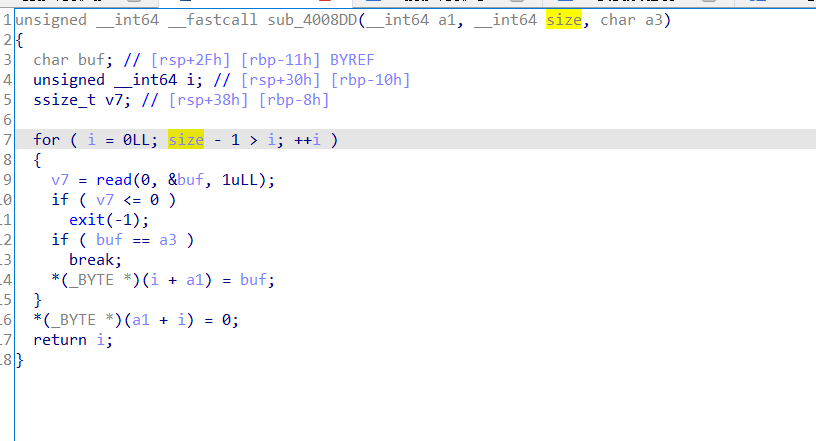
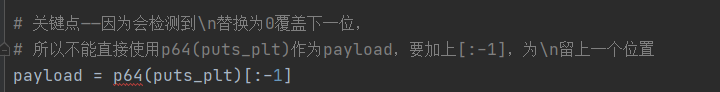
本题使用unlink进行攻击，但如何达到unlink的条件需要仔细观察，因为本题不存在常规意义上的溢出条件

1. size为0的无限制溢出

在read\_content的函数中可以看到当size为0时，size-1为超大无符号数，进而可以达到溢出修改下面的chunk的效果

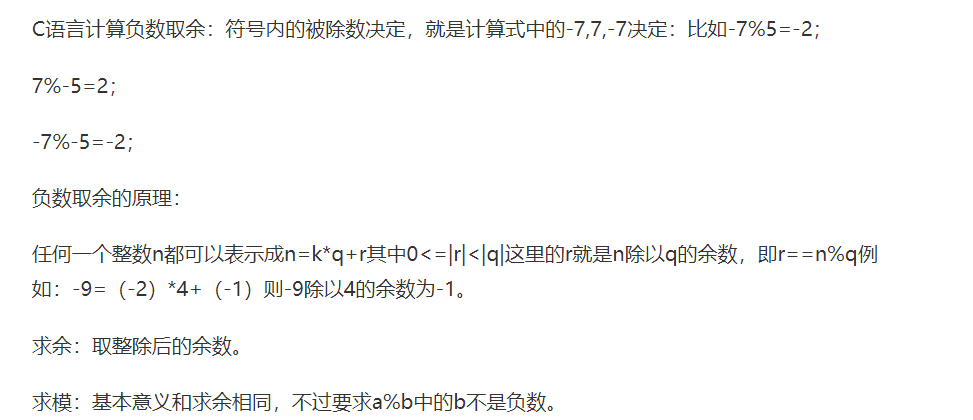


此外还有一点需要注意即在read\_content函数中，如果内容大小比size小时会遇\n终止输入并自动补0，故在unlink攻击成功后不能直接使用p64(puts\_plt)对free\_got进行替换，而是要添加[:-1]将最低位删掉为\n留一个位置，同理后面的system也是一样，当时就是因为这个问题卡了很久没发现为什么攻击失败

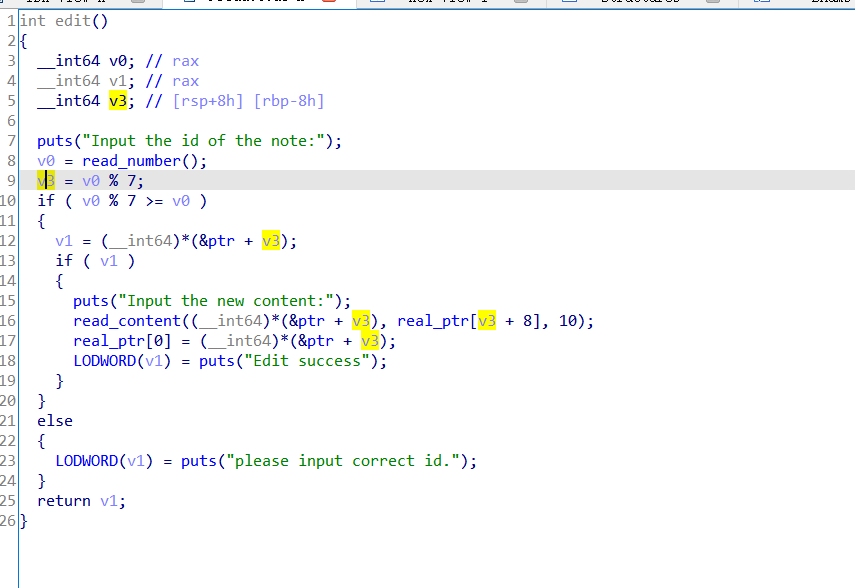


1. 整数转为负数的整数溢出

这个溢出在这道题则是很隐蔽，之前我也不知道这个知识点——c语言中负数取余还是负数



故如果输入特殊数字-0x8000000000000000，经过edit函数的read\_number后还是负数，%7后为-1，恰好本题在ptr的上方存在real\_ptr的存在，即ptr[-1]，所以可以通过构造chunk伪造ptr[-1]的size从而溢出



参考链接

<https://blog.csdn.net/seaaseesa/article/details/105588787>

<https://blog.csdn.net/One_p_Two_w/article/details/121142272>