《格式化字符串漏洞》实验报告

姓名：田晋宇　　学号：2212039　　班级：物联网班

实验名称：

格式化字符串漏洞——任意地址的数据获取

实验要求：

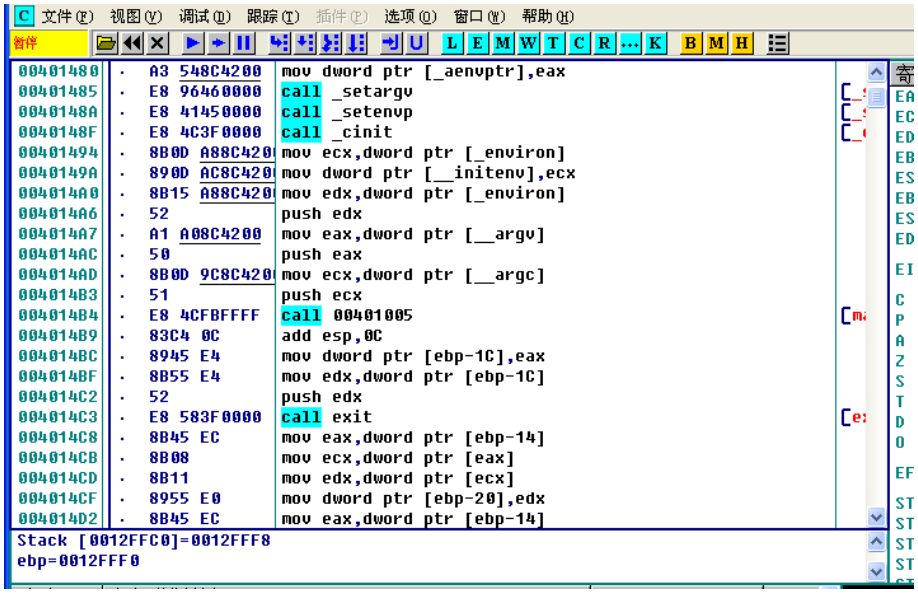
以第四章示例 4-7 代码，完成任意地址的数据获取，观察 Release 模式和 Debug 模式的差异，并进行总结。

实验过程：

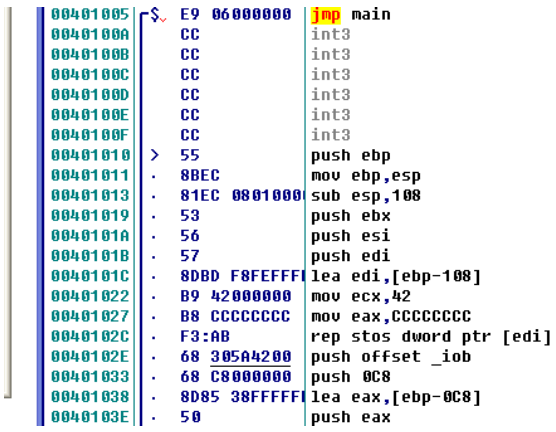
格式化字符串输出获取任意地址的数据是在调用格式化函数时，给出了格式化符号串，但没有提供对应参数时，这些函数会将格式化字符串后面的多个栈中的内容取出作为参数，并根据格式化符号将其输出。

Debug 模式：

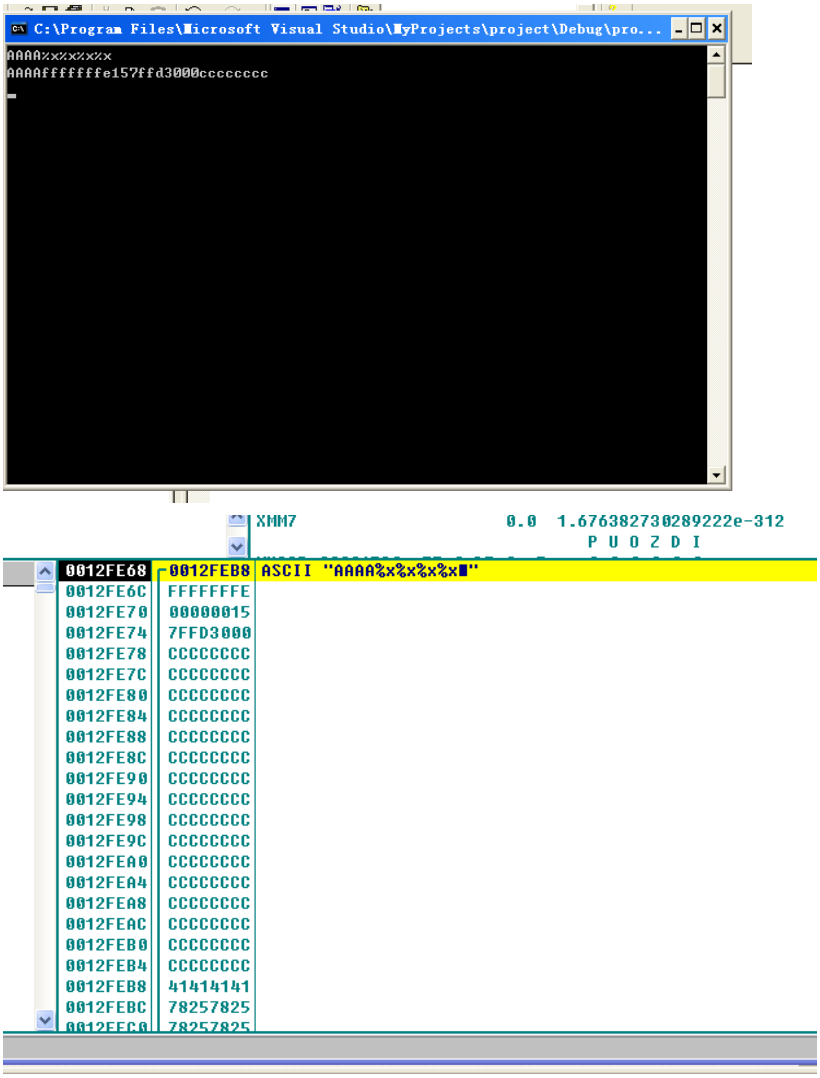
1. 首先生成 Debug 模式下的可执行文件，在 OllyDbg 中打开，找到主函数的入口，进入。



2. 进入主函数内部首先是对主函数调用之前的一些寄存器进行保存，并对栈空间进行初始化。可以看到在初始化栈时将栈抬高了 108 的空间，远远超过了字符串初始化的空间。在调用 fgets 前，通过计算将输入字符串的起始地址存入 eax 寄存器中。

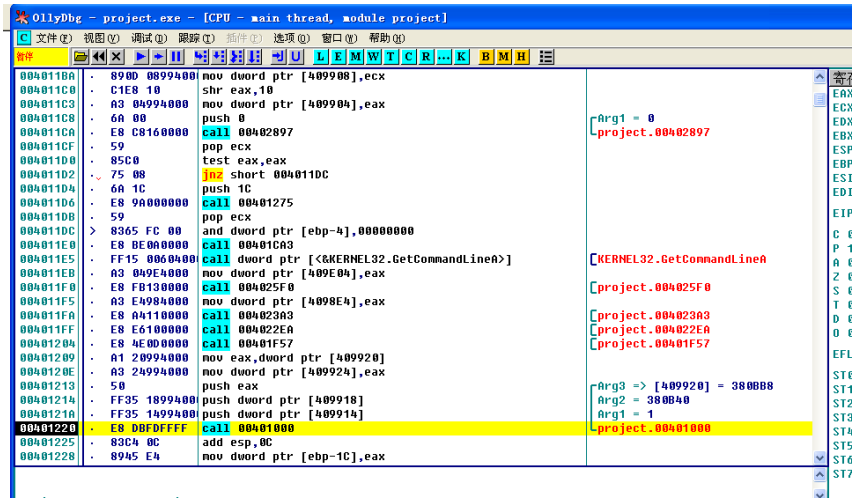


3. 接着在调试窗口输入格式化字符串“AAAA%x%x%x%x”，单步执行到 printf 函数的时候，调试窗口输出了输入的字符串以及一串地址信息。当“AAAA”输出完成后，函数继续在栈中找下一个参数，即接下来的连续四个地址，发生了内存的泄露。

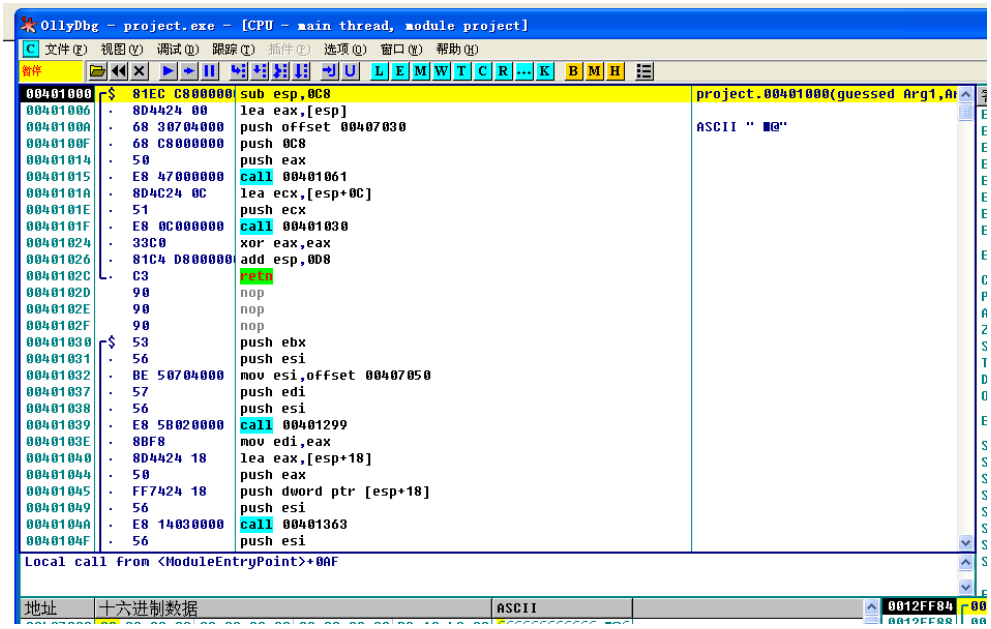


Release 模式：

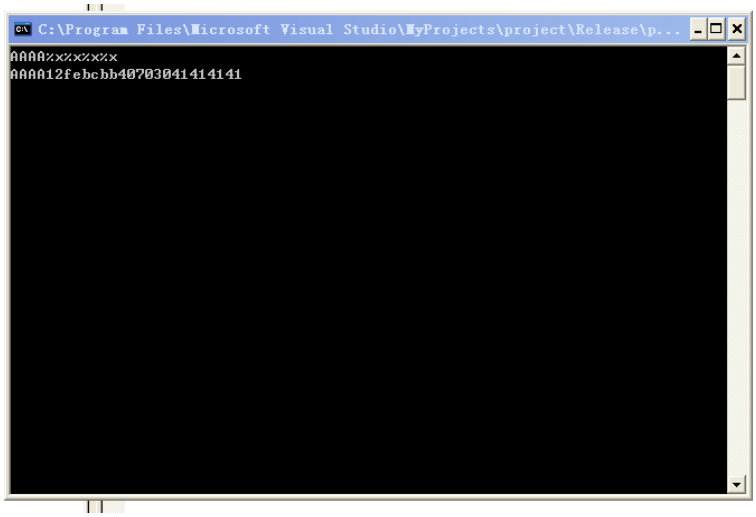
1. 生成 Release 模式下的可执行文件，在 OllyDbg 中打开，同样找到主函数的入口。

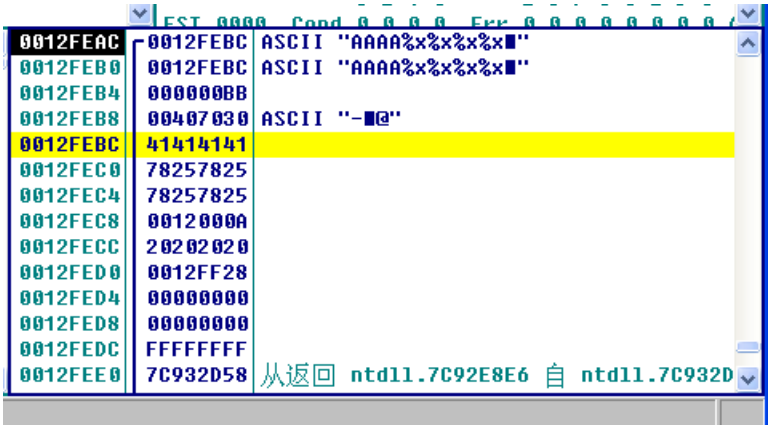


2. 进入主函数内部可以发现对栈空间进行初始化时只抬高了 0C8 的空间即局部变量的大小，也减少了寄存器的 push 指令，提高了程序的执行效率。



3. 执行到 fgets 函数入口时，我们在调试窗口输入“AAAA%x%x%x%x”，继续往下执行到 printf 函数入口时，我们可以发现最后一位输出了 41414141，这是由于在 Release 模式下没有申请额外的栈空间，不会输出“CCCCCCCC”，而是输出了输入字符串的起始地址所保存的值。





两种模式的差异：

再用两种模式对程序进行调试时，Debug 模式会首先会对栈初始化一个较大的空间，以及对寄存器的 push 操作，而且包含了调试信息，可以通过断点等操作快速找到错误信息，这样生成的代码更易于理解和调试，但由于没有对代码进行优化，运行速度较慢。

Release 模式的代码段比较紧凑，首先在初始化栈空间时只初始化了局部变量所需要的空间，并没有额外申请空间，对代码进行了优化，优点是代码体积更小，执行效率更高，缺点是不利于对代码的调试和跟踪，也容易导致内存泄漏等一系列问题。

心得体会：

理解了格式化字符串导致内存泄漏的基本原理，掌握了在 OllyDbg 中逐步调试的方法，同时通过对 Debug 模式和 Release 模式的比较，直观感受到了两种模式的差异。