《软件安全》实验报告

姓名：邢清画 学号： 2211999 班级：1023（物联网）

**实验名称：**

程序插桩及 Hook 实验

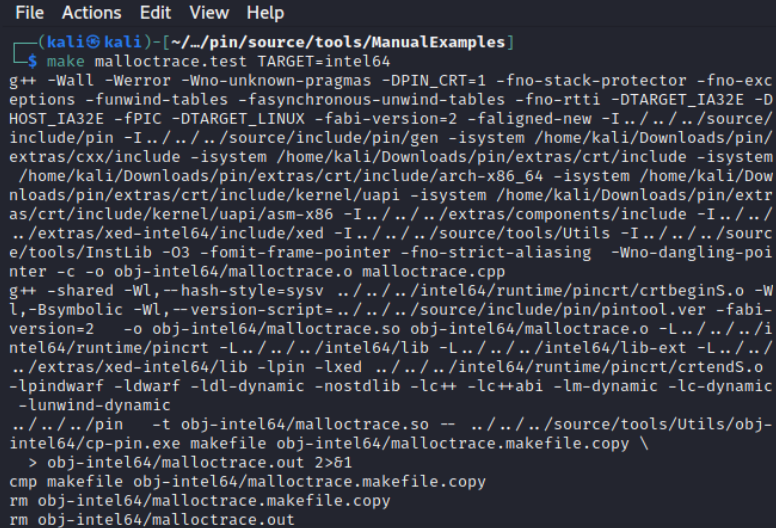
**实验要求：**

复现实验一，基于 Windows MyPinTool 或在 Kali 中复现 malloctrace 这个 PinTool， 理解 Pin 插桩工具的核心步骤和相关 API，关注 malloc 和 free 函数的输入输出信息。

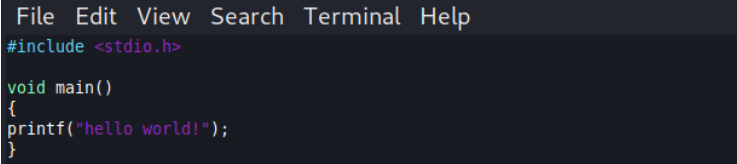
**实验过程：**

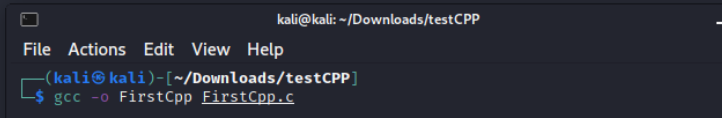
**一、malloctrace的复现**

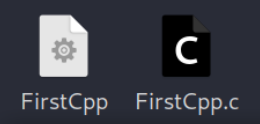
首先从官网https://software.intel.com/content/www/us/en/develop/articles/pin-abinary-instrumentation-tool-downloads.html下载pin（Linux版本），进行解压后添加到Linux虚拟机，在pin/source/tools/ManualExamples 文件夹中运行终端，编译运行**malloctrace.cpp**后生成动态链接库**malloctrace.so。**



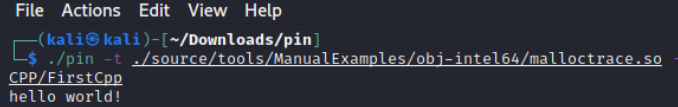
编写hello world！的c语言文件，并用gcc指令编译成可执行文件。





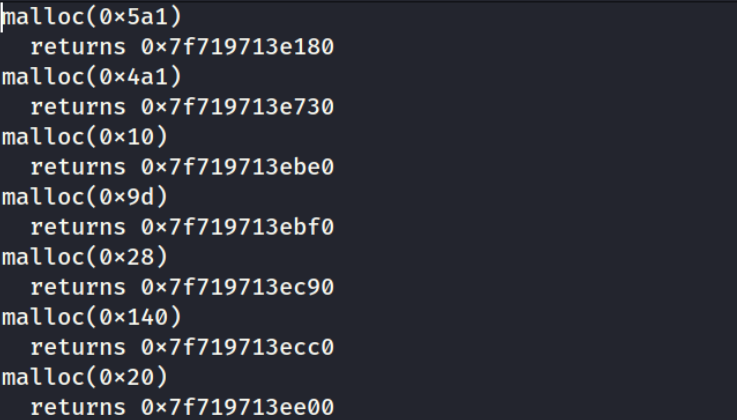


用刚刚编译生成的动态链接库**malloctrace.so**执行生成的可执行文件FirstCpp

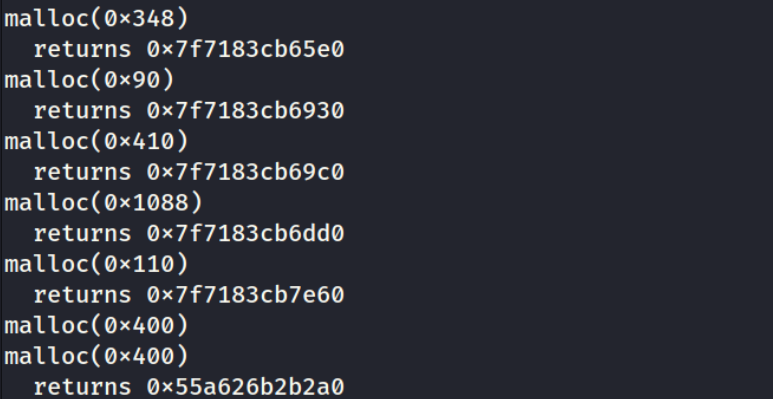


观察输出结果：

malloc(0x)表示调用了**malloctrace**的tool并分配了0xa字节的内存空间，其中a代表常数；

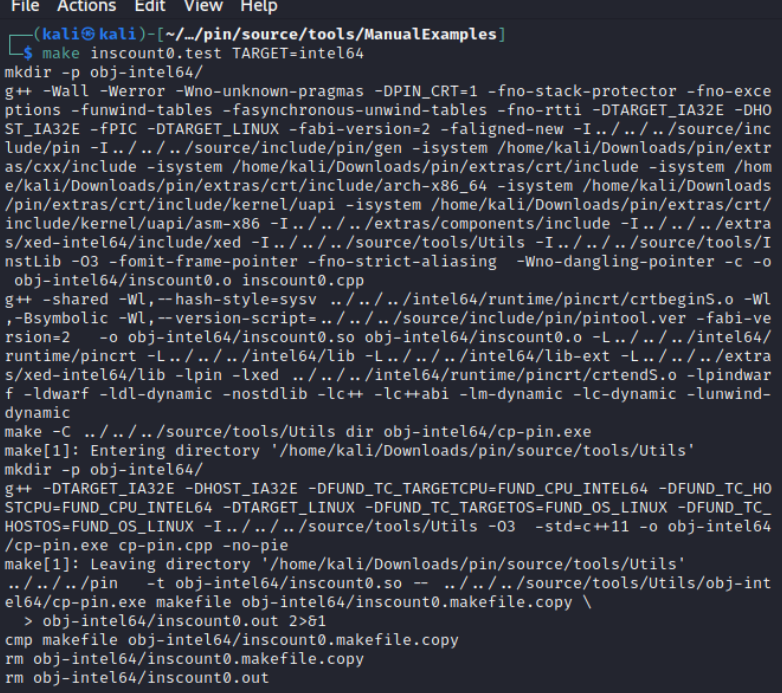


returns 0xbbbbbbbbbbbb表示**malloctrace**分配了内存并返回了指向该内存的指针。



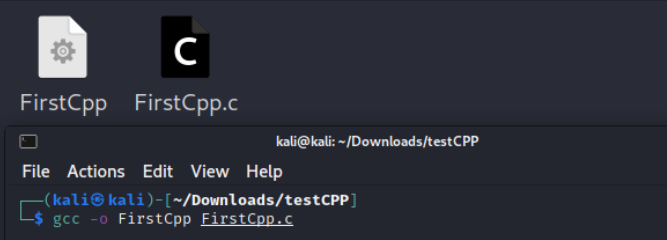
2. **Inscount0**的复现

首先从官网https://software.intel.com/content/www/us/en/develop/articles/pin-abinary-instrumentation-tool-downloads.html下载pin（Linux版本），进行解压后添加到Linux虚拟机，在pin/source/tools/ManualExamples 文件夹中运行终端，编译运行inscount0后生成动态链接库inscount0.so**。**

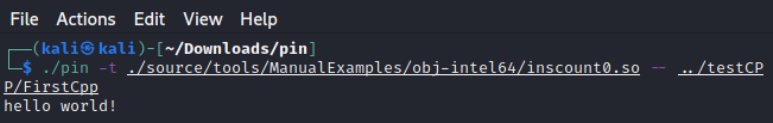


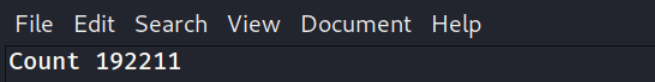
接下来与**malloctrace的复现**实验过程相同，编写hello world！的c语言文件，并用gcc指令编译成可执行文件。





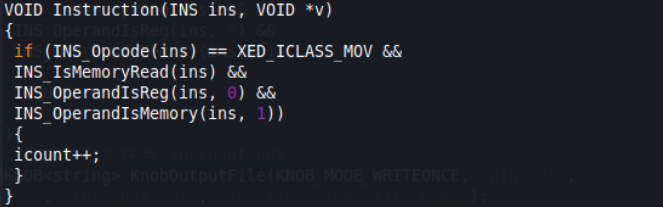
用刚刚编译生成的动态链接库inscount0.so执行生成的可执行文件FirstCpp，得到插桩后的计数结果count 192211



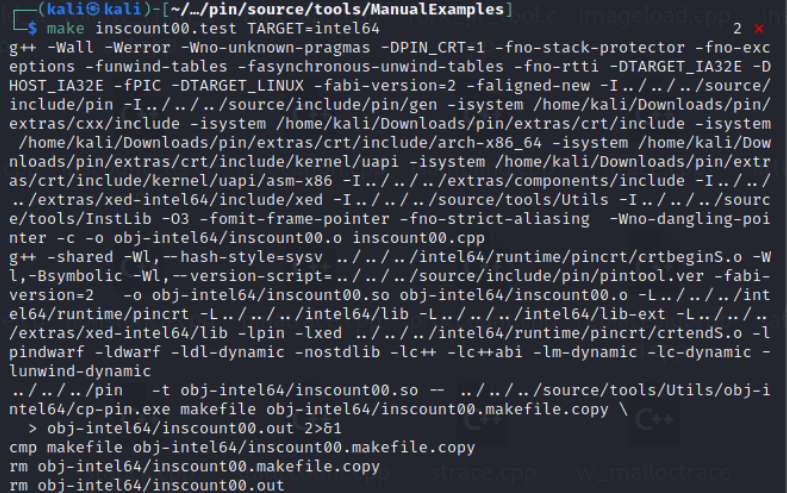


修改 inscount0.cpp中的代码，生成新的 tool inscount00.cpp，并编译成动态链接库 inscount00.so。

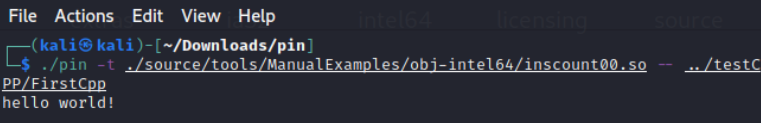
修改后的 inscount00.cpp ：

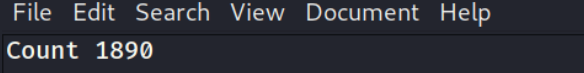


（pin工具每次遇到一个新指令都会调用该函数）



执行可执行文件，并得到修改后的插桩计数1890





（此处根据实际操作过程，留下具体操作步骤、附加一些自己的理解，即可）

**心得体会：**

通过本次程序插桩及 Hook 实验学习了如何进行使用pin工具的动态二进制插桩操作，同时在复现实验和实现自己定义的插桩工具过程中，进一步熟悉Linux环境下的指令操作，学习了不同的插桩工具之间的联系和各自的作用。