cases to test the robustness of the program.



```
#include <stdio.h>
#include <string.h>
int main () {
   char src[30], dest[30];
   gets(src);
   gets(dest);
   strcat(dest, src);
   printf("String after concatenation: |%s|", dest);
   return(0);
}
```

Problem 3 (10 pts). Interger overflow – spot the defect of the following codes (i.e., give an example that will cause the overflow).

```
bool fun(char *s1, int len1, char *s2, int len2){
    if(len1 + len2 + 1 > 1024){
        return false;
    }
    char * pBuf = new char[len1 + len2 + 1];
    if(pBuf == NULL){
        return false;
    }
    memcpy(pBuf, s1, len1);
    memcpy(pBuf + len1, s2, len2);
    return true;
}
```

Problem 4 (10 pts). Interger overflow – spot the defect of the following codes (i.e., give an example that will cause the overflow).

```
bool fun(int *name, long cbBuf){
    unsigned short cbCalculatedBufSize = cbBuf;
    int *buf = new int(cbCalculatedBufSize);
    if(buf == NULL)
    {
        return false;
    }
    memcpy(buf, name, cbBuf);
    return true;
}
```