《漏洞利用及渗透测试基础》实验报告

姓名：袁田 学号：2314022 班级：计科三班

**实验名称：**

格式化字符串漏洞实验

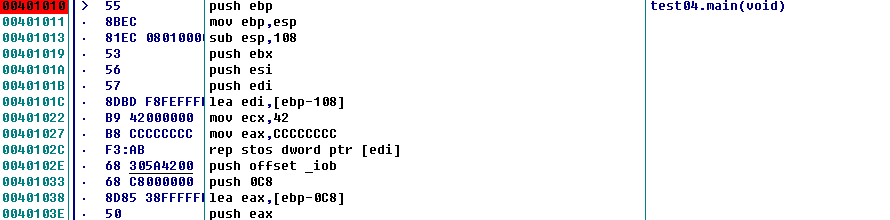
**实验要求：**

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：**

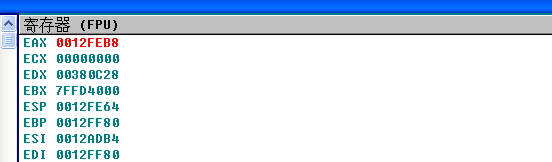
1. 将编译好的文件在OllYDBG中打开使用debug模式，进行该模式下的观察

(1)在debug模式下，首先ebp入栈，保存函数调用之前的栈底地址，将ebp赋值为esp,并将esp抬高，留出了108大小的空间，随后edi,ebx,esi等寄存器入栈，edi保存[ebp-108](即esp的地址)

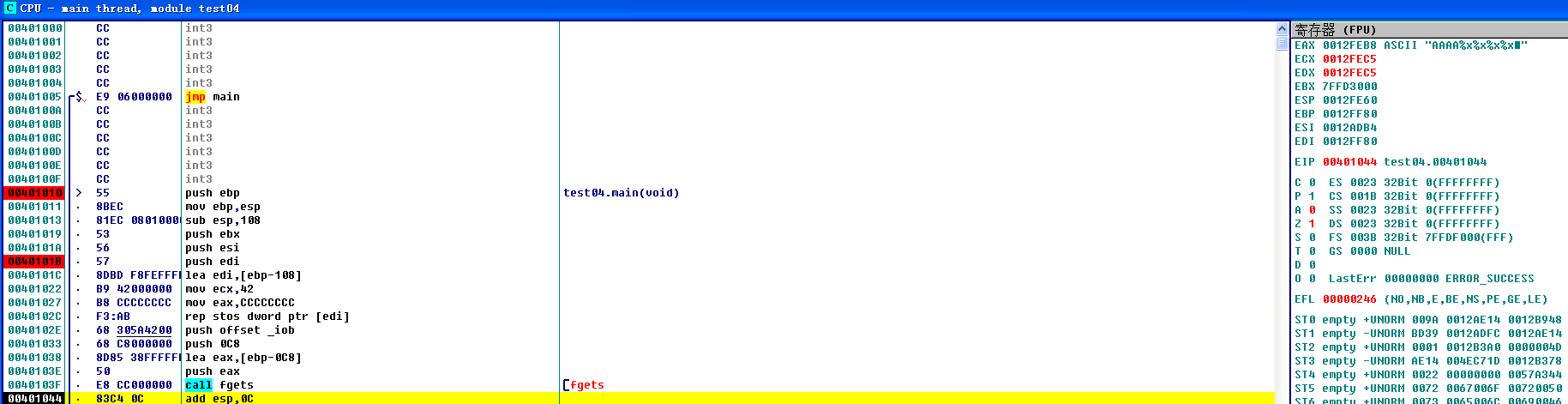


将eax寄存器初始化设置为cccccccc,即进行初始化，将函数栈帧用cccccccc填充。

用push将fgets函数的参数入栈。在call fgets之前，寄存器的情况为：



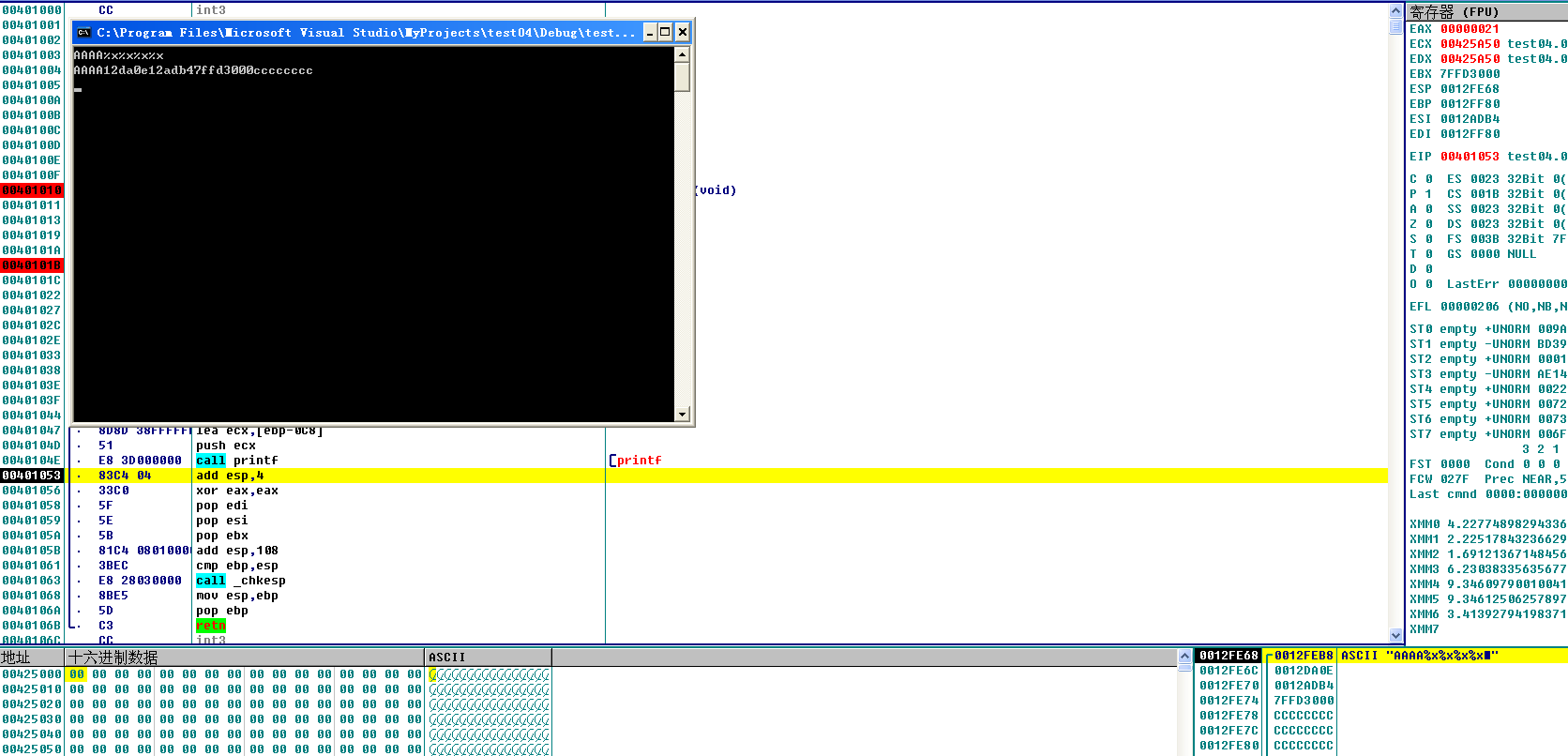
(2) 随后调用fgets函数，传入数据AAAA%x%x%x%x%x，此时可以发现，执行完fgets函数后，栈中存放着输入的字符串



（3）在调用了fgets函数并将数据传入栈中后，恢复栈帧并将printf的参数入栈



（4）随后调用printf函数，获得结果如下：



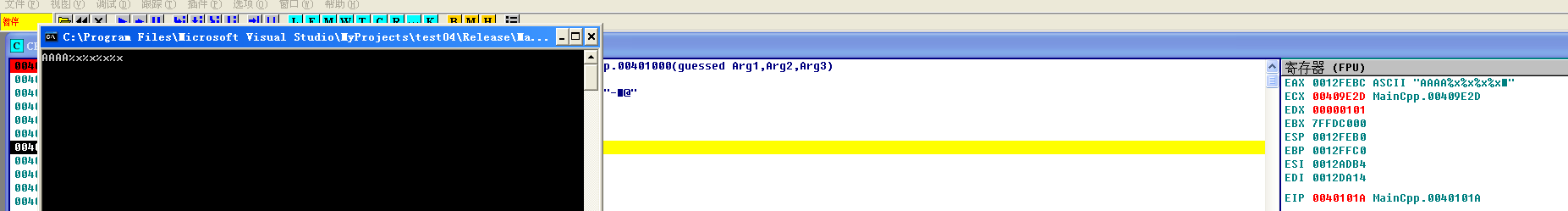
其中AAAA为正常的输出，由于fgets函数读取了四个x%，但printf函数并没有给出对应的十六进制数，因而printf函数将从栈中往高地址处继续读取，获得栈后续空间的地址，分别为0xFFFFFFFF,0x00000015，0x7FFDD000,0xcccccccc。另外，如果将AAAA替换为任意构造的地址而最后一个%x替换为%s,则可以将构造地址的数据打印出来，但是由于Debug模式下，可以看到由于sub esp 108指令，实际上的开辟空间远大于200，导致栈帧除了局部变量外还填充了填充了大量的cccccccc(初始化时将函数栈帧用cccccccc填充)，因此无法读到对应位置的数据。

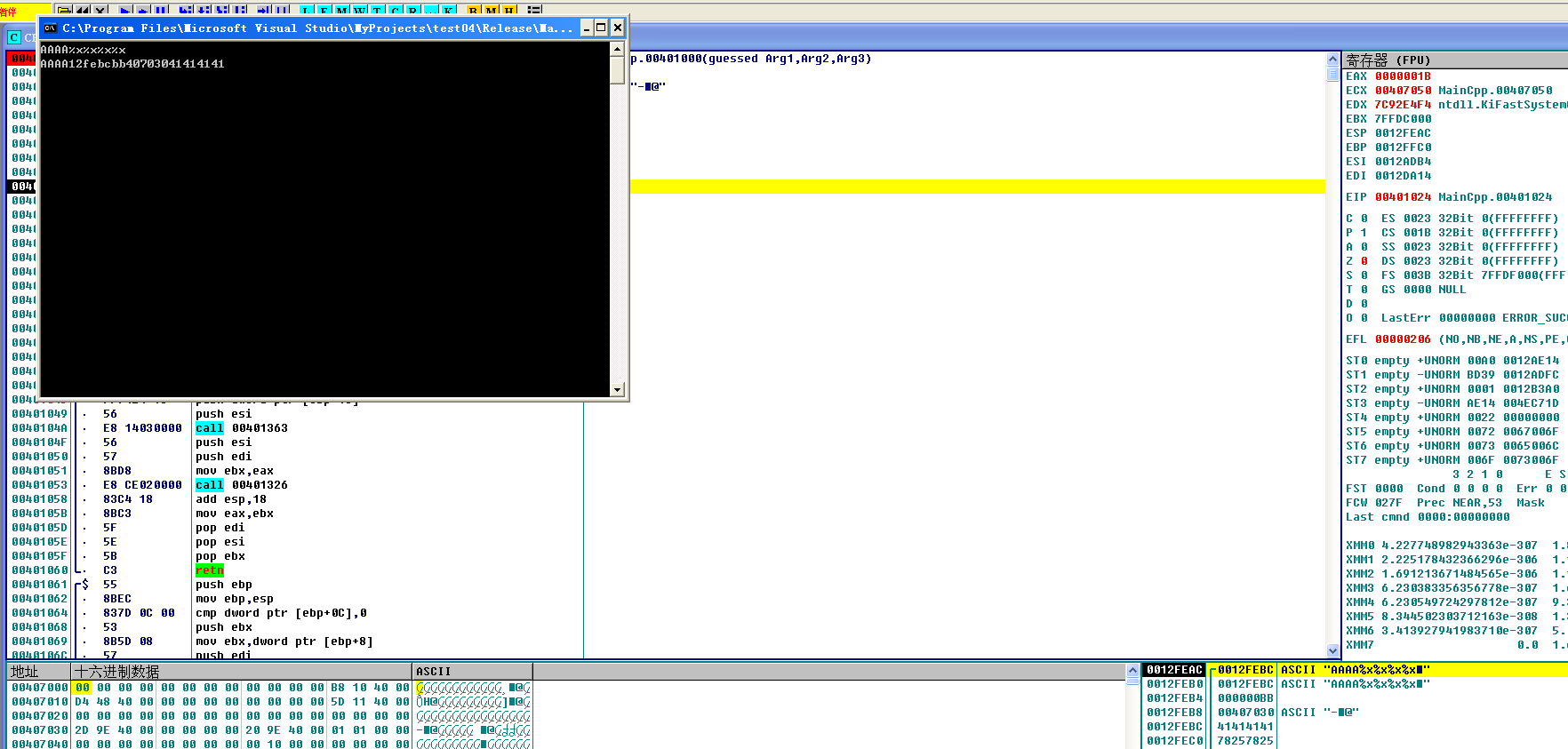
2.使用release模式

(1)在visual c++中通过设置release模式进行运行获得release下的maincpp文件，并将其导入至OLLYDBG，可以看到相对于debug模式，release模式下省略了edi,ebx等寄存器的入栈，并且没有ebp的入栈，栈帧也仅抬高了0c8(正好为200字节，用于容纳fgets输入的字符数组)



(2)然后通过fgets函数(即call 00401061指令)传入输入进去的字符串，此时栈中为：



(3)随后运行printf函数，输出结果最后为41414141，即AAAA的ASCII码的十六进制

**心得体会：**

**对于debug模式和release模式的区别有了更清楚的理解，并且知道了该如何利用printf等格式化的函数获取信息。**