

# Bugku ctf--逆向第三题游戏过关题解

原创

[Air\\_cat](#) 于 2019-04-01 22:12:23 发布 1675 收藏

分类专栏: [逆向工程](#) [二进制CTF](#) 文章标签: [Bugku ctf writeup](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: [https://blog.csdn.net/Air\\_cat/article/details/88958097](https://blog.csdn.net/Air_cat/article/details/88958097)

版权



[逆向工程](#) 同时被 2 个专栏收录

25 篇文章 0 订阅

订阅专栏



[二进制CTF](#)

21 篇文章 0 订阅

订阅专栏

题目链接: <https://ctf.bugku.com/challenges>

先让我们来打开文件

```
C:\Users\DELL\Downloads\ConsoleApplication4.exe
```

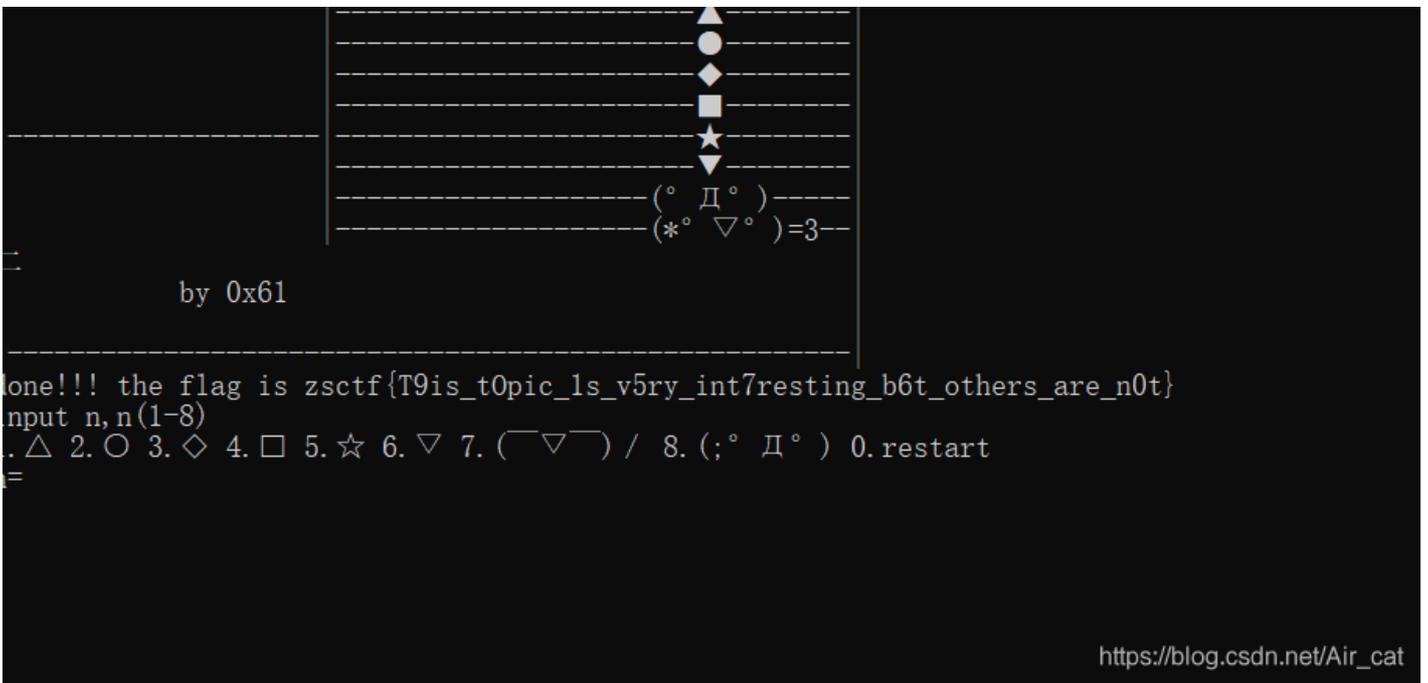
```
----- / ----- △
----- / ----- ○
----- / ----- ◇
----- / ----- □
----- / ----- ☆
----- / ----- ▽
----- / ----- (▽▽) / ---
----- / ----- (;° ▽°) -----

by 0x61

Play a game
The n is the serial number of the lamp, and m is the state of the lamp
If m of the Nth lamp is 1, it's on, if not it's off
At first all the lights were closed
Now you can input n to change its state
But you should pay attention to one thing, if you change the state of the Nth lamp, the state of (N-1)th and (N+1)th will
be changed too
When all lamps are on, flag will appear
Now, input n
input n, n(1-8)
1. △ 2. ○ 3. ◇ 4. □ 5. ☆ 6. ▽ 7. (▽▽) / 8. (;° ▽°) 0. restart
n=
```

[https://blog.csdn.net/Air\\_cat](https://blog.csdn.net/Air_cat)

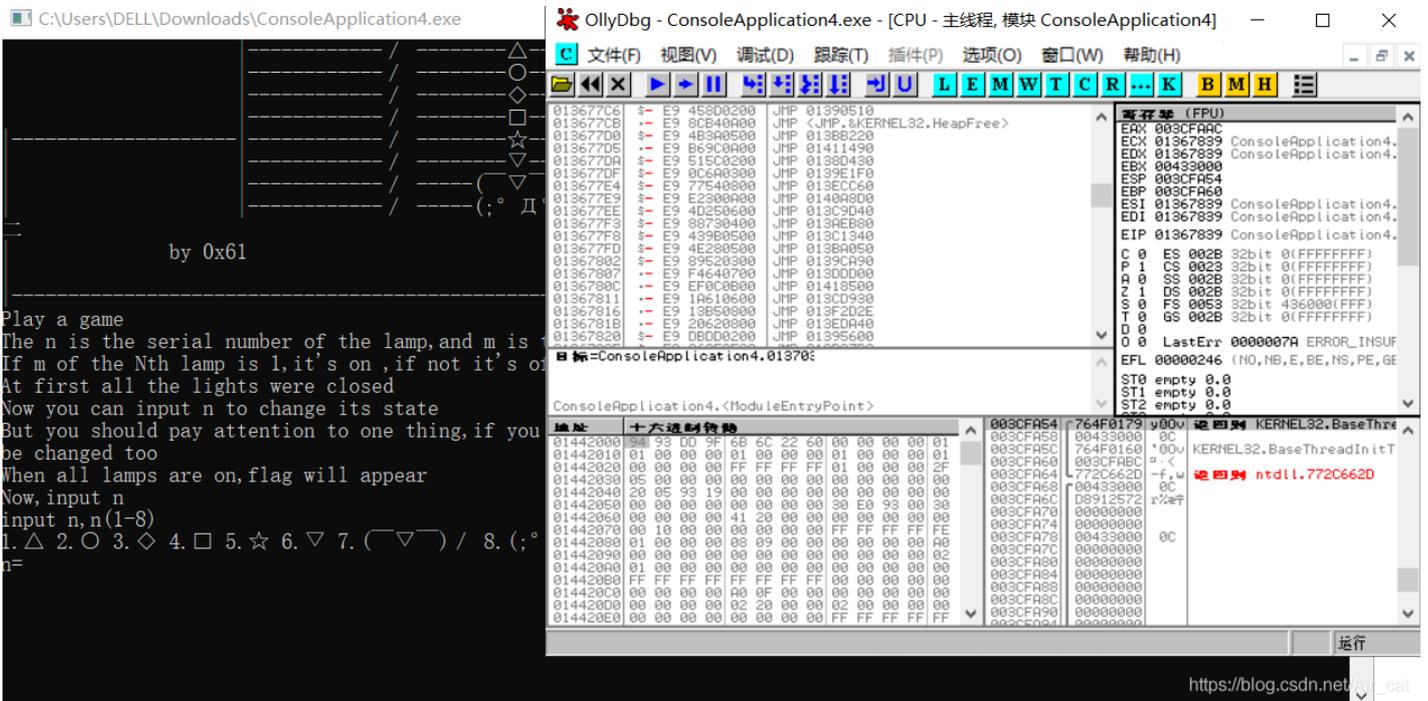
既然题目都说是游戏过关了, 那就玩玩这个游戏呗 (雾



游戏还是很简单的，但显然这道题应该用逆向的思维来做（废话

正式部分：

逆向的题拿到手一般能考虑的工具（方法）就是通过ida（静态分析）或者通过od（动态分析），因为ida还搞不太懂（菜鸟），然后看了下盲猜这题是要求运行的时候就能直接跳结果，开始od



开始的界面是这样的，字符串这么多，查个串呗

Address	命令	注释
0136E968	PUSH OFFSET 0141B0F0	ASCII "done!!! the flag is "
0136EBE8	PUSH OFFSET 0141B10C	ASCII "%e"
0136ED0E	PUSH OFFSET 0141AE00	ASCII "-----"
0136ED3D	PUSH OFFSET 0141AED0	ASCII "-----"
0136ED4A	PUSH OFFSET 0141AED4	ASCII "----- "
0136ED86	PUSH OFFSET 0141AEE8	ASCII "----- "
0136ED93	PUSH OFFSET 0141AED0	ASCII "-----"
0136EDC2	PUSH OFFSET 0141AED0	ASCII "-----"
0136EDDF	PUSH OFFSET 0141AED4	ASCII "----- "
0136EE0B	PUSH OFFSET 0141AEE8	ASCII "----- "
0136EE18	PUSH OFFSET 0141AED0	ASCII "-----"
0136EE46	PUSH OFFSET 0141AED0	ASCII "-----"
0136EE53	PUSH OFFSET 0141AED4	ASCII "----- "
0136EE8E	PUSH OFFSET 0141AEE8	ASCII "----- "
0136EE9B	PUSH OFFSET 0141AED0	ASCII "-----"
0136EECA	PUSH OFFSET 0141AED0	ASCII "-----"
0136ED07	PUSH OFFSET 0141AED4	ASCII "----- "
0136EF13	PUSH OFFSET 0141AEE8	ASCII "----- "
0136EF20	PUSH OFFSET 0141AF10	ASCII "----- "
0136EF4F	PUSH OFFSET 0141AED0	ASCII "-----"
0136EF5C	PUSH OFFSET 0141AED4	ASCII "----- "
0136EF98	PUSH OFFSET 0141AEE8	ASCII "----- "
0136EFA5	PUSH OFFSET 0141AF44	ASCII "----- "
0136EFD4	PUSH OFFSET 0141AED0	ASCII "-----"
0136EFE1	PUSH OFFSET 0141AED4	ASCII "----- "
0136F01D	PUSH OFFSET 0141AEE8	ASCII "----- "
0136F02A	PUSH OFFSET 0141AF44	ASCII "----- "
0136F059	PUSH OFFSET 0141AED0	ASCII "-----"
0136F066	PUSH OFFSET 0141AF78	ASCII "-----"
0136F0A2	PUSH OFFSET 0141AF40	ASCII "----- "
0136F0AF	PUSH OFFSET 0141AF44	ASCII "----- "
0136F0DE	PUSH OFFSET 0141AED0	ASCII "-----"
0136F0EB	PUSH OFFSET 0141AF78	ASCII "-----"
0136F127	PUSH OFFSET 0141AF08	ASCII "----- "
0136F141	PUSH OFFSET 0141B018	ASCII "-----  by 0x61  "
0136F14E	PUSH OFFSET 0141B060	ASCII "-----  by 0x61  "
0136F15B	PUSH OFFSET 0141B0A8	ASCII "-----  by 0x61  "
0136F41E	PUSH OFFSET 0141B110	ASCII "20"----- "
0136F42B	PUSH OFFSET 0141B158	ASCII "20"----- "
0136F438	PUSH OFFSET 0141B1A0	ASCII "20"----- "
0136F445	PUSH OFFSET 0141B1E8	ASCII "20"----- "
0136F493	PUSH OFFSET 0141B350	ASCII "-----  by 0x61  "
0136F498	PUSH OFFSET 0141B350	ASCII "-----  by 0x61  "
0136F4A0	PUSH OFFSET 0141B0A8	ASCII "----- "
0136F4B8	PUSH OFFSET 0141B398	ASCII "----- "
0136F4C7	PUSH OFFSET 0141B464	ASCII "Play a game!The n is the serial number of the lamp,and m is the state of the lamp!If m of th
0136F4D4	PUSH OFFSET 0141B498	ASCII "Now you can input n to change its state!"
0136F4E1	PUSH OFFSET 0141B53C	ASCII "But you should pay attention to one thing,if you change the state of the Nth lamp,th
0136F4EE	PUSH OFFSET 0141B56C	ASCII "When all lamps are on,flag will appear!"
0136F506	PUSH OFFSET 0141B57C	ASCII "Now,input n "
0136F51A	PUSH OFFSET 0141B590	ASCII "input n,n(1-8)"
0136F52B	PUSH OFFSET 0141B594	ASCII "n="
0136F551	PUSH OFFSET 0141B59C	ASCII "%d"
0136F5B7	PUSH OFFSET 0141B5BC	ASCII "sorry,n error,try again!"
01370554	PUSH OFFSET 0141BB10	ASCII "CLS"
01370579	PUSH OFFSET 0141BB68	ASCII "Stack area around _alloca memory reserved by this function is corrupted!"
01370587	PUSH OFFSET 0141BB74	ASCII "Data: <"
01370590	PUSH OFFSET 0141BB88	ASCII "Allocation"
01370593	PUSH OFFSET 0141BBB4	ASCII "Size: "
01370598	PUSH OFFSET 0141BBC8	ASCII "Address: 0"
0137059D	PUSH OFFSET 0141BCC0	ASCII "Stack area around _alloca memory reserved by this function is corrupted"
013705D4	PUSH OFFSET 0141BC38	ASCII "%s%s%p%s%d%s%d%s"
013705E3	PUSH OFFSET 0141BC3C	ASCII ">"
013705E6	CALL 0136A7BE	ASCII "%s%s%s"

[https://blog.csdn.net/Air\\_cat](https://blog.csdn.net/Air_cat)

这里查到input那里，设置断点

0136F445	E8 14B3FFFF	CALL 0136A7BE	
0136F44A	83C4 04	ADD ESP,4	
0136F44D	68 00B04101	PUSH OFFSET 0141B0A8	ASCII "!:----- "
0136F482	E8 07B3FFFF	CALL 0136A7BE	
0136F487	83C4 04	ADD ESP,4	
0136F48A	68 20B34101	PUSH OFFSET 0141B398	ASCII "Play a game!The n is the serial number of the lamp,and m is the state of the lamp!If
0136F48F	E8 F4B2FFFF	CALL 0136A7BE	
0136F4C4	83C4 04	ADD ESP,4	
0136F4C7	68 64B44101	PUSH OFFSET 0141B464	ASCII "Now you can input n to change its state!"
0136F4CC	E8 E0B2FFFF	CALL 0136A7BE	
0136F4D1	83C4 04	ADD ESP,4	
0136F4D4	68 20B44101	PUSH OFFSET 0141B498	ASCII "But you should pay attention to one thing,if you change the state of the Nth lamp,th
0136F4D9	E8 E0B2FFFF	CALL 0136A7BE	
0136F4DE	83C4 04	ADD ESP,4	
0136F4E1	68 20B54101	PUSH OFFSET 0141B53C	ASCII "When all lamps are on,flag will appear!"
0136F4E6	E8 D3B2FFFF	CALL 0136A7BE	
0136F4EB	83C4 04	ADD ESP,4	
0136F4EE	68 60B54101	PUSH OFFSET 0141B56C	ASCII "Now,input n "
0136F4F3	E8 C6B2FFFF	CALL 0136A7BE	
0136F4F8	83C4 04	ADD ESP,4	
0136F4FB	B8 01000000	MOV EAX,1	
0136F500	85C0	TEST EAX,EAX	
0136F502	0F84 6E010000	JZ 0136F676	
0136F506	68 70B54101	PUSH OFFSET 0141B57C	ASCII "input n,n(1-8)"
0136F512	E8 ACB2FFFF	CALL 0136A7BE	
0136F515	83C4 04	ADD ESP,4	
0136F518	E8 FE9EFFFF	CALL 01369418	
0136F51A	68 20B54101	PUSH OFFSET 0141B590	ASCII "n="
0136F51F	E8 9AB2FFFF	CALL 0136A7BE	
0136F524	83C4 04	ADD ESP,4	
0136F527	8045 F8	LEA EAX,[EBP-8]	
0136F52A	50	PUSH EAX	
0136F52B	68 24B54101	PUSH OFFSET 0141B594	ASCII "%d"
0136F530	E8 9FA1FFFF	CALL 013696D4	
0136F535	83C4 08	ADD ESP,8	
0136F538	68 20B54101	PUSH OFFSET 0141B598	
0136F53D	E8 7CB2FFFF	CALL 0136A7BE	
0136F542	83C4 04	ADD ESP,4	
0136F545	837D F8 00	CMP DWORD PTR SS:[EBP-8],0	
0136F549	7C 06	JL SHORT 0136F551	
0136F54B	837D F8 00	CMP DWORD PTR SS:[EBP-8],8	
0136F54F	7E 0F	JLE SHORT 0136F560	
0136F551	68 20B54101	PUSH OFFSET 0141B59C	ASCII "sorry,n error,try again!"
0136F556	E8 C3B2FFFF	CALL 0136A7BE	
0136F55B	83C4 04	ADD ESP,4	
0136F55E	EB 9B	JMP SHORT 0136F4FB	

同时可以发现第一行出现了"done!!! the flag is"的字符串，跳进去

```

3136F763 CC INT3
3136F764 CC INT3
3136F765 CC INT3
3136F766 CC INT3
3136F767 CC INT3
3136F768 CC INT3
3136F769 CC INT3
3136F76A CC INT3
3136F76B CC INT3
3136F76C CC INT3
3136F76D CC INT3
3136F76E CC INT3
3136F76F CC INT3
3136F770 55 PUSH EBP
3136F771 8BEC MOV EBP,ESP
3136F773 81EC 08000000 SUB ESP,008
3136F779 53 PUSH EBX
3136F77A 56 PUSH ESI
3136F77B 57 PUSH EDI
3136F77C 8080 28FFFFFF LEA EDI,[EBP-008]
3136F782 B9 36000000 MOV ECX,36
3136F787 B8 CCCCCCCC MOV EAX,CCCCCCCC
0136F78C F3:AB REP STOS DWORD PTR ES:[EDI]
3136F78E E8 9F9BFFFF CALL 01369332
3136F793 8D45 0C LEA EAX,[EBP+0C]
3136F796 8945 EC MOV DWORD PTR SS:[EBP-14],EAX
3136F799 8B45 EC MOV EAX,DWORD PTR SS:[EBP-14]
3136F79C 50 PUSH EAX
3136F79D 6A 00 PUSH 0
3136F79F 8B4D 08 MOV ECX,DWORD PTR SS:[EBP+8]
3136F7A2 51 PUSH ECX
3136F7A3 6A 01 PUSH 1
3136F7A5 E8 B084FFFF CALL 01367C67
3136F7AA 83C4 04 ADD ESP,4
3136F7AD 50 PUSH EAX
3136F7AE E8 4198FFFF CALL 01368FF4
3136F7B3 83C4 10 ADD ESP,10
3136F7B6 8945 F8 MOV DWORD PTR SS:[EBP-8],EAX
3136F7B9 8745 EC MOV DWORD PTR SS:[EBP-14],0
3136F7C8 8B45 F8 MOV EAX,DWORD PTR SS:[EBP-8]

```

[https://blog.csdn.net/Air\\_cat](https://blog.csdn.net/Air_cat)

选择复制0136F779，呆会我们要跳进这个地方，需要注意的是有些题解说跳进done flag的那行，但具体操作起来可能会行不通，（大概是代码段的入口点？）个人水平实在有限暂时无法解答，先将这个疑问记下来。

后来稍稍查证了一下，该处确实为该代码段的入口

```

136F444 83C4 04 ADD ESP,4
136F44D E8 A8B41010 PUSH OFFSET 0141B0A8
136F4B2 E8 07B3FFFF CALL 0136A7BE
136F4B7 83C4 04 ADD ESP,4
136F4BA E8 98B34101 PUSH OFFSET 0141B398
136F4BF E8 FAB2FFFF CALL 0136A7BE
136F4C4 83C4 04 ADD ESP,4
136F4C7 E8 4AB44101 PUSH OFFSET 0141B464
136F4CC E8 EDB2FFFF CALL 0136A7BE
136F4D1 83C4 04 ADD ESP,4
136F4D4 E8 98B44101 PUSH OFFSET 0141B498
136F4D9 E8 E8B2FFFF CALL 0136A7BE
136F4DE 83C4 04 ADD ESP,4
136F4E1 E8 2CB54101 PUSH OFFSET 0141B53C
136F4E6 E8 D3E2FFFF CALL 0136A7BE
136F4EB 83C4 04 ADD ESP,4
136F4EE E8 4CB54101 PUSH OFFSET 0141B56C
136F4F3 E8 C6B2FFFF CALL 0136A7BE
136F4F8 83C4 04 ADD ESP,4
136F4FB B8 01000000 MOV EAX,1
136F500 85C9 TEST EAX,EAX
136F502 0F84 6E010000 JZ 0136F676
0136F508 E8 33C4FFFF JMP 0136E940
136F50D E8 ACB2FFFF CALL 0136A7BE
136F512 83C4 04 ADD ESP,4
136F519 E8 FE9EFFFF CALL 01369418
136F51D E8 88B54101 PUSH OFFSET 0141B590
136F51F E8 9AB2FFFF CALL 0136A7BE
136F524 83C4 04 ADD ESP,4
136F527 8D45 F8 LEA EAX,[EBP-8]
136F52A 50 PUSH EAX
136F52D E8 94B54101 PUSH OFFSET 0141B594
136F530 E8 9FA1FFFF CALL 013696D4
136F531 83C4 04 ADD ESP,4

```

[https://blog.csdn.net/Air\\_cat](https://blog.csdn.net/Air_cat)

之后，让我们随便找一行代码改成跳转到刚才复制下来的地址，F8F8F8，完成。

The screenshot shows a debugger window with assembly code on the left and a hex editor on the right. The assembly code shows a jump instruction being modified. The hex editor shows the corresponding bytes being changed to F8F8F8. The program output is visible in the background.