《漏洞利用及渗透测试基础》实验报告

姓名： 艾明旭 学号：2111033 班级：信息安全

**实验名称：**

格式化字符串漏洞

**实验要求：**

**数据泄露利用实验（任意内存数据获取）：**可以利用格式化函数，精心设计输入，使得

输入的地址处的数据被获取

以第四章示例4-7代码，完成任意地址的数据获取，观察Release模式和Debug模式的差异，并进行总结。

**实验过程：**

**代码：**

#include <stdio.h>

int main(int argc, char \*argv[])

{

char str[200];

fgets(str,200,stdin);

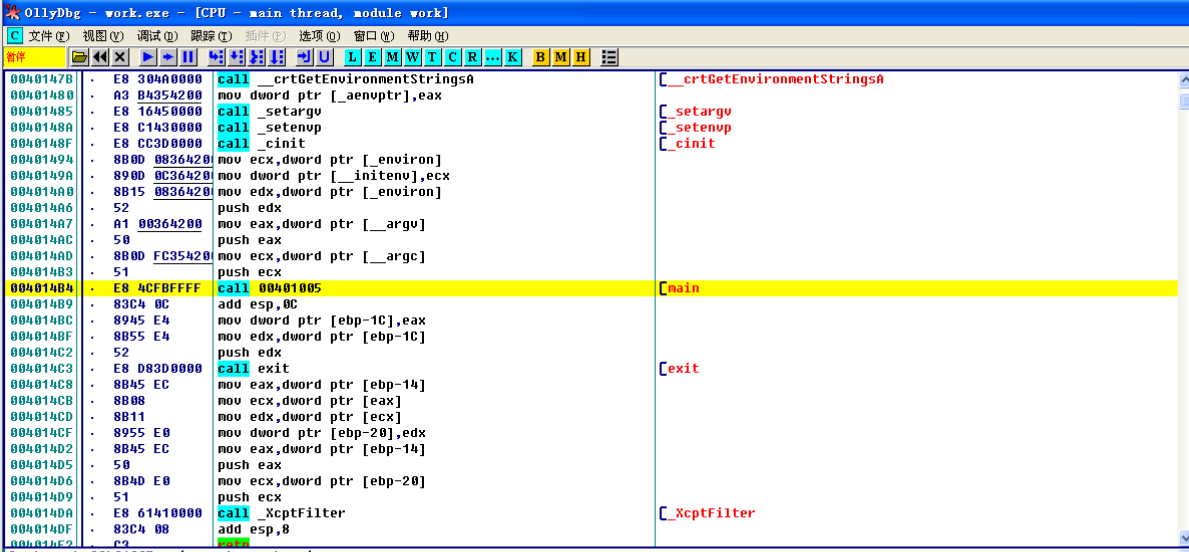
printf(str);

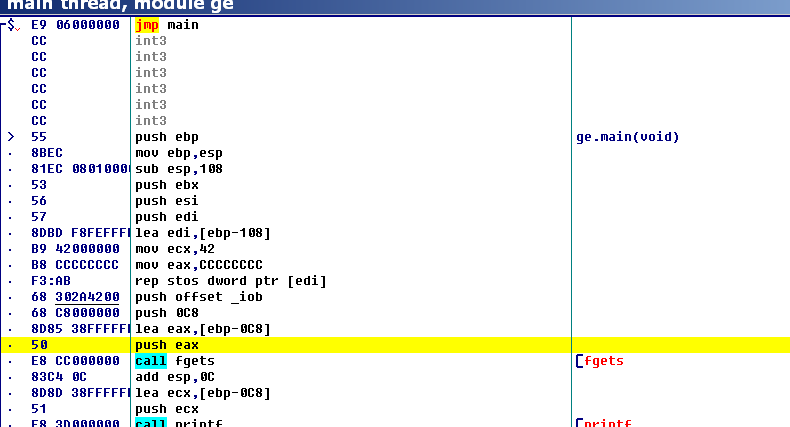
return 0;

}

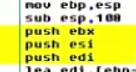
1. Debug模式：Debug模式下，因为开辟了足够大的栈帧并初始化，char str[200]是从靠近EBP的地址分配空间，如果要读到str的地址，需要很多的格式化字符；

找到main函数入口

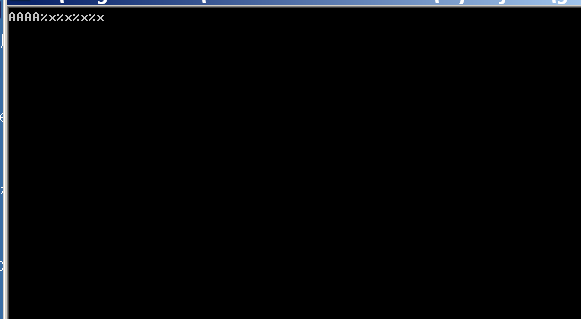




由Sub ebp 108 可知开辟了很大的一块区域



输入字符串



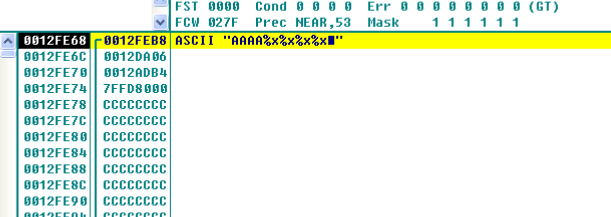
0x0012FB8中存取了字符串

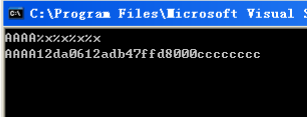


调用add以后还原栈帧状态

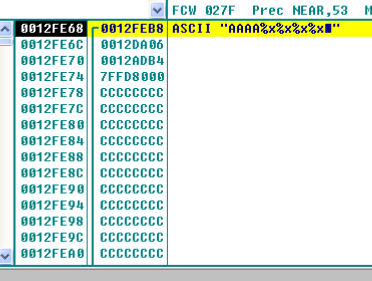


执行结果



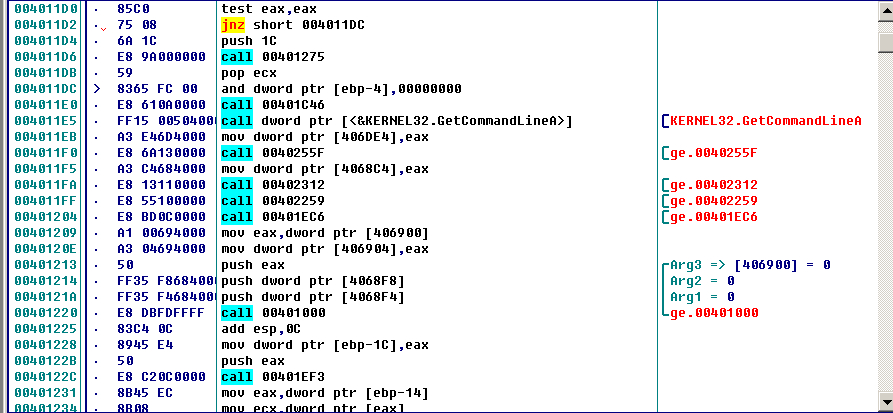


可以看到Debug模式下要读到str的地址，需要很多的格式化字符

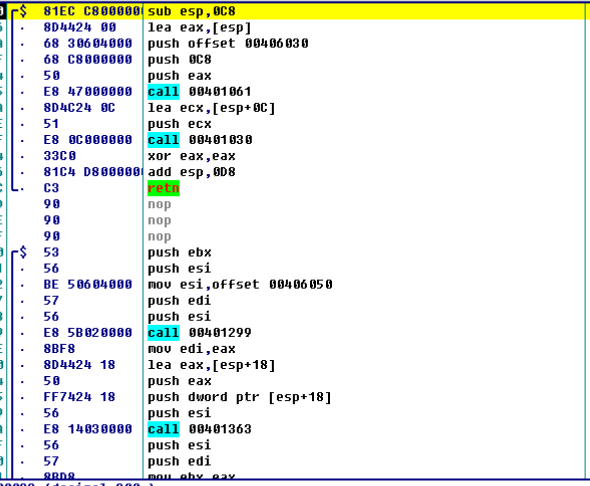


2.Release模式：Relase模式下，可以看到，并没有严格按照制式的栈帧分配，而是考虑运行性能，在执行到printf(str)的时候，栈区自顶到底部分为存着“printf函数参数|fgets函数参数|str数组”的内容，在Main函数的retn语句前，才有一个add esp XX的处理。

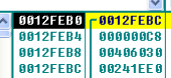
首先找到main函数，转入

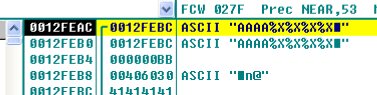


没有EBP入栈 栈顶抬高幅度比Debug模式 仅给局部变声明了空间



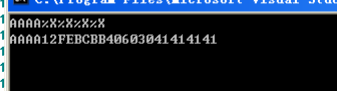
0x0012FEBC为要输入的字符串的地址

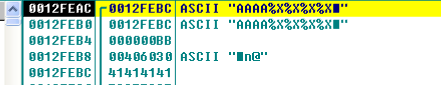




执行结果

输出结果（若把%x改为%s输出的将会是AAAA）





编译后运行（Release 模式）并输入：AAAA%x%x%x%x

我们成功读到了 AAAA：AAAA18FE84BB40603041414141（0x41 就是 ASCII 的字母 A 的值）。

思考：41414141是怎么读到的？考虑栈帧状态，参数入栈（字符串str的地址）后，通过%x依次读参数下面的内存数据时，很快就读到了原来函数的局部变量str的数据了。

执行printf(str)语句的时候，对比Debug模式和Relase模式的栈帧结构，如下图所示。



**心得体会：**

通过本次实验，我成功了解了Release模式和Debug模式的差异，在栈帧和参数上的不同。由此产生了存储方式以及读取数据时的差异。

同时通过本次实验，我还了解到了字符串溢出可能造成的危害