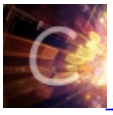
[ACTF新生赛2020]Universe_final_answer学习笔记

原创

mortal15 于 2021-02-10 23:35:35 发布 206 收藏
分类专栏： re

本文链接：https://blog.csdn.net/a5555678744/article/details/113787081


re 专栏收录该内容
6 篇文章 0 订阅
订阅专栏

[ACTF新生赛2020]Universe_final_answer

题目重述

熟练地打开main函数

```
__int64 __fastcall main(int a1, char **a2, char **a3)
{
  char v4[32]; // [rsp+0h] [rbp-A8h] BYREF
  char v5[104]; // [rsp+20h] [rbp-88h] BYREF
  unsigned __int64 v6; // [rsp+88h] [rbp-20h]

  v6 = __readfsqword(0x28u);
  __printf_chk(1LL, "Please give me the key string:", a3);
  scanf("%s", v5);
  if ( sub_860(v5) )
  {
    sub_C50(v5, v4);
    __printf_chk(1LL, "Judgement pass! flag is actf{%s_%s}\n", v5);
  }
  else
  {
    puts("False key!");
  }
  return 0LL;
}
```


程序的意思很明显：就是做一个判断，如果通过了就给你flag，但是可以看到，输出里面有两个%s而后面只有一个对应参量，这么说另一个隐藏起来的就不能单纯靠静态分析解决了。

关键函数

这里很显然sub_860是关键函数，只要它判断成功了，问题就得到解决了

```
bool __fastcall sub_860(char *a1)
{
  int v1; // ecx
  int v2; // esi
  int v3; // edx
  int v4; // er9
  int v5; // er11
  int v6; // ebp
  int v7; // ebx
  int v8; // er8
  int v9; // er10
  bool result; // al
  int v11; // [rsp+0h] [rbp-38h]

  v1 = a1[1];
  v2 = *a1;
  v3 = a1[2];
  v4 = a1[3];
  v5 = a1[4];
  v6 = a1[6];
  v7 = a1[5];
  v8 = a1[7];
  v9 = a1[8];
  result = 0;
  if ( -85 * v9 + 58 * v8 + 97 * v6 + v7 + -45 * v5 + 84 * v4 + 95 * v2 - 20 * v1 + 12 * v3 == 12613 )
  {
    v11 = a1[9];
    if ( 30 * v11 + -70 * v9 + -122 * v6 + -81 * v7 + -66 * v5 + -115 * v4 + -41 * v3 + -86 * v1 - 15 * v2 - 30 * v8 == -54400
      && -103 * v11 + 120 * v8 + 108 * v7 + 48 * v4 + -89 * v3 + 78 * v1 - 41 * v2 + 31 * v5 - (v6 << 6) - 120 * v9 == -10283
      && 71 * v6 + (v7 << 7) + 99 * v5 + -111 * v3 + 85 * v1 + 79 * v2 - 30 * v4 - 119 * v8 + 48 * v9 - 16 * v11 == 22855
      && 5 * v11 + 23 * v9 + 122 * v8 + -19 * v6 + 99 * v7 + -117 * v5 + -69 * v3 + 22 * v1 - 98 * v2 + 10 * v4 == -2944
      && -54 * v11 + -23 * v8 + -82 * v3 + -85 * v2 + 124 * v1 - 11 * v4 - 8 * v5 - 60 * v7 + 95 * v6 + 100 * v9 == -2222
      && -83 * v11 + -111 * v7 + -57 * v2 + 41 * v1 + 73 * v3 - 18 * v4 + 26 * v5 + 16 * v6 + 77 * v8 - 63 * v9 == -13258
      && 81 * v11 + -48 * v9 + 66 * v8 + -104 * v6 + -121 * v7 + 95 * v5 + 85 * v4 + 60 * v3 + -85 * v2 + 80 * v1 == -1559
      && 101 * v11 + -85 * v9 + 7 * v6 + 117 * v7 + -83 * v5 + -101 * v4 + 90 * v3 + -28 * v1 + 18 * v2 - v8 == 6308 )
    {
      result = 99 * v11 + -28 * v9 + 5 * v8 + 93 * v6 + -18 * v7 + -127 * v5 + 6 * v4 + -9 * v3 + -93 * v1 + 58 * v2 == -1697;
    }
  }
  return result;
}
```


这个乍一看挺吓人的，但是分析问题要先总体分析，这么一些v1-v10的参数，乘上参数线性组合得到一个值，这不就是矩阵吗？跟着这个思路，我们可以分析到，要想返回值为True，那么两个if的条件一定要满足，return中的等式一定要为True。现在为证实思路正确性，数一数方程个数和未知数个数，确认可解性，发现正好是10个未知数，10个方程，无疑是可解的。

数据处理

但是10*10的矩阵手算是会死的！这里推荐用Excel做，毕竟这个人人都有，只要会用两个函数就行。

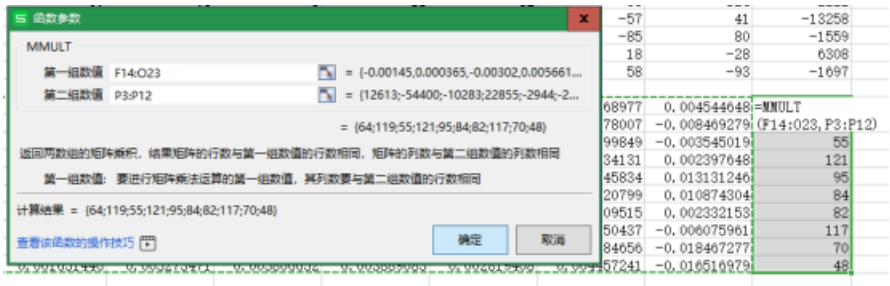
先把v11到v1的数据导入进去（这里我是按照v11的系数在前，v1的在后的顺序）

0	-85	58	1	97	-45	84	12	95	-20	12613
30	-70	-30	-81	-122	-66	-115	-41	-15	-86	-54400
-103	-120	120	108	-64	31	48	-89	-41	78	-10283
-16	48	-119	128	71	99	-30	-111	79	85	22855
5	23	122	99	-19	-117	10	-69	-98	22	-2944
-54	100	-23	-60	95	-8	-11	-82	-85	124	-2222
-83	-63	77	-111	16	26	-18	73	-57	41	-13258
81	-48	66	-121	-104	95	85	60	-85	80	-1559
101	-85	-1	117	7	-83	-101	90	18	-28	6308
99	-28	5	-18	93	-127	6	-9	58	-93	-1697

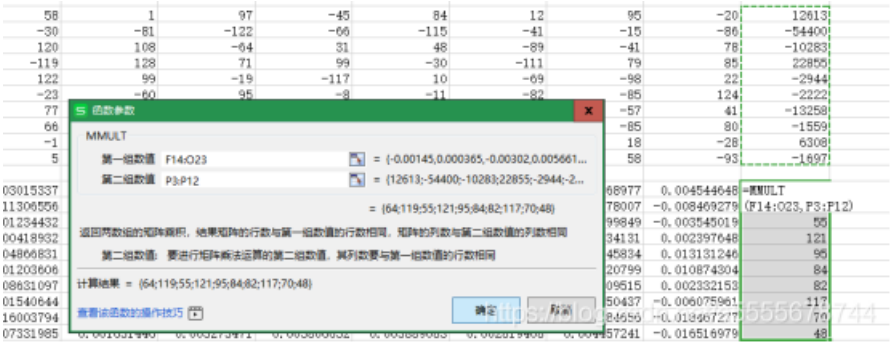
然后选中一个10*10的区域，对前面10个系数列使用MINVERSE函数

注意按确定的时候要同时按住ctrl和shift再按确定！

之后再选中一个10*1的列，使用MMULT函数



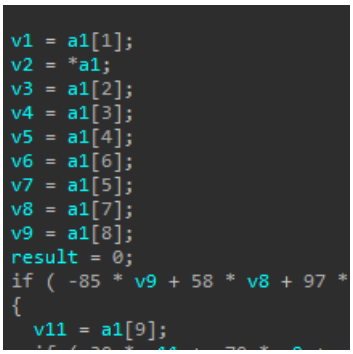
上面一组数据选择获得的逆矩阵（10*10），下面一组数据选择原矩阵的数据列



得到的结果

64
119
55
121
95
84
82
117
70
48

用python把ascii码转字符得到F0uRTy_7w@（其实直接解出来是0FuRT_y7w@）,但是ida里面定义v1-v11有打乱顺序



```
v1 = a1[1];
v2 = *a1;
v3 = a1[2];
v4 = a1[3];
v5 = a1[4];
v6 = a1[6];
v7 = a1[5];
v8 = a1[7];
v9 = a1[8];
result = 0;
if ( -85 * v9 + 58 * v8 + 97 *
{
  v11 = a1[9];
```

V1和V2,V6和V7都交换了位置，所以最后也要相应调整。

得到flag

但是开始分析的时候也说了，不可能通过静态调试得到完整的答案

```
if ( sub_860(v5) )
{
  sub_C50(v5, v4);
  __printf_chk(1LL, "Judgement pass! flag is actf{%s_%s}\n", v5);
}
else
```

因为第二部分的%s是没有给出来的，但是我们都知道正确的key了，直接运行这个程序不就知道了？

放到虚拟机里运行

```
[redacted]/tmp/UniverseFinalAnswer'
Please give me the key string: FOuRTy_7w@
Judgement pass! flag is actf{FOuRTy_7w@_42}
```

flag为actf{F0uRTy_7w@_42}