# 汇编语言与逆向技术实验报告

## Lab7 - CTF（Capture The Flag）夺旗赛

学号：2112492 姓名：刘修铭 专业：信息安全

## 一、实验目的

1.熟悉静态反汇编工具 IDA Freeware；

2.掌握对二进制代码内部逻辑关系的分析；

3.掌握对二进制代码的修改和保存。

## 二、逆向分析

### （一）主要逻辑结构

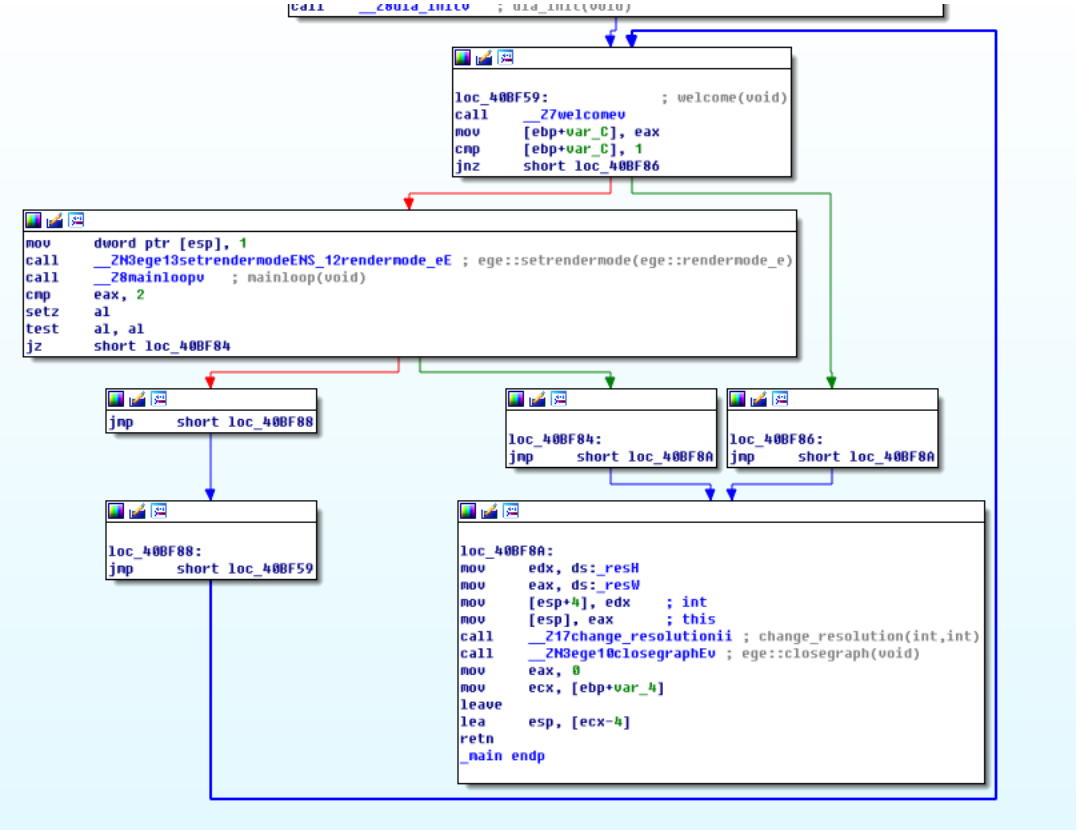
### 1.main 主函数

首先找到 main 主函数，观察程序主要结构。如图

```
; Attributes: bp-based frame

; int __cdecl main(int argc, const char **argv, const char **envp)
public _main
_main proc near

var_C= dword ptr -0Ch
var_4= dword ptr -4
argc= dword ptr  0Ch
argv= dword ptr  10h
envp= dword ptr  14h

lea     ecx, [esp+4]
and     esp, 0FFFFFFF0h
push    dword ptr [ecx-4]
push    ebp
mov     ebp, esp
push    ecx
sub     esp, 24h
call    ___main
mov     dword ptr [esp], offset aResourceSoundB ; "resource\\sound\\bgm.wma"
mov     ecx, offset _BGM
call    __ZN3ege5MUSIC8OpenFileEPKc ; ege::MUSIC::OpenFile(char const*)
sub     esp, 4
mov     eax, ds:dword_4EBC50
mov     [esp], eax      ; this
mov     ecx, offset _BGM
call    __ZN3ege5MUSIC9SetVolumeEf ; ege::MUSIC::SetVolume(float)
sub     esp, 4
mov     dword ptr [esp+4], 0FFFFFFFFh ; unsigned __int32
mov     dword ptr [esp], 0 ; this
mov     ecx, offset _BGM
call    __ZN3ege5MUSIC4PlayEmm ; ege::MUSIC::Play(ulong,ulong)
sub     esp, 8
mov     dword ptr [esp+8], 80000000h ; int
mov     dword ptr [esp+4], 80000000h ; int
mov     dword ptr [esp], 11h ; this
call    __ZN3ege11setinitmodeEiii ; ege::setinitmode(int,int,int)
mov     dword ptr [esp+4], 258h ; int
mov     dword ptr [esp], 320h ; int
call    __Z17change_resolutionii ; change_resolution(int,int)
mov     dword ptr [esp+8], 100h ; int
mov     dword ptr [esp+4], 258h ; char *
mov     dword ptr [esp], 320h ; this
call    __ZN3ege9initgraphEiii ; ege::initgraph(int,int,int)
mov     dword ptr [esp], 0 ; this
call    __ZN3ege9showmouseEi ; ege::showmouse(int)
mov     dword ptr [esp], offset String ; "CTFbug"
call    __ZN3ege10setcaptionEPKc ; ege::setcaption(char const*)
call    __ZN3ege9randomizeEv ; ege::randomize(void)
mov     dword ptr [esp], 0 ; time_t *
call    _time
mov     [esp], eax      ; unsigned int
call    _srand
call    __Z13resource_initv ; resource_init(void)
call    __Z8dia_initv   ; dia_init(void)
```

由上图可知，在 main 函数中，使用 call 指令调用了许多函数，结合其在开头的位置可知，这些函数会包含一系列的初始化函数。

后面调用 welcome 函数，进入游戏的欢迎页面。如图




接下来则主要调用循环函数 mainloop，以及进行游戏画面的切换与关闭。

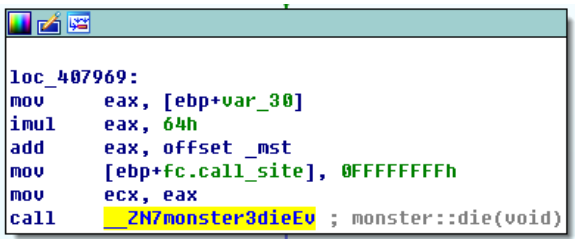
## 2.mainloop 主循环函数

mainloop 函数内部主要是一个大的循环结构，其中调用了许多重要函数跟结构。如图

```
; Attributes: bp-based frame

; _DWORD mainloop(void)
public __Z8mainloopv
__Z8mainloopv proc near

var_26C= dword ptr -26Ch
var_268= dword ptr -268h
var_264= dword ptr -264h
fc= SjLj_Function_Context ptr -260h
var_240= byte ptr -240h
var_22C= dword ptr -22Ch
var_228= dword ptr -228h
var_224= dword ptr -224h
var_220= dword ptr -220h
var_210= dword ptr -210h
```

（1）moster::die(void)

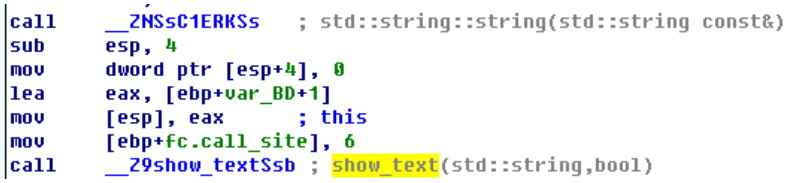
怪物死亡后，通过生成随机掉落的钻石或魔法瓶等道具，改变人物的力量等属性。



```
loc_407969:
mov     eax, [ebp+var_30]
imul    eax, 64h
add     eax, offset _mst
mov     [ebp+fc.call_site], 0FFFFFFFFh
mov     ecx, eax
call    __ZN7monster3dieEv ; monster::die(void)
```

（2）show_text(std::string,bool)

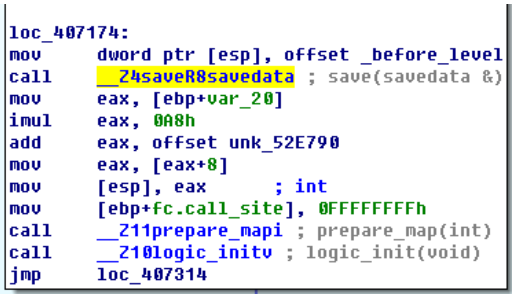
显示文本框。



```
call    __ZNSsC1ERKSs   ; std::string::string(std::string const&)
sub     esp, 4
mov     dword ptr [esp+4], 0
lea     eax, [ebp+var_BD+1]
mov     [esp], eax      ; this
mov     [ebp+fc.call_site], 6
call    __Z9show_textSsb ; show_text(std::string,bool)
```

（3）save(savedata &)

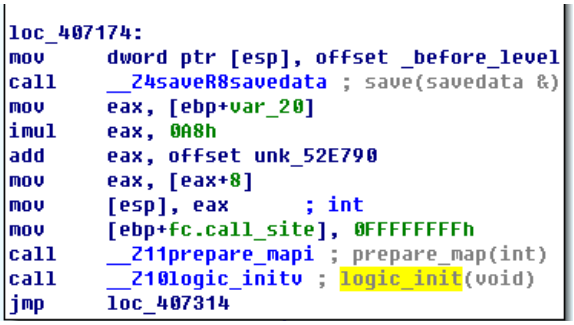
切换关卡时，保存主人公血量、技能点等属性。



```
loc_407174:
mov     dword ptr [esp], offset _before_level
call    __Z4saveR8savedata ; save(savedata &)
mov     eax, [ebp+var_20]
imul    eax, 0A8h
add     eax, offset unk_52E790
mov     eax, [eax+8]
mov     [esp], eax      ; int
mov     [ebp+fc.call_site], 0FFFFFFFFh
call    __Z11prepare_mapi ; prepare_map(int)
call    __Z10logic_initv ; logic_init(void)
jmp     loc_407314
```

（4）logic_init(void)

初始化地图内容，调用相关函数和 generate_monster(float,float,int,float,float)函数，生成不同类型的怪物。



```
loc_407174:
mov     dword ptr [esp], offset _before_level
call    __Z4saveR8savedata ; save(savedata &)
mov     eax, [ebp+var_20]
imul    eax, 0A8h
add     eax, offset unk_52E790
mov     eax, [eax+8]
mov     [esp], eax      ; int
mov     [ebp+fc.call_site], 0FFFFFFFFh
call    __Z11prepare_mapi ; prepare_map(int)
call    __Z10logic_initv ; logic_init(void)
jmp     loc_407314
```

## （二）重要数据

### 1.字符数组——存放待显示的字符串

如图

```
.rdata:004EB41C aZKeyDecrypted_ db 'Z KEY DECRYPTED. CONGRATULATIONS.',0Ah,0
.rdata:004EB41C                                         ; DATA XREF: mainloop(void)+E1↑o
.rdata:004EB43F                 align 10h
.rdata:004EB440 ; char aYouNeedToKillE[]
.rdata:004EB440 aYouNeedToKillE db 'You need to kill enough monsters!',0
.rdata:004EB440                                         ; DATA XREF: mainloop(void)+556↑o
.rdata:004EB462 ; CHAR aResourceSoundT[]
.rdata:004EB462 aResourceSoundT db 'resource\sound\tp.wav',0
.rdata:004EB462                                         ; DATA XREF: mainloop(void)+593↑o
.rdata:004EB478 ; char aZKeyDecrypting[]
.rdata:004EB478 aZKeyDecrypting db 'Z KEY DECRYPTING PROGRESS : 0%%',0
.rdata:004EB478                                         ; DATA XREF: mainloop(void)+645↑o
.rdata:004EB498 ; char aZKeyDecrypti_0[]
.rdata:004EB498 aZKeyDecrypti_0 db 'Z KEY DECRYPTING PROGRESS : 25%%',0
.rdata:004EB498                                         ; DATA XREF: mainloop(void)+668↑o
.rdata:004EB4B9                 align 4
.rdata:004EB4BC ; char aZKeyDecrypti_1[]
.rdata:004EB4BC aZKeyDecrypti_1 db 'Z KEY DECRYPTING PROGRESS : 50%%',0
.rdata:004EB4BC                                         ; DATA XREF: mainloop(void)+694↑o
.rdata:004EB4DD                 align 10h
.rdata:004EB4E0 ; char aZKeyDecrypti_2[]
.rdata:004EB4E0 aZKeyDecrypti_2 db 'Z KEY DECRYPTING PROGRESS : 75%%',0
.rdata:004EB4E0                                         ; DATA XREF: mainloop(void)+6C0↑o
.rdata:004EB501 ; char aYourHealthReco[]
.rdata:004EB501 aYourHealthReco db 'Your health recovered %d.',0
.rdata:004EB501                                         ; DATA XREF: mainloop(void)+731↑o
.rdata:004EB51B ; ege aResourceSoundH
.rdata:004EB51B aResourceSoundH db 'resource\sound\heal.wav',0
.rdata:004EB51B                                         ; DATA XREF: mainloop(void)+7C7↑o
.rdata:004EB51B                                         ; mainloop(void)+867↑o
```

### 2.重要参数

（1）主人公血量

```
.data:004E004C                 public _MAX_HP
.data:004E004C _MAX_HP         dd 0FFFFFFFFh            ; DATA XREF: save(savedata &)+18↑r
.data:004E006C _FIRE_SCOPE     dd 800000.0              ; DATA XREF: mainloop(void)+2A4B↑r
.data:004E0070                 public _INITIAL_HP
.data:004E0070 _INITIAL_HP     dd 0FFFFFFFFh            ; DATA XREF: data_init(void)+6↑r
```

（2）主人公移动速度

```
.data:004E0064                                          ; mainloop(void)+16FC↑r ...
.data:004E0068                 public _FIRE_SPEED
.data:004E0068 _FIRE_SPEED     dd 8.0                   ; DATA XREF: shot(float,float,float,float,int)+1D6↑r
.data:004E0068                                          ; shot(float,float,float,float,int)+1FE↑r
```

（3）主人公击杀怪物数量

```
.bss:00526F00                  public _kill_cnt
.bss:00526F00 _kill_cnt        dd ?                     ; DATA XREF: mainloop(void)+577↑o
.bss:00526F00                                           ; monster::die(void)+10↑r ...
```

## 三、修改代码——夺旗

### （一）实现思路

1.启动游戏后，主人公及其容易因为血量不足死亡，此时会弹出“U died”字样。

**解决方案：找到并修改主人公血量，使其血量无穷大，从而永远不会死亡。**

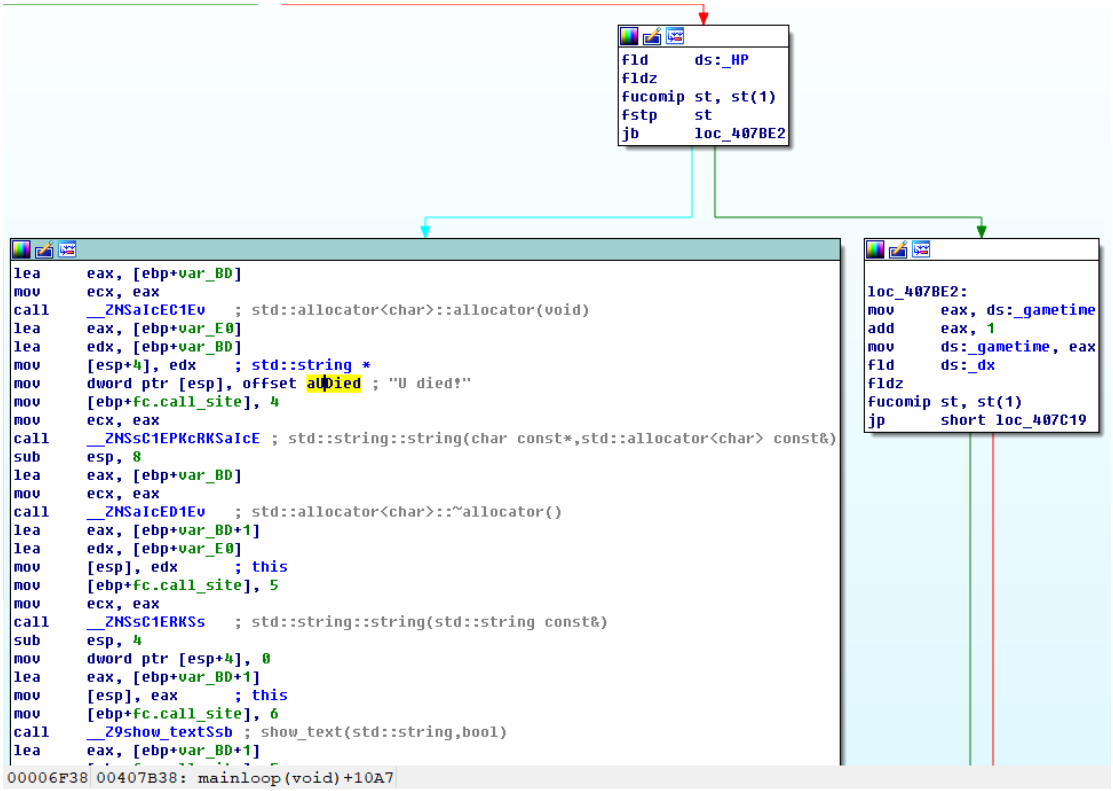
2.血量修改完毕后，继续游戏，发现游戏的通关条件是“杀死足够多的怪物”。

**解决方案：找到并修改通关条件，变为无条件通关。**

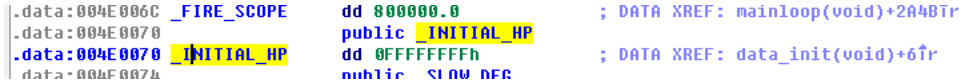
### （二）逆向修改

### 1.修改血量

寻找“U died”字符串，进而找到血量标识符。如图




在逆向得到的代码中寻找血量 HP，把原来的“_INITIAL_HP”和“_MAX_HP”修改为无穷大 0FFFFFFFFh。如图




_INITIAL_HP

```
.data:004E004C                 public _MAX_HP
.data:004E004C _MAX_HP         dd 0FFFFFFFFh                   ; DATA XREF: save(savedata &)+18↑r
```

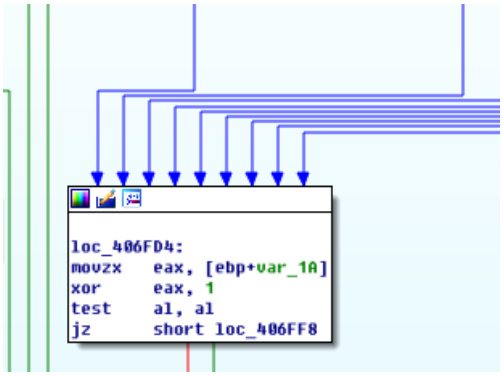
_MAX_HP

## 2.修改通关条件

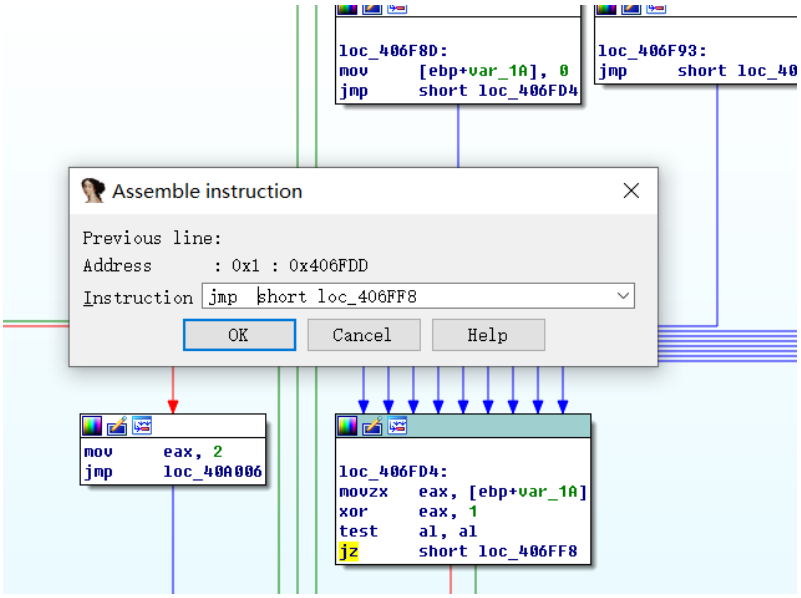
寻找“you need to kill enough monsters”的字符串，进而找到调用该字符串的代码段。如图

```
.text:00406FDF ; ---------------------------------------------------------------------------
.text:00406FDF                 mov     dword ptr [esp+4], 3Ch ; int
.text:00406FE7                 mov     dword ptr [esp], offset aYouNeedToKillE ; "You need to kill enough monsters!"
.text:00406FEE                 call    __Z5toastPci    ; toast(char *,int)
.text:00406FF3                 jmp     loc_407314
.text:00406FF8 ; ---------------------------------------------------------------------------
```

按空格键转到可视化结构图，找到跳转到该代码段的前一个代码段。如图




发现源代码中是 jz 指令，属于条件跳转。故将其修改为 jmp，设置为无条件跳转，使其能够无条件通关。如图



（三）运行游戏，夺旗