




170629 Pwn-XCTF决赛的Pwn视频分析

原创

奈沙夜影  于 2017-06-30 17:16:08 发布  982  收藏

分类专栏: [CTF](#)

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/whklhjh/article/details/74002423>

版权



[CTF 专栏收录该内容](#)

163 篇文章 4 订阅

订阅专栏

1625-5 王子昂 总结《2017年6月29日》【连续第270天总结】

A. XCTF总决赛 Pwn

B. 看二进制指导的时候误打误撞看了一个Pwn的视频writeup, 不过技术原理上与逆向相同的, 区别只在于Pwn主要对程序的反编译来找到程序的漏洞(如栈溢出、逻辑BUG等)从而获取隐藏的flag, 而Reverse是拆解理解程序的功能来获得flag

本题比较简单, 是通过栈溢出的漏洞; 环境为Linux

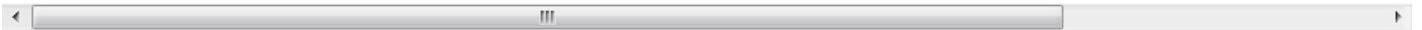
首先实现File命令来查看该文件的属性: 64位ELF

直接运行, 发现毫无输出退出; 使用strace来输出该程序所调用的API, 发现它试图打开了/home/flag-robot/flag文件

那么在该位置touch一个flag文件即可

再次运行, 显示文字, 表示需要输入指令

此时前期准备就无法继续了, 需要使用IDA进行静态反编译---由于竞赛文件已无处可寻, I春秋提供的实验环境中的IDA无法进



大体上思路为分析源码, 寻找可利用的漏洞

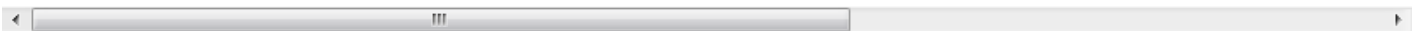
本文件先声明了三个数组dest、shadow和stack_cookie, 每次接收三个数字作为指令, 当满足一定条件时可以触发不同的指令

一共有三个漏洞:

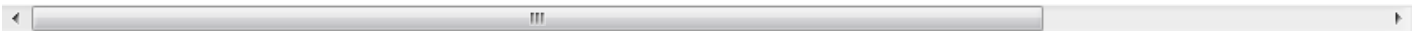
首先是char overflow 当一个变量为char型时, 当它的值发生溢出时将会变为负值。因此若将其作为数组的下标加数将会导致向前



第二个漏洞是stack memory leak。与上个漏洞类似, 但是当下标不可为负值时, 如果程序可以递归调用那么此时栈堆中将会出现



第三个漏洞, stack memory write。当对堆栈写入时如果发生溢出, 那么将可以修改函数返回地址。此时可以寻找内存中的system



C. 明日计划

春秋逆向简介