感想

题目很有意思，省略了很多前期的复杂步骤，让学生只要专注思考如何进行栈溢出的操作即可，（hex2raw函数可以保留下来以后留着用😀）。还得感谢一下那些国外的老师们，出了完美封装后的题目给学生练习

ctarget

（代码注入CI）

phase1

比较简单，只要跳到touch1函数即可

phase2

在跳到touch2前需要先把0x59b997fa放到rdi。

phase3

需要先把0x59b997fa的字符形式写到栈中，然后把地址赋给rdi寄存器，然后跳转到touch3

rtarget

（面向返回编程ROP）
这是一个比较有意思的东西，大概原理：利用若干个返回指令（ret 操作码:c7）前几个字节的代码，构造你想要的指令，然后执行。
这个技术的出现主要是为了解决栈随机化（CI就不能写入固定地址）以及使栈地址中的数据不可执行
难点在于如何构造ROP链（难怪会叫做ROP链，是若干条指令的链状结构）

phase4

还是要进入touch2，但是这时因为使用了站随机化和栈段内容不可执行，所以原phase2的代码无法使用。
这里使用的办法是构造
POP %RAX
MOV %RDI,%RAX
因为使用了POP指令，所以需要在两端指令的地址中间添加数据0x59b997fa

phase5

要进入touch3，难点在于用到了pdf中没有用到的指令，
这里构建了
MOV %RAX,%RSP
ADD %RAX,$0X37
MOV %RDI,%RAX
根据这些指令构造ROP链，详情见answer文件夹下的内容

注意

1.上述代码使用的是编写汇编语言的格式，与objdump反汇编后的内容，源寄存器和目的寄存器的放置顺序相反（看answer文件夹中的内容也请注意）
2.phase1~5在answer文件夹中的名称为touch1,touch2,touch3,level2,level3