OLLVM 混淆相关知识

编译

llvm 混淆开源框架 ollvm 支更新到 llvm4.0，下载略。

```
mkdir build

cd build

如果不想跑测试用例加上-DLLVM_INCLUDE_TESTS=OFF

cmake -DCMAKE_BUILD_TYPE=Release -DLLVM_CREATE_XCODE_TOOLCHAIN=ON ../ obfuscator-llvm-4.0/

make -j4
```

使用

这里原版提供了 3 种混淆方式分别是控制流扁平化,指令替换,虚假控制流程,用起来都是加 cflags 的方式。下面简单说下这几种模式。

控制流扁平化

这个模式主要是把一些 if-else 语句，嵌套成 do-while 语句

-mllvm -fla：激活控制流扁平化
-mllvm -split：激活基本块分割。在一起使用时改善展平。
-mllvm -split_num=3：如果激活了传递，则在每个基本块上应用 3 次。默认值：1

bcf 可以配合下面参数使用

-mllvm -perBCF=20: 对所有函数都混淆的概率是 20%，默认 100%

-mllvm -boguscf-loop=3: 对函数做 3 次混淆，默认 1 次

-mllvm -boguscf-prob=40: 代码块被混淆的概率是 40%，默认 30%

指令替换

这个模式主要用功能上等效但更复杂的指令序列替换标准二元运算符(+ , – , & , | 和 ^)

-mllvm -sub：激活指令替换
-mllvm -sub_loop=3：如果激活了传递，则在函数上应用 3 次。默认值：1

虚假控制流程

这个模式主要嵌套几层判断逻辑，一个简单的运算都会在外面包几层 if-else，所以这个模式加上编译速度会慢很多因为要做几层假的逻辑包裹真正有用的代码。

另外说一下这个模式编译的时候要浪费相当长时间包哪几层不是闹得!

-mllvm -bcf：激活虚假控制流程
-mllvm -bcf_loop=3：如果激活了传递，则在函数上应用 3 次。默认值：1
-mllvm -bcf_prob=40：如果激活了传递，基本块将以 40％的概率进行模糊处理。默认值：30

上面说完模式下面讲一下几种使用方式

直接用二进制文件

直接使用编译的二进制文件 build/bin/clang test.c -o test -mllvm -sub -mllvm -fla -mllvm -bcf

NDK 集成

这里分为工具链的制作和项目里的配置。

制作 Toolchains

n16

将 build/bin 里的 clang++ 、clang、clang-format、 clang-4.0、拷贝到 android-ndk-r16b/toolchains/llvm/prebuilt/linux-x86_64/bin（覆盖前请自行备份） 。 在 CMakeLists.txt 里的 SET(CMAKE_CXX_FLAGS 追加 -mllvm -fla ， CMAKE_C_FLAGS 也可以加，然后 重新编译你的项目。（我这里是用 cmake 编译， 如果你是 ndk-build 那么你可以在 LOCAL_CFLAGS、LOCAL_CPPFLAGS 里添加）

n18

老的 ndk 版本比这更容易都在 ndk-bundle/toolchains 里放着需要修改的文件。

```
复制 ndk 的 toolschain 里的 llvm

cp -r ndk-bundle/toolchains/llvm ndk-bundle/toolchains/ollvm

删除 prebuilt 文件夹下的文件夹的 bin 和 lib64，prebuilt 文件夹下根据系统不同命名也不同

rm -rf ndk-bundle/toolchains/ollvm/prebuilt/darwin-x86_64/bin

rm -rf ndk-bundle/toolchains/ollvm/prebuilt/darwin-x86_64/lib64

把我们之前编译好的 ollvm 下的 bin 和 lib 移到我们刚才删除 bin 和 lib64 的目录下

mv build/bin ndk-bundle/toolchains/ollvm/prebuilt/darwin-x86_64/

mv build/lib ndk-bundle/toolchains/ollvm/prebuilt/darwin-x86_64/

复制 ndk-bundle/build/core/toolchains 的文件夹，这里根据自己对 CPU 架构的需求自己复制然后修改

cp -r ndk-bundle/build/core/toolchains/arm-linux-androideabi-clang ndk-bundle/build/core/toolchains/arm-linux-androideabi-clang-ollvm
```

最后把 arm-linux-androideabi-clang-ollvm 里的 setup.mk 文件进行修改

```
TOOLCHAIN_NAME := ollvm
```

```
TOOLCHAIN_ROOT := $(call get-toolchain-root,$(TOOLCHAIN_NAME))

TOOLCHAIN_PREFIX := $(TOOLCHAIN_ROOT)/bin
```

config.mk 里是 CPU 架构,刚才是复制出来的所以不用修改，但如果要添加其他的自定义架构需要严格按照格式规范命名最初的文件夹，如 mips 的需要添加文件夹 mipsel-linux-android-clang-ollvm，setup.mk 和刚才的修改一样即可。

项目中配置

到了项目里还需要修改两个文件:
在 Android.mk 中添加混淆编译参数

```
LOCAL_CFLAGS += -mllvm -sub -mllvm -bcf -mllvm -fla
```

Application.mk 中配置 NDK_TOOLCHAIN_VERSION

#根据需要添加

```
根据需要添加

APP_ABI := x86 armeabi-v7a x86_64 arm64-v8a mips armeabi mips64

使用刚才我们做好的编译链

NDK_TOOLCHAIN_VERSION := ollvm
```

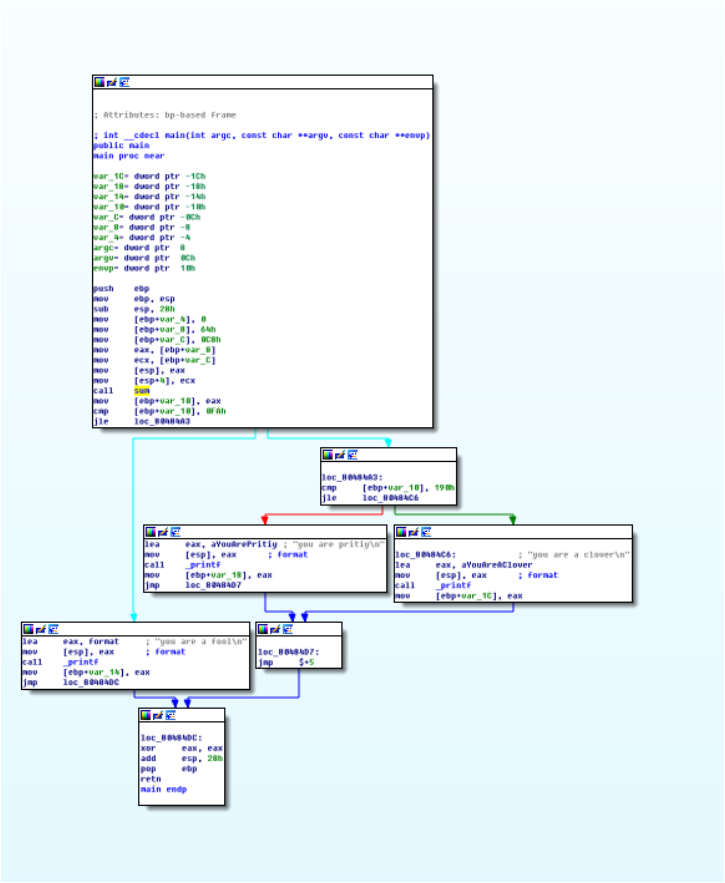
demo 演示

原程序

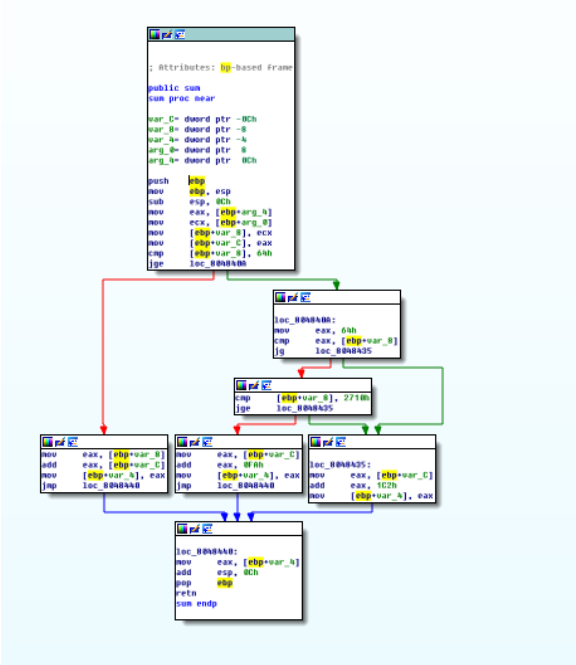
```c
#include <stdlib.h>
#include <stdio.h>

int sum(int a, int b) {
        if(a < 100) {
                return a + b;
        } else if(100 <=  a && a  < 10000) {
                return 250 + b;
        } else {
                return 450 + b;
        }
}

int main() {



        int a = 100;
        int b =200;
        int c  = sum(a, b);


        if (c > 250) {

                printf("you are a fool\n");
        } else if(c > 400) {
                printf("you are pretity\n");
        } else {
                printf("you are a clever\n");
        }

        return 0;
}
```

编译后控制流图

main 函数



sum 函数



ida f5 图

main 函数

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int v4; // [sp+18h] [bp-10h]@1

  v4 = sum(100, 200);
  if ( v4 <= 250 )
  {
    if ( v4 <= 400 )
      printf("you are a clover\n");
    else
      printf("you are pritiy\n");
  }
  else
  {
    printf("you are a fool\n");
  }
  return 0;
}
```

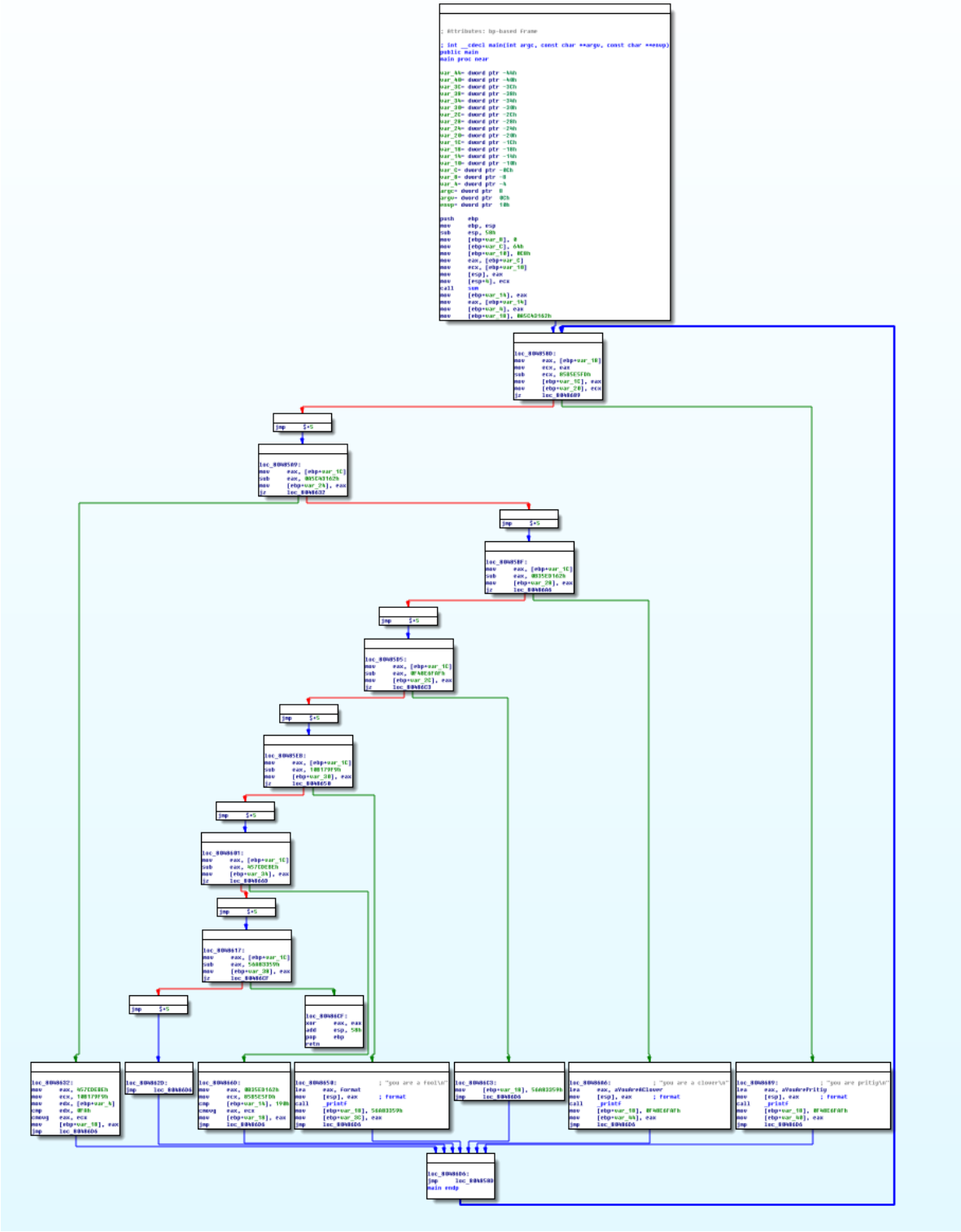
sum 函数

```
int __cdecl sum(signed int a1, int a2)
{
  int v3; // [sp+8h] [bp-4h]@2

  if ( a1 >= 100 )
  {
    if ( a1 < 100 || a1 >= 10000 )
      v3 = a2 + 450;
    else
      v3 = a2 + 250;
  }
  else
  {
    v3 = a2 + a1;
  }
  return v3;
}
```

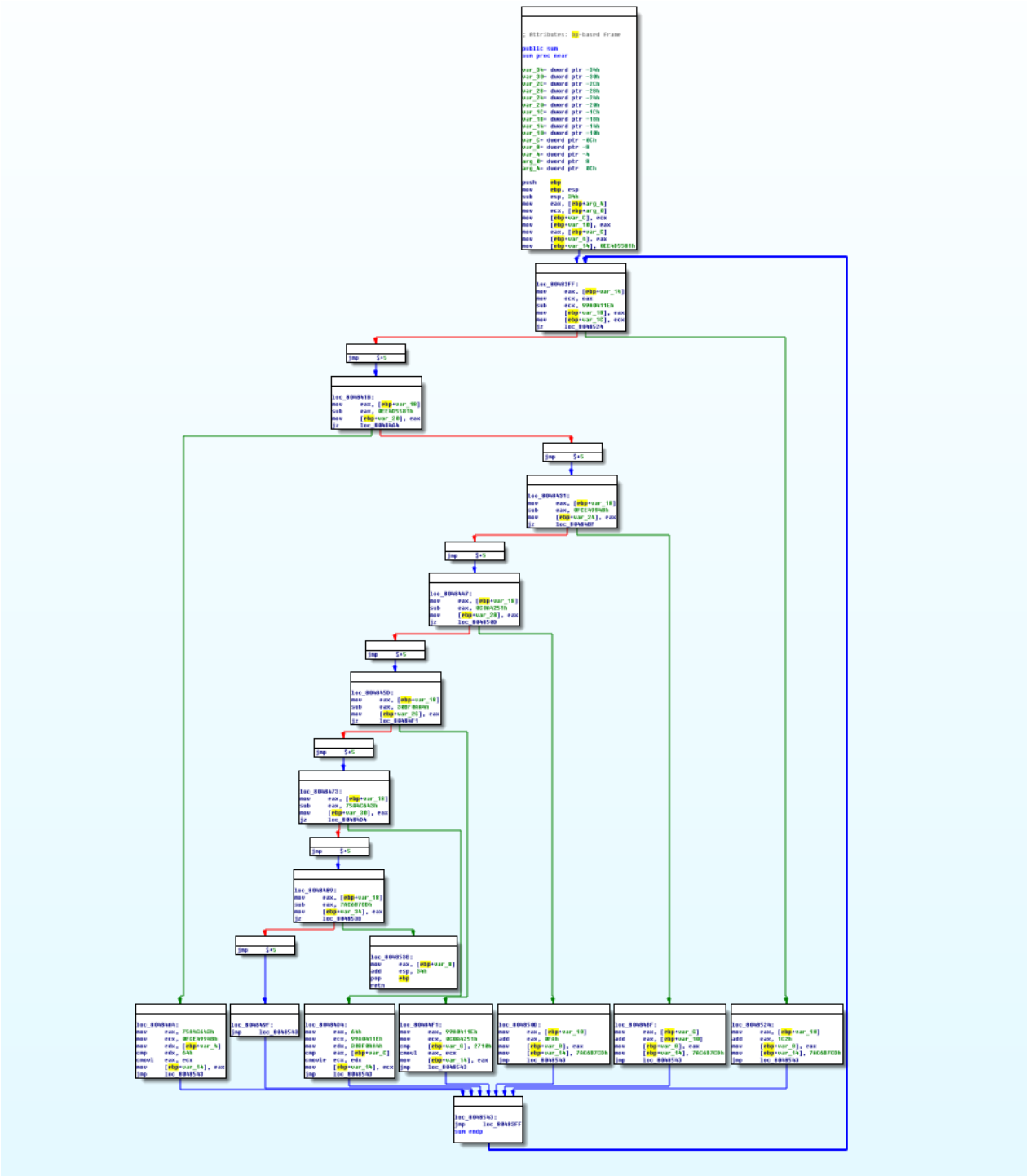
加平坦化效果

编译后控制流图

main 函数



sum 函数



ida f5 图

main 函数

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  signed int v3; // eax@10
  signed int v4; // eax@14
  signed int v6; // [sp+40h] [bp-18h]@1
  signed int v7; // [sp+44h] [bp-14h]@1

  v7 = sum(100, 200);
  v6 = -1513868958;
  do
  {
    while ( 1 )
    {
      while ( 1 )
      {
        while ( 1 )
        {
          while ( 1 )
          {
            while ( 1 )
            {
              while ( v6 == -2051676675 )
              {
                v6 = -191991889;
                printf("you are pritiy\n");
              }
              if ( v6 != -1513868958 )
                break;
              v3 = 1165811390;
              if ( v7 > 250 )
                v3 = 280066553;
              v6 = v3;
            }
            if ( v6 != -1285631646 )
              break;
            v6 = -191991889;
            printf("you are a clover\n");
          }
          if ( v6 != -191991889 )
            break;
          v6 = 1453863769;
        }
        if ( v6 != 280066553 )
          break;
        v6 = 1453863769;
        printf("you are a fool\n");
      }
      if ( v6 != 1165811390 )
        break;
      v4 = -1285631646;
      if ( v7 > 400 )
        v4 = -2051676675;
      v6 = v4;
    }
  }
  while ( v6 != 1453863769 );
  return 0;
}
```

sum 函数

```
int __cdecl sum(signed int a1, int a2)
{
  signed int v2; // eax@10
  signed int v3; // ecx@14
  signed int v4; // eax@17
  signed int v6; // [sp+20h] [bp-14h]@1
  int v7; // [sp+2Ch] [bp-8h]@0

  v6 = -296921727;
  do
  {
    while ( 1 )
    {
      while ( 1 )
      {
        while ( 1 )
        {
          while ( 1 )
          {
            while ( 1 )
            {
              while ( 1 )
              {
                while ( v6 == -1717550818 )
                {
                  v7 = a2 + 450;
                  v6 = 2059843533;
                }
                if ( v6 != -296921727 )
                  break;
                v2 = 1973732931;
                if ( a1 < 100 )
                  v2 = -52127413;
                v6 = v2;
              }
              if ( v6 != -52127413 )
                break;
              v7 = a2 + a1;
              v6 = 2059843533;
            }
            if ( v6 != 201998929 )
              break;
            v7 = a2 + 250;
            v6 = 2059843533;
          }
          if ( v6 != 817826468 )
            break;
          v4 = -1717550818;
          if ( a1 < 10000 )
            v4 = 201998929;
          v6 = v4;
        }
        if ( v6 != 1973732931 )
          break;
        v3 = -1717550818;
        if ( a1 >= 100 )
          v3 = 817826468;
        v6 = v3;
      }
    }
    while ( v6 != 2059843533 );
    return v7;
}
```

指令替换

这里指演示替换 sub 指令

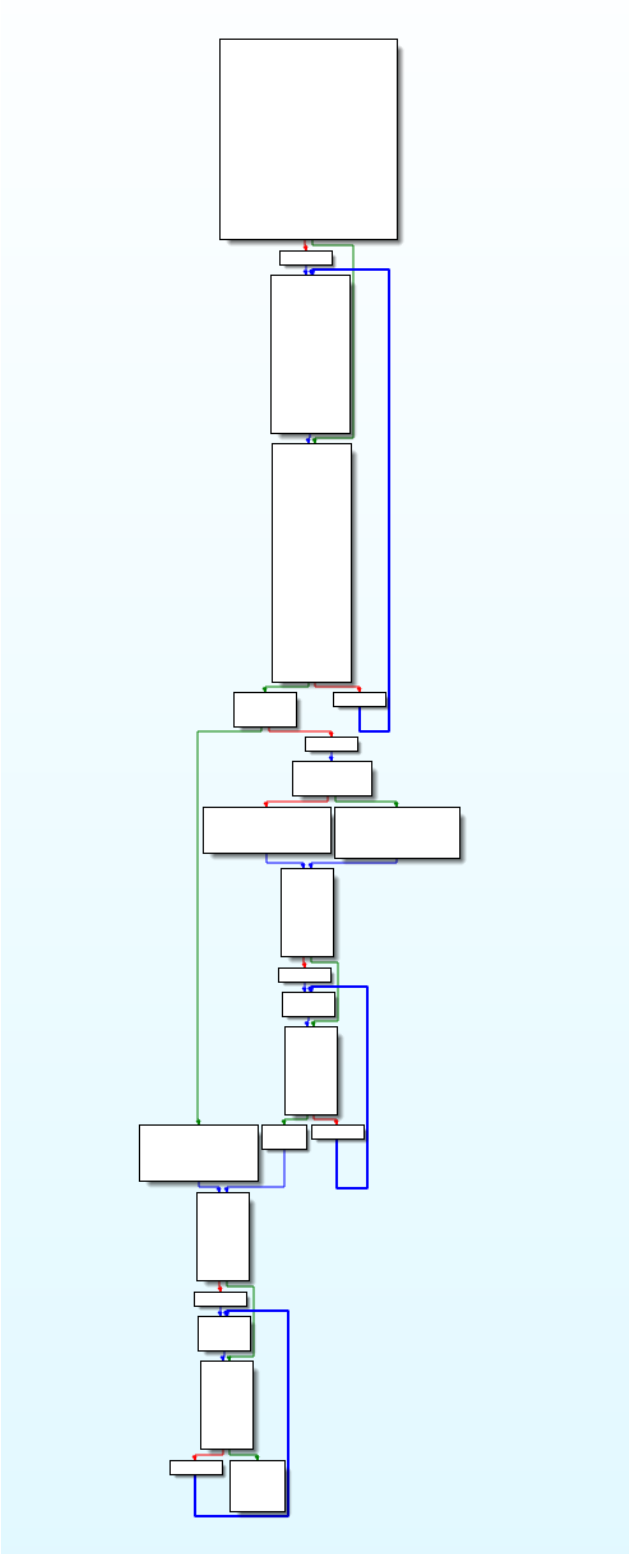
编译后的控制流图不变，f5 后因为只有 sum 函数存在加减运算，所以 sum 函数有变化

```
int __cdecl sum(signed int a1, int a2)
{
  int v3; // [sp+8h] [bp-8h]@2

  if ( a1 >= 100 )
  {
    if ( a1 < 100 || a1 >= 10000 )
      v3 = a2 - 156844907 + 156845357;
    else
      v3 = -(-250 - a2);
  }
  else
  {
    v3 = -(-a1 - a2);
  }
  return v3;
}
```

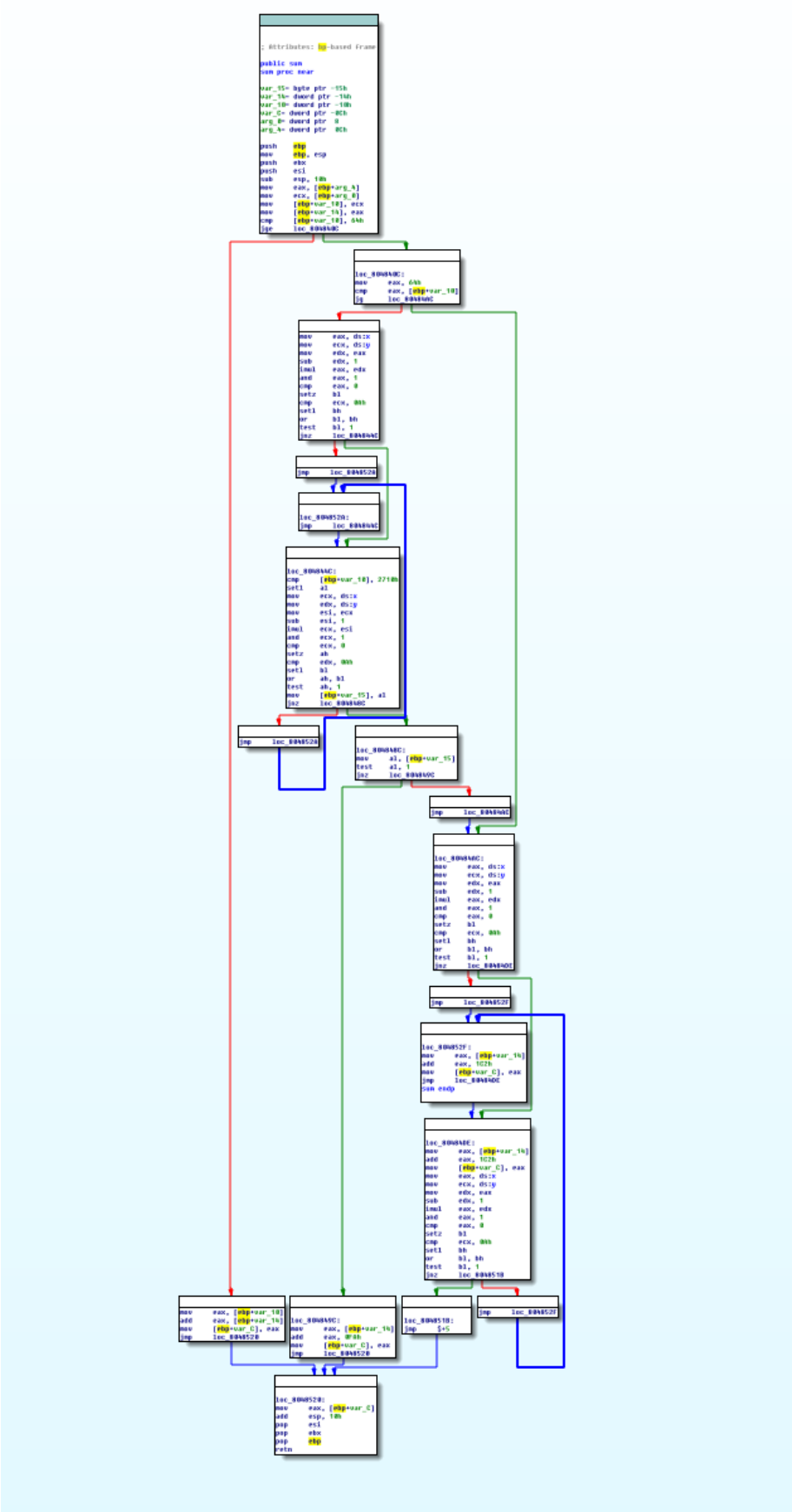
虚假控制流

编译后控制流图

main 函数



sum 函数



ida f5 图

main 函数

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int v3; // eax@2
  int *v4; // ecx@2
  int v6; // [sp-10h] [bp-38h]@2
  int *v7; // [sp+8h] [bp-20h]@11
  int v8; // [sp+Ch] [bp-1Ch]@7
  int v9; // [sp+10h] [bp-18h]@6
  int v10; // [sp+14h] [bp-14h]@4
  bool v11; // [sp+1Bh] [bp-Dh]@2
  int *v12; // [sp+1Ch] [bp-Ch]@2

  if ( y_4 >= 10 && (((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) != 0 )
    goto LABEL_11;
  while ( 1 )
  {
    v12 = &v6;
    v3 = sum(200, 200);
    v4 = v12;
    *v12 = v3;
    v11 = *v4 > 250;
    if ( y_4 < 10 || (((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) == 0 )
      break;
LABEL_11:
    v7 = &v6;
    v6 = sum(200, 200);
  }
  if ( v11 )
  {
    v10 = printf("you are a fool\n");
  }
  else
  {
    if ( *v12 <= 400 )
      v8 = printf("you are a clover\n");
    else
      v9 = printf("you are pritiy\n");
    while ( y_4 >= 10 && (((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) != 0 )
      ;
  }
  while ( y_4 >= 10 && (((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) != 0 )
    ;
  return 0;
}
```

sum 函数

```
int __cdecl sum(signed int a1, int a2)
{
  int v3; // [sp+Ch] [bp-Ch]@2

  if ( a1 >= 100 )
  {
    if ( a1 < 100 )
      goto LABEL_13;
    while ( y >= 10 && (((_BYTE)x - 1) * (_BYTE)x & 1) != 0 )
      ;
    if ( a1 >= 10000 )
    {
LABEL_13:
      do
        v3 = a2 + 450;
      while ( y >= 10 && (((_BYTE)x - 1) * (_BYTE)x & 1) != 0 );
    }
    else
    {
      v3 = a2 + 250;
    }
  }
  else
  {
    v3 = a2 + a1;
  }
  return v3;
}
```

平坦化+指令变换

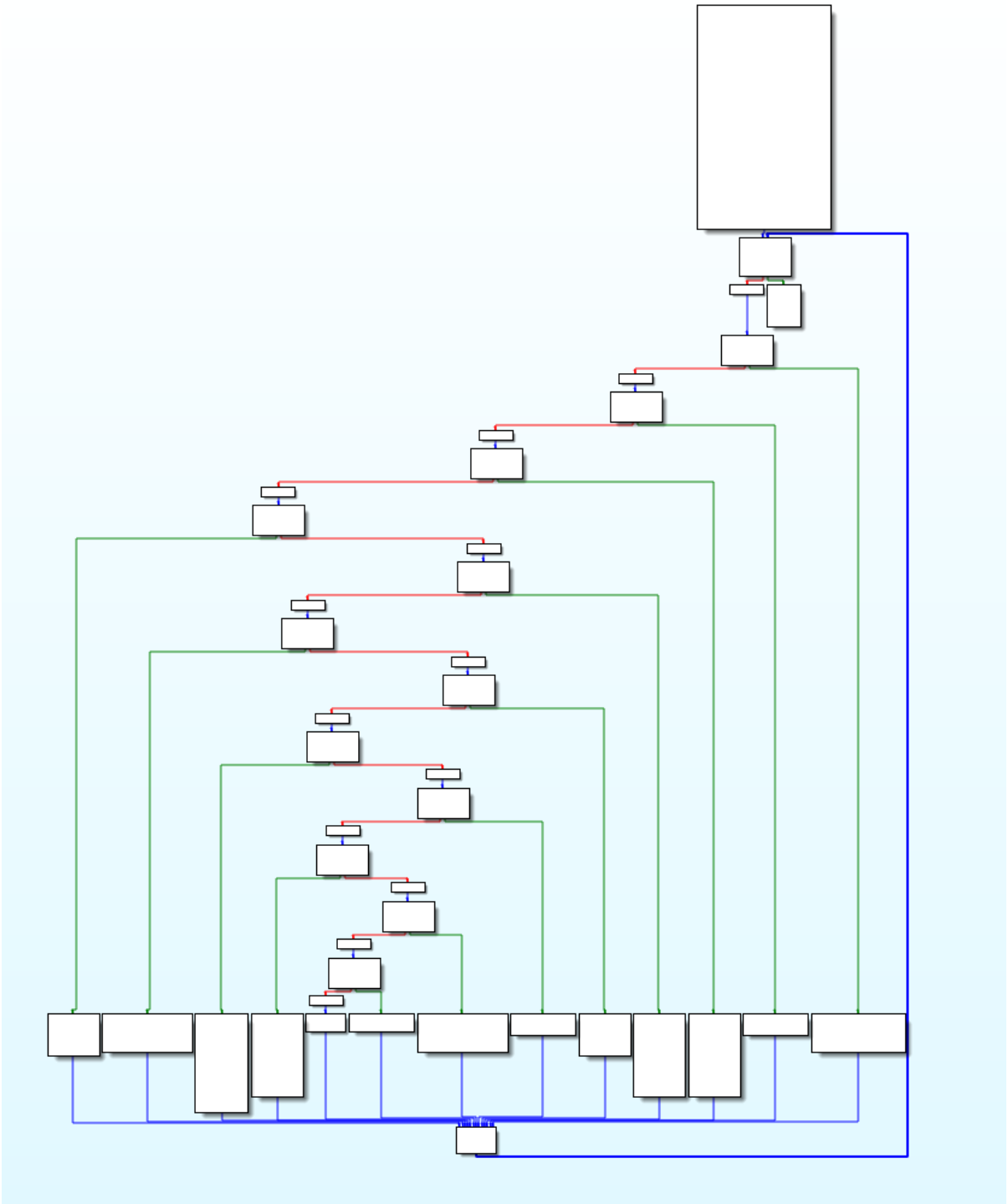
效果和平坦化控制流图差不多，f5 后 sum 函数稍有区别

```
int __cdecl sum(signed int a1, int a2)
{
  signed int v2; // eax@10
  signed int v3; // ecx@14
  signed int v4; // eax@17
  signed int v6; // [sp+20h] [bp-14h]@1
  int v7; // [sp+2Ch] [bp-8h]@0

  v6 = -878224646;
  while ( 1 )
  {
    while ( 1 )
    {
      while ( 1 )
      {
        while ( v6 == -878224646 )
        {
          v2 = -66762377;
          if ( a1 < 100 )
            v2 = 991721972;
          v6 = v2;
        }
        if ( v6 != -66762377 )
          break;
        v3 = 1358161861;
        if ( a1 >= 100 )
          v3 = 583626648;
        v6 = v3;
      }
      if ( v6 != 583626648 )
        break;
      v4 = 1358161861;
      if ( a1 < 10000 )
        v4 = 1643971560;
      v6 = v4;
    }
    if ( v6 == 925986251 )
      break;
    switch ( v6 )
    {
      case 991721972:
        v7 = a2 + a1 + 757610050 - 757610050;
        v6 = 925986251;
        break;
      case 1358161861:
        v7 = a2 + 450;
        v6 = 925986251;
        break;
      case 1643971560:
        v7 = a2 + 250;
        v6 = 925986251;
        break;
    }
  }
  return v7;
}
```

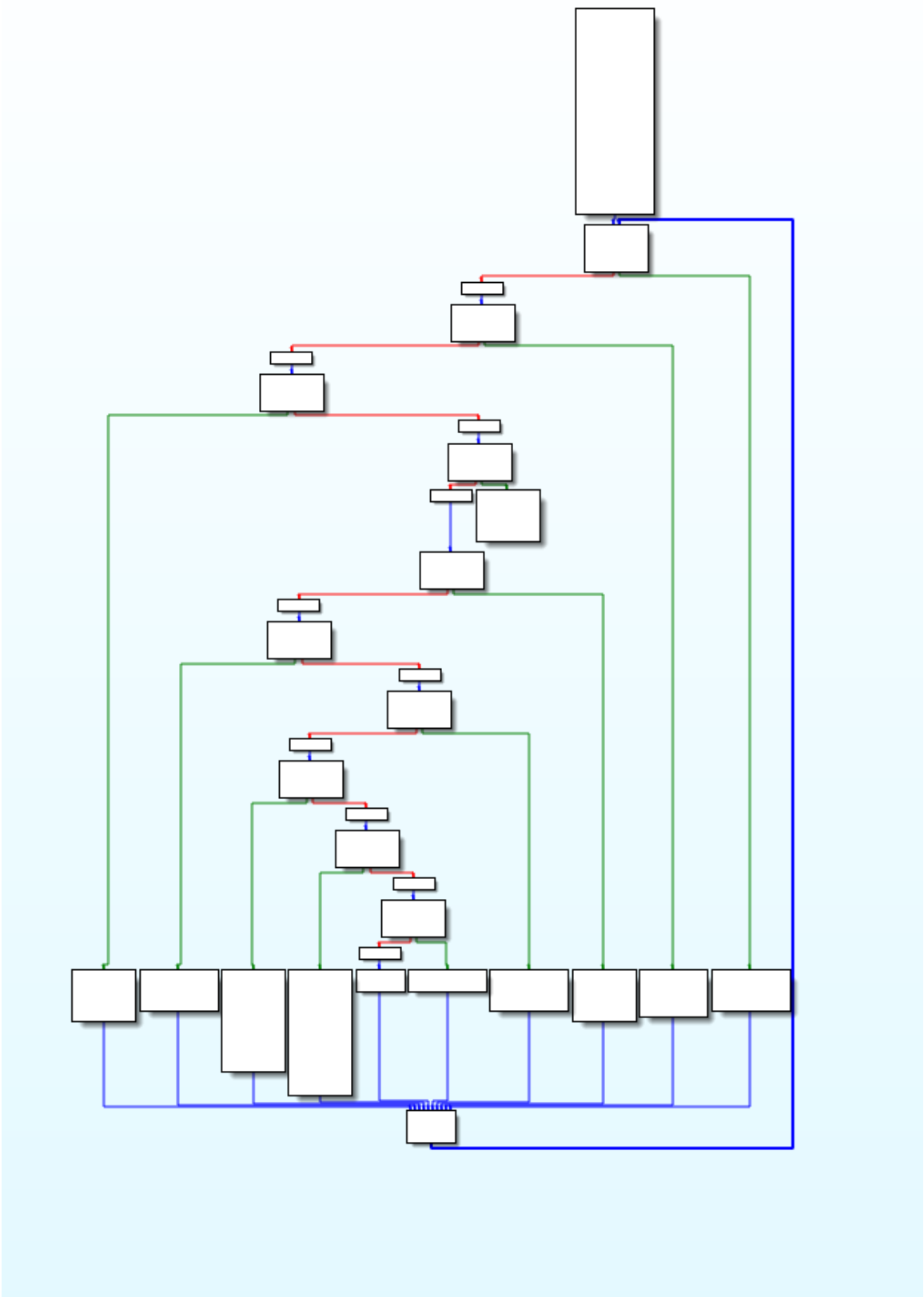
平坦化+虚假控制流

编译后控制流图

main 函数



sum 函数



ida f5 图

main 函数

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  signed int v3; // eax@16
  signed int v4; // eax@20
  signed int v5; // eax@23
  signed int v6; // eax@26
  signed int v7; // eax@32
  signed int v8; // eax@35
  signed int v10; // [sp+50h] [bp-28h]@1
  signed int v11; // [sp+54h] [bp-24h]@1
  bool v12; // [sp+6Bh] [bp-Dh]@0

  v11 = sum(100, 200);
  v10 = -764211651;
  while ( v10 != -1797298406 )
  {
    switch ( v10 )
    {
      case -1453465084:
        v10 = -249539738;
        printf("you are a clover\n");
        break;
      case -1329455460:
        v10 = -517509607;
        break;
      case -1060779968:
        v8 = 1546052436;
        if ( y_4 < 10 || (((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) == 0 )
          v8 = -1797298406;
        v10 = v8;
        break;
      case -764211651:
        v3 = 752430523;
        if ( v11 > 250 )
          v3 = -592616171;
        v10 = v3;
        break;
      case -691641183:
        v7 = 1546052436;
        if ( y_4 < 10 || (((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) == 0 )
          v7 = -1060779968;
        v10 = v7;
        break;
      case -592616171:
        v10 = -691641183;
        printf("you are a fool\n");
        break;
      case -536257094:
        v6 = -1453465084;
        if ( v12 )
          v6 = 1500924599;
        v10 = v6;
        break;
      case -517509607:
        v5 = -1329455460;
        v12 = v11 > 400;
        if ( y_4 < 10 || (((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) == 0 )
          v5 = -536257094;
        v10 = v5;
        break;
      case -249539738:
        v10 = -691641183;
        break;
      case 752430523:
        v4 = -1329455460;
        if ( y_4 < 10 || (((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) == 0 )
          v4 = -517509607;
        v10 = v4;
        break;
      case 1500924599:
        v10 = -249539738;
        printf("you are pritiy\n");
        break;
      case 1546052436:
        v10 = -1060779968;
        break;
    }
  }
  return 0;
}
```

sum 函数

```
int __cdecl sum(signed int a1, int a2)
{
  signed int v2; // eax@13
  signed int v3; // eax@17
  signed int v4; // eax@20
  signed int v5; // eax@23
  signed int v6; // eax@26
  signed int v8; // [sp+2Ch] [bp-24h]@1
  int v9; // [sp+38h] [bp-18h]@0
  bool v10; // [sp+43h] [bp-Dh]@0

  v8 = -1863910984;
  while ( 1 )
  {
    while ( 1 )
    {
      while ( 1 )
      {
        while ( v8 == -1977420134 )
        {
          v9 = a2 + 450;
          v8 = -1695259099;
        }
        if ( v8 != -1897342302 )
          break;
        v6 = -1977420134;
        if ( a1 < 10000 )
          v6 = 840245405;
        v8 = v6;
      }
      if ( v8 != -1863910984 )
        break;
      v2 = 1186888782;
      if ( a1 < 100 )
        v2 = 104270425;
      v8 = v2;
    }
    if ( v8 == -1695259099 )
      break;
    switch ( v8 )
    {
      case -367873085:
        v5 = -1977420134;
        if ( v10 )
          v5 = -1897342302;
        v8 = v5;
        break;
      case 104270425:
        v9 = a2 + a1;
        v8 = -1695259099;
        break;
      case 840245405:
        v9 = a2 + 250;
        v8 = -1695259099;
        break;
      case 1186888782:
        v3 = 1725198655;
        if ( y < 10 || (((_BYTE)x - 1) * (_BYTE)x & 1) == 0 )
          v3 = 1475356273;
        v8 = v3;
        break;
      case 1475356273:
        v4 = 1725198655;
        v10 = a1 >= 100;
        if ( y < 10 || (((_BYTE)x - 1) * (_BYTE)x & 1) == 0 )
          v4 = -367873085;
        v8 = v4;
        break;
      case 1725198655:
        v8 = 1475356273;
        break;
    }
  }
  return v9;
}
```

指令变换+虚假控制流

控制流图几乎和虚假控制流相同略

ida f5 图

main 函数

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int v3; // eax@2
  int *v4; // edx@2
  bool v5; // bh@2
  int v7; // [sp-10h] [bp-38h]@2
  int *v8; // [sp+4h] [bp-2Ch]@10
  int v9; // [sp+8h] [bp-28h]@7
  int v10; // [sp+Ch] [bp-1Ch]@6
  int v11; // [sp+10h] [bp-18h]@4
  bool v12; // [sp+17h] [bp-11h]@2
  int *v13; // [sp+18h] [bp-10h]@2

  if ( !(((unsigned __int8)((y_4 < 10) ^ ((((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) == 0)) | (y_4 < 10
                                                                                       && (((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) == 0)) & 1) )
    goto LABEL_10;
  while ( 1 )
  {
    v13 = &v7;
    v3 = sum(200, 200);
    v4 = v13;
    *v13 = v3;
    v5 = (((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) == 0;
    v12 = *v4 > 250;
    if ( ((unsigned __int8)((y_4 < 10) ^ v5) | (y_4 < 10 && v5)) & 1 )
      break;
LABEL_10:
    v8 = &v7;
    v7 = sum(200, 200);
  }
  if ( v12 )
  {
    v11 = printf("you are a fool\n");
  }
  else if ( *v13 <= 400 )
  {
    v9 = printf("you are a clover\n");
  }
  else
  {
    v10 = printf("you are pritiy\n");
  }
  while ( !((~((unsigned __int8)~(y_4 < 10) | (unsigned __int8)~((((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) == 0)) & 1 | (unsigned __int8)(~(y_4 < 10) ^ ~((((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) == 0))) & 1) )
    ;
  return 0;
}
```

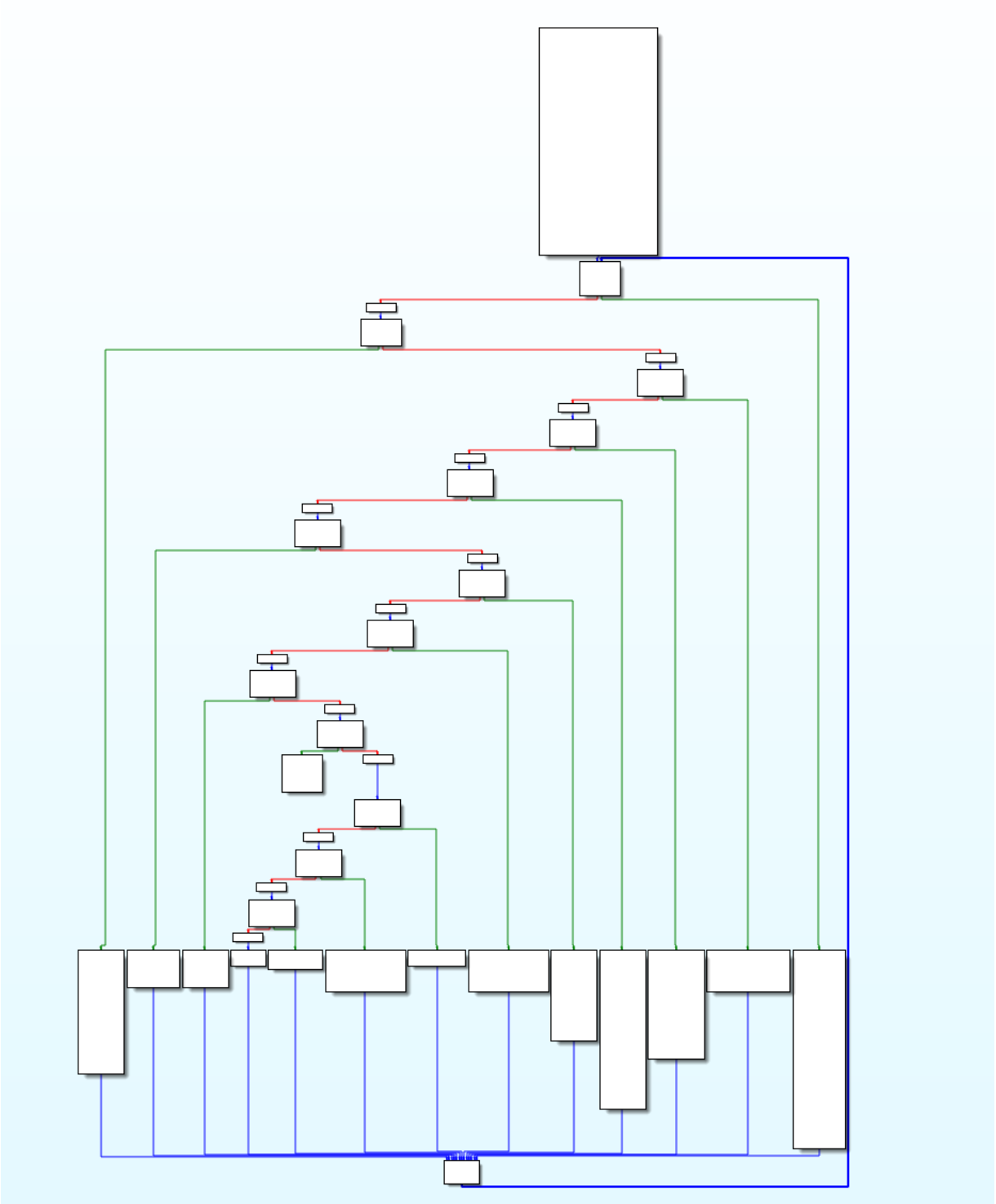
sum 函数

```
int __cdecl sum(signed int a1, int a2)
{
  unsigned __int8 v2; // dl@4
  int v4; // [sp+Ch] [bp-10h]@2

  if ( a1 >= 100 )
  {
    if ( a1 < 100 )
      goto LABEL_13;
    do
      v2 = ~((((_BYTE)x - 1) * (_BYTE)x & 1) == 0);
    while ( !((~((unsigned __int8)~(y < 10) | v2) & 1 | (unsigned __int8)(~(y < 10) ^ v2)) & 1) );
    if ( a1 >= 10000 )
    {
LABEL_13:
      do
        v4 = a2 + 450;
      while ( !((~((unsigned __int8)~(y < 10) | (unsigned __int8)~((((_BYTE)x - 1) * (_BYTE)x & 1) == 0)) & 1 | (unsigned __int8)((y < 10) ^ ((((_BYTE)x - 1) * (_BYTE)x & 1) == 0))) & 1) );
    }
    else
    {
      v4 = a2 - 1406211025 + 1406211275;
    }
  }
  else
  {
    v4 = a1 + a2;
  }
  return v4;
}
```

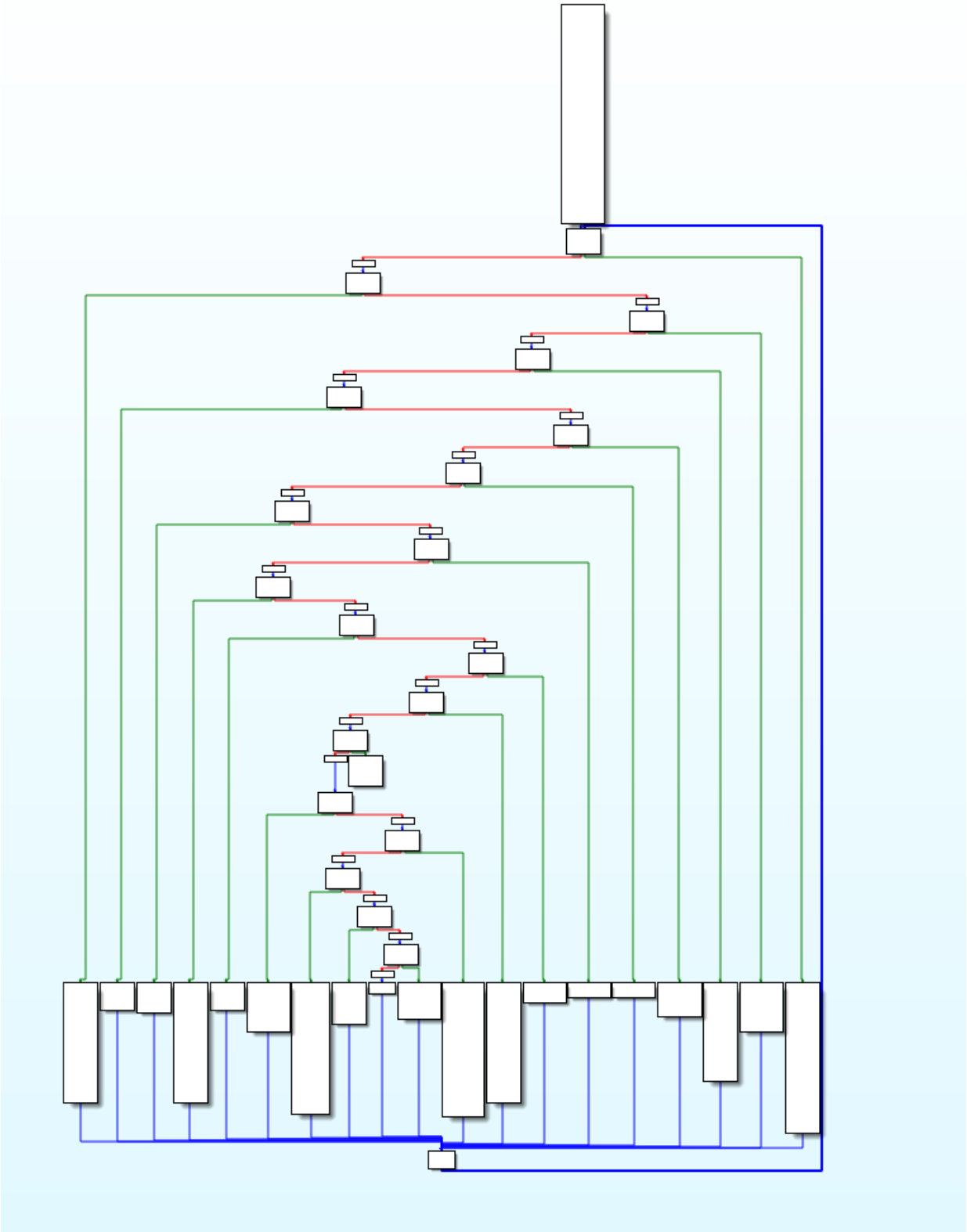
全开效果

编译后控制流图

main 函数



sum 函数



ida f5 图

main 函数

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  signed int v3; // esi@16
  int v4; // eax@19
  signed int v5; // ecx@19
  signed int v6; // eax@22
  int v7; // eax@25
  signed int v8; // eax@26
  int v9; // eax@29
  int v10; // eax@30
  signed int v11; // eax@31
  signed int v12; // esi@34
  int v14; // [sp-10h] [bp-88h]@19
  int *v15; // [sp+4h] [bp-74h]@39
  unsigned __int8 v16; // [sp+8h] [bp-6Dh]@34
  int v17; // [sp+Ch] [bp-6Ch]@34
  int v18; // [sp+10h] [bp-68h]@30
  int v19; // [sp+14h] [bp-64h]@29
  int v20; // [sp+18h] [bp-60h]@25
  char v21; // [sp+1Fh] [bp-59h]@16
  int v22; // [sp+20h] [bp-58h]@16
  int v23; // [sp+24h] [bp-54h]@14
  int v24; // [sp+28h] [bp-50h]@13
  int v25; // [sp+2Ch] [bp-4Ch]@12
  int v26; // [sp+30h] [bp-48h]@11
  int v27; // [sp+34h] [bp-44h]@10
  int v28; // [sp+38h] [bp-40h]@9
  int v29; // [sp+3Ch] [bp-3Ch]@8
  int v30; // [sp+40h] [bp-38h]@7
  int v31; // [sp+44h] [bp-34h]@6
  int v32; // [sp+48h] [bp-30h]@5
  int v33; // [sp+4Ch] [bp-2Ch]@4
  int v34; // [sp+50h] [bp-28h]@3
  int v35; // [sp+54h] [bp-24h]@2
  int v36; // [sp+58h] [bp-20h]@2
  int v37; // [sp+5Ch] [bp-1Ch]@1
  bool v38; // [sp+62h] [bp-16h]@1
  bool v39; // [sp+63h] [bp-15h]@1
  int *v40; // [sp+64h] [bp-14h]@19
  bool v41; // [sp+6Bh] [bp-Dh]@19

  v38 = (((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) == 0;
  v39 = y_4 < 10;
  v37 = -1634193910;
  while ( 1 )
  {
    while ( 1 )
    {
      while ( 1 )
      {
        while ( 1 )
        {
          while ( 1 )
          {
            while ( 1 )
            {
              while ( 1 )
              {
                while ( 1 )
                {
                  while ( 1 )
                  {
                    while ( 1 )
                    {
                      while ( 1 )
                      {
                        while ( 1 )
                        {
                          while ( 1 )
                          {
                            v36 = v37;
                            v35 = v37 + 1880022957;
                            if ( v37 != -1880022957 )
                              break;
                            v40 = &v14;
                            v4 = sum(200, 200);
                            v5 = -1042671526;
                            *v40 = v4;
                            v41 = *v40 > 250;
                            LOBYTE(v4) = (((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) == 0;
                            if ( ((unsigned __int8)((y_4 < 10) ^ v4) | (y_4 < 10) & (unsigned __int8)v4) & 1 )
                              v5 = 87875210;
                            v37 = v5;
                          }
                          v34 = v36 + 1634193910;
                          if ( v36 != -1634193910 )
                            break;
                          v22 = -1042671526;
                          v21 = ~v39;
                          v3 = -1042671526;
                          if ( (~((unsigned __int8)~v39 | (unsigned __int8)~v38) & 1 | (unsigned __int8)(~v39 ^ ~v38)) & 1 )
                            v3 = -1880022957;
                          v37 = v3;
                        }
```

1

```
                v37 = v3;
              }
              v33 = v36 + 1516924001;
              if ( v36 != -1516924001 )
                break;
              v10 = printf("you are a clover\n");
              v37 = -13467984l;
              v18 = v10;
            }
            v32 = v36 + 1042671526;
            if ( v36 != -1042671526 )
              break;
            v15 = &v14;
            v14 = sum(200, 200);
            v37 = -1880022957;
          }
          v31 = v36 + 617416175;
          if ( v36 != -617416175 )
            break;
          v17 = 351354523;
          v16 = ~(y_4 < 10);
          v12 = 351354523;
          if ( (~(v16 | (unsigned __int8)~((((_BYTE)x_3 - 12 + 11) * (_BYTE)x_3 & 1) == 0)) & 1 | (unsigned __int8)((y_4 < 10) ^ ((((_BYTE)x_3 - 12 + 11) * (_BYTE)x_3 & 1) == 0))) & 1 )
            v12 = 893410117;
          v37 = v12;
        }
        v30 = v36 + 398958931;
        if ( v36 != -398958931 )
          break;
        v8 = -1516924001;
        if ( *v40 > 400 )
          v8 = 427385819;
        v37 = v8;
      }
      v29 = v36 + 13467984l;
      if ( v36 != -13467984l )
        break;
      v11 = 351354523;
      if ( ((unsigned __int8)((y_4 < 10) ^ ((((_BYTE)x_3 - 1) * (_BYTE)x_3 & 1) == 0)) | (y_4 < 10
                                                                                        && (((_BYTE)x_3 - 1)
                                                                                          * (_BYTE)x_3 & 1) == 0)) & 1 )
        v11 = -617416175;
      v37 = v11;
    }
    v28 = v36 + 44821166;
    if ( v36 != -44821166 )
      break;
    v7 = printf("you are a fool\n");
    v37 = 1121318142;
    v20 = v7;
  }
  v27 = v36 - 87875210;
  if ( v36 != 87875210 )
    break;
  v6 = -398958931;
  if ( v41 )
    v6 = -44821166;
  v37 = v6;
 }
 v26 = v36 - 1121318142;
 if ( v36 == 112131842 )
  break;
 v25 = v36 - 351354523;
 if ( v36 == 351354523 )
 {
  v37 = -617416175;
 }
 else
 {
  v24 = v36 - 427385819;
  if ( v36 == 427385819 )
  {
    v9 = printf("you are pritiy\n");
    v37 = -13467984l;
    v19 = v9;
  }
  else
  {
    v23 = v36 - 893410117;
    if ( v36 == 893410117 )
      v37 = 112131842;
  }
 }
}
return 0;
}
```

2

sum 函数

```
int __cdecl sum(int a1, int a2)
{
  signed int v2; // eax@22
  signed int v3; // eax@25
  signed int v4; // eax@28
  signed int v5; // ecx@32
  bool v6; // dh@35
  signed int v7; // esi@35
  signed int v8; // esi@38
  signed int v9; // eax@41
  bool v10; // dh@44
  signed int v11; // esi@44
  signed int v12; // esi@47
  bool v13; // dh@51
  signed int v14; // esi@51
  signed int v15; // eax@54
  int *v17; // [sp+0h] [bp-A8h]@25
  unsigned __int8 v18; // [sp+7h] [bp-A1h]@51
  int v19; // [sp+8h] [bp-A0h]@51
  unsigned __int8 v20; // [sp+Eh] [bp-9Ah]@47
  bool v21; // [sp+Fh] [bp-99h]@47
  int v22; // [sp+10h] [bp-98h]@47
  unsigned __int8 v23; // [sp+17h] [bp-91h]@44
  int v24; // [sp+18h] [bp-90h]@44
  unsigned __int8 v25; // [sp+1Fh] [bp-89h]@38
  int v26; // [sp+20h] [bp-88h]@38
  unsigned __int8 v27; // [sp+27h] [bp-81h]@35
  int v28; // [sp+28h] [bp-80h]@35
  int v29; // [sp+2Ch] [bp-7Ch]@20
  int v30; // [sp+30h] [bp-78h]@19
  int v31; // [sp+34h] [bp-74h]@18
  int v32; // [sp+38h] [bp-70h]@17
  int v33; // [sp+3Ch] [bp-6Ch]@16
  int v34; // [sp+40h] [bp-68h]@15
  int v35; // [sp+44h] [bp-64h]@14
  int v36; // [sp+48h] [bp-60h]@13
  int v37; // [sp+4Ch] [bp-5Ch]@12
  int v38; // [sp+50h] [bp-58h]@11
  int v39; // [sp+54h] [bp-54h]@10
  int v40; // [sp+58h] [bp-50h]@9
  int v41; // [sp+5Ch] [bp-4Ch]@8
  int v42; // [sp+60h] [bp-48h]@7
  int v43; // [sp+64h] [bp-44h]@6
  int v44; // [sp+68h] [bp-40h]@5
  int v45; // [sp+6Ch] [bp-3Ch]@4
  int v46; // [sp+70h] [bp-38h]@3
  int v47; // [sp+74h] [bp-34h]@2
  int v48; // [sp+78h] [bp-30h]@2
  int *v49; // [sp+7Ch] [bp-2Ch]@1
  int *v50; // [sp+80h] [bp-28h]@1
  int v51; // [sp+84h] [bp-24h]@1
  bool v52; // [sp+8Ah] [bp-1Eh]@1
  bool v53; // [sp+8Bh] [bp-1Dh]@1
  int *v54; // [sp+8Ch] [bp-1Ch]@25
  int *v55; // [sp+90h] [bp-18h]@25
  int *v56; // [sp+94h] [bp-14h]@25
  bool v57; // [sp+9Ah] [bp-Eh]@25
  bool v58; // [sp+9Bh] [bp-Dh]@38

  v52 = (((_BYTE)x - 1) * (_BYTE)x & 1) == 0;
  v53 = y < 10;
  v51 = 1813646441;
  v50 = (int *)a2;
  v49 = (int *)a1;
  while ( 1 )
  {
    while ( 1 )
    {
      while ( 1 )
      {
        while ( 1 )
        {
          while ( 1 )
          {
            while ( 1 )
            {
              while ( 1 )
              {
                while ( 1 )
                {
                  while ( 1 )
                  {
                    while ( 1 )
                    {
                      while ( 1 )
                      {
```

1

```
                while ( 1 )
                {
                  while ( 1 )
                  {
                    while ( 1 )
                    {
                      v48 = v51;
                      v47 = v51 + 1904509565;
                      if ( v51 != -1904509565 )
                        break;
                      *v54 = *v56 + 250;
                      v22 = 1859786781;
                      v21 = y < 10;
                      v20 = ~(y < 10);
                      v12 = 1859786781;
                      if ( (~(v20 | (unsigned __int8)~((((_BYTE)x - 1) * (_BYTE)x & 1) == 0)) & 1 | (unsigned __int8)(v20 ^ ~((((_BYTE)x - 1) * (_BYTE)x & 1) == 0))) & 1 )
                        v12 = -422562557;
                      v51 = v12;
                    }
                    v46 = v48 + 1778328355;
                    if ( v48 != -1778328355 )
                      break;
                    v28 = 23485419;
                    v6 = (((_BYTE)x - 55 + 54) * (_BYTE)x & 1) == 0;
                    v27 = ~(y < 10);
                    v7 = 23485419;
                    if ( (~(v27 | (unsigned __int8)~v6) & 1 | (unsigned __int8)((y < 10) ^ v6)) & 1 )
                      v7 = 1249156110;
                    v51 = v7;
                  }
                  v45 = v48 + 1041264831;
                  if ( v48 != -1041264831 )
                    break;
                  *(&v17 - 4) = v49;
                  *(&v17 - 4) = v50;
                  v51 = 1281026285;
                }
                v44 = v48 + 984908924;
                if ( v48 != -984908924 )
                  break;
                v15 = -609108537;
                *v54 = -(-450 - *v56);
                if ( ((unsigned __int8)((y < 10) ^ ((((_BYTE)x - 1) * (_BYTE)x & 1) == 0)) | (y < 10 && (((_BYTE)x - 1) * (_BYTE)x & 1) == 0)) & 1 )
                  v15 = -374562900;
                v51 = v15;
              }
              v43 = v48 + 640344842;
              if ( v48 != -640344842 )
                break;
              v4 = -398494098;
              if ( v57 )
                v4 = 6046504460;
              v51 = v4;
            }
            v42 = v48 + 609108537;
            if ( v48 != -609108537 )
              break;
            *v54 = *v56 + 450;
            v51 = -984908924;
          }
          v41 = v48 + 422562557;
          if ( v48 != -422562557 )
            break;
          v51 = 441312340;
        }
        v40 = v48 + 398494098;
        if ( v48 != -398494098 )
          break;
        v5 = 293814475;
        if ( *v55 >= 100 )
          v5 = -1778328355;
        v51 = v5;
      }
      v39 = v48 + 374562900;
      if ( v48 != -374562900 )
        break;
      v51 = 441312340;
    }
    v38 = v48 + 241265571;
    if ( v48 != -241265571 )
      break;
    v24 = 1859786781;
    v10 = (((_BYTE)x - 65 + 64) * (_BYTE)x & 1) == 0;
    v23 = ~(y < 10);
    v11 = 1859786781;
    if ( (~(v23 | (unsigned __int8)~v10) & 1 | (unsigned __int8)((y < 10) ^ v10)) & 1 )
      v11 = -1904509565;
    v51 = v11;
```

2

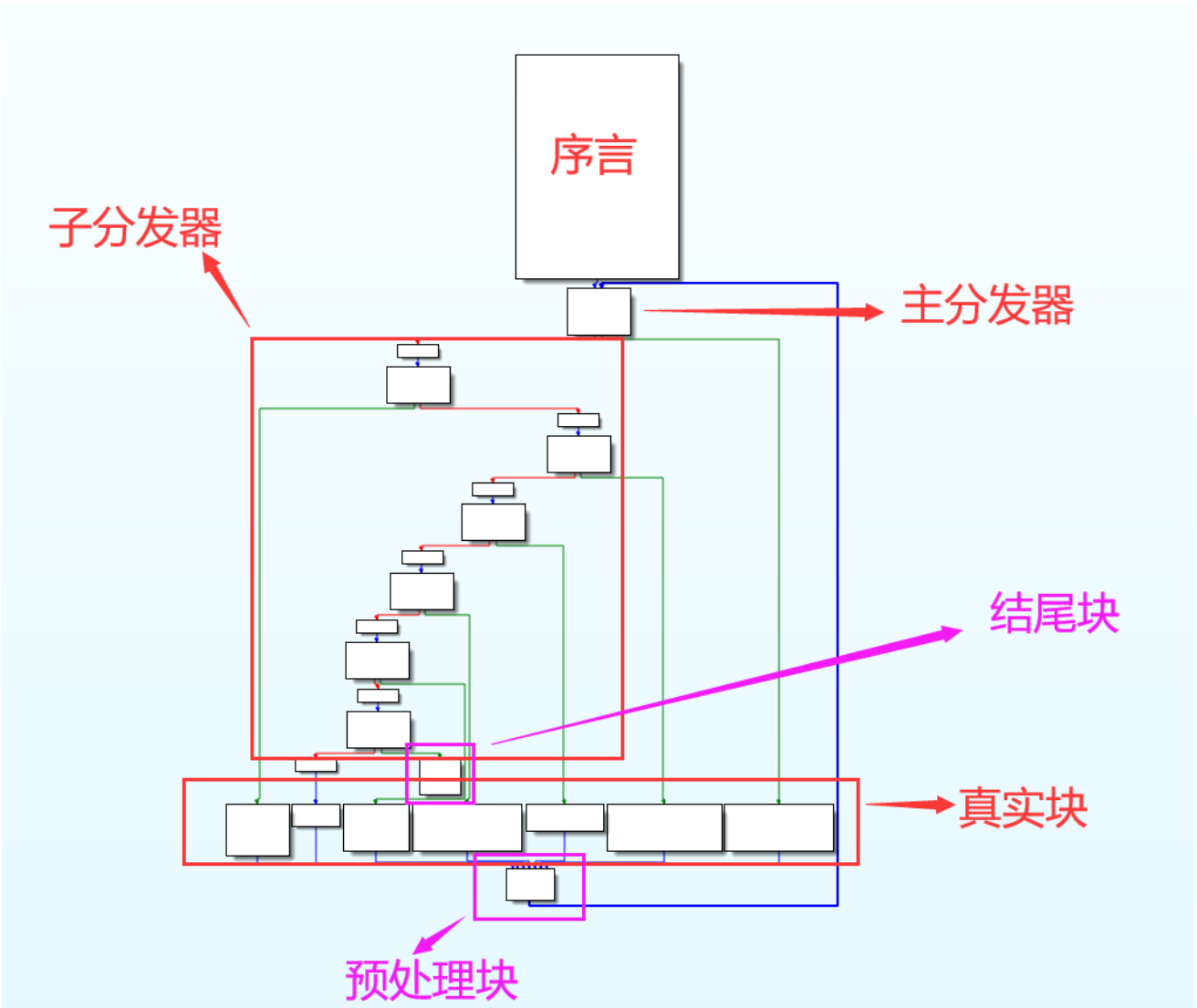
```
        if ( v48 != -121082555 )
          break;
        v9 = 293814475;
        if ( v58 )
          v9 = -241265571;
        v51 = v9;
      }
      v36 = v48 - 23485419;
      if ( v48 != 23485419 )
        break;
      v51 = 1249156110;
      v17 = v55;
    }
    v35 = v48 - 293814475;
    if ( v48 != 293814475 )
      break;
    v19 = -609108537;
    v13 = (((_BYTE)x - 96 + 95) * (_BYTE)x & 1) == 0;
    v18 = ~(y < 10);
    v14 = -609108537;
    if ( (~(v18 | (unsigned __int8)~v13) & 1 | (unsigned __int8)((y < 10) ^ v13)) & 1 )
      v14 = -984908924;
    v51 = v14;
  }
  v34 = v48 - 441312340;
  if ( v48 == 441312340 )
    break;
  v33 = v48 - 604650460;
  if ( v48 == 604650460 )
  {
    *v54 = -(-*v55 - *v56);
    v51 = 441312340;
  }
  else
  {
    v32 = v48 - 1249156110;
    if ( v48 == 1249156110 )
    {
      v58 = *v55 < 10000;
      v26 = 23485419;
      v25 = ~(y < 10);
      v8 = 23485419;
      if ( (~(v25 | (unsigned __int8)~((((_BYTE)x - 1) * (_BYTE)x & 1) == 0)) & 1 | (unsigned __int8)((y < 10) ^ ((((_BYTE)x - 1) * (_BYTE)x & 1) == 0))) & 1 )
        v8 = -121082555;
      v51 = v8;
    }
    else
    {
      v31 = v48 - 1281026285;
      if ( v48 == 1281026285 )
      {
        v3 = -1041264831;
        v54 = (int *)(&v17 - 4);
        v55 = (int *)(&v17 - 4);
        v56 = (int *)(&v17 - 4);
        *v55 = (int)v49;
        *v56 = (int)v50;
        v57 = *v55 < 100;
        if ( ((unsigned __int8)((y < 10) ^ ((((_BYTE)x - 1) * (_BYTE)x & 1) == 0)) | (y < 10
                                                                                     && (((_BYTE)x - 1) * (_BYTE)x & 1) == 0)) & 1 )
          v3 = -640344842;
        v51 = v3;
      }
      else
      {
        v30 = v48 - 1813646441;
        if ( v48 == 1813646441 )
        {
          v2 = -1041264831;
          if ( ((unsigned __int8)(v53 ^ v52) | (v53 && v52)) & 1 )
            v2 = 1281026285;
          v51 = v2;
        }
        else
        {
          v29 = v48 - 1859786781;
          if ( v48 == 1859786781 )
          {
            *v54 = *v56 + 250;
            v51 = -1904509565;
          }
        }
      }
    }
  }
}
return *v54;
}
```

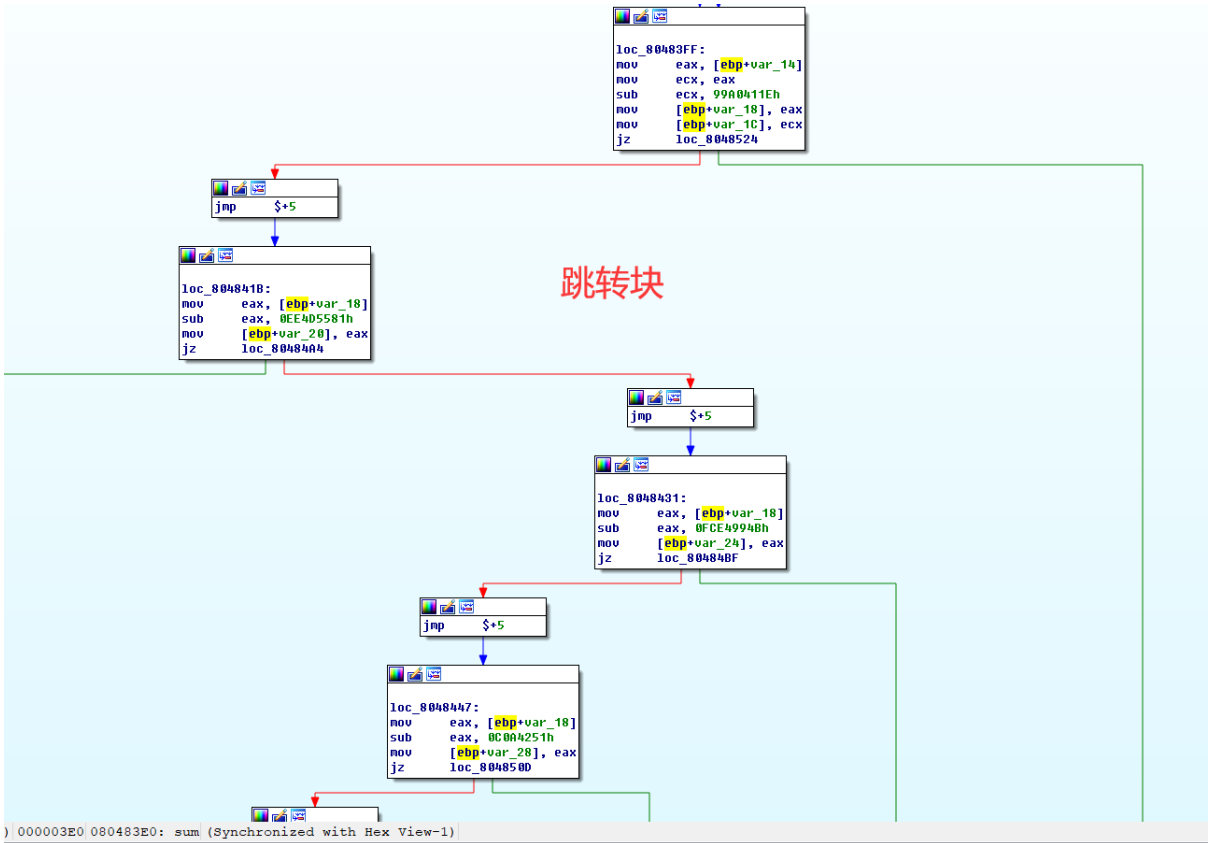
3

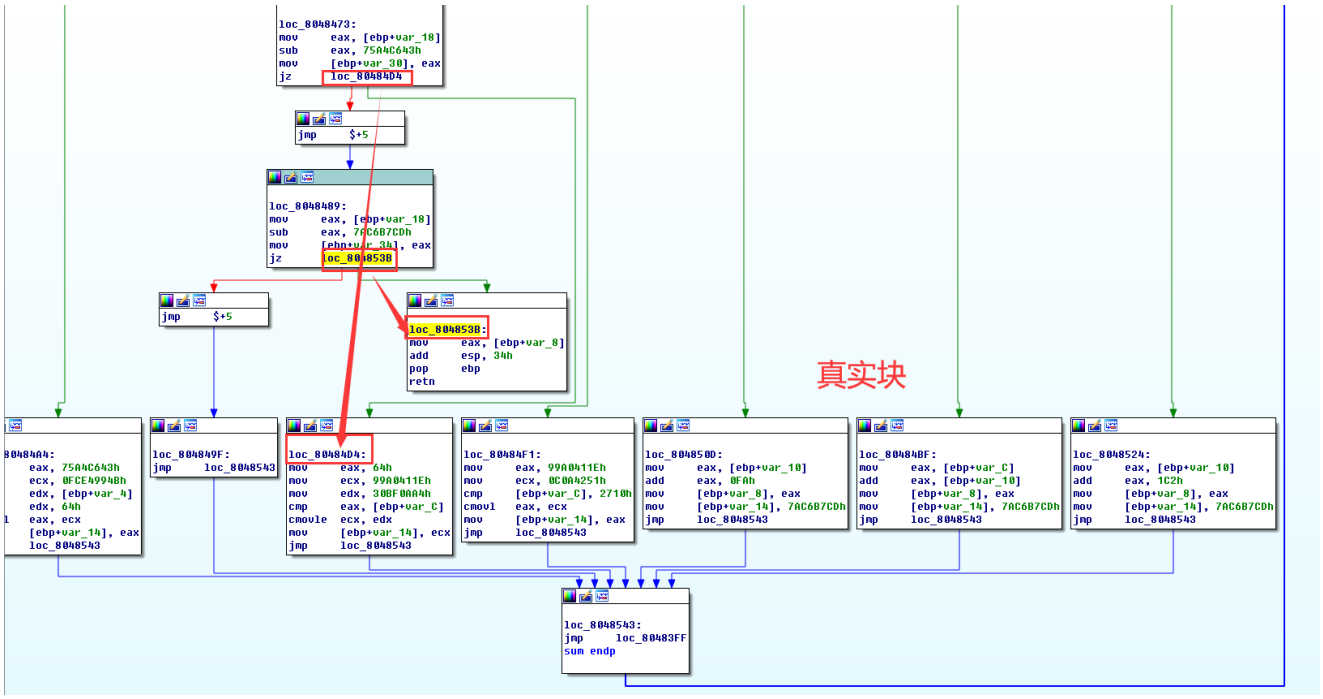
和原程序区别

指令平坦化

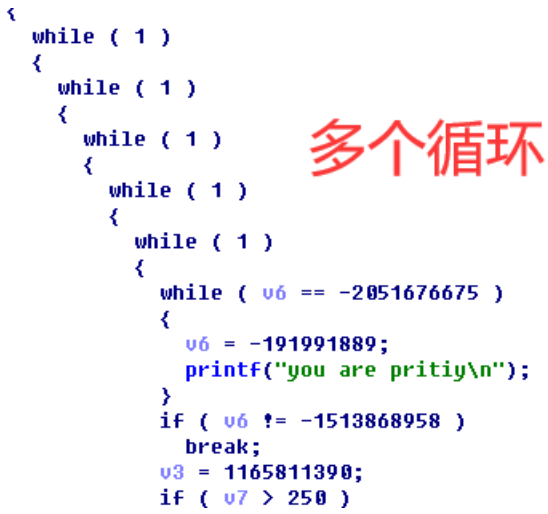
控制流图中，很多比较跳转块、switch case 块，这些块并不处理真实逻辑，每个块的末尾跳转的地方就是真实块；


序言
子分发器
主分发器
结尾块
真实块
预处理块



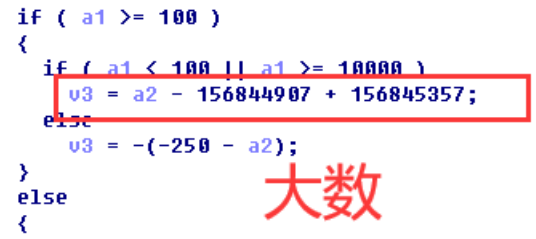


代码中你会看见很多 while 循环，很多 switch。



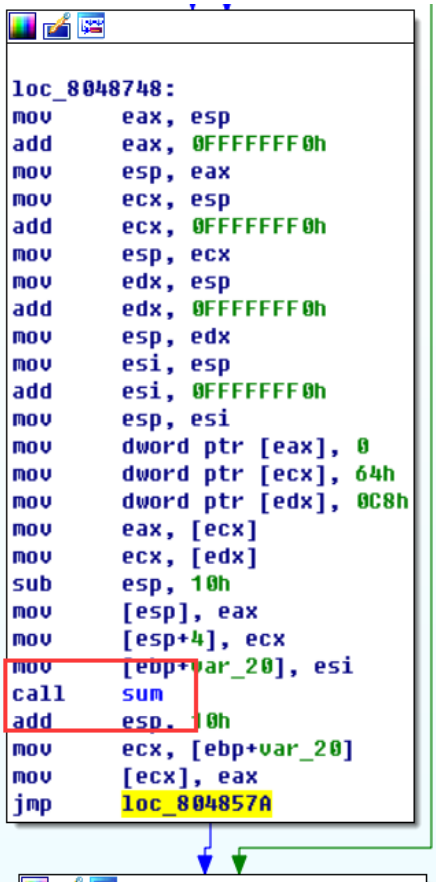
指令替换

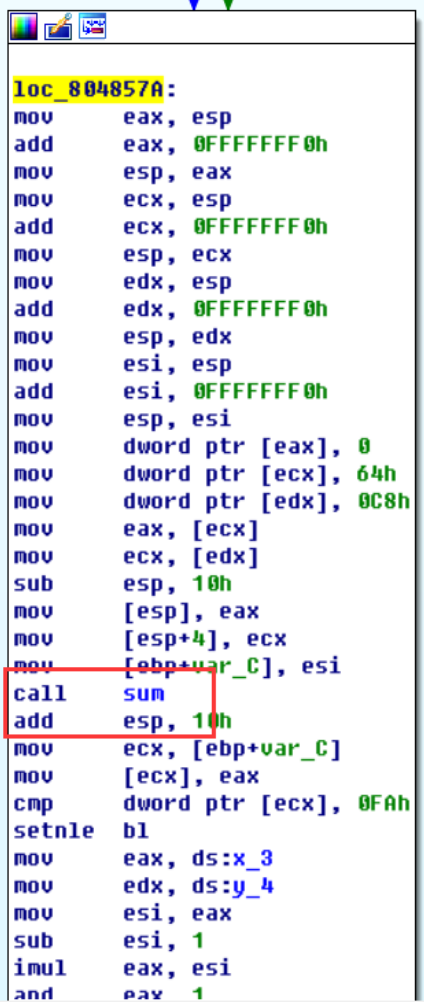
控制流程图变化不大；代码中本来很小的数，都被替换成了很大的数相加减，大的离谱。



虚假控制流

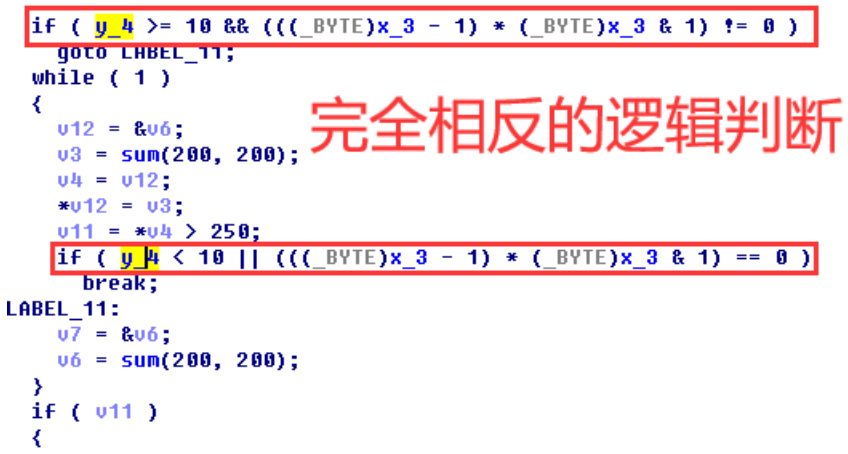
控制流图中会看见很多类似的逻辑块，例如原程序中只调用一次 sum 函数，而虚假控制流会出现多个调用 sum 函数的逻辑块；具体那个块被虚假好像是随机的。





原程序仅一个地方调用，这里存在一个虚假的块，并且逻辑块的核心逻辑类似

代码中可以看到多个地方代码类似，但是肯定有一个条件是真，其他都为假，也就是说其他类似的代码逻辑根本不会被执行；从下图我们可以清楚的看到这里存在两处相同的 sum 逻辑（参数都相同）；并且我们发现执行它的条件正好完全相反，我们不管哪个为真，只有一个为真，另一个一定为假，也就是说无论何时都只可能执行一次 sum；而另一个只是用来迷惑你。

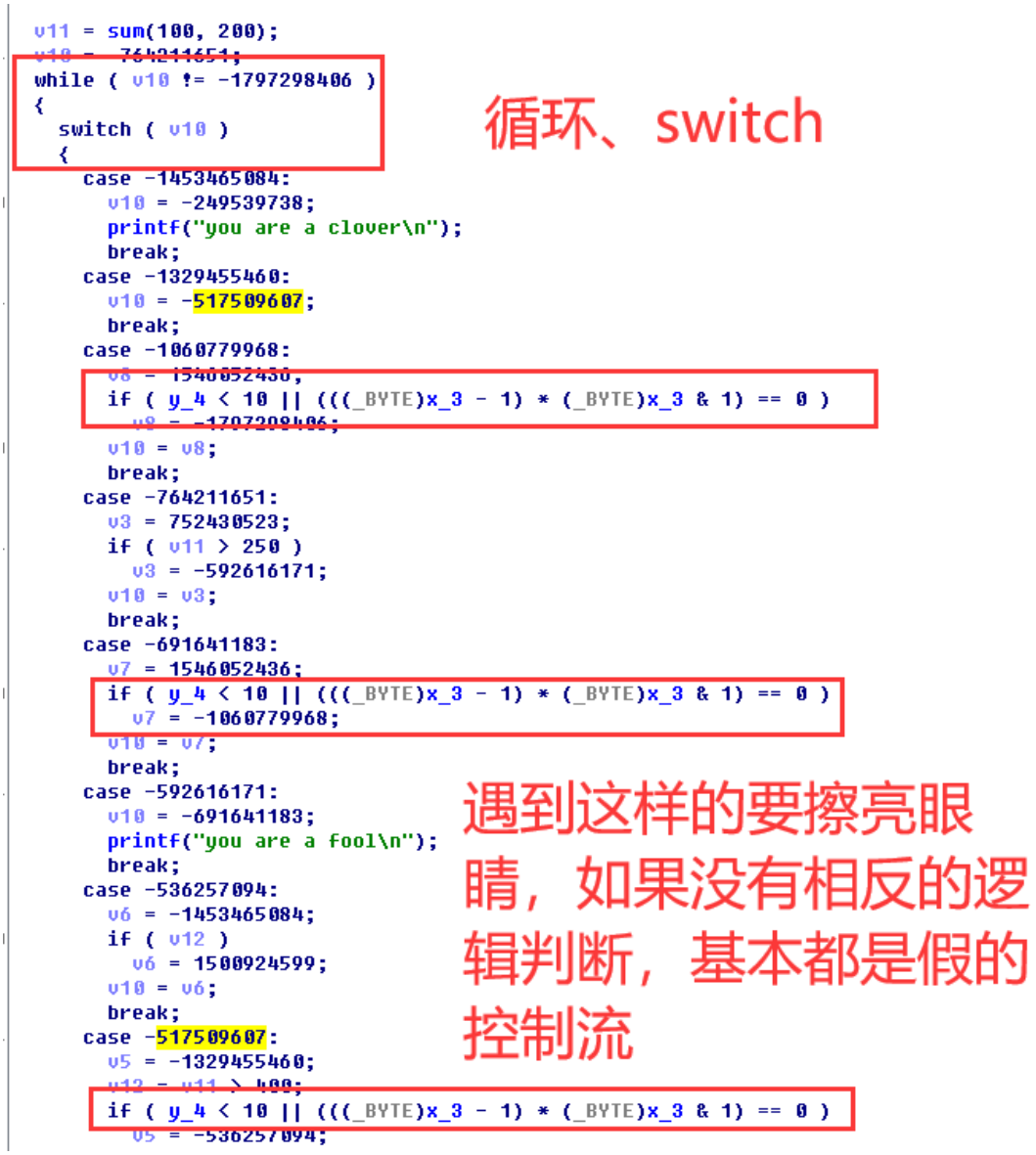


平坦化+指令变换

流程图和平坦化类似；指令变换使 sum 代码多了 switch 逻辑（里面都是类似大数操作）

平坦化+虚假控制流

控制流图是平坦化和虚假控制流的结合体，很多跳转块、switch 块、虚假块的结合，特点也同上面说的；代码风格是循环+switch 块+if 判断（一定同时一个真一个假，如过只出现一中判断基本都是假逻辑块）。



指令变换+虚假控制流

特征基本+虚假控制流类似，略。

全开效果

特征同平坦化+虚假控制流，但是因为加了指令替换所以可能会出现很大大数或者奇怪的逻辑运算等，略。

反混淆

这里容我说下自己浅薄的想法，实际应该远比这复杂（特别是全开后，另外暂时不知道有没有安全公司研发出更牛逼的基于 ollvm 的混淆），大佬请绕过；网上类似的文章和开源项目也不少，依据这个思想你应该可以实现自己的反混淆。

指令平坦化

一个事实在一层深度中，跳转块不执行核心逻辑，顺着跳转块找下去最后的一个块一定是程序的结尾块（真实块）；第一个块基本都是真实块；每个跳转块衔接的块（跳转指令指向的块）一定是真实块（跳转块中仅仅存在一个跳转指令且它衔接的块也仅仅存在一个跳转指令，这样以此类推，它多半是无用的块）。

依据此原理我们需要解析每个跳转块得到它所指向的真实块，得到真实块把它们衔接（除头尾块，其他的块需要注意遍历顺序）。

另一种办法（摘自网络）：

1. 函数的开始地址为序言的地址
2. 序言的后继为主分发器
3. 后继为主分发器的块为预处理器
4. 后继为预处理器的块为真实块
5. 无后继的块为 retn 块
6. 剩下的为无用块

指令替换

视情况而定，需要找出规律，但一般不影响分析。

虚假控制流

找到条件（判断分支的条件，一般比较好找，看起来很长并且很奇怪，大多包含逻辑或与非甚至移位运算都有可能；可能存在 n 个完全一样的逻辑判断；或者成对出现）。

出现 n 个完全相同的条件判断大多都是假（非真即假，非假即真），成对出现的肯定是一真一假。

依据此原理，找出真的去掉假的。

平坦化+指令替换

基本和平台化类似

平坦化+虚假控制流

结合平坦化和虚假控制流

指令替换+虚假控制流

基本和控制流类似

全开

结合平坦化和虚假控制流

示例

静态分析级别

这种去混淆的结果可能导致堆栈不平衡，不过核心代码被保留下来，可以静态看看逻辑，作为辅助。

借助 ida 脚本实现 HexRaysDeob
反汇编->中间代码->cfg->符号执行->patch
符号执行 Triton、barf
angr、Miasm（对 arm 支持不好）
Binary Ninja（商业）、RetDec
模拟执行 Unicorn
选择执行

动态分析级别

这种混淆的结果是使反混淆后的代码接近原始代码并且不破坏堆栈平衡，不影响调用和调试。
反汇编->中间代码->优化（反混淆）->编译