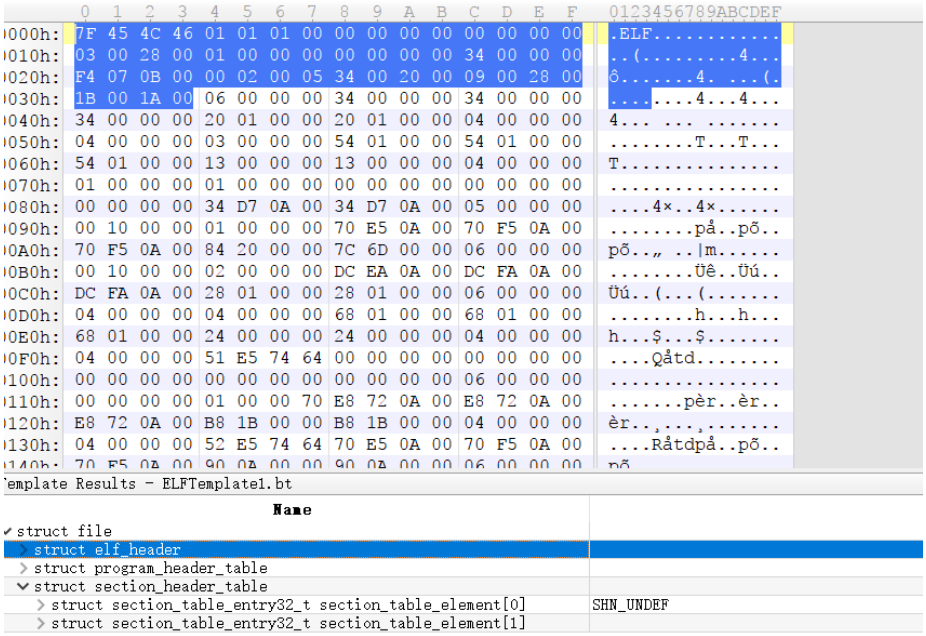
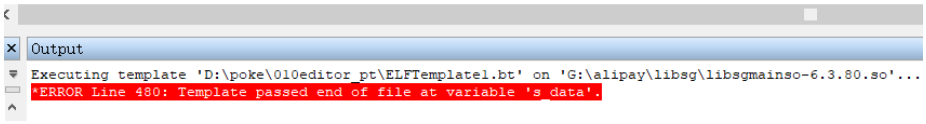
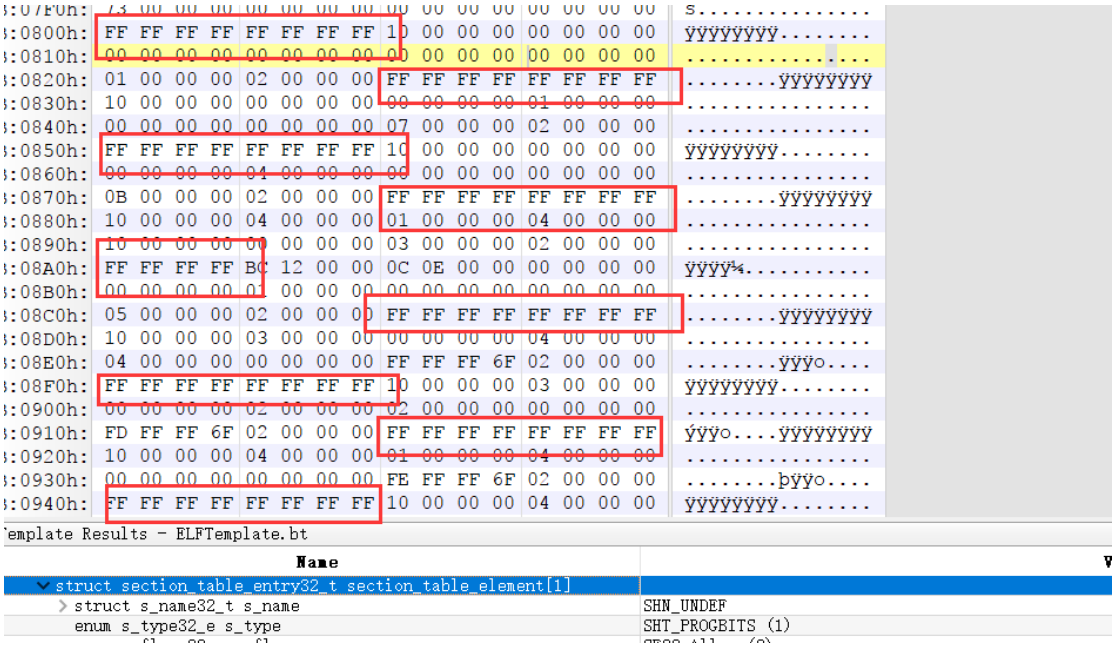
安全 sdk-so 保护

抹掉 Section 节表信息

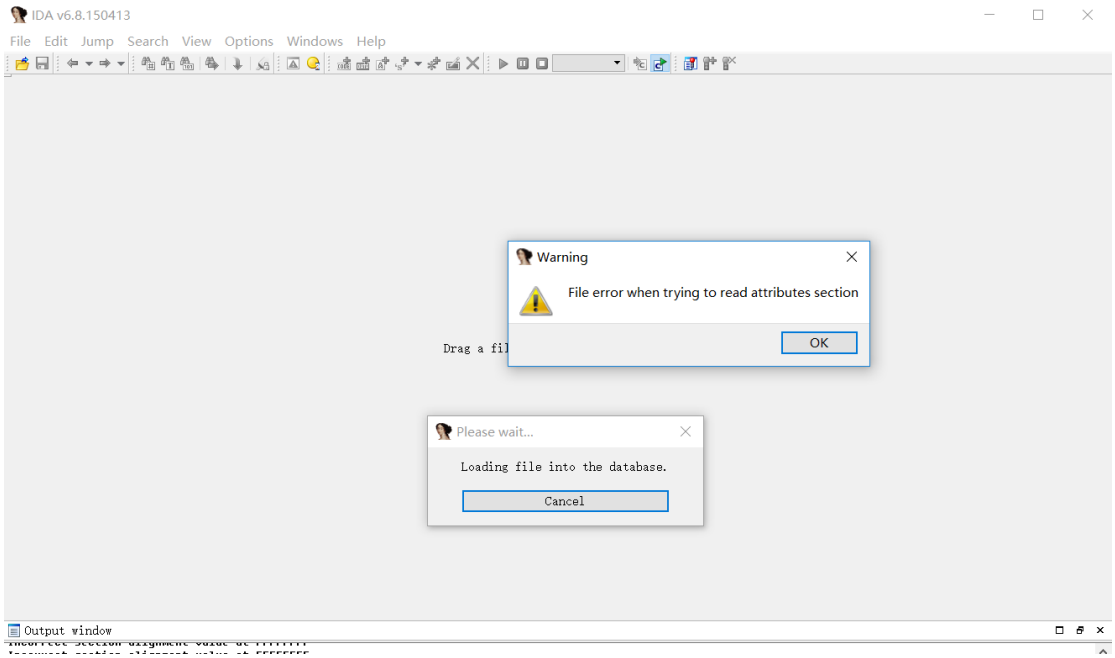




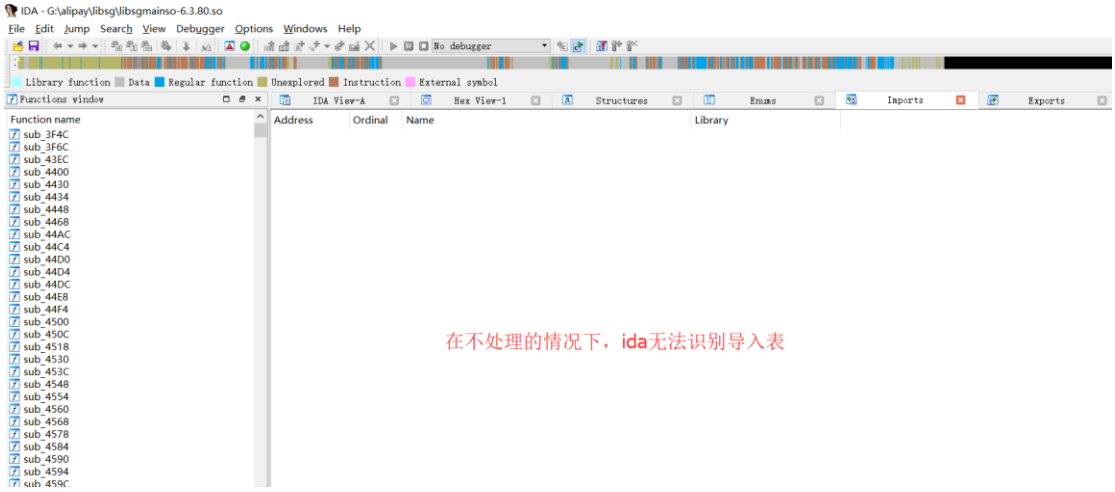
节头表信息:

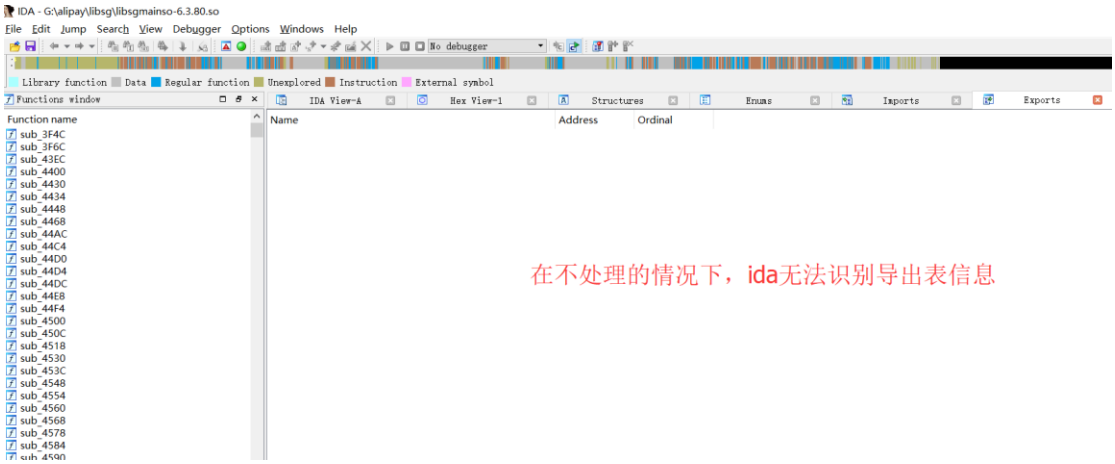


抹掉节头表导致 IDA（类型的基于节头表解析的逆向工具）加载失败



IDA 无法识别导入导出表





垃圾数据、指令

```
libsgmainso_6.3.80.so:E93DDA68
libsgmainso_6.3.80.so:E93DDA68 000 70 B5       PUSH            {R4-R6,LR}
libsgmainso_6.3.80.so:E93DDA6A 010 09 A4       ADR             R4, unk_E93DDA90
libsgmainso_6.3.80.so:E93DDA6C 010 02 25 20 35 MOVS            R5, #0x22
libsgmainso_6.3.80.so:E93DDA70 010 64 19       ADDS            R4, R4, R5
libsgmainso_6.3.80.so:E93DDA72 010 64 1C       ADDS            R4, R4, #1
libsgmainso_6.3.80.so:E93DDA74 010 01 D0       BEQ             loc_E93DDA7A
libsgmainso_6.3.80.so:E93DDA76 010 00 D1       BNE             loc_E93DDA7A
libsgmainso_6.3.80.so:E93DDA78 010 7F BD       POP             {R0-R6,PC}
libsgmainso_6.3.80.so:E93DDA7A                 ; ---------------------------------------------------------------------------
libsgmainso_6.3.80.so:E93DDA7A
libsgmainso_6.3.80.so:E93DDA7A                 loc_E93DDA7A                            ; CODE XREF: sub_E93DDA68+CTj
libsgmainso_6.3.80.so:E93DDA7A                                                         ; sub_E93DDA68+ETj
libsgmainso_6.3.80.so:E93DDA7A 010 03 94       STR             R4, [SP,#0x10+var_4]
libsgmainso_6.3.80.so:E93DDA7C 010 00 9C       LDR             R4, [SP,#0x10+var_10]
libsgmainso_6.3.80.so:E93DDA7E 010 01 9D       LDR             R5, [SP,#0x10+var_C]
libsgmainso_6.3.80.so:E93DDA80 010 6E 46       MOV             R6, SP
libsgmainso_6.3.80.so:E93DDA82 010 08 36       ADDS            R6, #8
libsgmainso_6.3.80.so:E93DDA84 010 B5 46       MOV             SP, R6
libsgmainso_6.3.80.so:E93DDA86 010 40 BD       POP             {R6,PC}
libsgmainso_6.3.80.so:E93DDA86                 ; End of function sub_E93DDA68
libsgmainso_6.3.80.so:E93DDA86
libsgmainso_6.3.80.so:E93DDA86                 ; ---------------------------------------------------------------------------
```

迷惑代码，根本不会被执行

不管是否相等，都跳转到同一个地方

```
libsgmainso_6.3.80.so:E93DDA68
libsgmainso_6.3.80.so:E93DDA68 000 70 B5       PUSH            {R4-R6,LR}
libsgmainso_6.3.80.so:E93DDA6A 010 09 A4       ADR             R4, unk_E93DDA90
libsgmainso_6.3.80.so:E93DDA6C 010 02 25 20 35 MOVS            R5, #0x22
libsgmainso_6.3.80.so:E93DDA70 010 64 19       ADDS            R4, R4, R5
libsgmainso_6.3.80.so:E93DDA72 010 64 1C       ADDS            R4, R4, #1
libsgmainso_6.3.80.so:E93DDA74 010 01 D0       BEQ             loc_E93DDA7A
libsgmainso_6.3.80.so:E93DDA76 010 00 D1       BNE             loc_E93DDA7A
libsgmainso_6.3.80.so:E93DDA76                 ; ---------------------------------------------------------------------------
libsgmainso_6.3.80.so:E93DDA78 010 7F BD       DCW 0xBD7F
libsgmainso_6.3.80.so:E93DDA7A                 ; ---------------------------------------------------------------------------
libsgmainso_6.3.80.so:E93DDA7A
libsgmainso_6.3.80.so:E93DDA7A                 loc_E93DDA7A                            ; CODE XREF: sub_E93DDA68+CTj
libsgmainso_6.3.80.so:E93DDA7A                                                         ; sub_E93DDA68+ETj
libsgmainso_6.3.80.so:E93DDA7A 010 03 94       STR             R4, [SP,#0x10+var_4]
libsgmainso_6.3.80.so:E93DDA7C 010 00 9C       LDR             R4, [SP,#0x10+var_10]
libsgmainso_6.3.80.so:E93DDA7E 010 01 9D       LDR             R5, [SP,#0x10+var_C]
libsgmainso_6.3.80.so:E93DDA80 010 6E 46       MOV             R6, SP
libsgmainso_6.3.80.so:E93DDA82 010 08 36       ADDS            R6, #8
libsgmainso_6.3.80.so:E93DDA84 010 B5 46       MOV             SP, R6
libsgmainso_6.3.80.so:E93DDA86 010 40 BD       POP             {R6,PC}
libsgmainso_6.3.80.so:E93DDA86                 ; End of function sub_E93DDA68
libsgmainso_6.3.80.so:E93DDA86
```

应该这样，条件为真

类似的迷惑代码

```
sgmainso_6.3.80.so:EF416A88 010 97 46 04 B5 DCD 0xB5044697
sgmainso_6.3.80.so:EF416A8C 010 07 BD C0 46 DCD 0x46C0BD07
sgmainso_6.3.80.so:EF416A90                 ; ---------------------------------------------------------------------------
sgmainso_6.3.80.so:EF416A90
sgmainso_6.3.80.so:EF416A90                 loc_EF416A90                            ; DATA XREF: JNI_OnLoad+2To
sgmainso_6.3.80.so:EF416A90 010 11 60       STR             R1, [R2]
sgmainso_6.3.80.so:EF416A92 010 70 BD       POP             {R4-R6,PC}
sgmainso_6.3.80.so:EF416A92                 ; End of function JNI_OnLoad
sgmainso_6.3.80.so:EF416A92
sgmainso_6.3.80.so:EF416A94                 ; ---------------------------------------------------------------------------
sgmainso_6.3.80.so:EF416A94 11 60           STR             R1, [R2]
sgmainso_6.3.80.so:EF416A96 08 BD           POP             {R3,PC}
sgmainso_6.3.80.so:EF416A98                 ; ---------------------------------------------------------------------------
sgmainso_6.3.80.so:EF416A98 B7 46           MOV             PC, R6
sgmainso_6.3.80.so:EF416A9A                 ; ---------------------------------------------------------------------------
sgmainso_6.3.80.so:EF416A9A 0C B5           PUSH            {R2,R3,LR}
sgmainso_6.3.80.so:EF416A9C 0B 00           MOVS            R3, R1
sgmainso_6.3.80.so:EF416A9E 0C B5           PUSH            {R2,R3,LR}
sgmainso_6.3.80.so:EF416AA0 F0 47           BLX             LR
sgmainso_6.3.80.so:EF416AA0                 ; ---------------------------------------------------------------------------
sgmainso_6.3.80.so:EF416AA2                 CODE32
sgmainso_6.3.80.so:EF416AA2 38              DCB 0x38 ; 8
sgmainso_6.3.80.so:EF416AA3 BD              DCB 0xBD ;
sgmainso_6.3.80.so:EF416AA4 11              DCB 0x11
```

垃圾数据，迷惑反汇编器

迷惑代碼:

```
SUB             SP, SP, #8
SUB             SP, SP, #8
PUSH            {R0,R1,LR}
BLX             sub_4964
LSLS            R2, R5, #3
MOVS            R0, R0
POP             {R1}
CMP             R0, #0
ADD             R1, PC
LDR             R1, [R1]
PUSH            {R0,R1,LR}
```

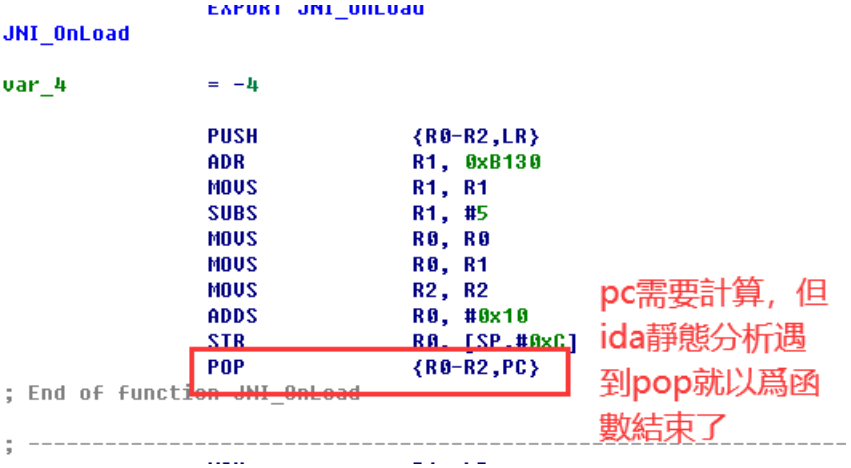
混淆指令

另一種迷惑代碼:

```
ar_4            = -4

                PUSH            {R0-R2,LR}
                ADR             R1, 0xB130
                MOVS            R1, R1
                SUBS            R1, #5
                MOVS            R0, R0
                MOVS            R0, R1
                MOVS            R2, R2
                ADDS            R0, #0x10
                STR             R0, [SP,#0xC]
                POP             {R0-R2,PC}
End of function JNI_OnLoad

-----------------------------------------------------------------------------

                MOV             R1, LR
                POP             {R5,R6,PC}
-----------------------------------------------------------------------------
                PUSH            {R2,R3}
                BLX             loc_B130
                NOP

oc_B130                                 ; CODE XREF: LOAD:0000
                MOVS            R3, R3
                POP             {R2,PC}
-----------------------------------------------------------------------------
                PUSH            {R1,R2}
                MOV             PC, R4
-----------------------------------------------------------------------------
                MOVS            R0, R2
                PUSH            {R0,R1,LR}
```

垃圾指令，擾亂ida

擾亂 ida 分析



函數之前插入計算 pc 和垃圾數據，擾亂 ida

```
                PUSH            {R0-R2,LR}
                ADR             R1, 0x18FF8
                MOVS            R1, R1
                SUBS            R1, #5
                MOVS            R0, R0
                MOVS            R0, R1
                MOVS            R2, R2
                ADDS            R0, #0x28 ; '('
                STR             R0, [SP,#0xC]
                POP             {R0-R2,PC}
End of function sub_18FD8

-------------------------------------------------------------------------------

                POP             {R2-R5,PC}
-------------------------------------------------------------------------------

                PUSH            {R1,R5}
                MOV             R4, LR
                BLX             loc_18FF8
                NOP

oc_18FF8                                ; CODE XREF: LOAD:00018FF2↑p
                CMP             R3, #5
                MOV             PC, R1
-------------------------------------------------------------------------------

                POP             {R1,R4-R6,PC}
-------------------------------------------------------------------------------

                MOV             PC, R0
-------------------------------------------------------------------------------

                MOV             R5, LR
                STR             R1, [R2]
                MOV             PC, R1
-------------------------------------------------------------------------------

                MOV             PC, R0
-------------------------------------------------------------------------------

                MOV             PC, R6
-------------------------------------------------------------------------------

                MOV             LR, R3
                POP             {R4,R5,PC}
-------------------------------------------------------------------------------

                MOVS            R0, R1
                MOV             R6, LR
                PUSH            {R2,R4}
                POP             {R5,PC}
-------------------------------------------------------------------------------

                POP             {R2-R5,PC}
-------------------------------------------------------------------------------

                MOVS            R0, R1
                PUSH.W          {R4-R8,LR}
```

pc需要計算，不是顯示的，擾亂ida分析，并且在函數之前，在函數之前還插入了一些垃圾數據做指令，擾亂ida分析

真正函數開始

pop xxx ， push xxx 垃圾數據指令

```
-------------------------------------------------------------------------------

                POP             {R0,PC}
-------------------------------------------------------------------------------

                POP             {R0,PC}
-------------------------------------------------------------------------------

                PUSH            {R0,R2}
                NOP
                BX              R6
-------------------------------------------------------------------------------

                BX              R4
-------------------------------------------------------------------------------

                PUSH            {R1,R3}
                MOV             PC, R6
-------------------------------------------------------------------------------

                LDR             R4, [R5]
                PUSH            {R1,R2}
                MOVS            R0, R5
                PUSH            {R0,R5}
                POP             {R1-R3,PC}
```

無用的垃圾數據，特徵
都是push xxx,
pop xxx這個樣子

动态计算目标地址

```
libsgmainso_6.3.80.so:EED8D848                                          ; libsgmainso_6.3.80.so:off_EEE2CAE4↓o
libsgmainso_6.3.80.so:EED8D848 01 10 CE E3    BIC           R1, LR, #1
libsgmainso_6.3.80.so:EED8D84C 00 01 91 E7    LDR           R0, [R1,R0,LSL#2]
libsgmainso_6.3.80.so:EED8D850 0E E0 80 E0    ADD           LR, R0, LR
libsgmainso_6.3.80.so:EED8D854 08 10 9D E5    LDR           R1, [SP,#8]
libsgmainso_6.3.80.so:EED8D858 08 E0 8D E5    STR           LR, [SP,#8]
libsgmainso_6.3.80.so:EED8D85C 01 E0 A0 E1    MOV           LR, R1
libsgmainso_6.3.80.so:EED8D860 03 80 BD E8    LDMFD         SP!, {R0,R1,PC}
libsgmainso_6.3.80.so:EED8D864                ; ---------------------------------------------------------------------------
libsgmainso_6.3.80.so:EED8D864 0E 10 A0 E1    MOV           R1, LR
libsgmainso_6.3.80.so:EED8D868 A1 10 A0 E1    MOV           R1, R1,LSR#1
libsgmainso_6.3.80.so:EED8D86C 81 10 A0 E1    MOV           R1, R1,LSL#1
libsgmainso_6.3.80.so:EED8D870 01 00 A0 E1    MOV           R0, R1
libsgmainso_6.3.80.so:EED8D874 00 10 91 E5    LDR           R1, [R1]
libsgmainso_6.3.80.so:EED8D878 00 10 81 E0    ADD           R1, R1, R0
libsgmainso_6.3.80.so:EED8D87C 00 00 91 E5    LDR           R0, [R1]
libsgmainso_6.3.80.so:EED8D880 10 00 8D E5    STR           R0, [SP,#0x10]
libsgmainso_6.3.80.so:EED8D884 04 E0 8E E2    ADD           LR, LR, #4
libsgmainso_6.3.80.so:EED8D888 0C E0 8D E5    STR           LR, [SP,#0xC]
libsgmainso_6.3.80.so:EED8D88C 03 40 BD E8    LDMFD         SP!, {R0,R1,LR}
libsgmainso_6.3.80.so:EED8D890 04 F0 9D E4    LDR           PC, [SP],#4
libsgmainso_6.3.80.so:EED8D894                ; ---------------------------------------------------------------------------
libsgmainso_6.3.80.so:EED8D894 03 00 2D E9    STMFD         SP!, {R0,R1}
libsgmainso_6.3.80.so:EED8D898 0E 10 A0 E1    MOV           R1, LR
libsgmainso_6.3.80.so:EED8D89C A1 10 A0 E1    MOV           R1, R1,LSR#1
libsgmainso_6.3.80.so:EED8D8A0 00 01 A0 E1    MOV           R0, R0,LSL#2
libsgmainso_6.3.80.so:EED8D8A4 81 10 A0 E1    MOV           R1, R1,LSL#1
libsgmainso_6.3.80.so:EED8D8A8 00 10 91 E7    LDR           R1, [R1,R0]
libsgmainso_6.3.80.so:EED8D8AC 81 10 A0 E1    MOV           R1, R1,LSL#1
```

很多，这里只是截三幅图

这样会导致调用不连续，只有计算出 pc 值才能分析清楚程序走向。
pc 非顯示的

```
                EXPORT JNI_OnLoad
JNI_OnLoad

var_4           = -4

                PUSH            {R0-R2,LR}
                ADR             R1, 0xB130
                MOVS            R1, R1
                SUBS            R1, #5
                MOVS            R0, R0
                MOVS            R0, R1
                MOVS            R2, R2
                ADDS            R0, #0x10
                STR             R0, [SP,#0xC]
                POP             {R0-R2,PC}
; End of function JNI_OnLoad

; ---------------------------------------------------------------------------
```

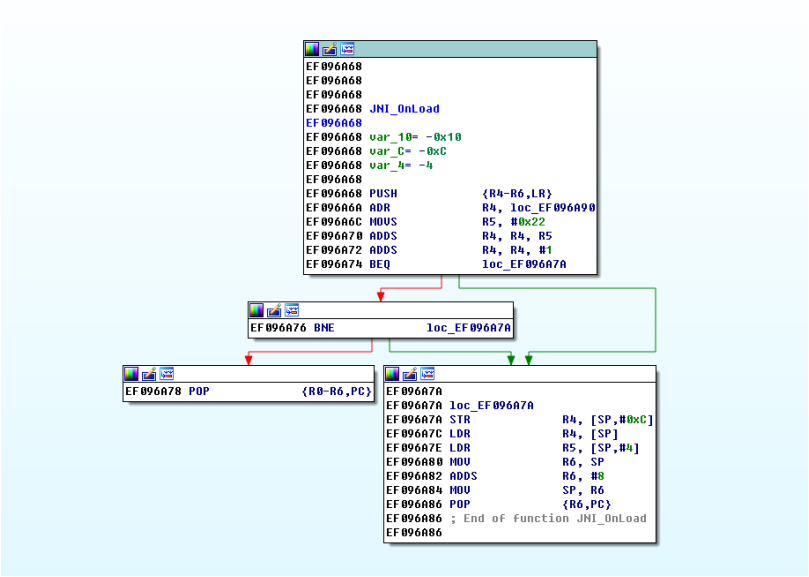
pc需要計算，但ida靜態分析遇到pop就以爲函數結束了

Ida 无法解析出正确的流程图和参数（垃圾代码）

下面的 JNI_OnLoad 是我动态调试找到的，但是 ida 明显解析错了参数，和函数流程。

```
void __fastcall JNI_OnLoad(int a1, int a2, int a3, int a4, int a5, int a6, int a7, int a8)
{
  int v8; // [sp+4h] [bp-Ch]@0

  if ( &loc_EF096AB2 )
  {
    if ( !&loc_EF096AB2 )
      JUMPOUT(__CS__, a8);
  }
  JUMPOUT(__CS__, v8);
}
```

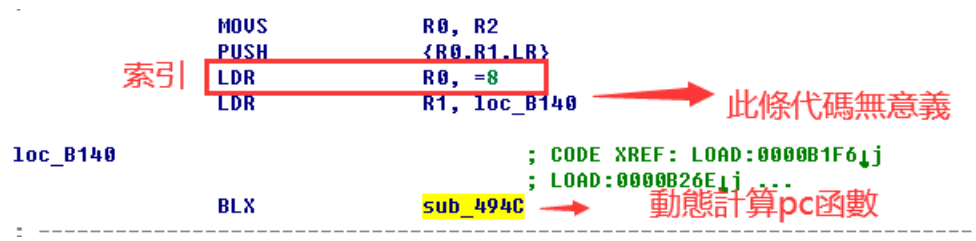
流程图：



这导致你必须单步调试，一边调试，一边帮助 ida 修正错误。

非顯示 pc，動態跳轉

pc 是非顯示的，需要進行查表計算；同時有多種計算 pc 的方式，其中一種跳轉特徵：



跳轉表：

```
LOAD:0000B140                 BLX             sub_494C
LOAD:0000B140 ; ---------------------------------------------------------------------------
LOAD:0000B144 dword_B144      DCD 8                   ; DATA XREF: LOAD:0000B13C↑r
LOAD:0000B148                 DCD 0xA8
LOAD:0000B14C                 DCD 0xBC
LOAD:0000B150                 DCD 0xCC
LOAD:0000B154                 DCD 0xE0
LOAD:0000B158                 DCD 0xF0
LOAD:0000B15C                 DCD 0x100
LOAD:0000B160                 DCD 0x110
LOAD:0000B164                 DCD 0x120
LOAD:0000B168                 DCD 0x134
LOAD:0000B16C                 DCD 0x14C
LOAD:0000B170                 DCD 0x168
LOAD:0000B174                 DCD 0x17C
LOAD:0000B178                 DCD 0x194
LOAD:0000B17C                 DCD 0x1AC
LOAD:0000B180                 DCD 0x1D4
LOAD:0000B184                 DCD 0x1EC
LOAD:0000B188                 DCD 0x208
LOAD:0000B18C                 DCD 0x224
LOAD:0000B190                 DCD 0x240
LOAD:0000B194                 DCD 0x268
LOAD:0000B198                 DCD 0x27C
LOAD:0000B19C                 DCD 0x290
LOAD:0000B1A0                 DCD 0x2A4
LOAD:0000B1A4                 DCD 0x2B8
LOAD:0000B1A8                 DCD 0x2CC
LOAD:0000B1AC                 DCD 0x2E0
LOAD:0000B1B0                 DCD 0x2FC
LOAD:0000B1B4                 DCD 0x310
LOAD:0000B1B8                 DCD 0x324
LOAD:0000B1BC                 DCD 0x33C
LOAD:0000B1C0                 DCD 0x358
```

另一種跳轉特徵:

```
; ---------------------------------------------------------------------------
                SUB             SP, SP, #8
                PUSH            {R0,R1,LR}
                BLX             sub_4964
                LSLS            R2, R5, #3
                MOVS            R0, R0
                POP             {R1}
                CMP             R0, #0
                ADD             R1, PC
                LDR             R1, [R1]
                PUSH            {R0,R1,LR}
```

字符串加密

幾乎所有 class 的字符串都被加密了

多種加密算法

該 so 用到了多種加密算法，如 rc4， aes 等，而且為了防止這些加密算法被直接識別，它還被稍作了修改。

补充

由于写这些文档时已经停止了分析工作，so 保护应该还有我没有发现的部分，此文档只是提供我发现的保护措施供大家分享。