《漏洞利用及渗透测试基础》实验报告

姓名：张洋　　学号：2111460　　班级：信安二班

实验名称：

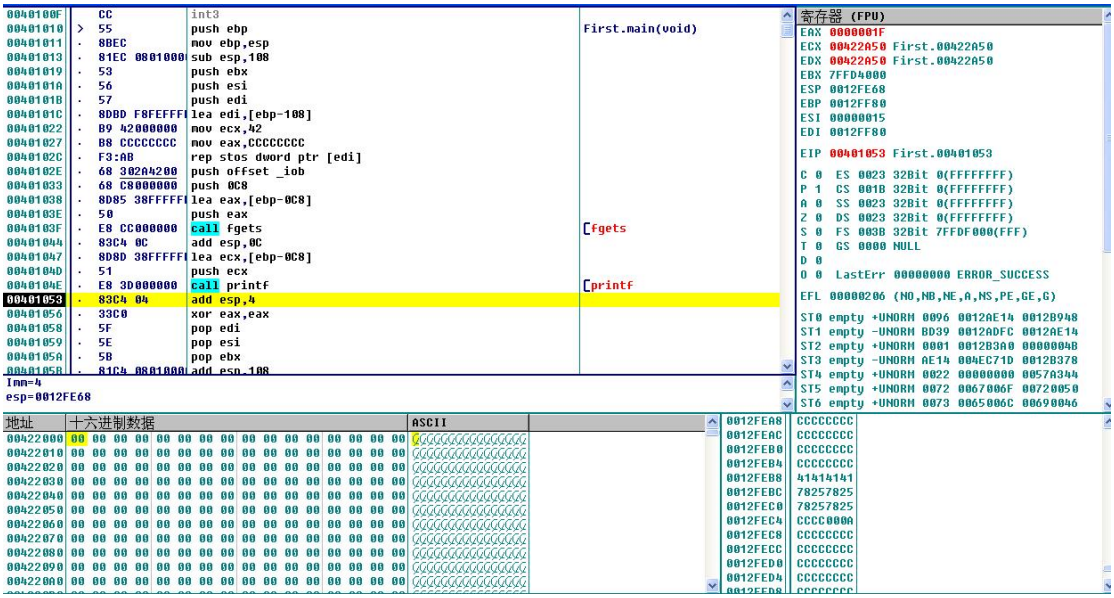
格式化字符串漏洞实验

实验要求：

根据第四章示例 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：

一、Debug 模式



逐过程分析：

1. 进入主函数，栈帧初始化，设置大小为 0x108

```
00401010 > 55          push ebp                    First.main(void)
00401011 · 8BEC        mov ebp,esp
00401013 · 81EC 0801000 sub esp,108
```

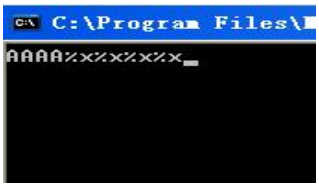
2. 存放原栈帧信息,利用 ecx 和 rep 指令把大小为 0x108 的栈空间初始化为 CCCCCCCC。

```
00401019 · 53          push ebx
0040101A · 56          push esi
0040101B · 57          push edi
0040101C · 8DBD F8FEFFF lea edi,[ebp-108]
00401022 · B9 42000000 mov ecx,42
00401027 · B8 CCCCCCCC mov eax,CCCCCCCC
0040102C · F3:AB       rep stos dword ptr [edi]
```

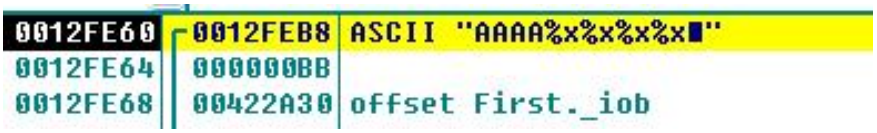
3. 初始化结束后，因为 str 字符数组的大小为 200，所以将栈帧存入 200(0xC8)位空间后的地址赋值给 eax，eax 为 0012FED8。

```
0040102E · 68 302A4200 push offset _iob
00401033 · 68 C8000000 push 0C8
00401038 · 8D85 38FFFFF lea eax,[ebp-0C8]
0040103E · 50          push eax
```

4. 调用 fgets 函数，输入一个字符串。

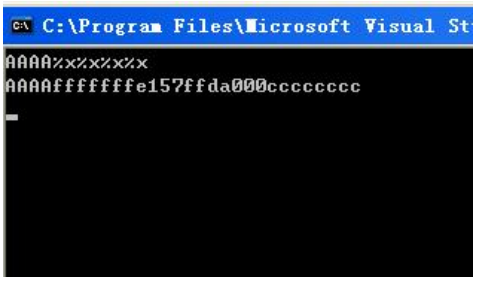


输入“AAAA%x%x%x%x”可以发现地址 0012FED8 上存放着输入的字符串。

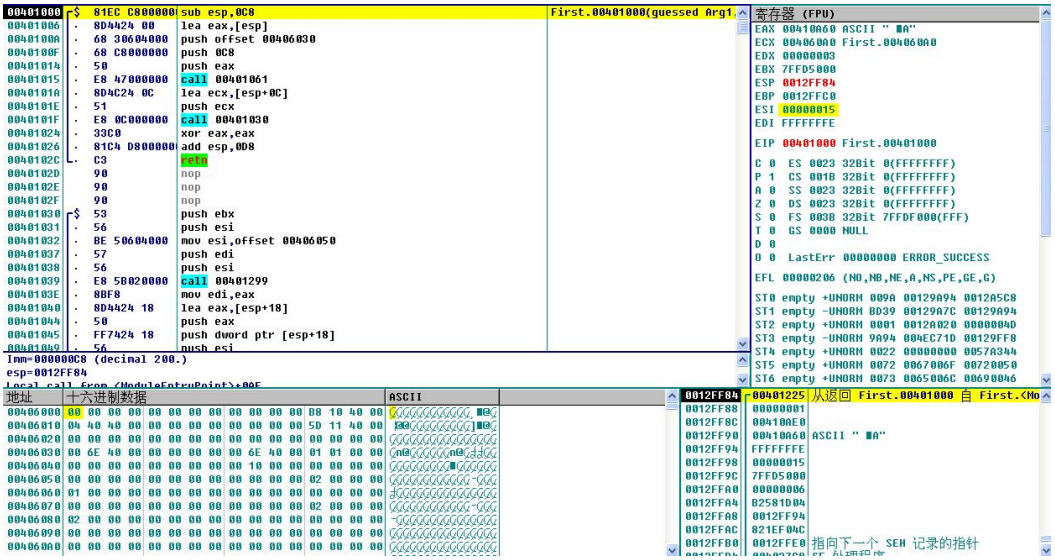


5. 调用完 fgets 函数后恢复 esp 的值，并将字符数组的起始地址作为 printf 函数的参数入栈。

6. 调用 printf 函数，打印出的内容如下图所示。打印出了输入的字符串“AAAA”，遇到%x 时会自动依次打印出栈中接下来的地址中存放的内容（依次打印 edi,esi,ebx 的存储内容和 CCCCCCCC）



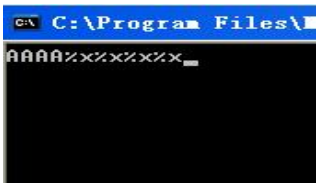
二、Release 模式



1. 进入主函数，release 模式下没有栈帧的切换，不将 ebp 入栈，只为字符数组声明 200 字节的空间，从而将 esp 抬高 200 字节，再将此时 esp 的值赋值给 eax，eax=0012FEBC。

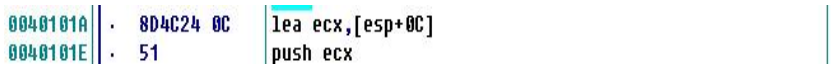
2. 把地址，立即数，eax 寄存器入栈。

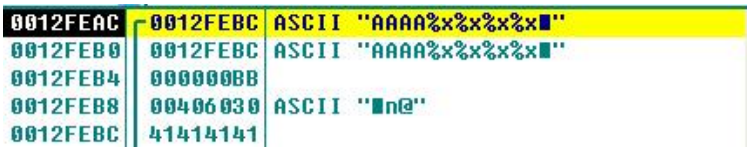
3. 调用 fgets 函数，输入“AAAA%x%x%x%x”。



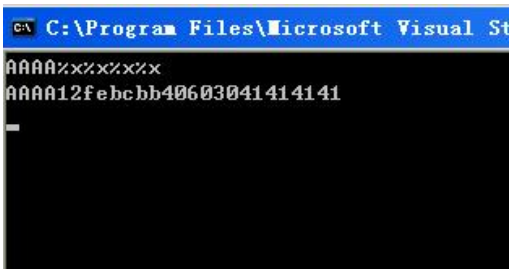
此时栈顶（0012FEB0）的存储内容为已经输入的字符串的地址（0012FEBC）。

4. 将字符串的地址赋值给 ecx，此时 ecx=0012FEBC，将 ecx 入栈，此时栈中存放了两个字符串的地址。





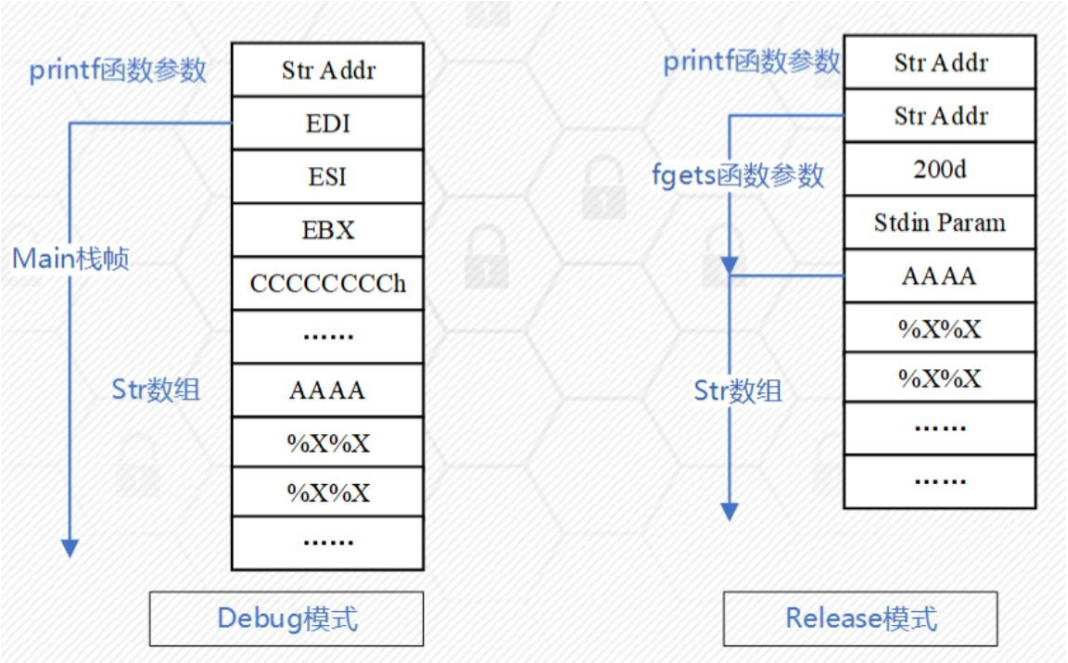
5. 调用 printf 函数，可以看到打印出了输入的字符串“AAAA”，遇到%x 时会自动依次打印出栈中接下来的地址中存放的内容（字符串地址和压入的变量）。



6. eax 寄存器中的内容清空，esp 恢复成原来的地址，最后 return。

心得体会：

执行 printf(str)语句的时候，对比 Debug 模式和 Release 模式的栈帧结构：



结合此题而言，我记录了 Debug 模式和 Release 模式下栈的地址和内容，以便更加直观地查看：

1. Debug 模式

0012FE68	0012FEB8（字符串地址）
0012FE6C	FFFFFFFE（最开始 edi 的值）
0012FE70	00000015（最开始 esi 的值）
0012FE74	7FFDA000（最开始 ebx 的值）
0012FE78	CCCCCCCC
……	CCCCCCCC
0012FEB4	CCCCCCCC
0012FEB8	41414141（“AAAA”）
0012FEBC	78257825（‘%’->25，‘x’->78）
0012FEC0	78257825（‘%’->25，‘x’->78）
0012FEC4	CCCC000A（0A 代表输入结束）
0012FEC8	CCCCCCCC
……	CCCCCCCC
0012FE7C	CCCCCCCC

2. Release 模式

0012FEAC	0012FEBC（字符串地址）
0012FEB0	0012FEBC（字符串地址）
0012FEB4	000000BB
0012FEB8	00406030
0012FEBC	41414141（“AAAA”）
0012FEC0	78257825（‘%’->25，‘x’->78）
0012FEC4	78257825（‘%’->25，‘x’->78）

通过本次实验，我深入地了解了 Debug 模式和 Release 模式下栈是如何变化的，以及他们之间存在怎样的区别。Debug 模式下栈帧 ebp 会随着函数不断调整，并且会初始化一部分空间为 CCCCCCCC；在 Release 模式下 ebp 不会变化，并且不用初始化空间，直接抬高 esp 来分配需要的字符数组的空间。