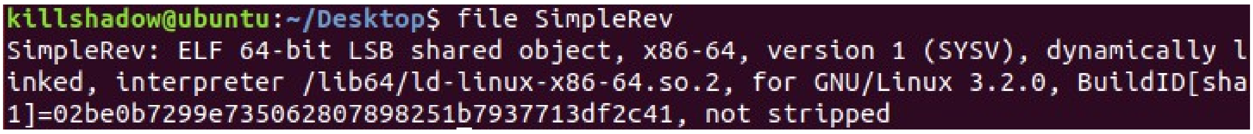
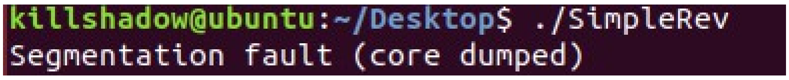
SimpleRev

1. 一开始现在Linux下用`file`命令看看是什么文件:



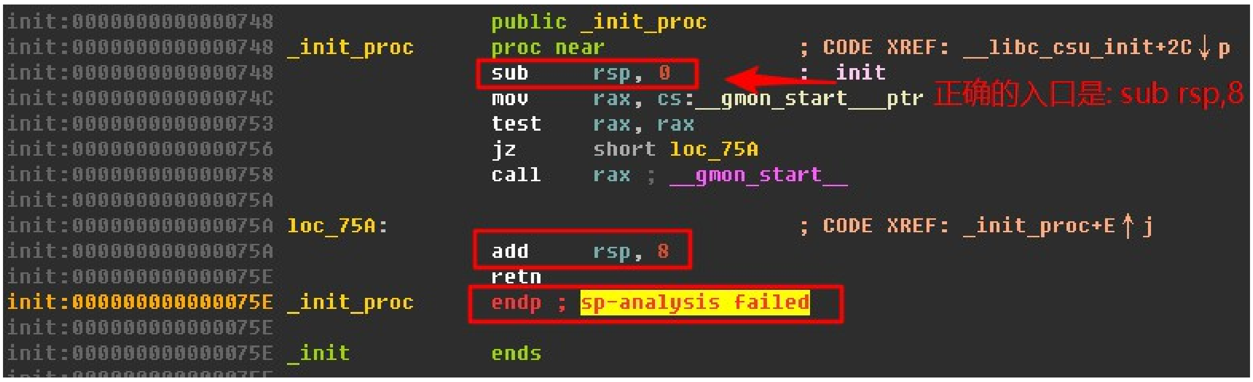
```
killshadow@ubuntu:~/Desktop$ file SimpleRev
SimpleRev: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically l
inked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 3.2.0, BuildID[sha
1]=02be0b7299e735062807898251b7937713df2c41, not stripped
```

2. 然后执行一下程序, 一看究竟, 发现执行不了`Segmentation fault (core dumped)`段错误:



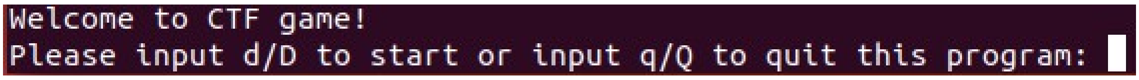
```
killshadow@ubuntu:~/Desktop$ ./SimpleRev
Segmentation fault (core dumped)
```

3. 用IDA打开, 从程序初始化开始, 在_init_proc中出现了`sp-analysis failed`的错误, 通过对比两个rsp可知, 是因为函数开头的rsp被修改了:



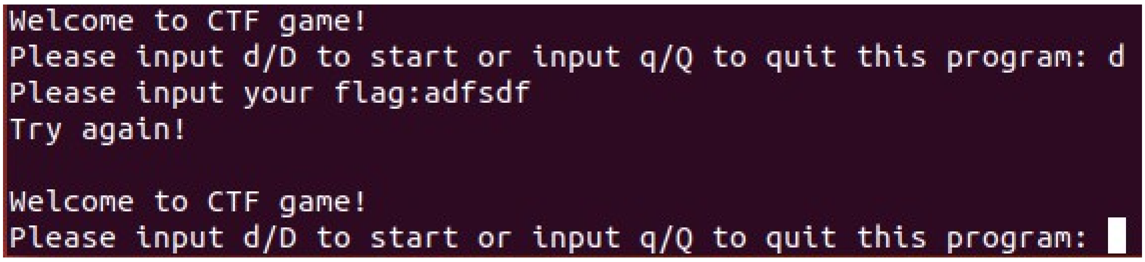
```
init:0000000000000748                 public _init_proc
init:0000000000000748 _init_proc      proc near               ; CODE XREF: __libc_csu_init+2C↓p
init:0000000000000748                 sub     rsp, 0          ;   init
init:000000000000074C                 mov     rax, cs:__gmon_start___ptr
init:0000000000000753                 test    rax, rax
init:0000000000000756                 jz      short loc_75A
init:0000000000000758                 call    rax ; __gmon_start__
init:000000000000075A
init:000000000000075A loc_75A:                                ; CODE XREF: _init_proc+E↑j
init:000000000000075A                 add     rsp, 8
init:000000000000075E                 retn
init:000000000000075E _init_proc      endp ; sp-analysis failed
init:000000000000075E
init:000000000000075E _init           ends
```

4. patch之后, 能正常运行:



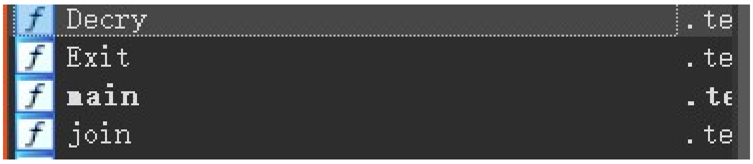
```
Welcome to CTF game!
Please input d/D to start or input q/Q to quit this program:
```

5. 可以大致猜测, 这是一道分析算法的逆向题:



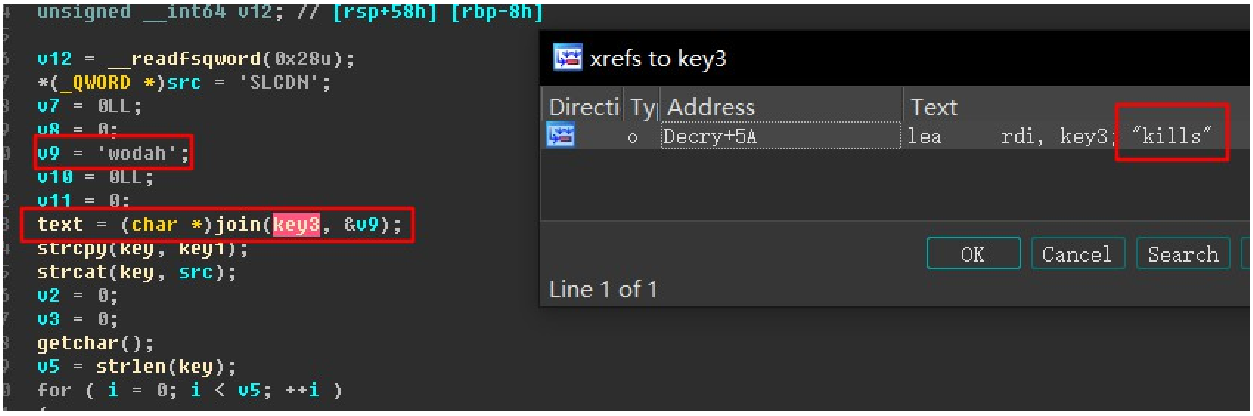
```
Welcome to CTF game!
Please input d/D to start or input q/Q to quit this program: d
Please input your flag:adfsdf
Try again!

Welcome to CTF game!
Please input d/D to start or input q/Q to quit this program:
```

6. 再回到IDA，看看特征函数:



```
f Decry    .te
f Exit     .te
f main     .te
f join     .te
```

7. 通过下面反汇编内容，就已经可以看出key和text分别是`ADSFKSLCDN`和`killshadow`:

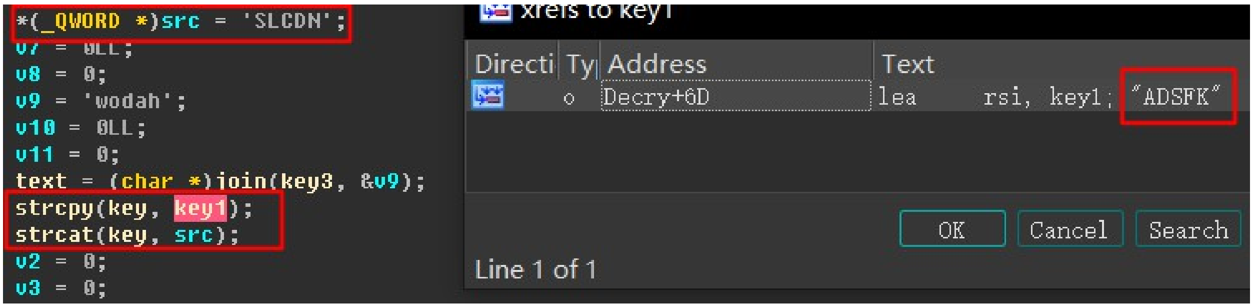


```
unsigned __int64 v12; // [rsp+58h] [rbp-8h]

v12 = __readfsqword(0x28u);
*(_QWORD *)src = 'SLCDN';
v7 = 0LL;
v8 = 0;
v9 = 'wodah';
v10 = 0LL;
v11 = 0;
text = (char *)join(key3, &v9);
strcpy(key, key1);
strcat(key, src);
v2 = 0;
v3 = 0;
getchar();
v5 = strlen(key);
for ( i = 0; i < v5; ++i )
```

xrefs to key3

Directi	Ty	Address	Text
	o	Decry+5A	lea rdi, key3; "kills"

Line 1 of 1



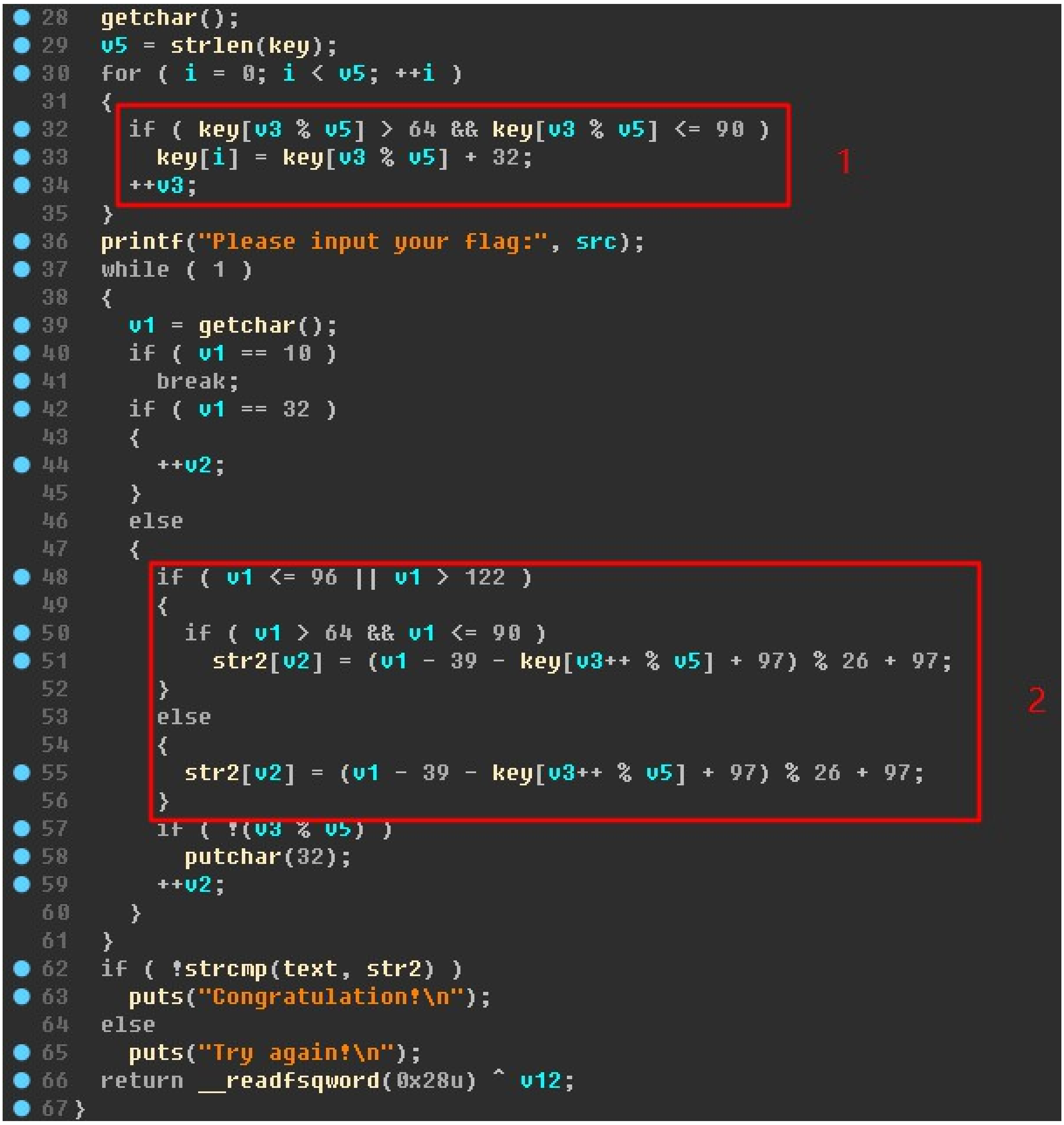
```
*(_QWORD *)src = 'SLCDN';
v7 = 0LL;
v8 = 0;
v9 = 'wodah';
v10 = 0LL;
v11 = 0;
text = (char *)join(key3, &v9);
strcpy(key, key1);
strcat(key, src);
v2 = 0;
v3 = 0;
```

xrefs to key1

Directi	Ty	Address	Text
	o	Decry+6D	lea rsi, key1; "ADSFK"

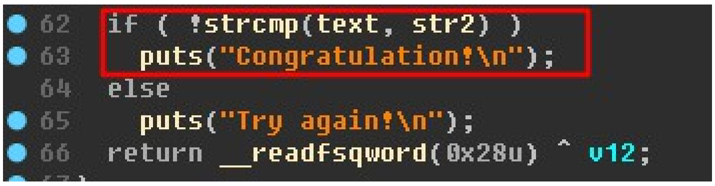
Line 1 of 1

8. 再看看函数逻辑：



```
getchar();
v5 = strlen(key);
for ( i = 0; i < v5; ++i )
{
  if ( key[v3 % v5] > 64 && key[v3 % v5] <= 90 )
    key[i] = key[v3 % v5] + 32;
  ++v3;
}
printf("Please input your flag:", src);
while ( 1 )
{
  v1 = getchar();
  if ( v1 == 10 )
    break;
  if ( v1 == 32 )
  {
    ++v2;
  }
  else
  {
    if ( v1 <= 96 || v1 > 122 )
    {
      if ( v1 > 64 && v1 <= 90 )
        str2[v2] = (v1 - 39 - key[v3++ % v5] + 97) % 26 + 97;
    }
    else
    {
      str2[v2] = (v1 - 39 - key[v3++ % v5] + 97) % 26 + 97;
    }
    if ( !(v3 % v5) )
      putchar(32);
    ++v2;
  }
}
if ( !strcmp(text, str2) )
  puts("Congratulation!\n");
else
  puts("Try again!\n");
return __readfsqword(0x28u) ^ v12;
}
```

可以看到，首先是对key操作，然后是跟text比较：

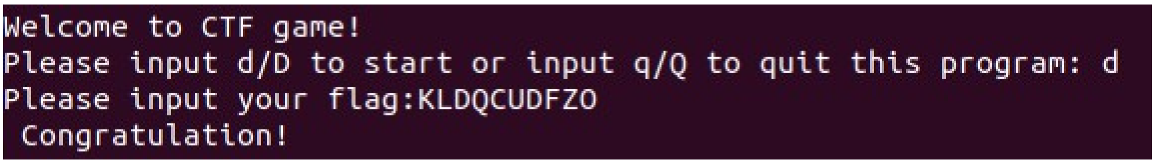


```
if ( !strcmp(text, str2) )
  puts("Congratulation!\n");
else
  puts("Try again!\n");
return __readfsqword(0x28u) ^ v12;
```

9. 写出解密算法即可解出flag：

```
void Encry()
{
    char key[100];
    char ch,temp;
    int L,i=0,j=0;int t;
    if(getchar()=='\n')
        temp=' ';

    printf("Input key: ");
    fgets(key,100,stdin);
    L=strlen(key);
    for(t=0;t<L;t++)
    {
        if (key[j%L] >= 'A'&&key[j%L] <= 'Z' || key[j%L] >= 'a'&&key[j%L] <= 'z')
        {
           key[t] = key[j%L] + 32;
       }
        else if(key[j%L] >= ' ' && key[j%L] <= '@')
       {
            key[t] = key[j%L] + 32;
        }
        j++;

    }

    while((ch=getchar())!='\n')
    {
        if(ch==' ')
        {
            i++;
            continue;
        }
        if(ch>='a'&&ch<='z')
        {
            str1[i] = (ch - 'a' + key[j%L] - 'a') % 26 + 'A';
            printf("%c",str1[i]);

            j++;
        }
        if(ch>='A'&&ch<='Z')
        {
            str1[i] = (ch - 'a' + key[j%L] - 'a') % 26 + 'A';
            printf("%c", str1[i]);
            j++;
        }
        if (ch >= ' '&& ch <= '@')
        {
            str1[i] = (ch - 'a' + key[j%L] - 'a') % 26 + 'A';
            printf("%c", str1[i]);
            j++;
        }
        if(j%L==0)
            printf(" ");
        i++;
    }
    putchar(ch);
}
```

10. 最后解得flag为：`KLDQCUDFZO`



```
Welcome to CTF game!
Please input d/D to start or input q/Q to quit this program: d
Please input your flag:KLDQCUDFZO
 Congratulation!
```