**电子科技大学**

**计算机科学与工程学院**

**标 准 实 验 报 告**

**（实验）课程名称 信息对抗综合设计实验II**

**电子科技大学教务处制表**

**电 子 科 技 大 学**

**实 验 报 告**

# 学生姓名：黄鑫 学号：2021050901013 指导教师：李忻洋

**一、实验项目名称：机器码探索与病毒初见**

**二、实验目的：**

通过调试器，自己设计实验分析机器码的 格式，并设计实验修改机器码完成实证。本实验包含对x86指令集 中比较简单的mov指令的分析，并在此基础上分析在函数调用和二进制安全领域有较大作用的call指令

1. **实验原理：**

实验通过反汇编和机器码的深入探索，揭示计算机底层代码的执行机制。实验将二进制安全与恶意代码攻防实践结合，旨在展示高级语言与机器码的映射关系，探讨如何通过调试技术监视和修改内存操作。通过修改机器码和内存，验证程序执行流程，模拟病毒行为，并研究程序指令和数据的操控。实验揭示了内存中的代码变化对执行流程的影响，涵盖病毒模型中的执行、破坏、感染和潜伏机制，展示了机器码如何实现执行权的转移与恢复，深化对计算机系统中汇编指令和病毒机制的理解，具体原理如下所示

1. 学习基本反汇编技术，理解高级语言语句如何转换为机器码。实践中，通过反汇编窗口查看机器码，识别并修改赋值语句的机器码。观察机器码如何影响内存中的数据，并验证地址和指令的一致性。调试技术：

运用调试器的监视和内存查看功能，观察高级语言变量及其在内存中的表现。

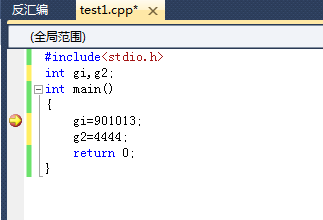
1. 使用断点和单步调试，追踪指令执行流程，验证机器码对内存的直接影响。通过内存窗口修改指令字节，实时监控内存变化，深入理解mov指令的执行。
2. 模拟计算机病毒的行为，包括窃取和归还执行权。通过修改内存中的机器码实现病毒的跳转和指令覆盖。使用JMP指令控制指令流转，跳转到恶意代码并返回正常执行路径。
3. 学习如何在内存和硬盘中修改机器码，验证修改对程序行为的影响。使用工具如C32ASM在文件中编辑机器码，理解内存和硬盘操作的一致性。
4. **实验环境（设备、元器件）：**

操作系统：Windows XP及之后的版本，如Windows 7、Windows 10等应用软件：VC++ 6.0及Visual Studio更高版本

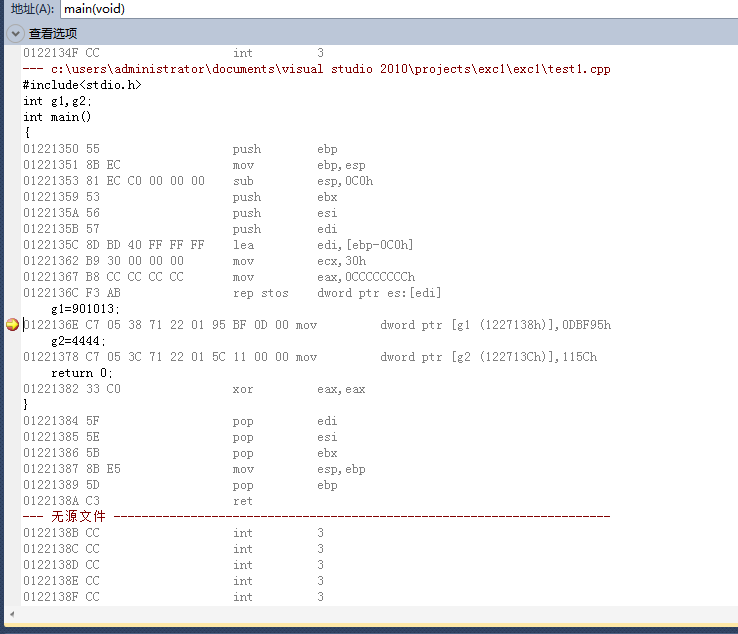
1. **实验步骤：**

**任务1：修改机器码的执行**

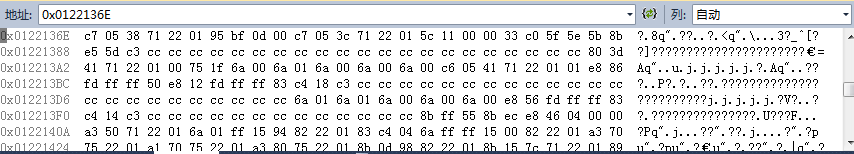
1. 定义两个全局变量g1,g2,写一个c语言的赋值语句g1=学号后6位(**901013**)。



1. 在“g1=学号后6位” 下断点，运行后查看对应mov指令机器码的指令地址。



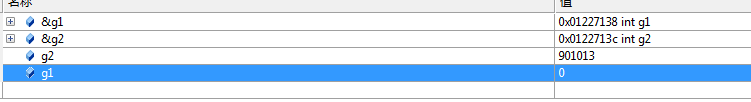
1. 用内存窗体找到（4）中的mov指令机器码存放的指令地址



1. 将c705后的四个字节，如后的下划线c7 05\_\_\_\_，修改成g2的内存地址

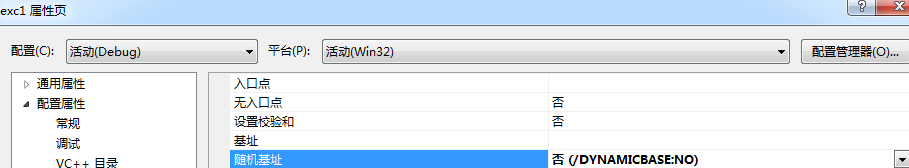
5

1. 单步执行mov，用watch窗口查看g2的内容将变成901013

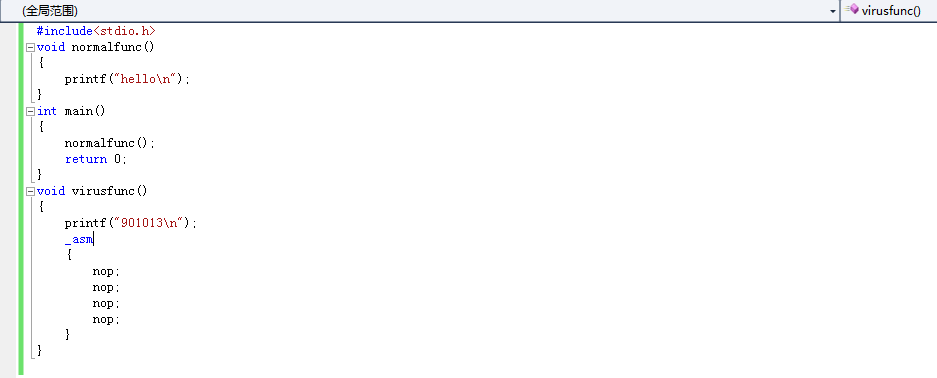


**任务2：在内存和硬盘上重现以上病毒模型的执行过程**

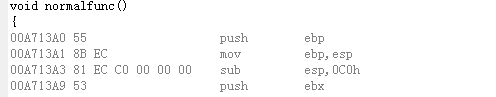
1. 避免重定位表带来的影响，链接器--高级--随机基址，选择禁止随机化属性

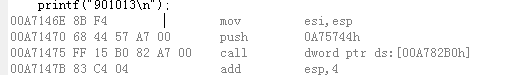


1. 编写正常代码以及用于病毒代码

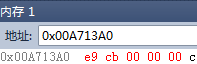


1. 计算jmp的偏移，从0xA713A0开始，占5字节，其后的指令起始为0xA713A5，从0xA713A5跳到0xA71470，这样偏移是0xA71470 − 0xA713A5 = cb。其小端机内存表示为cb 00 00 00





1. 用eb cb 00 00 00覆盖Normalfunc的入口， 即0xA713A0开始的5字节。用内存窗体完成。先在入口打断点，运行中断后，找到反汇编窗体，并在内存窗体中地址栏输入0xA713A0找到该指令内存，修改之



1. 此时，反汇编窗体中normalfunc的入口指令变成了eb cb 00 00 00。在观察显示的汇编指令，跳向了virusfunc入口+20h即0xA71470

2.6



**六、实验结论：**

**七、总结及心得体会：**

**八、对本实验过程及方法、手段的改进建议：**

**报告评分：**

**指导教师签字：**