

反编译

```
public void onGoClick(View arg5) {  
    if(this.getSecret(this.getFlag()).equals(this.getSecret(this.encrypt(this.etFlag.getText().toString())))) {  
        Toast.makeText(((Context)this), "Success", 1).show();  
    }  
    else {  
        Toast.makeText(((Context)this), "Failed", 1).show();  
    }  
}  
}
```

抵消掉了

要这两个相等

查找 so

```
1 // 点击a1, 按下y, 输入JNIEnv*, 就能得到以下的伪代码效果  
2 int __fastcall Java_com_ph0enix_android_1crackme_MainActivity_encrypt(JNIEnv *a1)  
3 {  
4     JNIEnv *v1; // r6  
5     const char *v2; // r4  
6     const char *i; // r5  
7  
8     v1 = a1;  
9     v2 = (const char *)((int (*)(void))(*a1)->GetStringUTFChars()); // 传入的参数  
10    for ( i = v2; i - v2 < strlen(v2); ++i ) // 对传入字符串的每一位的ascii值减一  
11        --*i;  
12    return ((int)(__fastcall*)(JNIEnv *, const char *))(*v1)->NewStringUTF(v1, v2);  
13 }
```

```

1 int __fastcall Java_com_ph0en1x_android_1crackme_MainActivity_getFlag(int a1)
2 {
3     char *v1; // r4
4     int v2; // r7
5     char *v3; // r3
6     int v4; // r0
7     int v5; // r1
8     char *v6; // r2
9     const char *v7; // r3
10    int v8; // r0
11    int v9; // r1
12    int v10; // r4
13    int v11; // r0
14    __int16 v12; // r3
15    signed int v13; // r8
16    signed int v14; // r0
17    char *v15; // r9
18    char v16; // r3
19    char v17; // t1
20    int v18; // r1
21    char s; // [sp+4h] [bp-5Ch]
22    char v21[40]; // [sp+14h] [bp-4Ch]
23    char v22; // [sp+40h] [bp-20h]
24
25    v1 = v21;
26    v2 = a1;
27    v3 = (char *)& dword_2770;
28    do
29    {
30        v4 = *(_DWORD *)v3;
31        v3 += 8;
32        v5 = *((_DWORD *)v3 - 1);
33        *(_DWORD *)v1 = v4;
34        *((_DWORD *)v1 + 1) = v5;
35        v1 += 8;
36    }
37    while ( v3 != "Hello Ph0en1x" );
38    v6 = &s;
39    v7 = "Hello Ph0en1x";
40    do
41    {
42        v8 = *(_DWORD *)v7;
43        v7 += 8;
44        v9 = *((_DWORD *)v7 - 1);
45        *(_DWORD *)v6 = v8;
46        *((_DWORD *)v6 + 1) = v9;
47        v10 = (int)(v6 + 8);
48        v6 += 8;
49    }
50    while ( v7 != "0en1x" );
51    v11 = *(_DWORD *)v7;

```

00000EE0 Java_com_ph0en1x_android_1crackme_MainActivity_getFlag:1 (EE0)

其实这个 getFlag 函数可以不用看，应为此函数是一个无输入的固定值，我们想办法把他回显出来其实就行了。

拿 getFlag 的值

这里有两种方法。

第一种是动态调试，

第二种是修改 smail 文件然后回编译

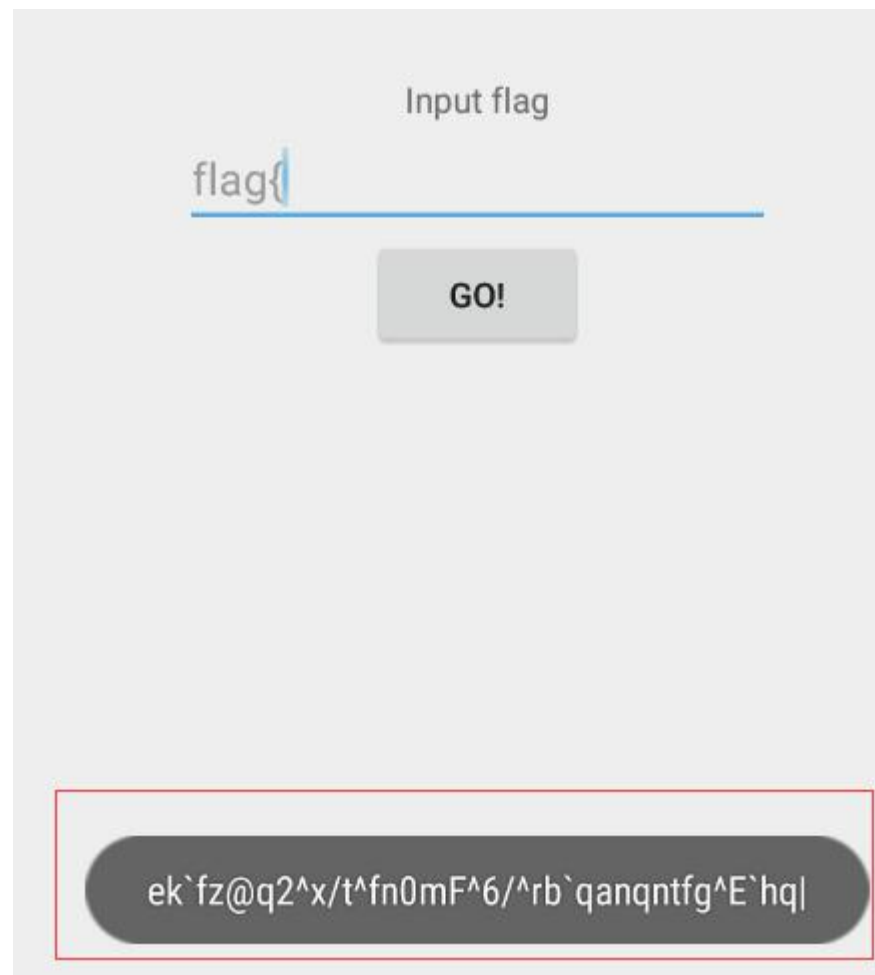
运行该程序时我们知道，如果我们的值时错误的会回显一个 Failed，我们在 MainActivity.smail 中查找到 Failed 的位置，我们要用 getFlag 的返回值替代掉 Failed 的值。

```
.line 37
:cond_0
const-string v1, "Failed"
invoke-virtual {p0}, Lcom/ph0en1x/android_crackme/MainActivity; -> getFlag()Ljava/lang/String;
move-result-object v1

invoke-static {p0, v1, v3}, Landroid/widget/Toast; -> makeText(Landroid/content/Context;Ljava/lang/CharSequence;I)L
```

我们添加的内

然后回编译，签名



最终结果和脚本

我们已经知道 `encrypt` 函数的作用是将 `string` 中的每一个字符的 `ascii` 值减一。所以我们可以编写脚本

```
str1 = "ek`fz@q2^x/t^fn0mF^6/^rb`qanqntfg^E`hq|"  
flag = ''  
for i in str1:  
    flag += chr(ord(i)+1)  
print(flag)
```

结果

```
flag{Ar3_y0u_g01nG_70_scarborough_Fair}
```