**《软件安全》实验报告**

姓名：胡博浩 学号：2212998 班级：信息安全

**一、实验名称：**

AFL模糊测试实验

**二、实验要求：**

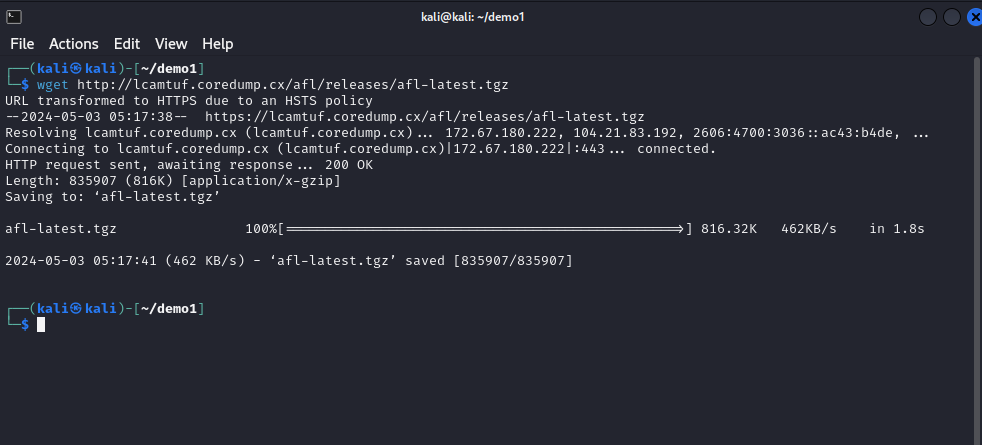
根据课本7.4.5章节,复现AFL在KALI下的安装、应用,查阅资料理解覆盖引导和文件变异的概念和含义。

**三、实验过程：**

（一）安装AFL

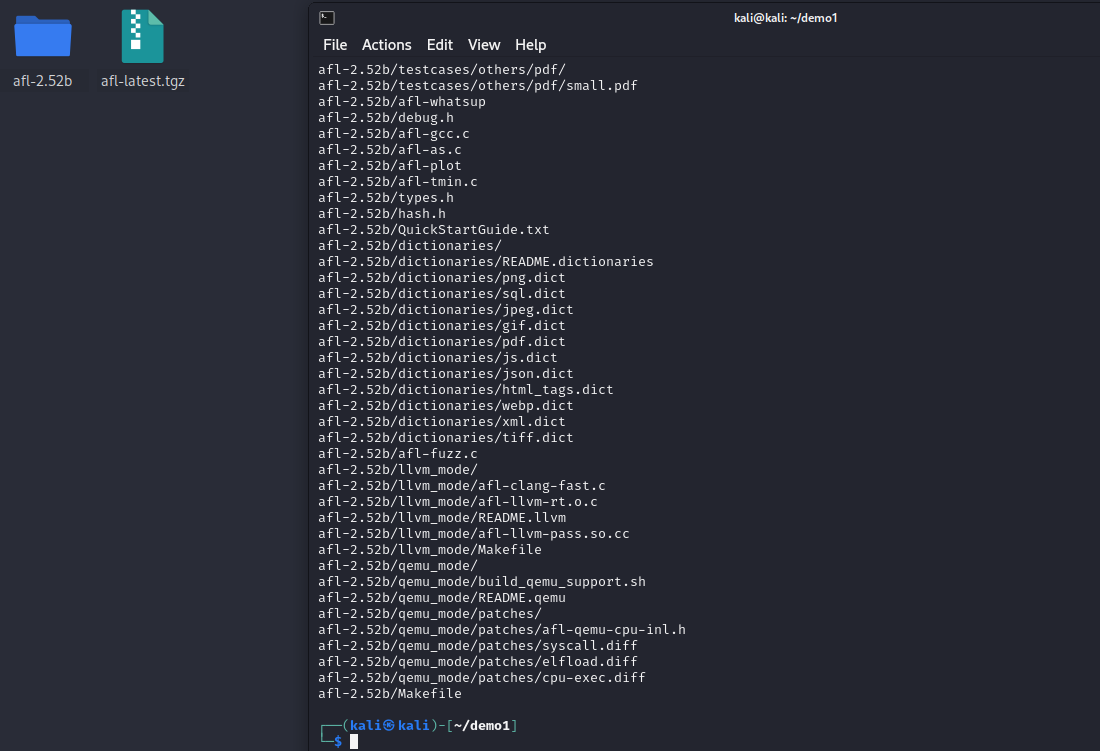
1.首先使用命令下载安装包：

wget http://lcamtuf.coredump.cx/afl/releases/afl-latest.tgz



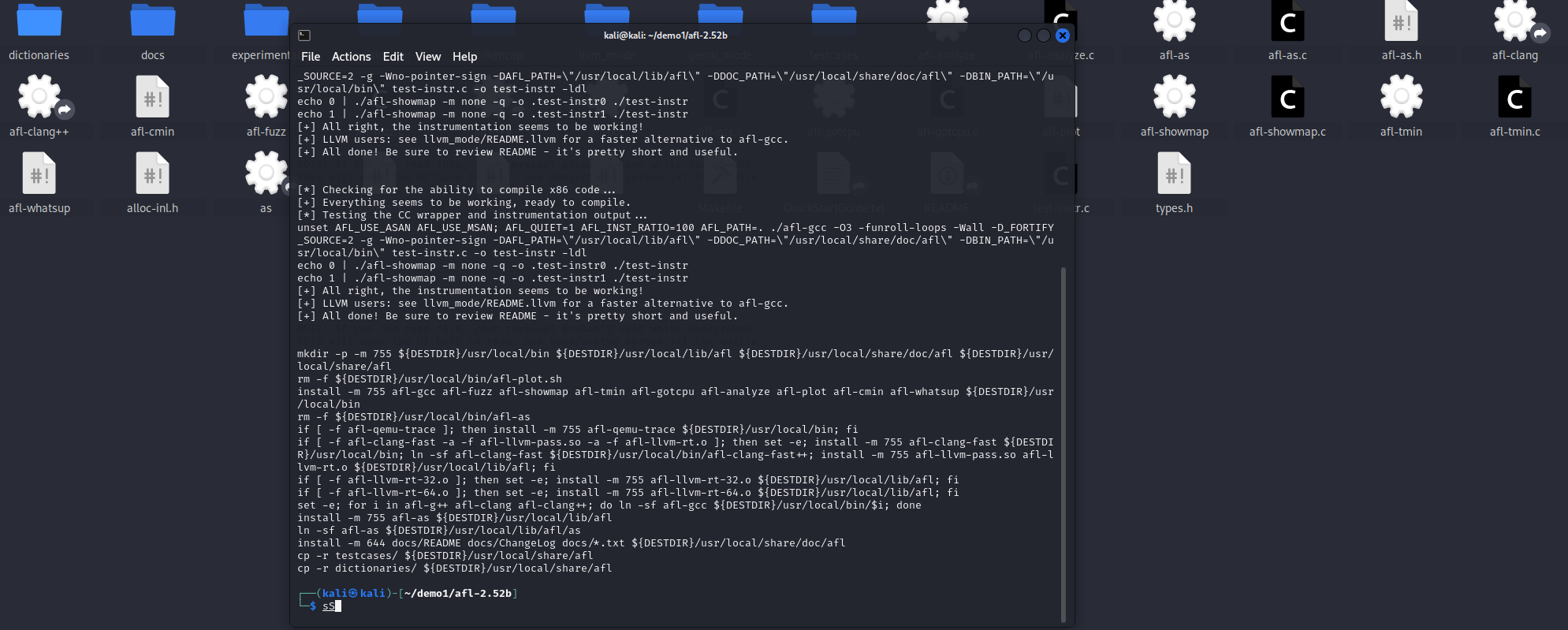
2.解压安装包：

tar xvf afl-latest.tgz



3.进入AFL目录，编译安装AFL：

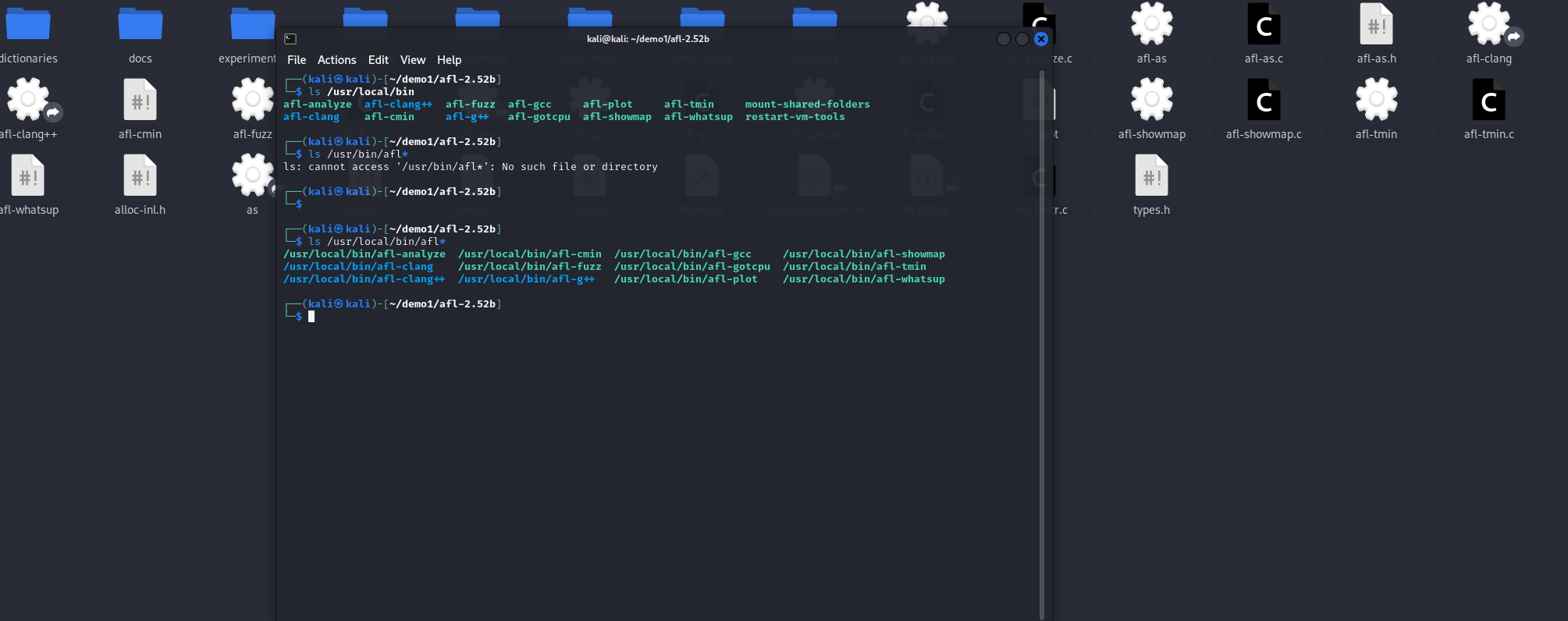
sudo make && sudo make install



4. 查看路径可以看到afl安装的文件：

ls /usr/local/bin

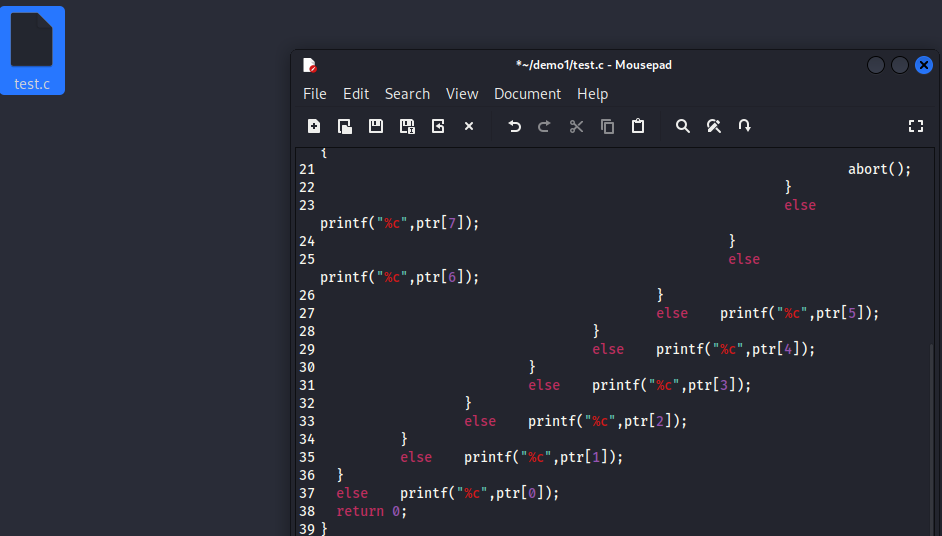
ls /usr/local/bin/afl\*



（二）AFL应用

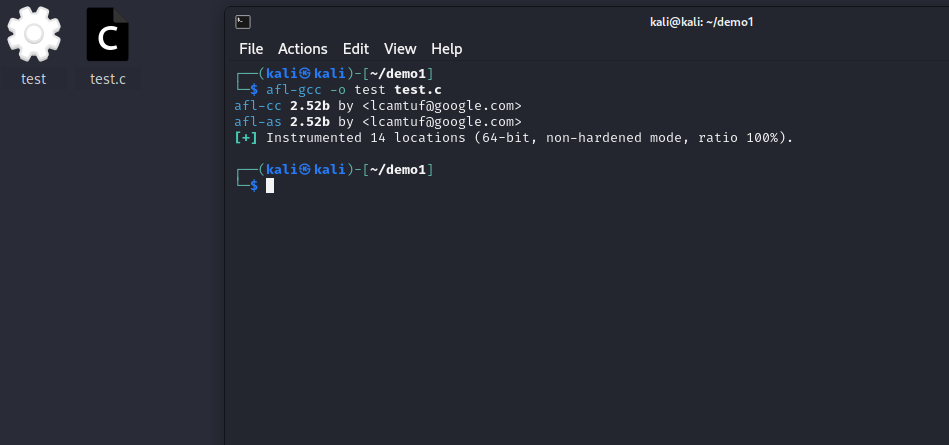
1. 创建本次实验的程序：

新建文件夹demo，并创建本次实验的程序test.c，该代码编译后得到的程序如果被传入 “deadbeef”则会终止，如果传入其他字符会原样输出：

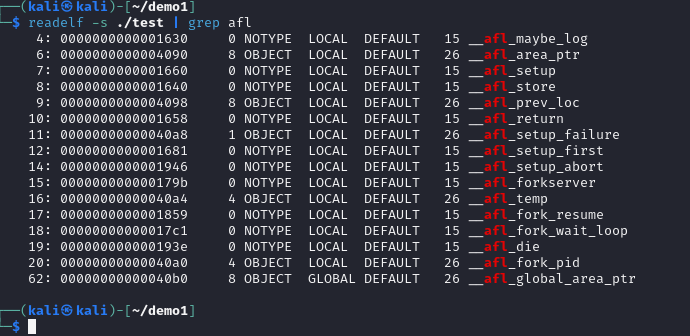


使用afl的编译器编译，可以使模糊测试过程更加高效：

afl-gcc-o test test.c

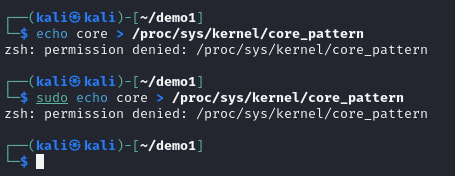


编译后会有插桩符号，使用如下命令验证：readelf-s ./test | grep afl



输入如下命令指示系统将coredumps输出为文件，而不是将它们发送到特定的崩溃处理程序应用程序。

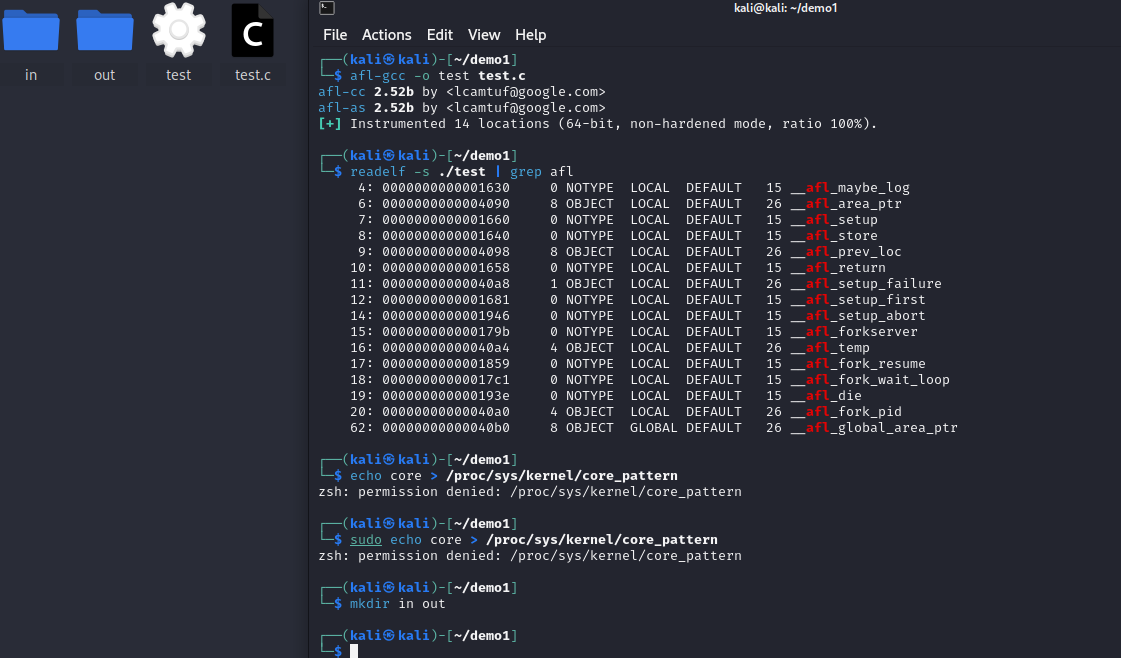
命令：echo core > /proc/sys/kernel/core\_pattern



2.创建测试用例：

首先，创建两个文件夹in和out，分别存储模糊测试所需的输入和输出相关的内容。

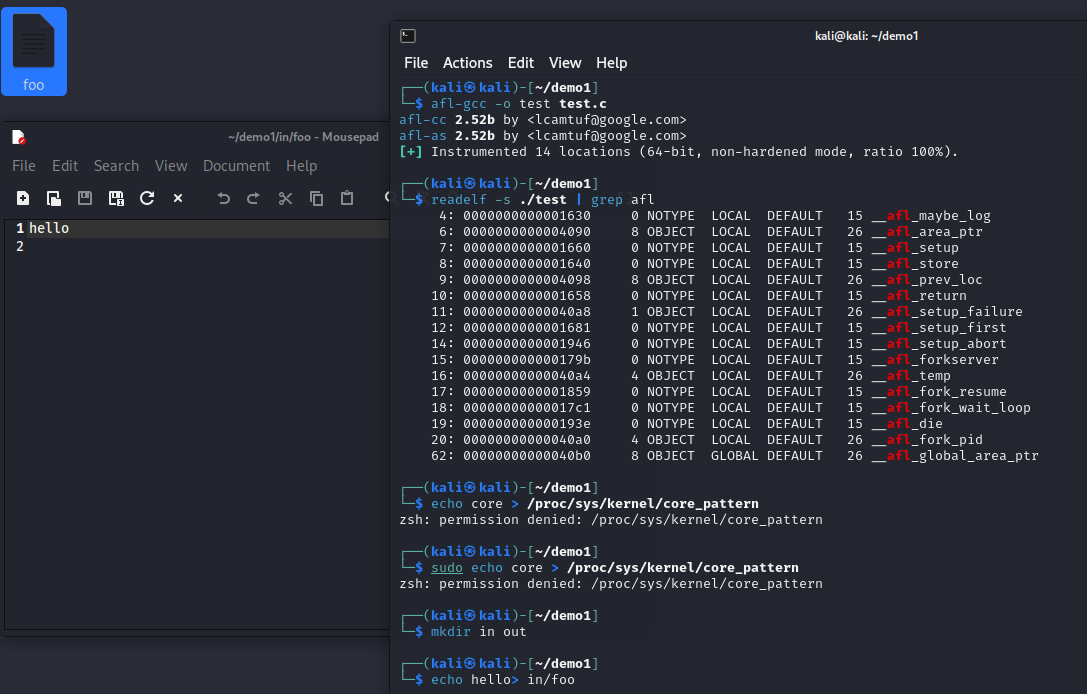
命令：mkdir in out



然后，在输入文件夹中创建一个包含字符串“hello”的文件。

命令：echo hello> in/foo

foo就是我们的测试用例，里面包含初步字符串hello。AFL会通过这个语料进行变异，构造更多的测试用例。

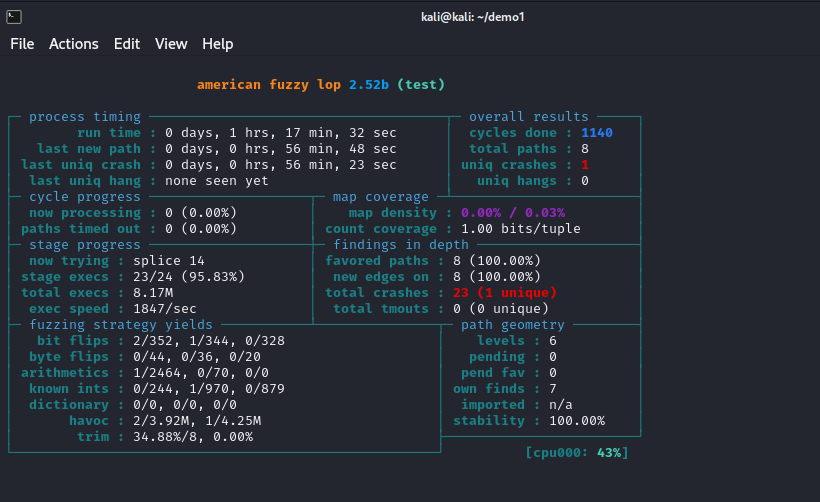


3.启动模糊测试：

运行如下命令，开始启动模糊测试：

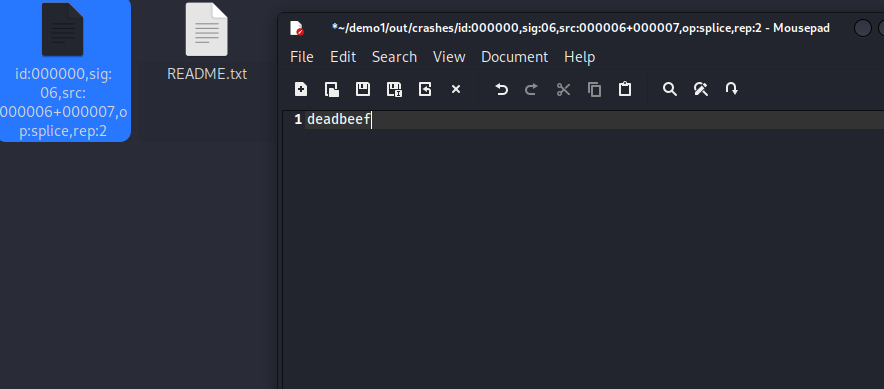
命令：afl-fuzz -i in -o out -- ./test @@

运行一段时间后，结果如下图：



4.分析crash：

在out文件夹下的crashes子文件夹里面是我们产生crash的样例，如下图所示：



与源代码对比可知，这正是使我们代码产生崩溃的输入。实验成功！

（三）覆盖引导

覆盖引导是一种测试策略，旨在利用代码覆盖率信息来指导测试过程，以提高测试的效率和效果。在软件测试中，覆盖率是指测试用例执行时覆盖到的源代码行数或代码块的百分比。覆盖引导通过监测程序执行过程中覆盖到的代码路径，并据此指导测试用例生成或变异的方向，从而增加发现潜在软件缺陷的可能性。

在AFL的fuzzing过程中，维护一个测试用例队列，每次从队列中取出文件后，对其进行变异。如果这个变异导致代码覆盖率提高了，说明测试用例更有效，那么就将这个文件放回队列中。通过这种方式，AFL能够在不断变异测试用例的过程中，引导变异的方向，使得测试用例更有可能触发程序中尚未执行的代码路径，从而提高发现漏洞的概率。

（四）文件变异

文件变异是指对测试用例文件进行各种修改操作，以生成新的测试用例，从而增加测试的多样性和覆盖范围。在AFL的fuzzing过程中，文件变异是通过对测试用例文件进行一系列变异操作来生成新的测试用例，以更好地探索程序的不同执行路径，并发现潜在的软件缺陷。

在AFL中，文件变异的主要类型包括：

1.按位翻转（bitflip）：将文件中的某些位按位翻转，即将1变为0，0变为1，以改变文件的内容。

2.算术运算（arithmetic）：对文件中的数值进行加减乘除等算术运算，以改变数值的大小或者值的类型。

3.特殊内容替换（interest）：将文件中的某些内容替换为特殊字符、关键字或者其他特殊内容，以增加测试的覆盖范围。

4.词典变异（dictionary）：将自动生成或者用户提供的词典中的token替换或者插入到原文件中，以改变文件的内容。

5.大破坏（havoc）：对文件进行大量随机变异，包括删除、插入、替换等操作，以生成更多样化的测试用例。

6.绞接（splice）：将两个或者多个文件拼接起来，生成一个新的文件，以尝试组合不同的测试用例内容。

通过文件变异，AFL能够生成大量多样化的测试用例，覆盖更多的代码路径，从而提高发现潜在软件缺陷的可能性。这种测试方法能够有效地帮助发现程序中的漏洞和错误，提高软件的安全性和稳定性。

**四、心得体会：**

通过本次实验，我学会了如何运用AFL工具进行模糊测试，这是一项非常实用和重要的技能。模糊测试是发现软件漏洞的一种有效方法，而AFL作为一款强大的模糊测试工具，能够帮助我自动化地处理大量的测试用例，并且通过记录和覆盖引导来增加发现漏洞的概率。

在实验中，我还了解了覆盖引导和文件变异的概念和含义。覆盖引导可以帮助指导模糊测试过程，使得测试用例更有针对性，从而提高测试的效率和准确性。文件变异则是通过修改和变换输入文件，生成更多样化的测试用例，从而增加测试的广度和深度。

通过这次实验，我对模糊测试的整个过程有了初步的了解，也对AFL工具的强大功能有了更深入的认识。这些知识的掌握将为我今后在软件安全领域的学习和工作提供很大的帮助。我会继续努力学习，提升自己在信息安全领域的技能，为保障软件的安全贡献自己的力量。