Mountain climbing

转载

有人_295 于 2019-04-14 00:45:34 发布 1355 收藏

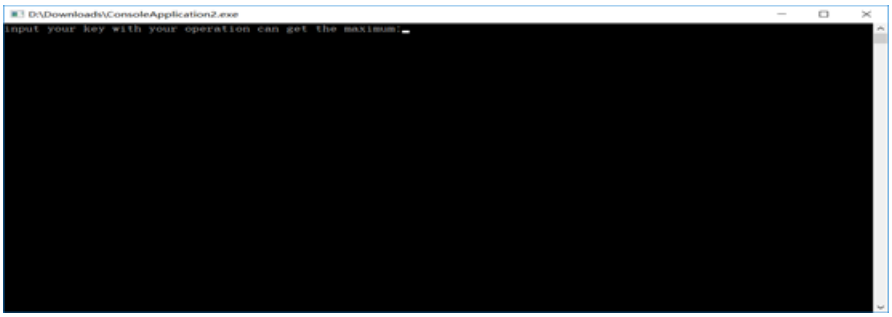
分类专栏： Writeup 文章标签： Writeup


Writeup 专栏收录该内容

3 篇文章 0 订阅

订阅专栏

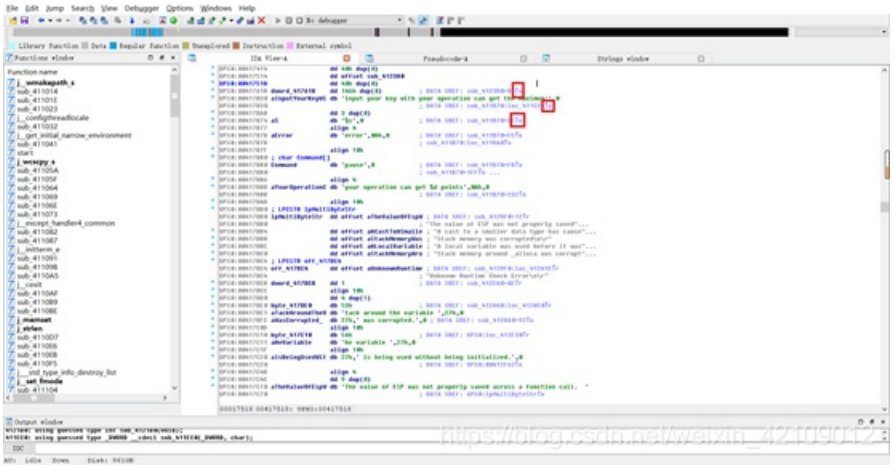
先是运行下程序，发现要输入一串东西。



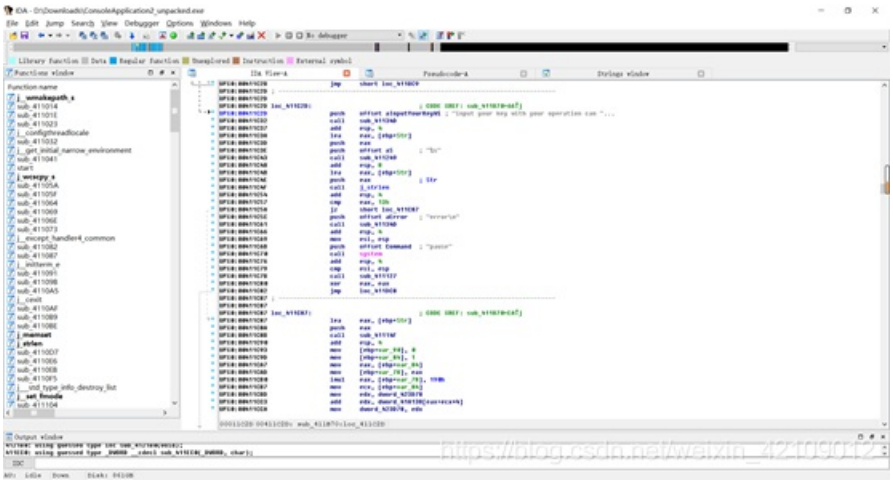
接着用查壳工具进行查壳，发现这个软件有个 UPX 的壳。

然后我用万能的脱壳工具进行脱壳，脱壳后得到能够使用IDA和OD打开的程序

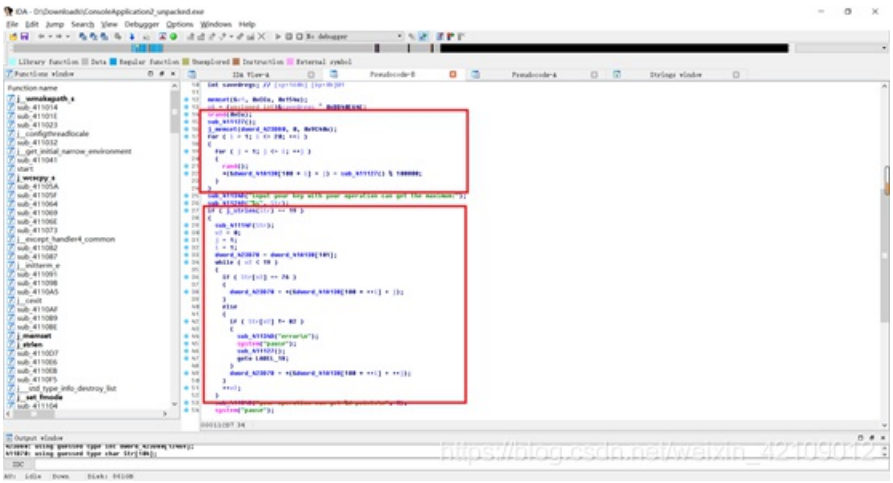
于是我用IDA打开，按F12查看字符串，点击开头的input开始查看，根据要输入的提示双击打开input字符串，汇编混杂，现在还不能使用反编译，点击那个箭头，跳转到全是汇编界面



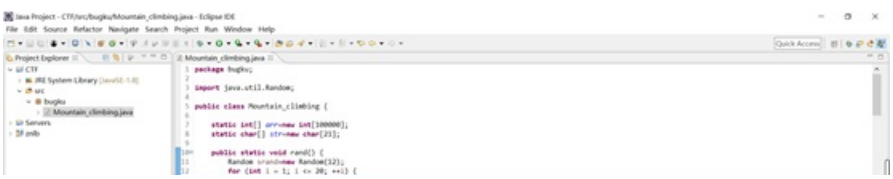
然后点击Tab或F5，就可以得到反汇编

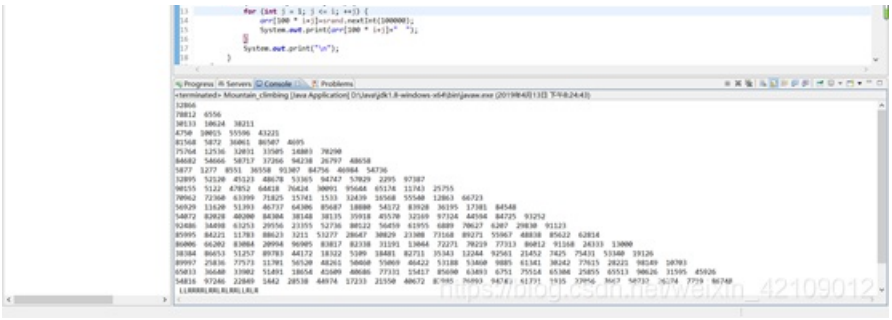


然后就是分析代码的时候了。

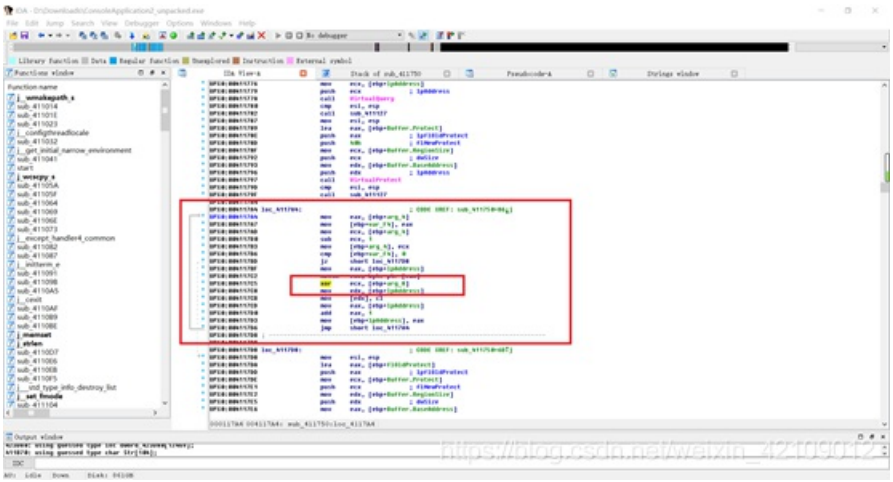


第一个红框表示生成随机数，srand代表随机数种子，0xCu表示12，生成一个100000以内的三角形随机数，一共19行。
第二个红框表示76=L，82=R，如果等于76L就直接向下移动一位，如果等于82R就向下和向右移动一位。
然后就进行反向编程，我用的是Java，编译后的结果如下，然后我就用结果去试，显而易见的错了，不可能这么简单的。。。。。

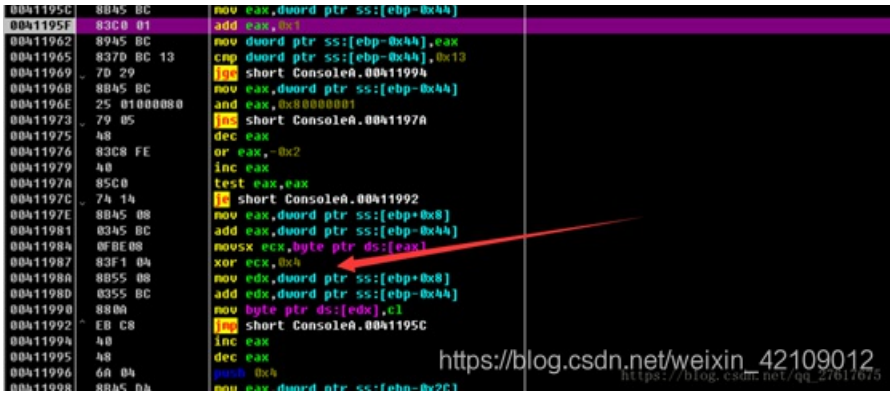




然后我就不知道怎么做了，于是百度了一下，发现原来还要进行一部操作，可是我照着网上的IDA做就是不能找到想要的东西。。。。。以下第一幅图是我的，没找打xor的0x4运算，而别人的有。。。。。
都找得到00411750即开始标志，push，不过我这个xor看不懂。



接着我又换一个软件OD用，使用网上的过程，还是不行。。。。。
都可以找到0041114F，然后跟进就始终找不到00411987 xor ecx 0x4.。。。。。。



最后整不出来，就思考了了一下思路，即在偶数位置需要与4进行异或xor运算

最后在整理一下思路

1、生成随机数，srand代表随机数种子，0xCu表示12，生成一个100000以内的三角形随机数，一共19行。

2、76=L，82=R，如果等于76L就直接向下移动一位，如果等于82R就向下和向右移动一位。

反编译过来就是下一行左边大得到L，右边大得到R。

3、偶数位置需要与4进行异或xor运算

然后使用Java逆向编程出来即可

```
import java.util.Random;

public class Mountain_climbing {
 //全局变量
 static int[] arr=new int[100000];
 static char[] str=new char[21];
 //生成随机数
 public static void rand() {
  Random srand=new Random(12);
  for (int i = 1; i <= 20; ++i) {
   for (int j = 1; j <= i; ++j) {
    arr[100 * i+j]=srand.nextInt(100000);
    System.out.print(arr[100 * i+j]+"  ");
   }
   System.out.print("\n");
  }
 }
 //进行位移L或R，
 public static void DFS() {
  int i=1;
  int j=1;
  int max=0;
  while(true) {
   int ii=i+1;
   int jj=j+1;
   if(arr[100 * ii+j]>arr[100 * ii + jj]) {
    str[i] = 'L';
            i++;
            max += arr[100 * i + j];
   }
   else {
    str[i] = 'R';
            i++;
            j++;
            max += arr[100 * i + j];
   }
   if (i
```

```
  if (i == 20){
          str[i] = '\0';
          break;
   }
  }
  System.out.println(str);
  System.out.println(max);
 }
 //改变偶数位置异或运算
 public static void change()
 {
     int i;
     for (i = 1; i <= 19; i++)
     {
         if (i % 2==0)
         {
             str[i] ^= 4;
         }
     }
     System.out.println(str);
 }
//主函数
 public static void main(String[] args) {
  rand();
  DFS();
  change();
 }
}
```

如果你只看了前面，恭喜你只得到思路，因为**TMD-------Java**和**C**语言随机数种子一样，但生成的随机数不一样。。。。。

拿去试还是不得行，经过长时间的比对，发现没有错误。。。。。。
最后发现我用**Java**和别人用**C**语言编的随机生成数不一样，可是**Java**中应该是这样写的，最后才发觉两个语言的随机算法不一样。。。。。。。。。。
于是我最后使用**C**语言编程得到最后答案，即**flag**

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

int main()
{
 srand(0xCu);  // 随机数种子
 int arr[2020];  // 存放随机数的数组
 printf("76 = %c  82 = %c  76^4 = %c  82^4 = %c\n", 76, 82, 76 ^ 4, 82 ^ 4);
 for (int i = 0; i < 20; i++)  // 总长度19
 {
  printf("%2d ", i);
  for (int j = 0; j <= i; j++)
  {
   arr[100 * i + j] = rand() % 100000;  // 依次生成随机数
   printf("%5d ", arr[100 * i + j]);
  }
  printf("\n");
 }
 int all = 0;  // 最终总和
 all += arr[100 * 0 + 0];  // 第一行第一个赋值给all
 char str[21];
 int i = 0, j = 0;
```

```
while (1)
{
 printf("%d ", i);
 int i_ = i + 1;
 int j_ = j + 1;
 if (arr[100 * i_ + j] > arr[100 * i_ + j_])  // 正下方的一个数大于其右边的数
 {
  str[i] = 'L';
  if (i % 2 == 1)
  {
   str[i] ^= 4;
  }
  i++;
  all += arr[100 * i + j];
 }
 else
 {
  str[i] = 'R';
  if (i % 2 == 1)
  {
   str[i] ^= 4;
  }
  i++;
  j++;
  all += arr[100 * i + j];
 }
 if (i == 19)
 {
  str[i] = '\0';
  break;
 }
}
printf("\n 输入为： %s\n", str);
printf("maxnum = %d\n", all);

system("pause");
return 0;
}
```

flag为：zsctf{RVRVRHLVRVLVLVRVLVL}