《软件安全》实验报告

姓名：冯思程 学号：2112213 班级： 计算机科学与技术二班

**实验名称：**

程序插桩及Hook实验

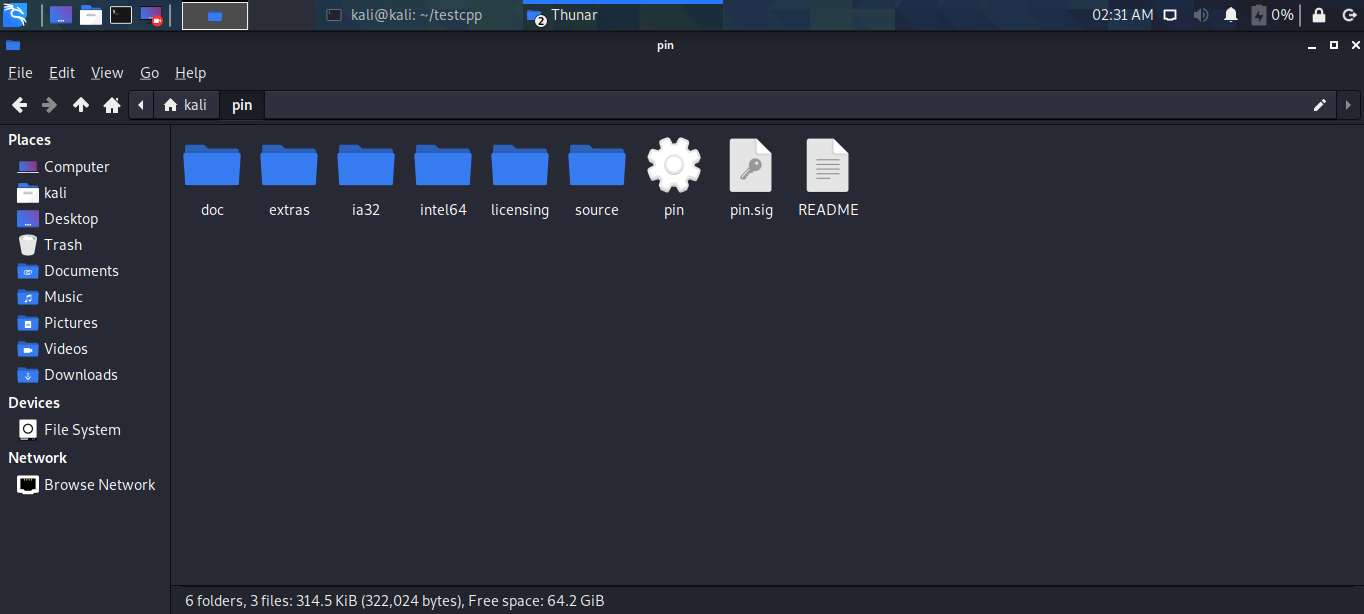
**实验要求：**

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：**

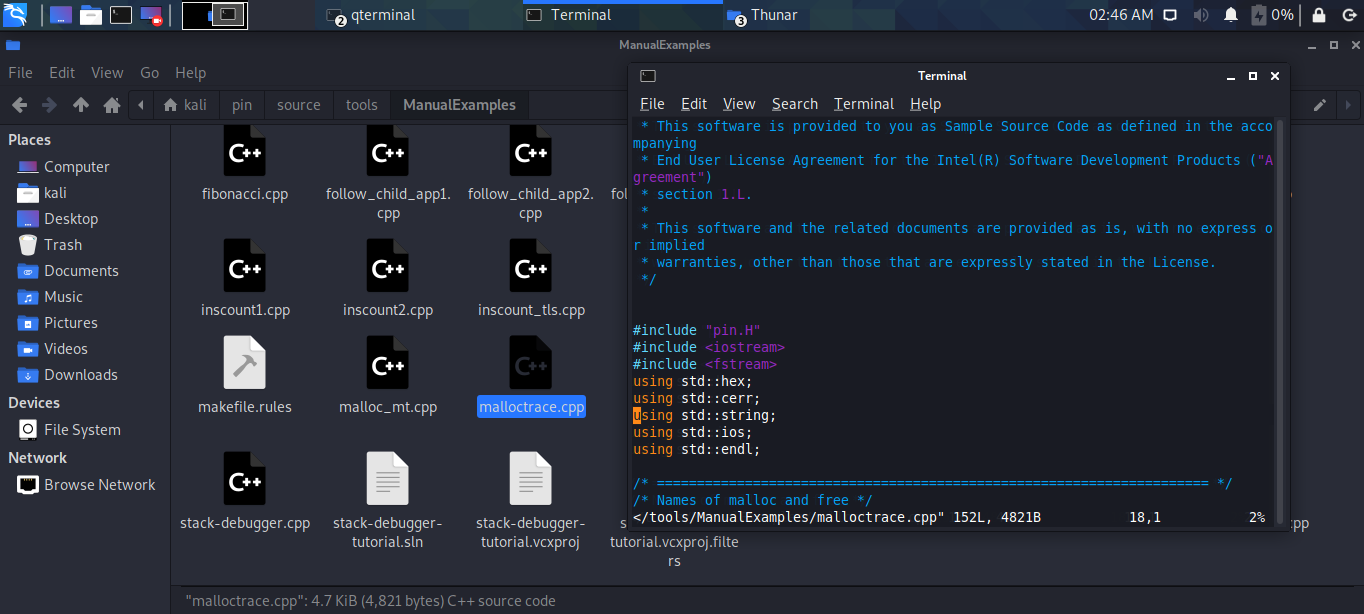
**第一部分：在kali中复现malloctrace**

1. **进行pin的安装：**打开VMware进入kali，然后打开火狐浏览器，输入pin下载的网址，然后进行pin的安装，这里为了为了配合实验，选择下载的版本是pin3.18。下载后解压打开文件夹，并将其位置放在kali主界面下，如下：

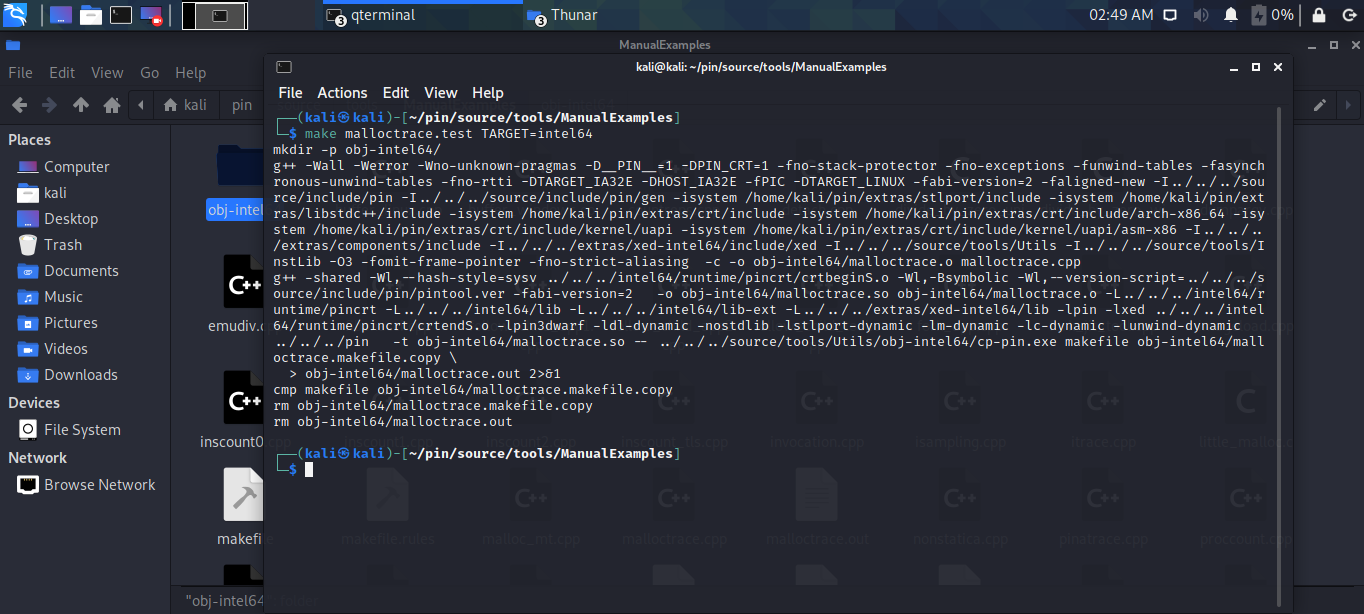


1. **在文件夹中找到malloctrace.cpp：**

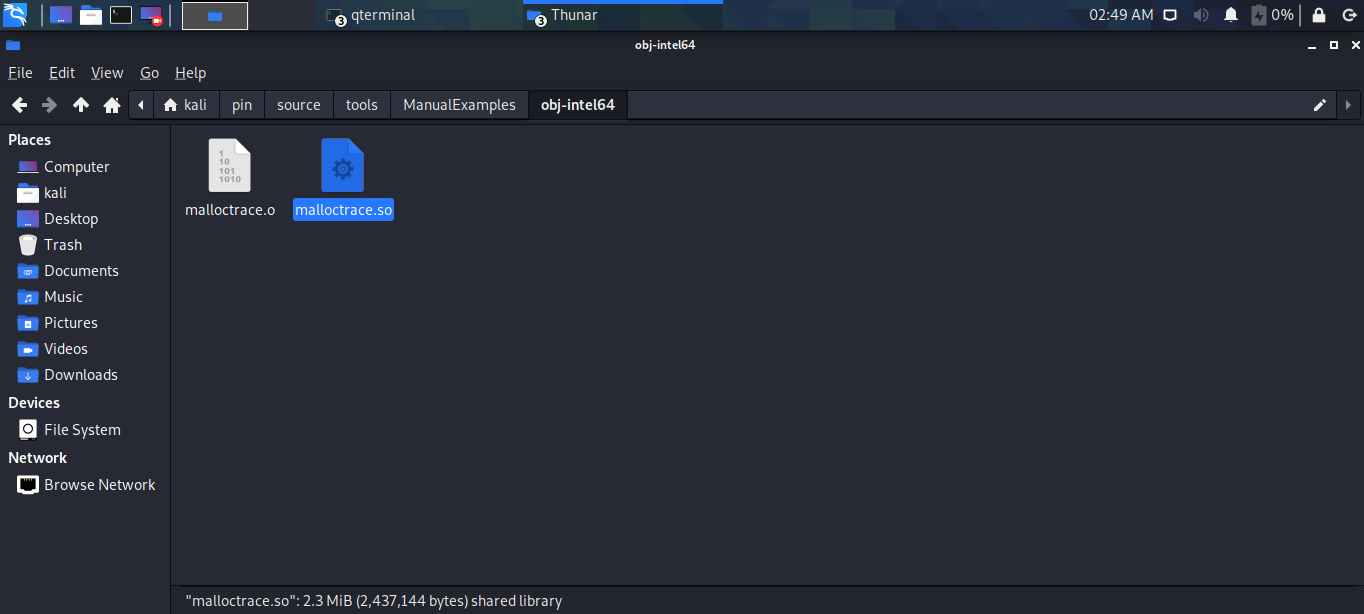
malloctrace.cpp在source/tools/ManualExamples路径下;



1. **对malloctrace.cpp进行编译，产生其动态链接库文件**，所用命令是：make malloctrace.test TARGET=intel64，截图如下：

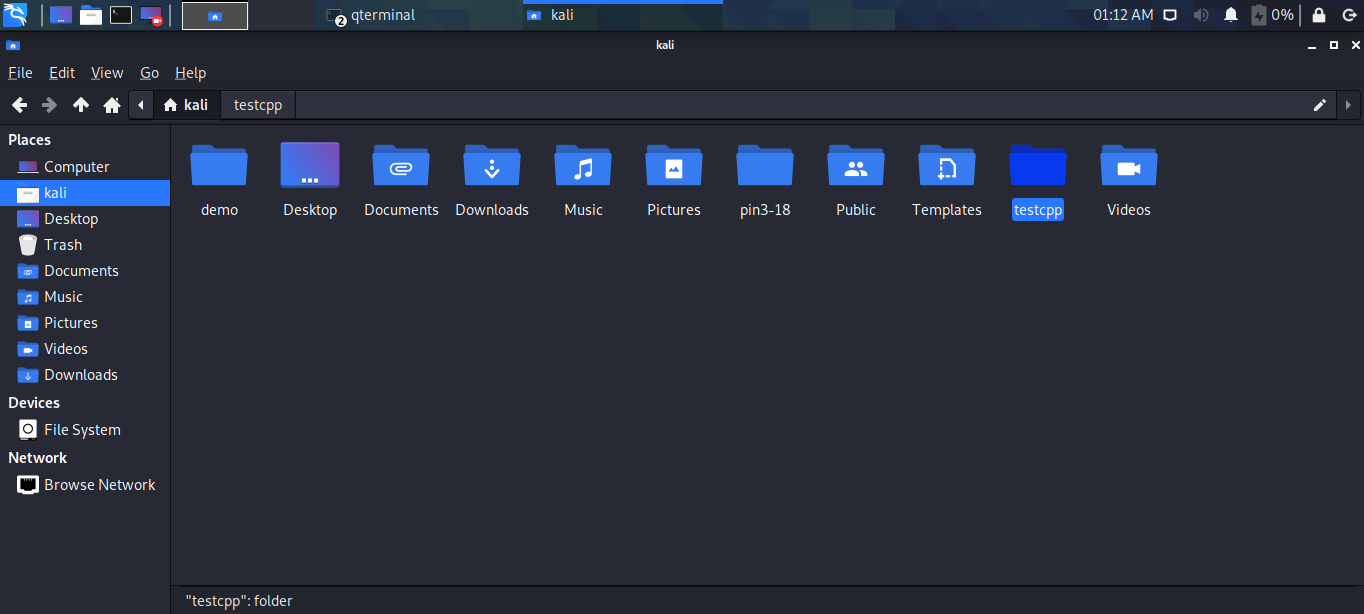


进入 obj-intel64 文件夹确认一下，已经生成了 malloctrace.so 这个动态链接库文件，截图如下：

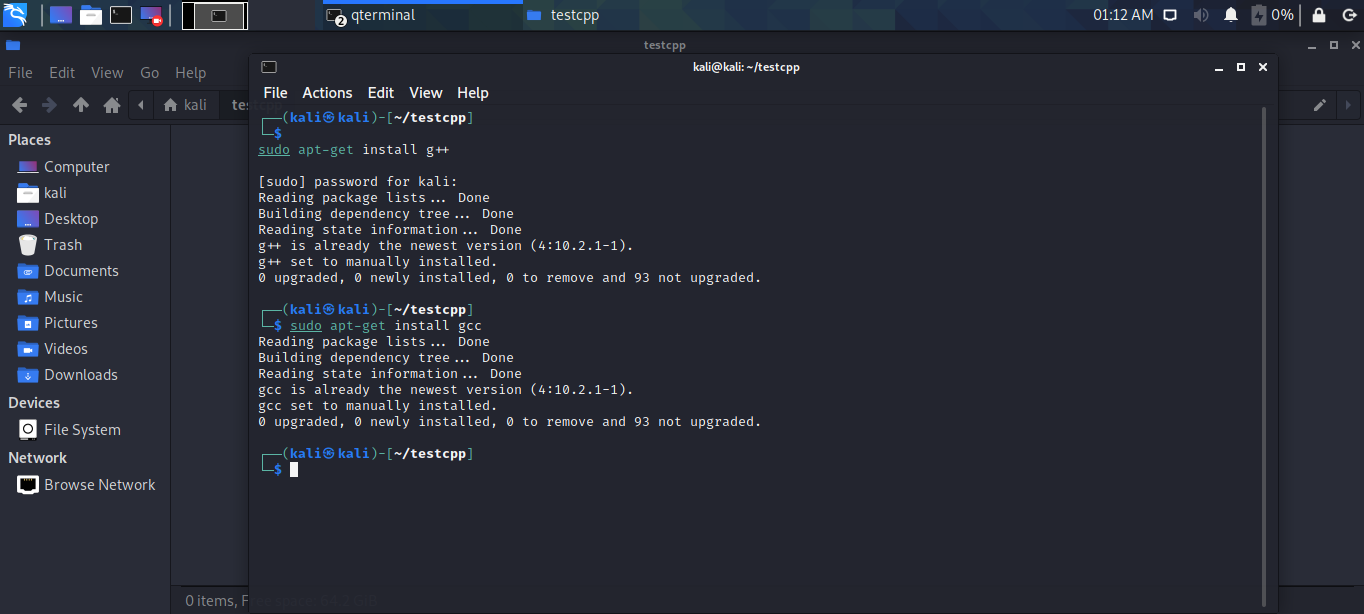


1. **新建一个testcpp文件夹，并在其中写一个简单的测试程序：**

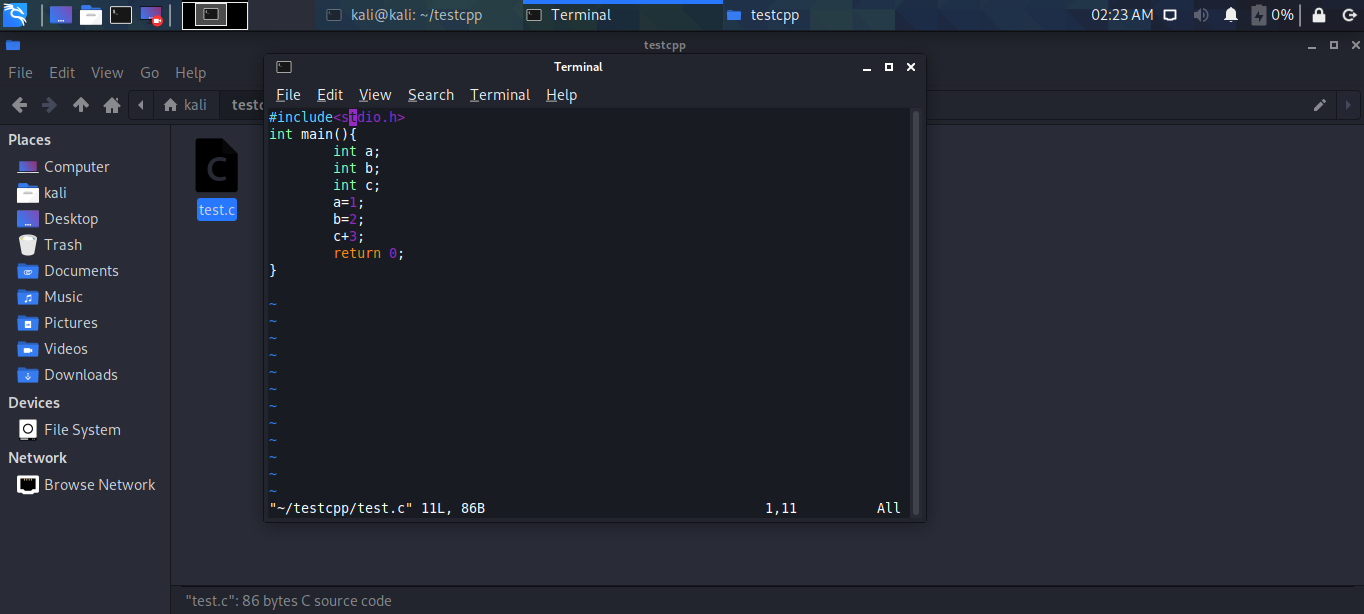
首先先新建一个testcpp文件夹，如下图：



然后确定一下编译器已经安装成功，如下图：

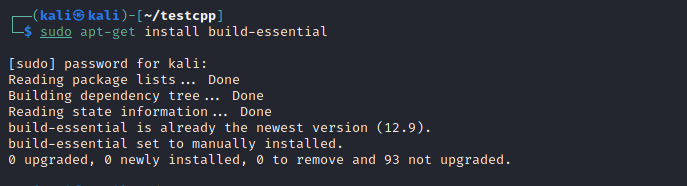


然后在testcpp文件夹中新建一个测试cpp文件，命令是touch test.c，然后用命令vim test.c输入程序代码，然后esc，然后：wq保存后退出，可以成功在当前文件夹下新建一个test.c，最后结果截图如下：

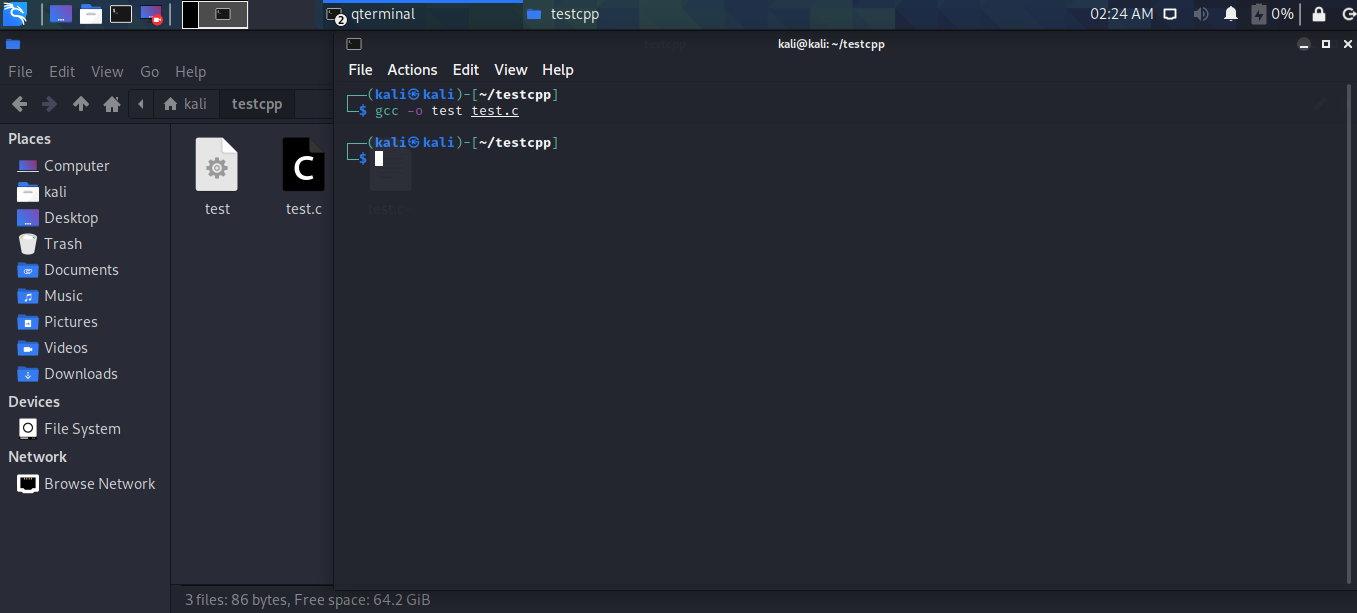


1. **编译生成可执行文件，并用上面的生成的malloctrace.so来检测**

首先利用gcc -o test test.c进行编译，发现无法编译，查阅资料之后发现可以通过安装build-essential来解决，通过命令sudo apt-get install build-essential来进行安装，截图如下：

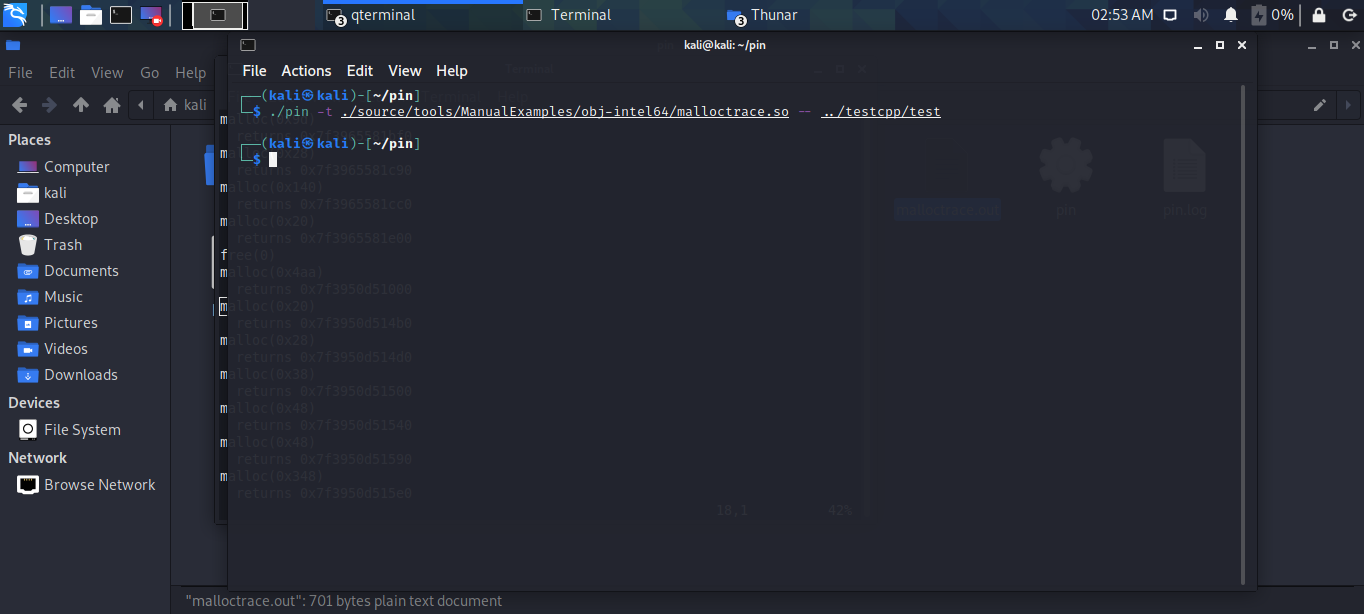


然后再进行编译，可以正常编译，截图如下：

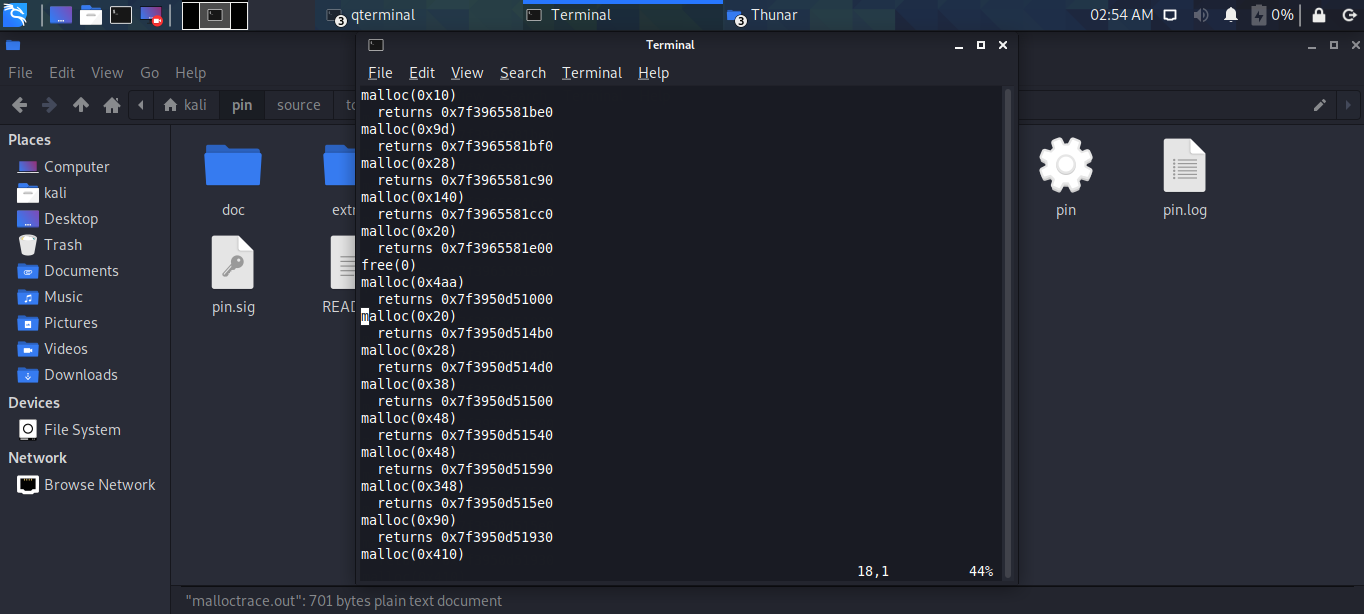


然后利用对 test可执行程序进行程序插桩的 Pin 命令为：

./pin -t ./source/tools/ManualExamples/obj-intel64/malloctrace.so -- ../testcpp/test



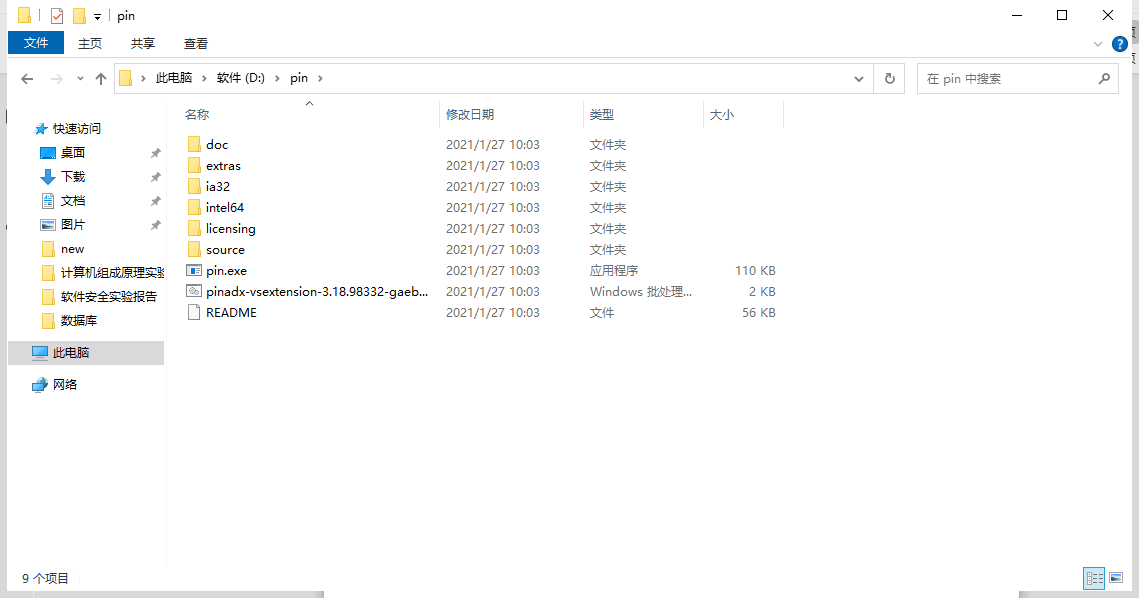
在 pin 路径下增加了一个输出文件 malloctrace.out，成功的实现了插桩，打开如下图：



Malloctrace的作用是打印出程序中调用malloc时传递的参数和返回值，以及调用free时传递的参数。

**第二部分：在windows10环境下，进行malloctrace的复现**，这里和在kali中复现有所区别，在 Windows 下，Pintool 需要基于 MyPinTool 框架来生成相应的 DLL 文件，之后基于生成的 DLL 文件来进行插桩。

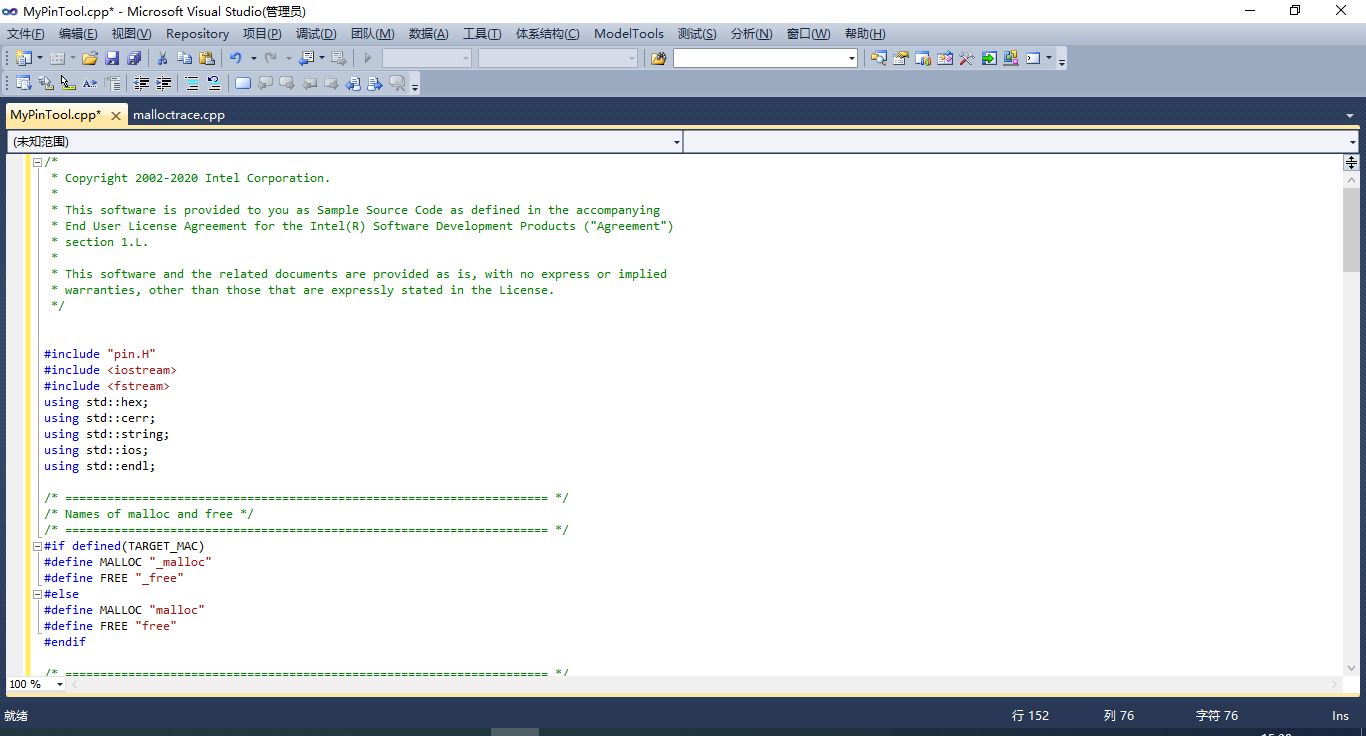
1. **进行pin的安装：**进入官网搜索pin，下载pin3.18。下载后解压打开文件夹，并将其位置放在D盘下，如下：



1. **在文件夹中找到malloctrace.cpp，将其复制到MyPinTool.cpp：**

malloctrace.cpp在source/tools/ManualExamples路径下;

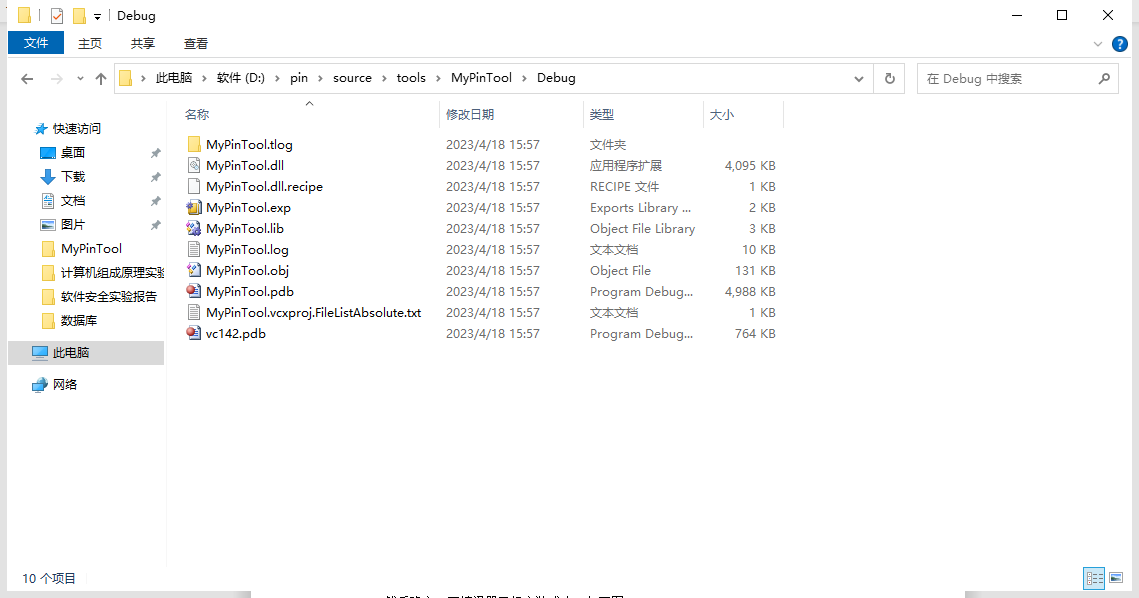
MyPinTool.cpp在source/tools/MyPinTool路径下。



1. **产生其动态链接库文件:**

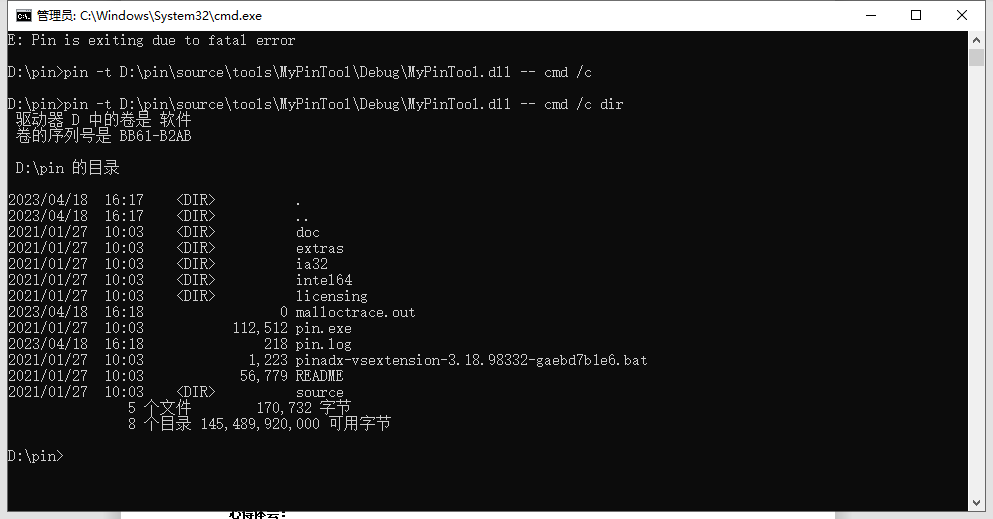
在 D:\pin\source\tools\MyPinTool 路径下找到 Visual studio 项目文件

MyPinTool.vcxproj，然后打开就可以看到 MyPinTool.cpp 包含了一个轨迹级插桩的程序。 点击运行，整个项目生成后，在 debug 路径下生成了一个 MyPinTool.dll 文件。

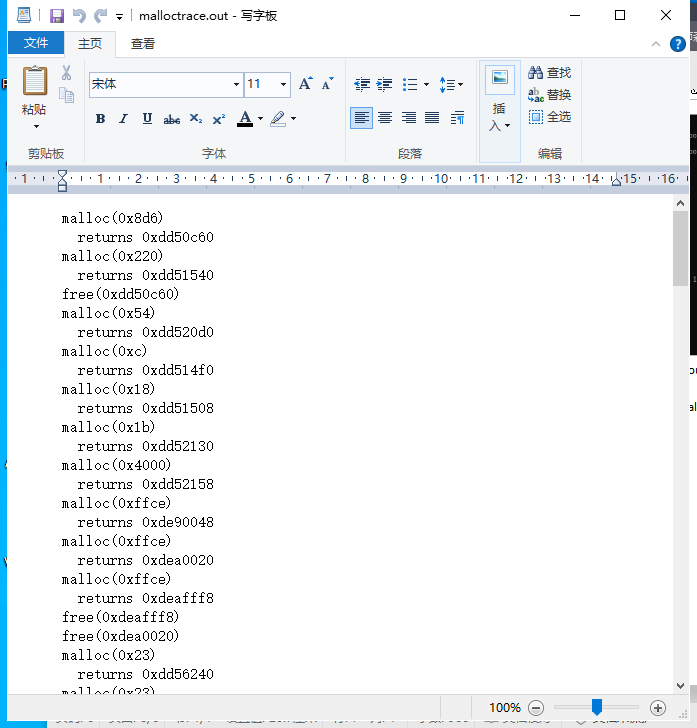


1. **利用pin命令进行程序插桩：**

命令：pin -t D:\pin\source\tools\MyPinTool\Debug\MyPinTool.dll -- cmd /c dir，通过-t 给定的 Pintool 是我们生成的动态链接库文件，--给定的目标程序cmd.exe。



发现在pin路径下生成了一个malloctrace.out文件，打开，如下图：



其中Malloctrace的作用是打印出程序中调用malloc时传递的参数和返回值，以及调用free时传递的参数。

**第三部分：对pin插桩的核心步骤和相关API以及malloctrace的输入输出信息的理解：**

1.Pin插桩工具的核心步骤：

加载目标程序：使用 Pin 工具加载目标程序，此时会创建一个新的进程并将目标程序加载到其中。

插桩：在加载目标程序时，利用 Pin 提供的插桩API，在目标程序的代码中插入指令，以实现对目标程序的监控和控制。

运行目标程序：运行已插桩的目标程序，此时插桩代码会被执行，并收集相应的信息。

分析结果：根据插桩代码收集到的信息，进行相应的结果分析和处理。

2.Pin插桩工具的相关API：

IMG\_AddInstrumentFunction()：注册函数，当 Pin 加载目标程序时，自动调用指定的函数进行插桩操作。

RTN\_CreateAt()：在指定的地址创建一个 RTN 对象（即函数对象），用于指定该函数的插桩操作。

INS\_InsertCall()：在指令前或后插入一个回调函数，用于对指令进行监控和控制。

PIN\_SafeCopy()：在指定的地址中读取或写入数据，并提供一定的安全保障，避免了指针越界等问题。

PIN\_GetTid()：获取当前线程的 ID。

PIN\_ExitApplication()：提前终止 Pin 工具的执行。

1. malloctrace 工具的输出信息：

Memory Allocation（内存分配）: 该信息表示内存分配的相关信息，包括分配的内存大小、分配时的函数调用栈信息等。例如：

Memory Allocation: 0x7f84b3802a00 [size=16] @ 0x4016c9 main + 9 malloc

该信息表示在地址 0x7f84b3802a00 处分配了 16 字节大小的内存，该内存的分配操作发生在 main 函数的第 9 行代码处，调用的函数是 malloc。

Memory Deallocation（内存释放）: 该信息表示内存释放的相关信息，包括释放的内存地址、释放时的函数调用栈信息等。例如：

Memory Deallocation: 0x7f84b3802a00 @ 0x4016fa main + 39 free

该信息表示释放了地址为 0x7f84b3802a00 的内存，该内存的释放操作发生在 main 函数的第 39 行代码处，调用的函数是 free。

Memory Re-allocation（内存重新分配）: 该信息表示内存重新分配的相关信息，包括重新分配前后的内存地址、分配的内存大小、重新分配时的函数调用栈信息等。例如：

Memory Re-allocation: 0x7f84b3802a00 -> 0x7f84b3802a20 [size=32] @ 0x401732 main + 82 realloc

该信息表示将地址为 0x7f84b3802a00 的内存重新分配为大小为 32 字节的内存，并返回新分配的内存地址 0x7f84b3802a20，该操作发生在 main 函数的第 82 行代码处，调用的函数是 realloc。

这些信息可以帮助我们更加深入地了解程序的内存使用情况，找出内存泄漏等问题，进一步优化和改善程序的性能和稳定性。

**心得体会：**

本次实验我学习了如何利用Pin插桩工具进行动态二进制插桩，以跟踪程序在运行时的内存分配和释放操作。我主要分为两个部分进行了实验：在Kali中利用malloctrace进行内存分析，在Windows10下利用MyPinTool框架进行malloctrace分析cmd.exe可执行文件。

在实验过程中，我学习了Pin工具的基本概念、安装方法、以及Pin Tool的编写方法，掌握了利用Pin进行程序调试和分析的基本技能。在利用malloctrace工具进行内存分析时，我深入理解了程序运行时内存分配和释放的行为，并了解了malloctrace输出信息的含义，包括内存分配和释放函数的调用参数和返回值、堆栈信息等。在分析cmd.exe可执行文件时，我进一步加深了对Pin插桩工具的理解，并学会了利用MyPinTool框架进行Pin Tool的编写和调试。

总的来说，本次实验对我的学习和实践能力都有很大的帮助。通过实验，我不仅学会了如何使用Pin插桩工具进行程序分析，而且深入了解了程序运行时内存分配和释放的行为，掌握了分析程序内存泄漏、重复释放等问题的方法和技巧。这将对我的软件开发和安全测试等工作有很大的帮助。同时，我也发现Pin工具具有很强的可扩展性，可以通过编写自己的Pin Tool来实现特定的功能，这为我今后的研究和开发工作提供了很多启示和灵感。