软件安全实验报告

学号: 2310764 姓名: 王亦辉 班次: 计科一班

1 实验名称：

AFL 模糊测试

2 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。

3 实验过程：

3.1 安装 Kali

首先在官网下载给虚拟机预构建的 Kali 操作系统，我们选择的是 VMware 的版本。Get Kali | Kali Linux

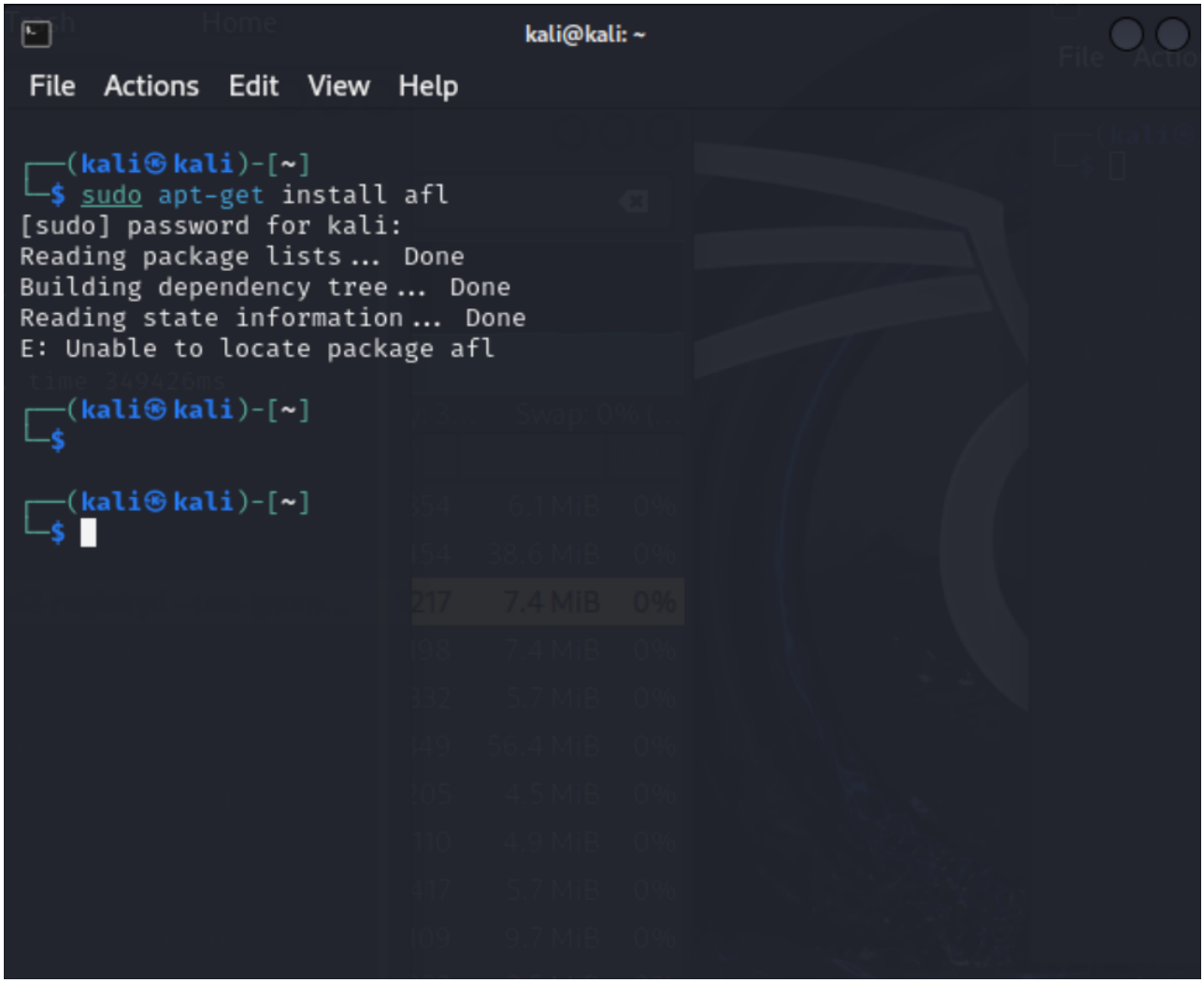
随后按照官网的教程进行操作系统的导入。Import Pre-Made Kali VMware VM | Kali Linux Documentation

使用 kali 作为用户名和密码登录

3.2 复现 AFL 在 KALI 下的安装、应用

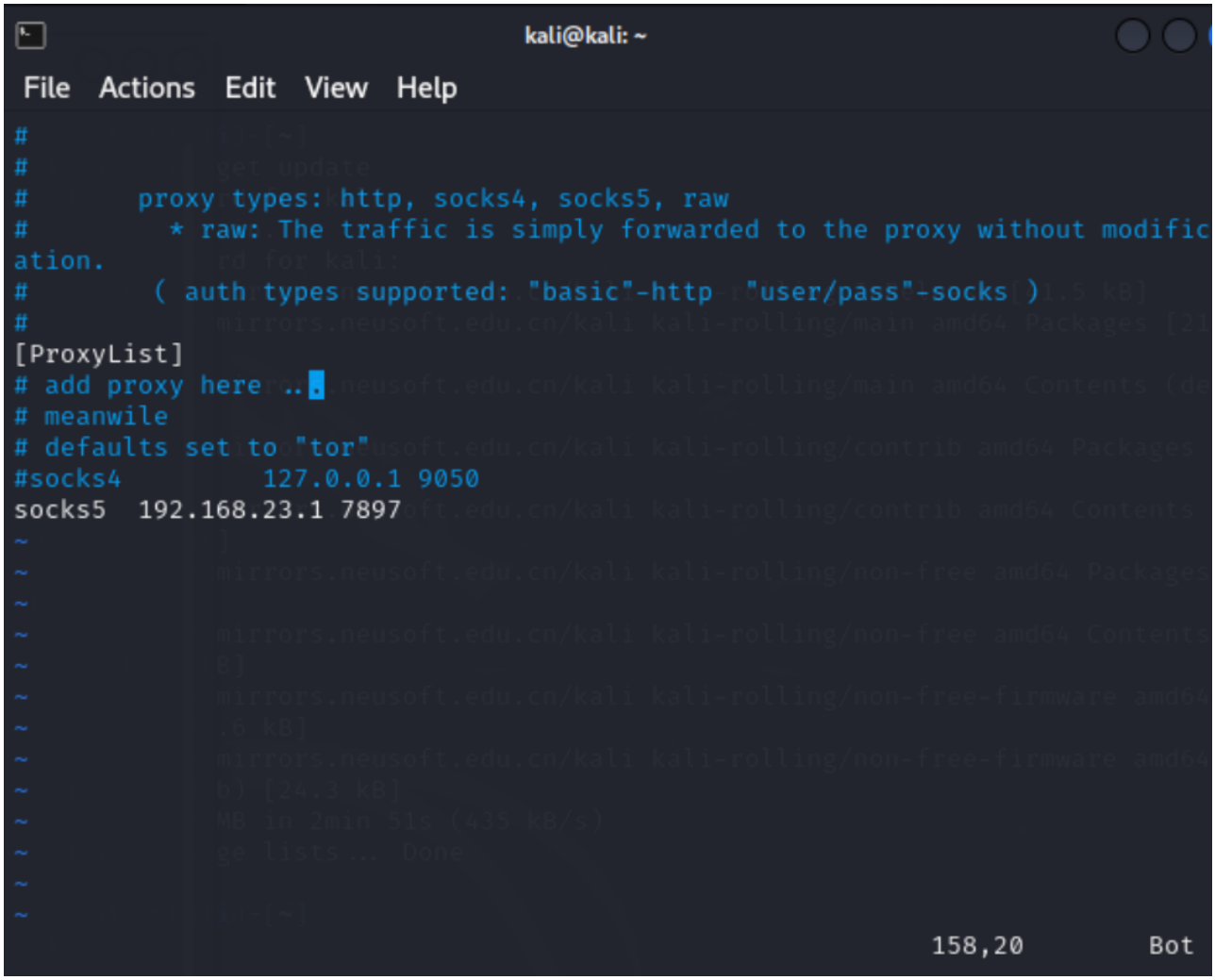
3.2.1 在安装之前，配置网络

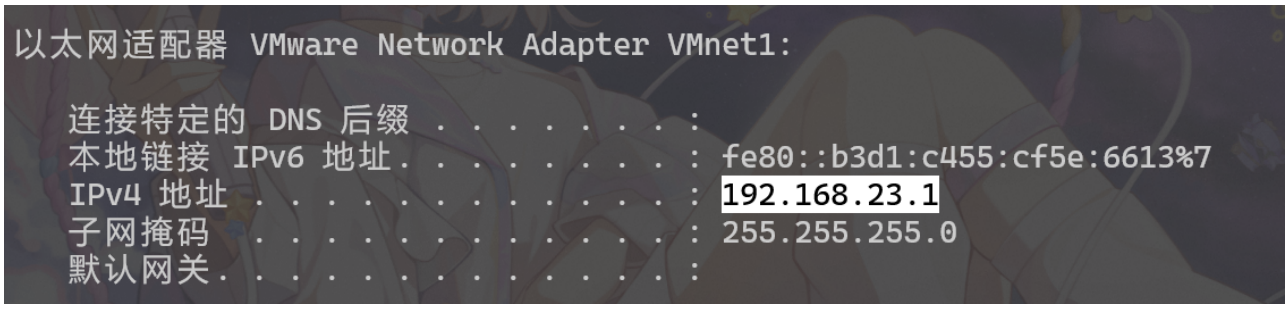
由于下载 AFL 需要使用 `sudo apt-get install afl` 命令，直接运行，可以看到无法获取到包，这是网络问题，接下来我们尝试连接主机的代理。



我下载的 Kali 虚拟机中已经内置了 `proxychains4` 工具，我们使用此工具配置代理。

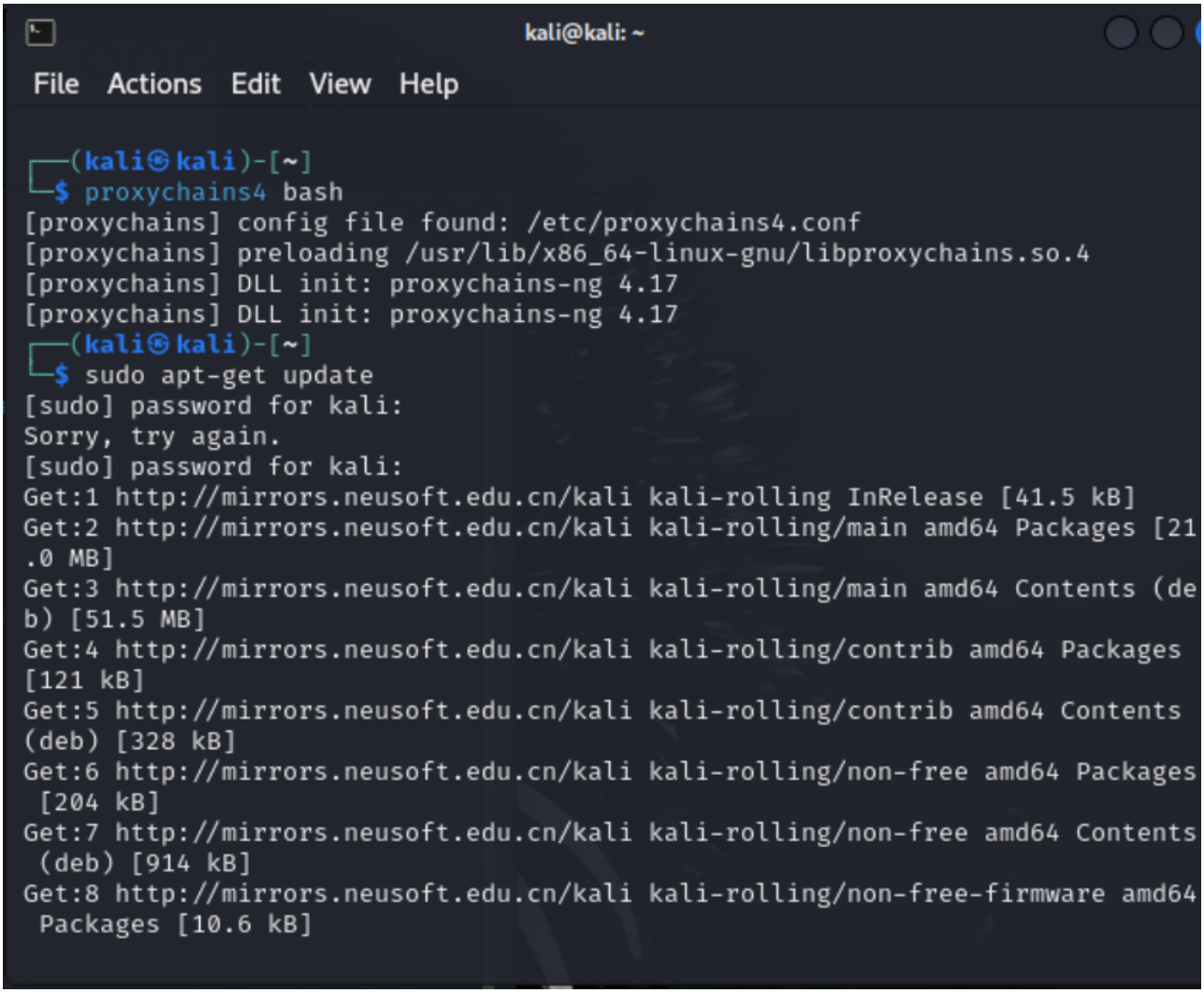
我们运行 `sudo vim /etc/proxychains4.conf` 打开该工具的配置文件，然后在 ProxyList 最后，将 `socks4` 那行注释掉(没用)。再加上一行，`socks5 192.168.23.1 7897`。其中，`192.168.23.1` 是在主机上用 `ipconfig` 查到的 VMnet1 的 `IPV4` 地址; `7897` 是主机上启用的代理的监听端口。使用 socks5 代理协议。参考教程1 教程2





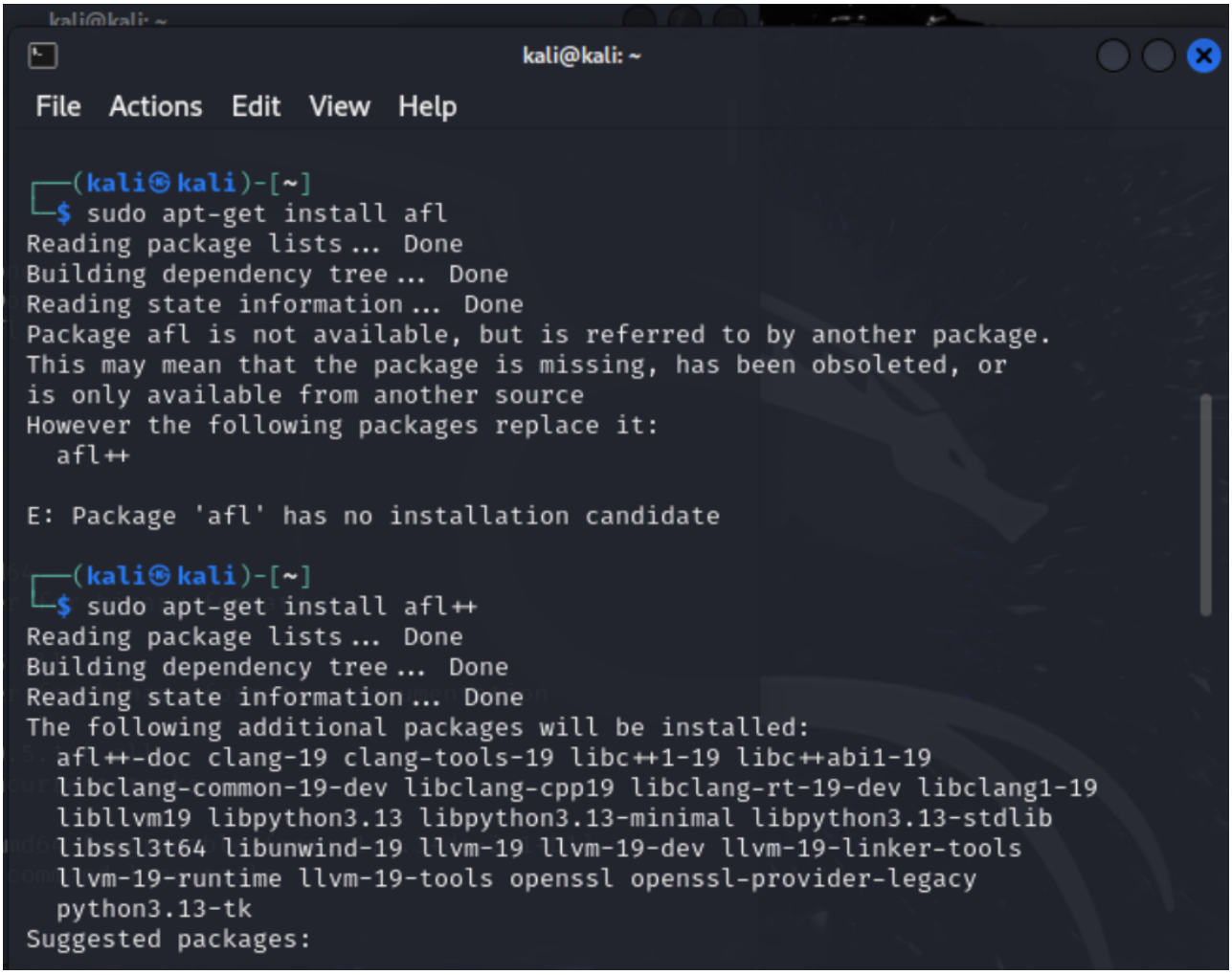
`proxychains4` 的使用方法是，在需要通过代理的命令之前加上 `proxychains4`。

可以使用 `proxychains4 bash` 启动一个流量全部由代理接管的终端，避免重复输入 `proxychains4`。在 `install` 之前，先使用 `update` 更新一下

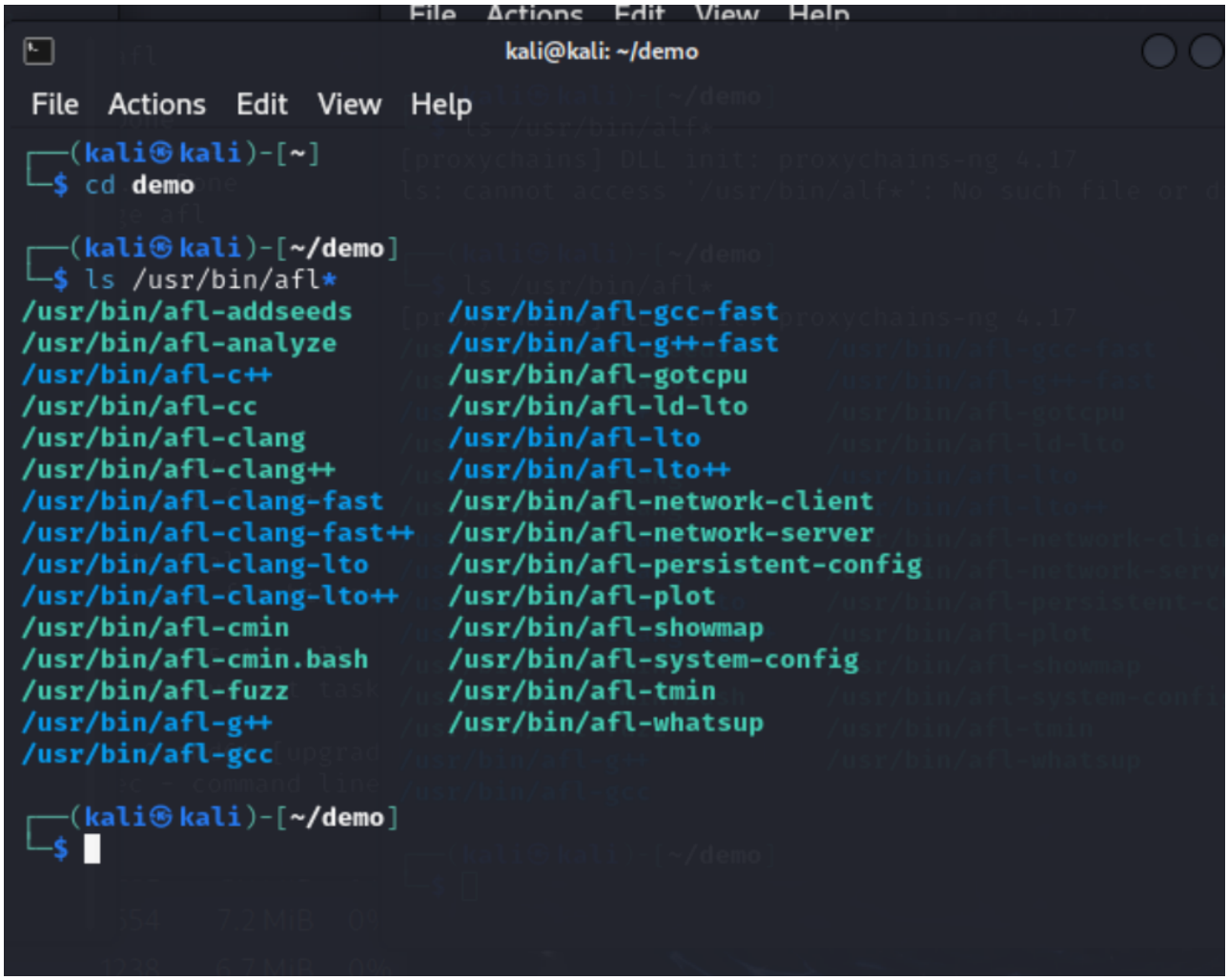


3.2.2 安装

运行命令，可以看到，afl 已经被更先进的 afl++取代了。并且 afl 的 github 仓库已经被 google 归档了。于是尝试安装 afl++.



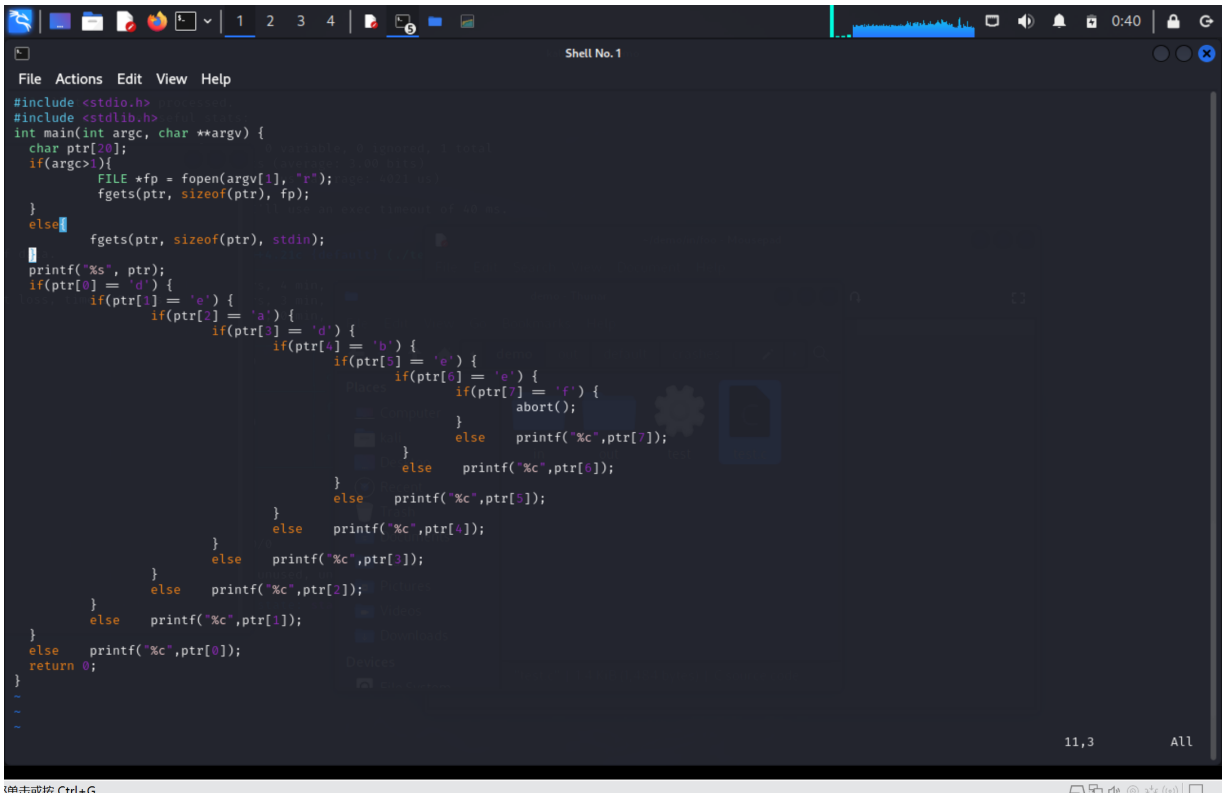
可以看到，安装完成后，`/usr/bin/` 下有了这些文件。



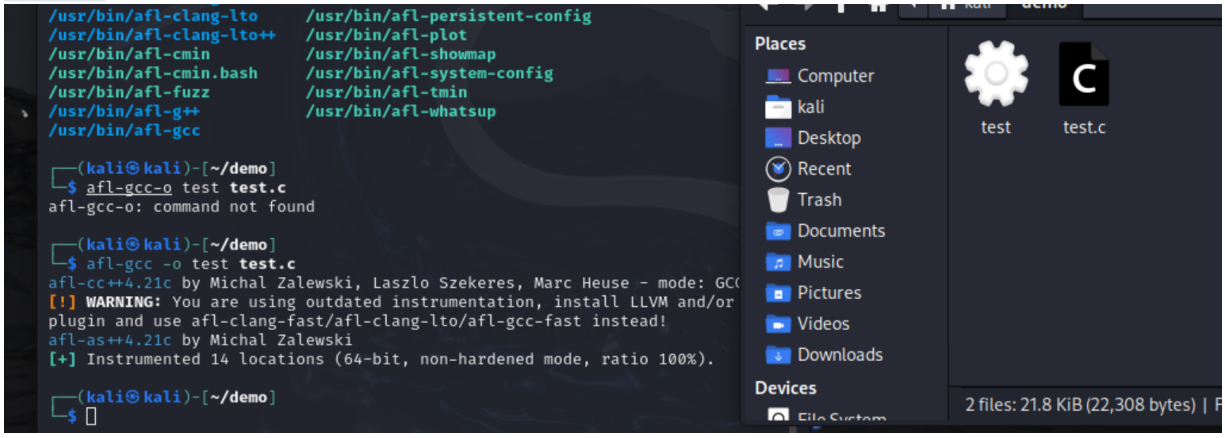
3.2.3 应用

接下来我们使用 afl++ 进行测试，看看它的能力，顺便感受下模糊测试。

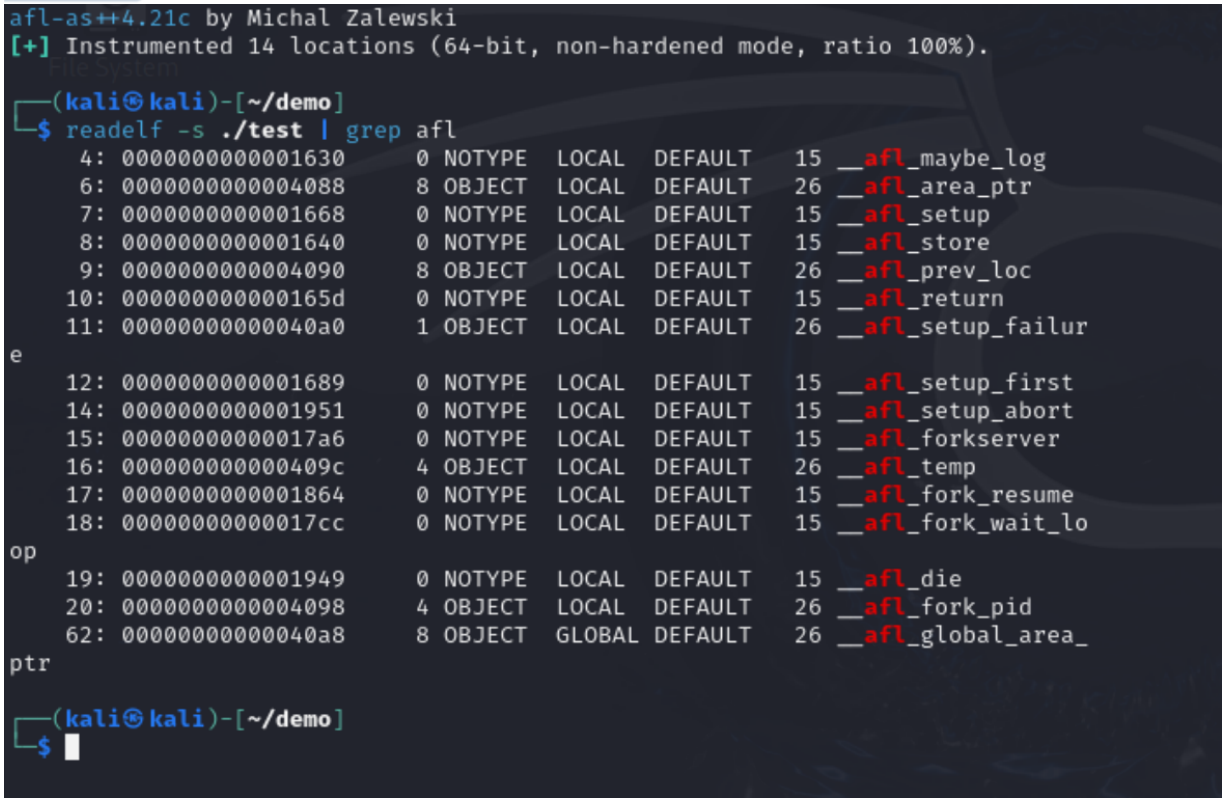
1. 创建本次实验的程序：新建文件夹 demo，并创建实验的程序 Test. c，该代码编译后得到的程序如果被传入 “deadbeef”则会终止，如果传入其他字符会原样输出。



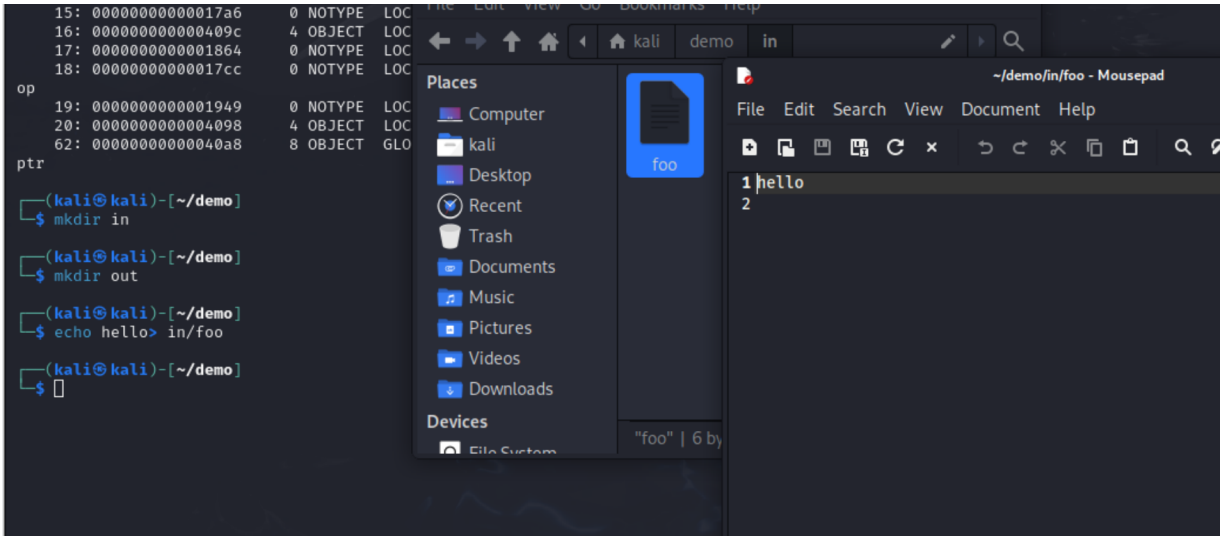
2. 使用 afl 的编译器编译，可以使模糊测试过程更加高效。命令：`afl-gcc -o test test.c`。编译出可执行文件，可以看到提示说有更好的编译方式。



3. 编译后会有插桩符号，使用下面的命令可以验证这一点。命令：`readelf -s ./test | grep afl`



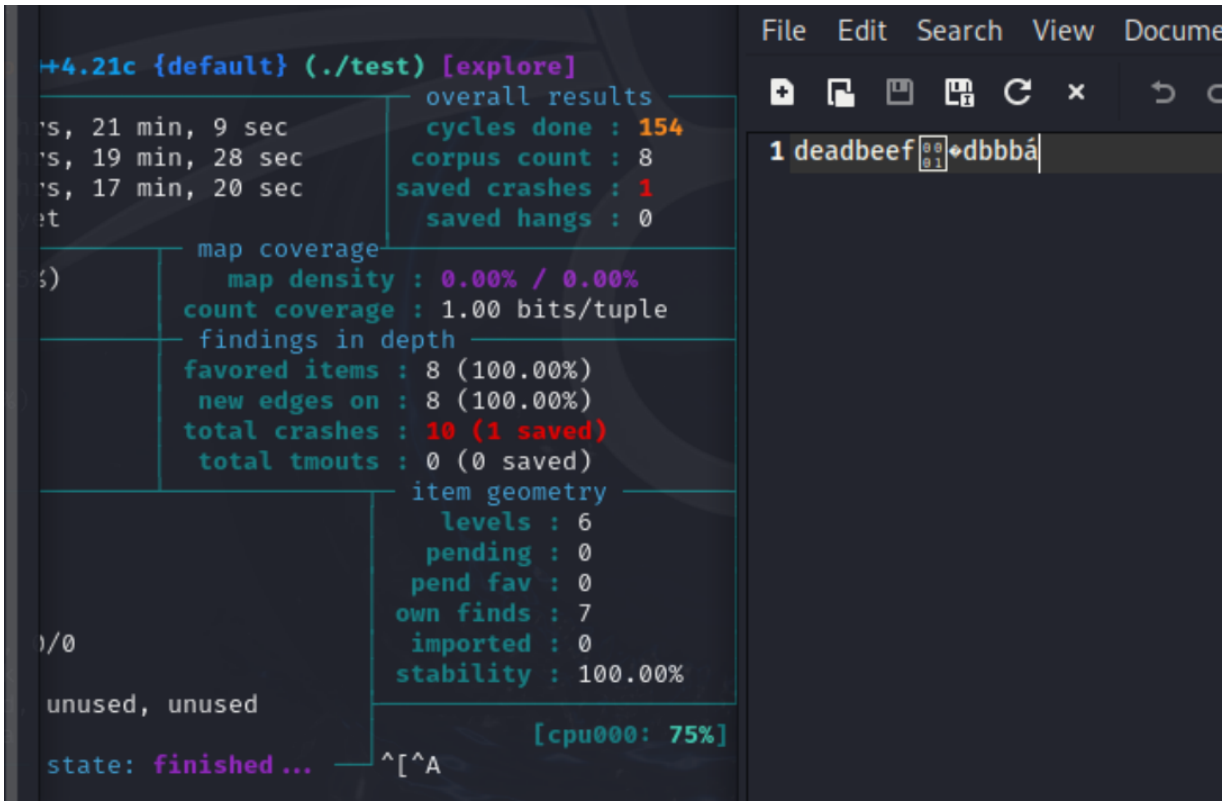
4. 创建测试用例。首先，创建两个文件夹 in 和 out，分别存储模糊测试所需的输入和输出相关的内容。然后，在输入文件夹中创建一个包含字符串 “hello”的文件。就是我们的测试用例，里面包含初步字符串 hello。AFL 会通过这个语料进行变异，构造更多的测试用例。

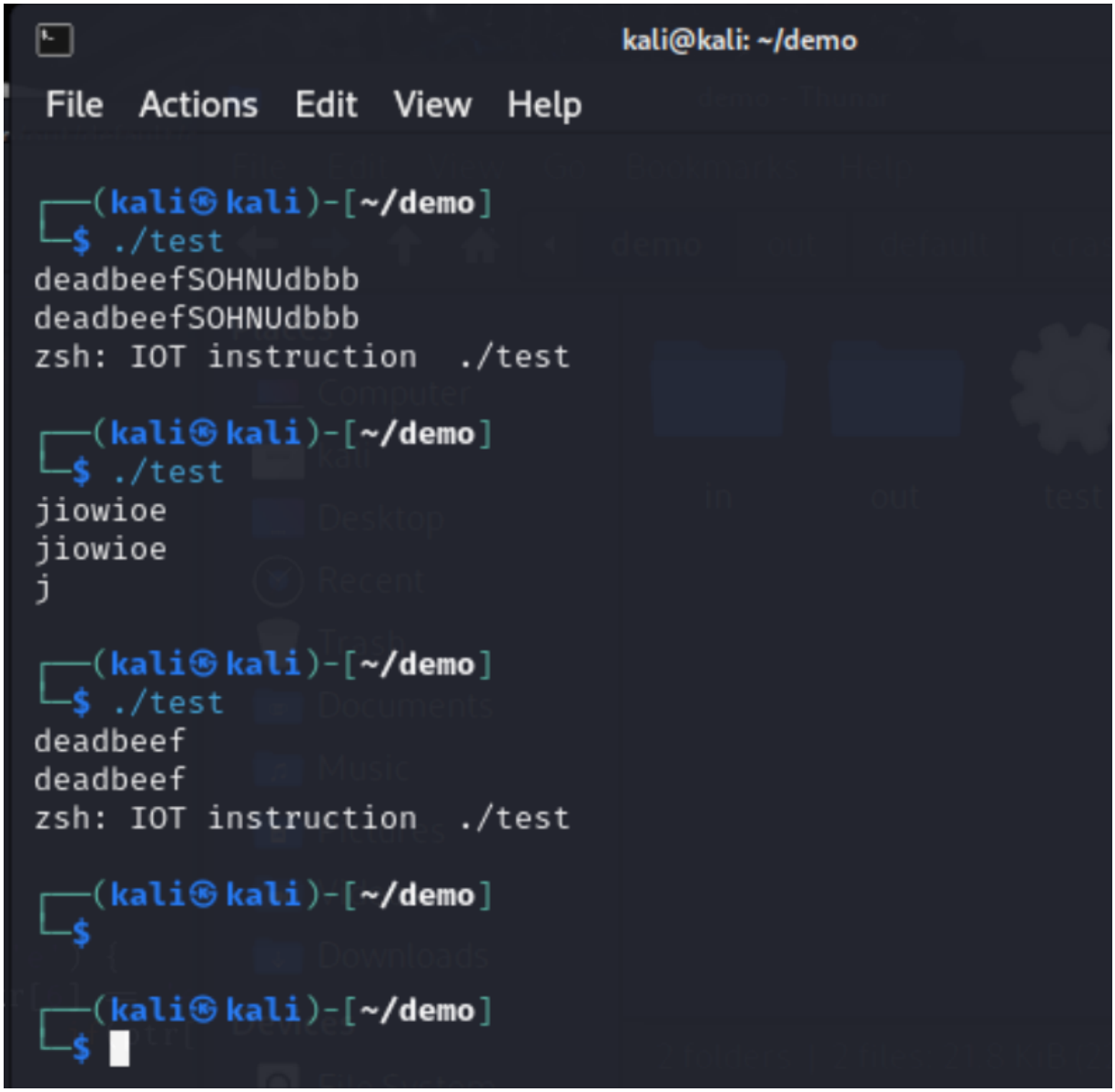


5. 启动模糊测试。运行如下命令，开始启动模糊测试（@@表示目标程序需要从文件读取输入）： 命令：`afl-fuzz -i in -o out -- ./test @@`

6. 分析 crash 观察 fuzzing 结果，如有 crash，定位问题。
   (a) 在 out 文件夹的 crashes 子文件夹里面是产生 crash 的样例，hangs 里面是产生超时的样例。
   (b) 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。

7. 可以看到，前 8 个字符确实是 deadbeef, 分析原始程序可以发现，只要前面 8 个字符正确，就会引发错误。经过手动测试也可以验证确实是这样。





3.3 覆盖引导的概念和含义

覆盖引导模糊测试是一种智能模糊测试技术，它会监控目标程序的代码覆盖情况，并基于覆盖率的反馈来生成新的测试用例，从而不断探索程序中尚未被测试到的路径。

思想：

1. Code Coverage（代码覆盖）：观察测试用例触发了哪些代码路径。
2. Guided（引导）：用新的测试用例尝试覆盖更多路径。

优点：相比传统“随机乱测”的模糊测试，它是“有目的地乱测”，效率高出很多倍。

3.3.1 文件变异的概念和含义

文件变异就是指对原始的测试输入文件（seed）做出各种扰动、修改、变形，以便触发程序新的行为甚至发现 bug。

- 基本变异方式（bit/byte level）
  1. 翻转一个 bit/byte
  2. 插入随机字节
  3. 删除某一段内容
  4. 重复一个字段或块
  5. 改变文件长度

4 心得体会：

1. 熟悉了一些 Linux 下的操作，面对命令行和 vim 显得不那么生疏了。
2. Kali 的下载，让我对 `.torrent` 文件的使用方式更了解。
3. AFL，是一款模糊测试工具，通过安装、尝试使用，我对模糊测试的基本流程和此工具更了解了。
4. 配置网络环境的过程，让我对有了更多网络方面的实践经验，可能对计算机网络的学习有一定帮助。了解到代理协议分为 `http (s)` 和 `socks`，后者功能更丰富；了解了虚拟机里的一些网络结构，比如 NAT、桥接的概念以及它们大概是怎么运作的。