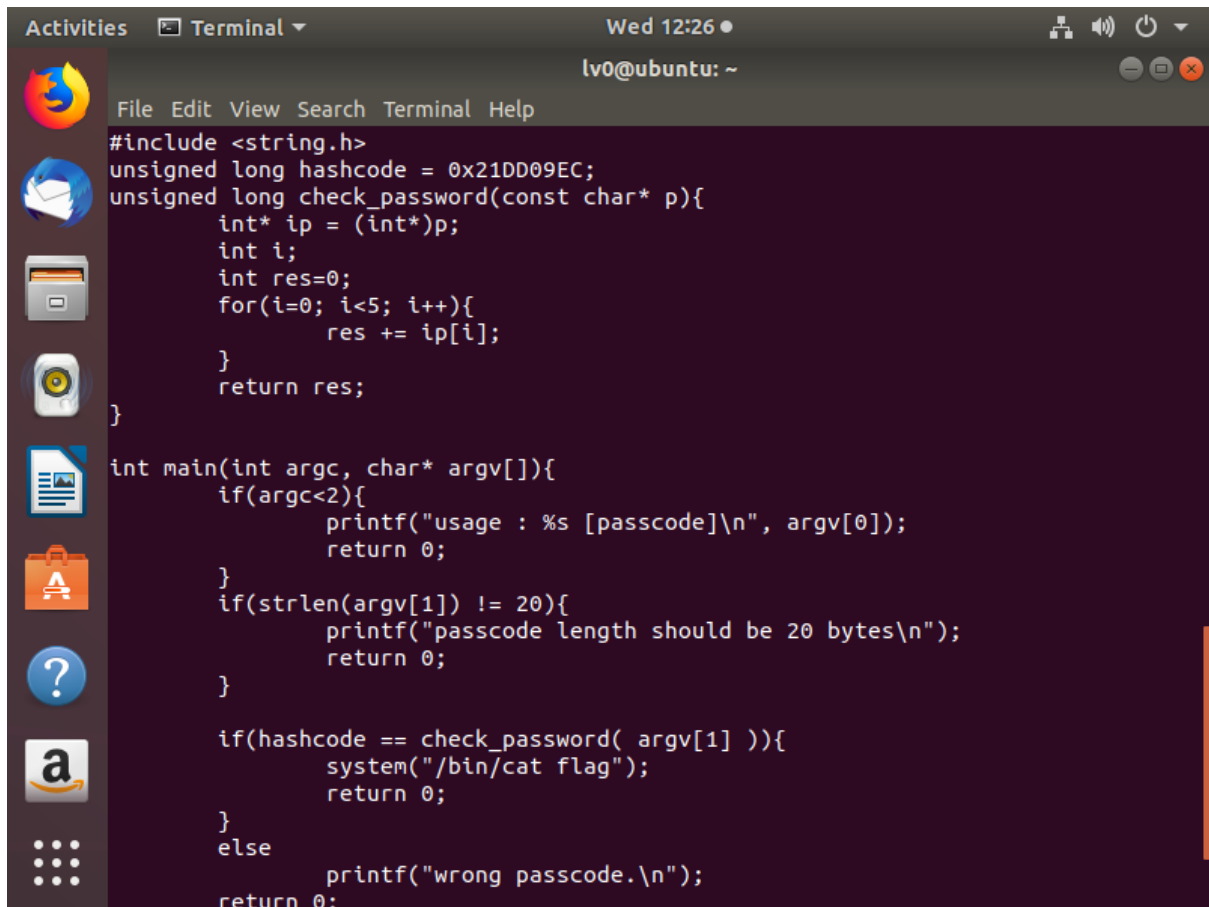


collision Write Up

윤준혁



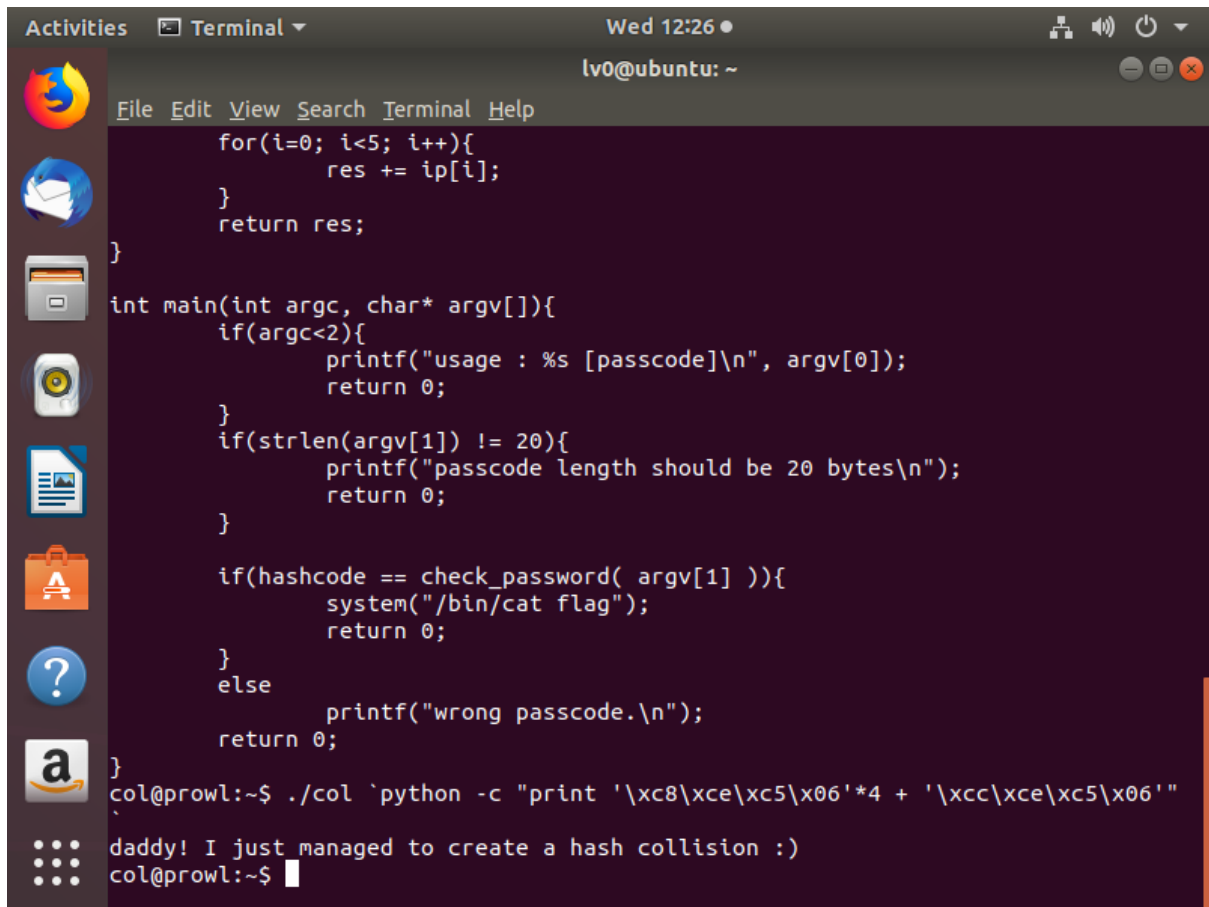
```
Activities Terminal Wed 12:26 lv0@ubuntu: ~
File Edit View Search Terminal Help
#include <string.h>
unsigned long hashcode = 0x21DD09EC;
unsigned long check_password(const char* p){
    int* ip = (int*)p;
    int i;
    int res=0;
    for(i=0; i<5; i++){
        res += ip[i];
    }
    return res;
}

int main(int argc, char* argv++){
    if(argc<2){
        printf("usage : %s [passcode]\n", argv[0]);
        return 0;
    }
    if(strlen(argv[1]) != 20){
        printf("passcode length should be 20 bytes\n");
        return 0;
    }

    if(hashcode == check_password( argv[1] )){
        system("/bin/cat flag");
        return 0;
    }
    else
        printf("wrong passcode.\n");
    return 0;
}
```

c 코드를 살펴보니 check_password(argv[1])와 hashcode가 일치해야 flag를 출력하는듯 하다. 또한 argv[1]의 길이는 20byte여야 한다.

check_password함수를 살펴보면 ip 변수가 char* 타입 변수인 p를 인자로 받아 int* 타입으로 타입 캐스팅을 한 뒤 res에 저장하고 res를 return한다. 이 때 int는 4바이트, char는 한 문자 당 1바이트이므로 우리가 입력한 문자열의 4자리가 ip[i] 하나로 저장이 된다. 따라서, argv[1]를 4byte씩 나누어 int형태로 더한 값이 hashcode와 같으면 된다. 0x21DD09EC를 5로 나뉘보면 0x6C5CEC8 4개와 0x6C5CECC 한 개를 더하면 된다.



The screenshot shows a terminal window titled 'Terminal' with the user 'lv0@ubuntu: ~'. The window contains a C program for password validation. The program defines a function `check_password` that sums the first five characters of a password. The `main` function checks the number of arguments, prints usage if there's only one, checks if the password length is 20 bytes, and then compares the hash of the password with a hardcoded value. If they match, it runs `cat flag`; otherwise, it prints 'wrong passcode.\n'. The user runs the program with a specific command, and the output is 'daddy! I just managed to create a hash collision :)'.

```
for(i=0; i<5; i++){
    res += ip[i];
}
return res;
}

int main(int argc, char* argv[]){
    if(argc<2){
        printf("usage : %s [passcode]\n", argv[0]);
        return 0;
    }
    if(strlen(argv[1]) != 20){
        printf("passcode length should be 20 bytes\n");
        return 0;
    }

    if(hashcode == check_password( argv[1] )){
        system("/bin/cat flag");
        return 0;
    }
    else
        printf("wrong passcode.\n");
    return 0;
}

col@prowl:~$ ./col `python -c "print '\xc8\xce\xc5\x06'*4 + '\xcc\xce\xc5\x06'"`
daddy! I just managed to create a hash collision :)
col@prowl:~$
```

플래그를 알아낼 수 있었다.

flag{daddy! I just managed to create a hash collision :)}