《漏洞利用及渗透测试基础》实验报告

姓名：姜志凯 学号：2011937 班级：信安班

**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. 启动虚拟机：



解压



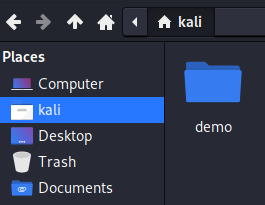
进入，用VM打开下面的文件



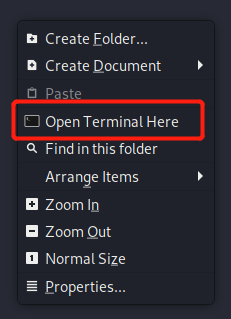
即可进入Linux-kali操作系统的虚拟机，账户和密码都是kali，准备实验。

1. 安装AFL：

进入kali文件夹，新建demo文件夹，进入demo

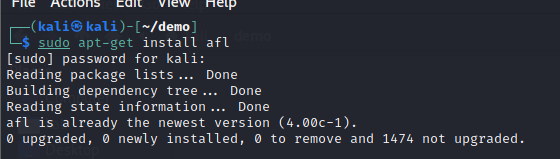


右键打开命令行

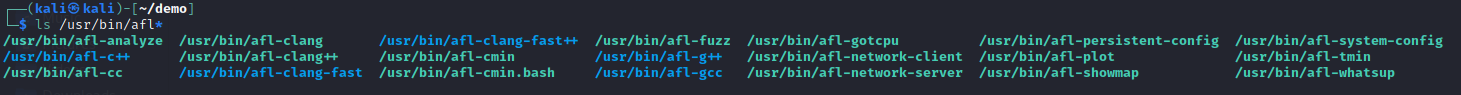


输入命令sudo apt-get install afl安装AFL

（已经安装过了，所以显示已经安装，没有安装过程）

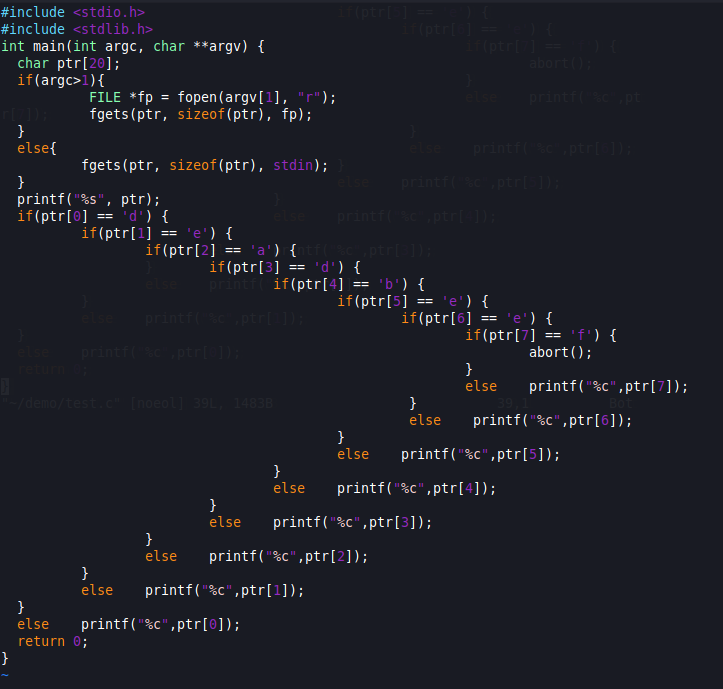


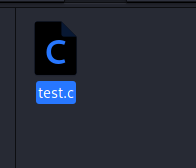
命令ls /usr/bin/afl\*可以看到AFL安装的文件



1. 创建本次实验的程序，即要进行模糊测试的目标程序：

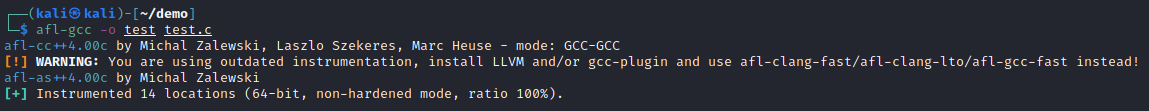
在demo中创建本次实验的程序test.c（该代码编译后得到的程序如果被传入“deadbeef”则会终止，如果传入其他字符会原样输出）



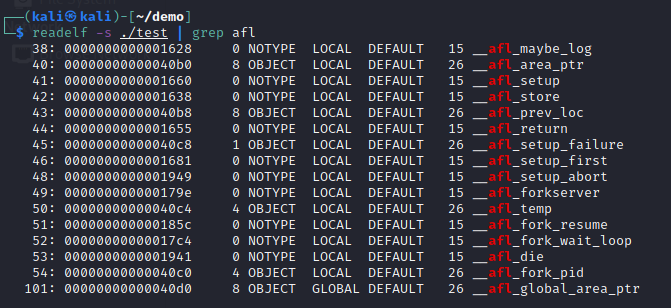


1. 使用afl的编译器编译：

输入命令afl-gcc -o test test.c



编译后会有插桩符号，用readelf -s ./test | grep afl命令验证

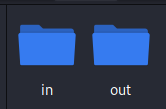


使用命令echo core > /proc/sys/kernel/core\_pattern指示系统将coredumps输出为文件



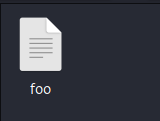
使用命令mkdir in out创建两个文件夹in和out，分别存储模糊测试所需的输入和输出相关的内容





命令echo hello> in/foo在输入文件夹中创建一个包含字符串“hello”的文件，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。

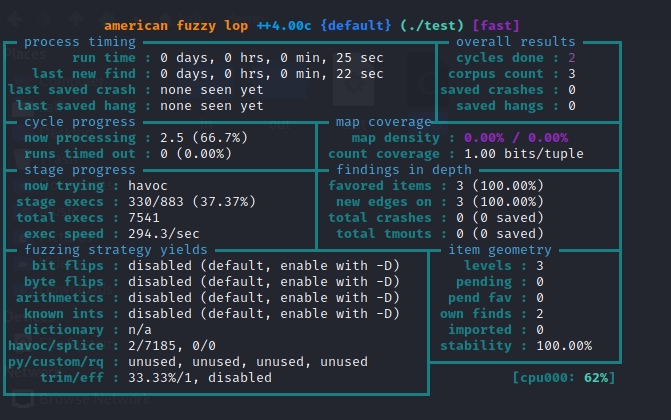






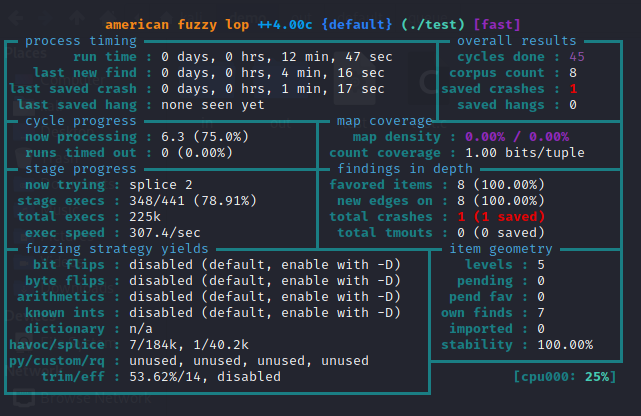
1. 启动模糊测试：

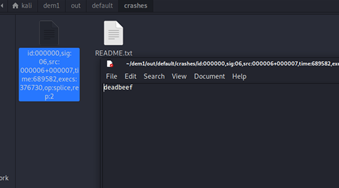
命令afl-fuzz -i in -o out -- ./test @@，开始模糊测试，出现如下运行界面



观察fuzzing结果，如有crash，定位问题。total crashes有值时证明找到crash。

在out文件夹下的crashes子文件夹里面是我们产生crash的样例。





通常，得到crash样例后，可以将这些样例作为**目标测试程序的输入**，重新触发**异常**并**跟踪运行状态**，进行分析、定位程序出错的原因或确认存在的漏洞类型，进而进行漏洞挖掘利用或打补丁。

**心得体会：**

**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，引导样本向提高代码覆盖率的方向改变，以增加发现漏洞的概率。

代码覆盖率通过编译时插桩来记录。

**文件变异**：将一个包含一定文本的文件内容按特定的策略进行改变，达到某种功能或目的后完成变异。文件变异就是覆盖引导中的“调整输入样本”的过程。

选择一些输入文件，作为初始测试集加入输入队列（queue）；将队列中的文件按一定的策略进行“突变”；如果经过变异文件更新了覆盖范围，则将其保留添加到队列中。上述过程，一直循环进行，期间触发了crash的文件会被记录下来，作为crash样例，用来重新输入，挖掘漏洞。

几个变异策略：

Bitflip：按位翻转

Arithmetic：从 8bit 级别开始，每次进行加减操作

Interest：把一些有意思的东西“interesting values”对文件内容进行替换

等等...

AFL模糊测试就是通过大量的输入样例，来测试数据，期望某些输入可以使程序出现crash，再用这些crash重新分析程序运行过程，期望发现程序漏洞。

此外，通过本实验，掌握了多个汇编语言的用法