《漏洞利用及渗透测试基础》实验报告

**实验名称：**

AFL模糊测试实验

**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。

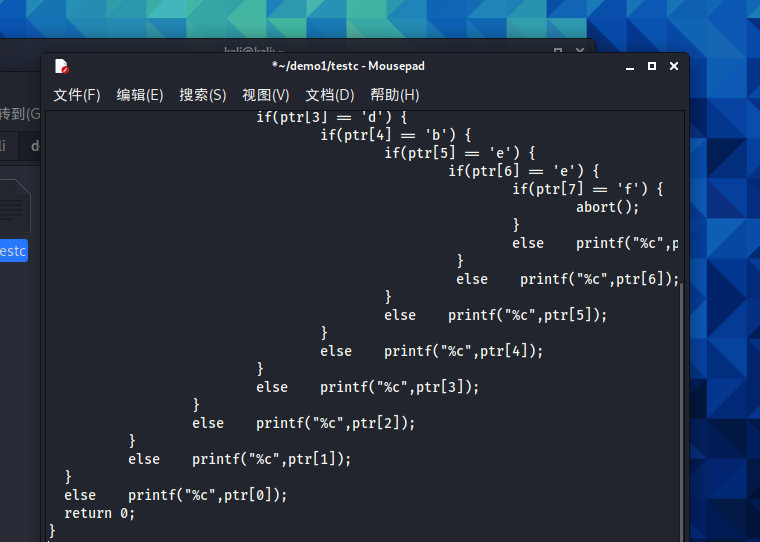
**实验过程：**

**1.安装AFL**

在终端中输入命令sudo apt-get install afl即可安装afl，可能需要首先执行sudo apt-get update指令更新软件源。

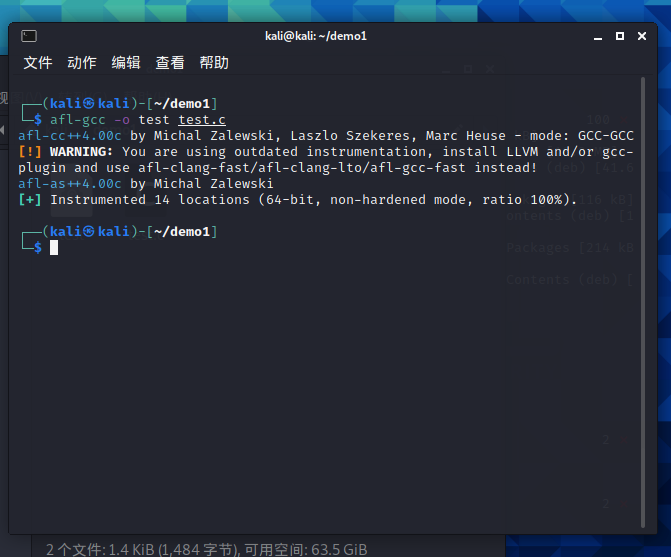
**2.程序编写**

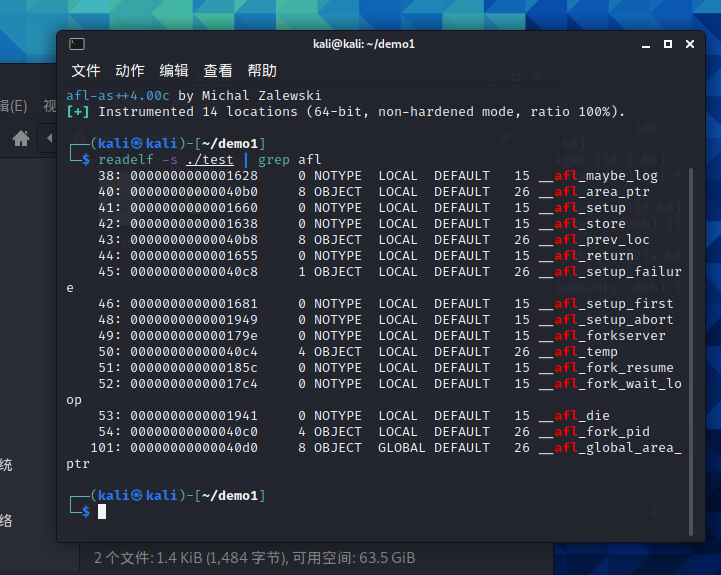
下图代码主要实现了当用户输入“deadbeef”字符串时，程序会调用abort函数结束进程，并返回错误代码。可以作为程序的一个漏洞进行ALF模糊测试实验。



**3.编译源程序**

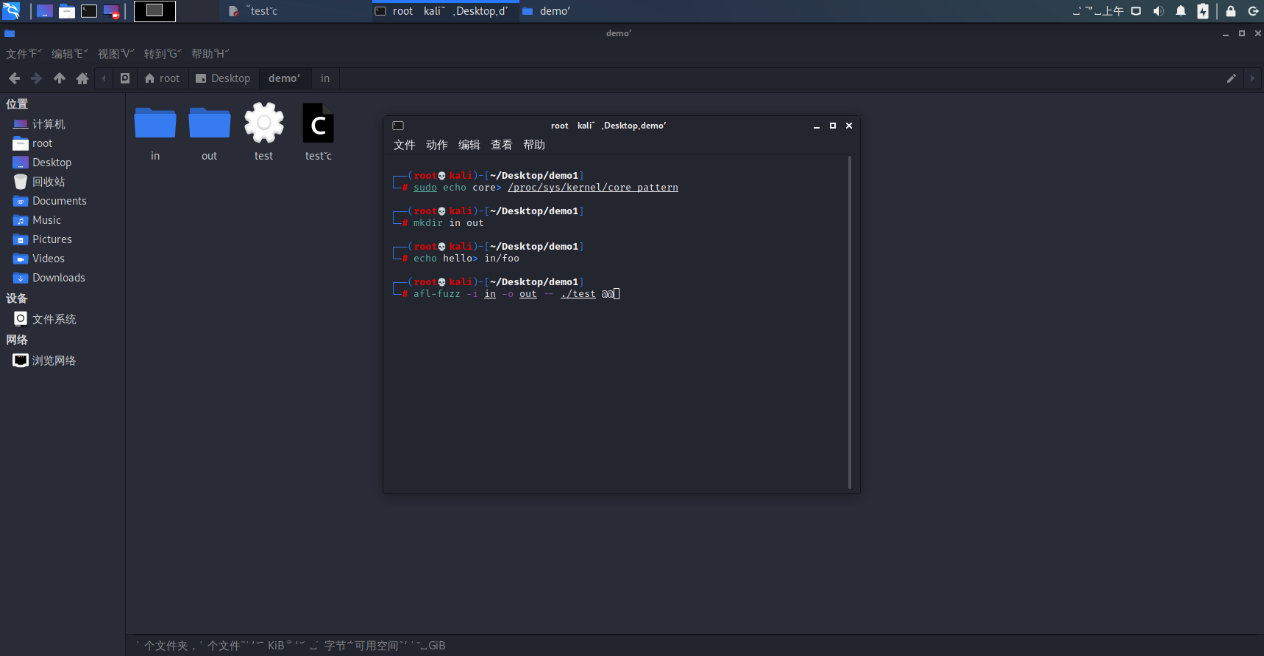
我们利用afl-gcc编译器对程序源代码进行编译。afl-gcc编译器会在编译时插桩，从而便于测试程序运行过程中的一系列信息。

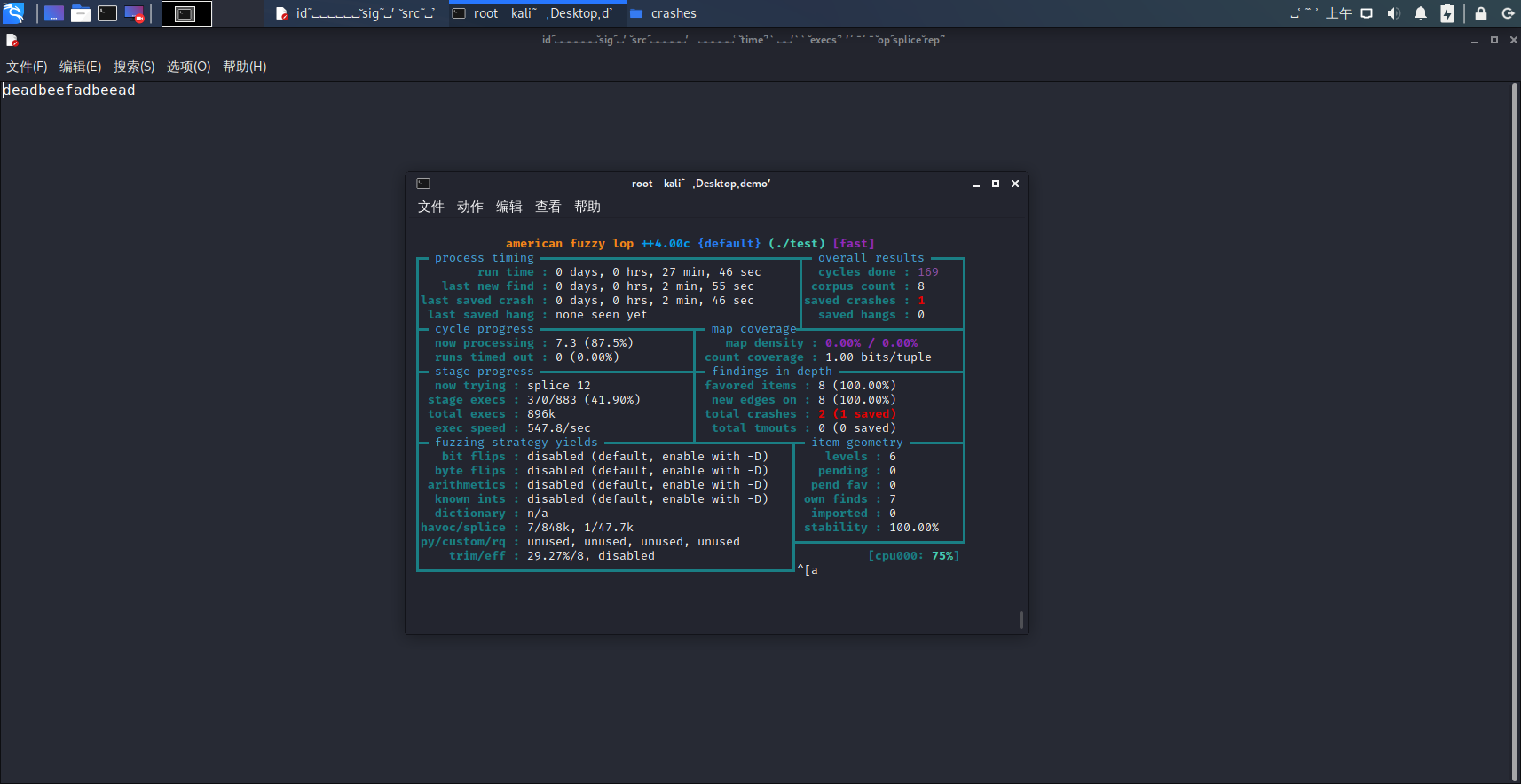




**4.进行模糊测试**

在进行模糊测试前，我们首先需要向终端输入echo core> /proc/sys/kernel/core\_pattern命令重定向core dumps到core\_pattern文件中，而不是将它们发送到特定的崩溃处理程序。接下来，我们还需要创建in和out文件夹分别用来存储模糊查询的输入信息和输出信息。然后我们在in文件夹中创建foo文件，并将其内容初始化为“hello”，AFL模糊测试会通过该文件变异产生其他的输入文件进行测试。之后便可以开始进行AFL模糊测试，在终端中输入命令afl-fuzz -i in -o out -- ./test @@开始执行模糊测试。我们重点关注测试过程中total crashes的值，total crashes值表示找到的总的使程序崩溃的输入样例数目。我们可以在/out/default/crashes路径下找到保存的使程序crash的输入样例，从而达到我们进行AFL模糊测试的目的。





**5.概念理解**

覆盖引导：

通过记录输入样本的**代码覆盖率**，从而调整输入样本以提高覆盖率，增加发现漏洞的概率。

文件变异：

模糊测试过程中，通过将队列中的文件取出并对其进行变异，来生成不同的测试样例。 有如下不同的文件变异策略：

* + bit flips：AFL采用的第一个也是最基本的策略,执行顺序的、有序的位翻转。步距总是一位；连续翻转的位数从1到4不等。
  + byte flips：位翻转方法的自然扩展，该方法依赖于8、16或32位宽的位翻转，并且具有一个字节的恒定步距。
  + arithmetic：与位翻转不同的是，从字节级别开始，而且每次进行的是加减操作，而不是翻转；
  + dictionary：用户提供的字典里有token，用来替换要进行变异的文件内容，如果用户没提供就使用bitflip自动生成的token；
  + havoc：进行很大程度的杂乱破坏，规则很多，基本上换完就是面目全非的新文件；
  + splice：通过将两个文件按一定规则进行拼接，得到一个效果不同的新文件。