

# 《AFL 模糊测试》实验报告

姓名：汤清云 学号：2013536 班级： 1075

## 实验名称：

AFL 模糊测试实验

## 实验要求：

复现 AFL 在 KALI 下的安装、应用；理解覆盖引导和文件变异的概念和含义

## 实验过程：

### 1. AFL 在 KALI 下的安装

使用语句 `sudo apt-get update` 更新系统：

```
(kali㉿kali)-[~/Chapter7_AFL]
$ sudo apt-get update
Get:1 http://kali.download/kali kali-rolling InRelease [30.6 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [18.2
Get:3 http://kali.download/kali kali-rolling/main amd64 Contents (deb
MB]
Get:4 http://kali.download/kali kali-rolling/contrib amd64 Packages [11
Get:5 http://kali.download/kali kali-rolling/contrib amd64 Contents (de
5 kB]
Get:6 http://kali.download/kali kali-rolling/non-free amd64 Packages [2
Get:7 http://kali.download/kali kali-rolling/non-free amd64 Contents (d
,005 kB]
Fetched 61.3 MB in 39s (1,567 kB/s)
Reading package lists... Done
```

使用语句 `sudo apt-get install afl` 安装 AFL：

```
(kali㉿kali)-[~/Chapter7_AFL]
$ sudo apt-get install afl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
afl++ afl++-doc clang-13 gcc-11-base gcc-12-base icu-devtools libasan6
libatomic1 libc-bin libc-dev-bin libc-l10n libc6 libc6-dev libc6-i386
libclang-common-13-dev libclang-cpp13 libclang1-13 libffi8 libgcc-11-dev
libgcc-s1 libgomp1 libicu-dev libicu67 libitm1 libllvm13 liblsan0
libobjc-11-dev libobjc4 libpython3.10 libpython3.10-minimal
libpython3.10-stdlib libquadmath0 libstdc++-11-dev libstdc++6 libtsan0
libubsan1 libxml2 libxml2-dev libz3-4 libz3-dev llvm-13 llvm-13-dev
llvm-13-linker-tools llvm-13-runtime llvm-13-tools locales rpcsvc-proto
Suggested packages:
gnuplot clang-13-doc glibc-doc libnss-nis libnss-nisplus manpages-dev
icu-doc libstdc++-11-doc pkg-config llvm-13-doc
Recommended packages:
manpages-dev libc-devtools
The following NEW packages will be installed:
```

使用语句 `ls /usr/bin/afl*` 打开查看 afl 目录下文件。

```
(kali㉿kali)-[~/Chapter7_AFL]
└─$ ls /usr/bin/afl*
/usr/bin/afl-analyze      /usr/bin/afl-gcc
/usr/bin/afl-c++          /usr/bin/afl-gotcpu
/usr/bin/afl-cc           /usr/bin/afl-network-client
/usr/bin/afl-clang        /usr/bin/afl-network-server
/usr/bin/afl-clang++      /usr/bin/afl-persistent-config
/usr/bin/afl-clang-fast   /usr/bin/afl-plot
/usr/bin/afl-clang-fast++ /usr/bin/afl-showmap
/usr/bin/afl-cmin         /usr/bin/afl-system-config
/usr/bin/afl-cmin.bash    /usr/bin/afl-tmin
/usr/bin/afl-fuzz         /usr/bin/afl-whatsup
/usr/bin/afl-g++
```

## 2. AFL 在 KALI 下的应用

添加空白 c 语言文件命名为 test.c，复制所给代码并保存，  
使用语句 `afl-gcc -o test test.c` 编译

```
(kali㉿kali)-[~/Chapter7_AFL]
└─$ afl-gcc -o test test.c
afl-cc++4.00c by Michal Zalewski, Laszlo Szekeres, Marc Heuse - mode: GCC-GCC
[!] WARNING: You are using outdated instrumentation, install LLVM and/or gcc-
plugin and use afl-clang-fast/afl-clang-lto/afl-gcc-fast instead!
afl-as++4.00c by Michal Zalewski
[+] Instrumented 14 locations (64-bit, non-hardened mode, ratio 100%).
```

使用语句 `readelf -s ./test | grep afl` 查看插桩符号

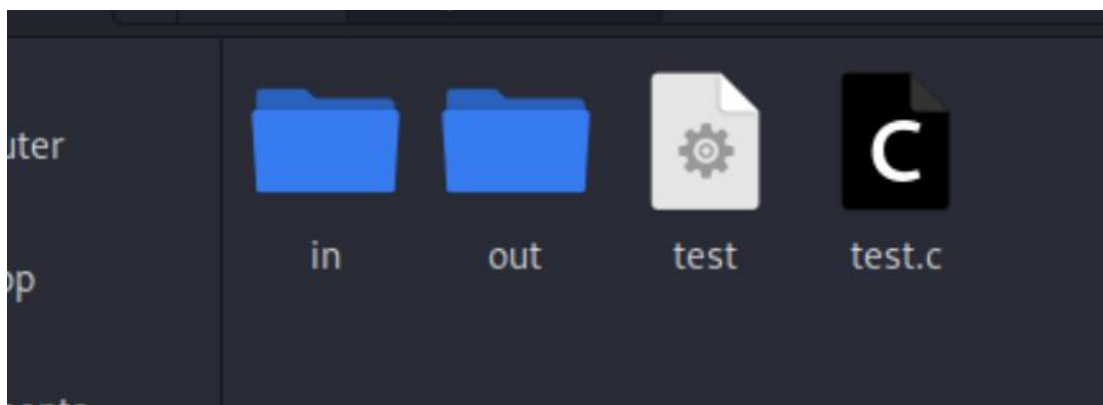
```
(kali㉿kali)-[~/Chapter7_AFL]
└─$ readelf -s ./test | grep afl
38: 00000000000001628      0 NOTYPE  LOCAL  DEFAULT 15  __afl_maybe_log
40: 000000000000040b0      8 OBJECT  LOCAL  DEFAULT 26  __afl_area_ptr
41: 00000000000001660      0 NOTYPE  LOCAL  DEFAULT 15  __afl_setup
42: 00000000000001638      0 NOTYPE  LOCAL  DEFAULT 15  __afl_store
43: 000000000000040b8      8 OBJECT  LOCAL  DEFAULT 26  __afl_prev_loc
44: 00000000000001655      0 NOTYPE  LOCAL  DEFAULT 15  __afl_return
45: 000000000000040c8      1 OBJECT  LOCAL  DEFAULT 26  __afl_setup_failur
e
46: 00000000000001681      0 NOTYPE  LOCAL  DEFAULT 15  __afl_setup_first
48: 00000000000001949      0 NOTYPE  LOCAL  DEFAULT 15  __afl_setup_abort
49: 0000000000000179e      0 NOTYPE  LOCAL  DEFAULT 15  __afl_forkserver
50: 000000000000040c4      4 OBJECT  LOCAL  DEFAULT 26  __afl_temp
51: 0000000000000185c      0 NOTYPE  LOCAL  DEFAULT 15  __afl_fork_resume
52: 000000000000017c4      0 NOTYPE  LOCAL  DEFAULT 15  __afl_fork_wait_lo
op
53: 00000000000001941      0 NOTYPE  LOCAL  DEFAULT 15  __afl_die
54: 000000000000040c0      4 OBJECT  LOCAL  DEFAULT 26  __afl_fork_pid
101: 000000000000040d0      8 OBJECT  GLOBAL  DEFAULT 26  __afl_global_area_
ptr
```

使用语句 `echo core > /proc/sys/kernel/core_pattern`，将信息输出到 `core_pattern` 位置

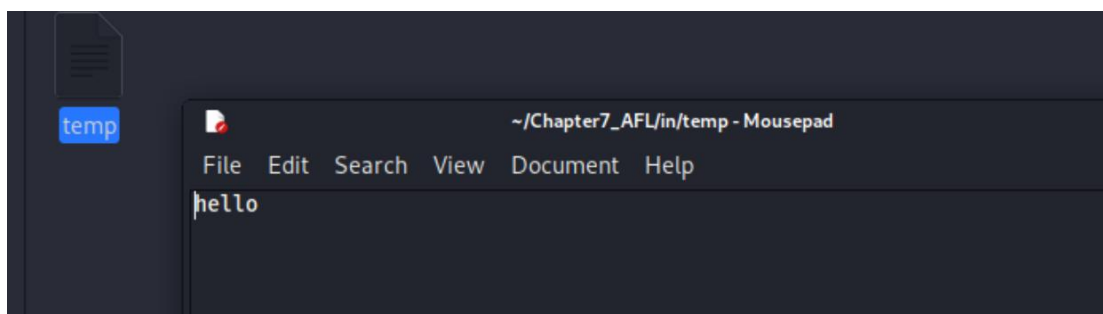
```
(kali㉿kali)-[~/Chapter7_AFL]
└─$ echo core > /proc/sys/kernel/core_pattern
zsh: permission denied: /proc/sys/kernel/core_pattern
```

使用语句 `mkdir in out` 创建文件夹 `in` 和 `out`

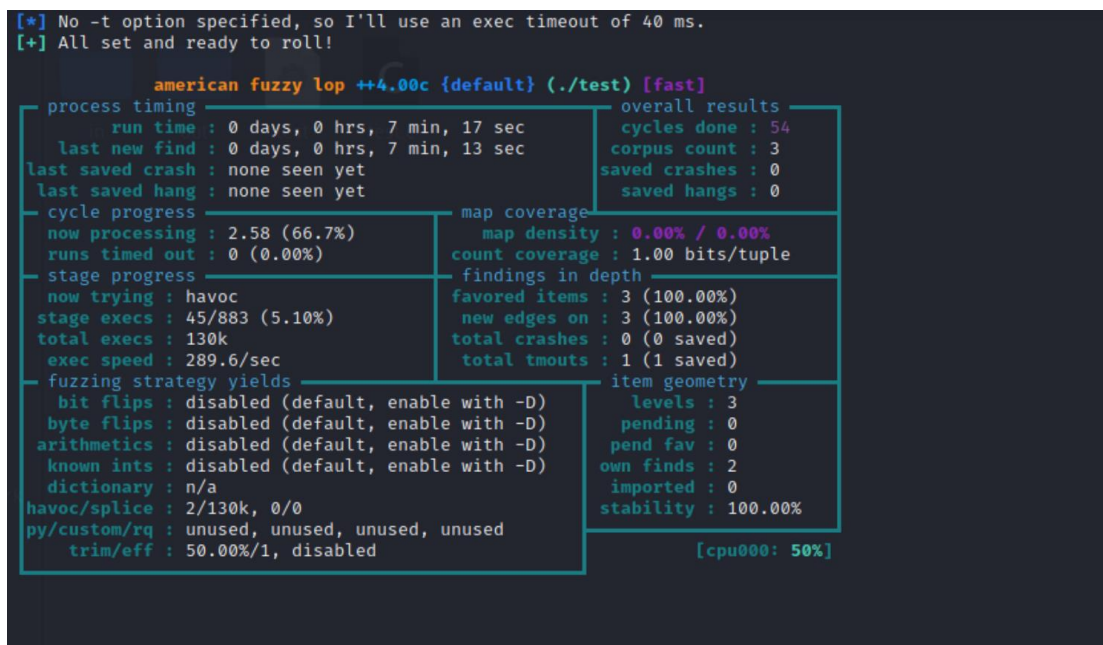
```
(kali㉿kali)-[~/Chapter7_AFL]
└─$ mkdir in out
```



使用语句 `echo hello> in/temp` 在 in 文件夹中添加文件 temp，并输入 hello 到 temp 文件中：



使用语句 `afl-fuzz -i in -o out -- ./test @@` 查看系统运行



运行结果为：



```
kali@kali: ~/Chapter7_AFL
File Actions Edit View Help

american fuzzy lop ++4.00c {default} (./test) [fast]

process timing
  run time : 0 days, 0 hrs, 28 min, 17 sec
  last new find : 0 days, 0 hrs, 5 min, 29 sec
  last saved crash : 0 days, 0 hrs, 0 min, 15 sec
  last saved hang : none seen yet

cycle progress
  now processing : 6.3 (75.0%)
  runs timed out : 0 (0.00%)

stage progress
  now trying : havoc
  stage execs : 1590/3532 (45.02%)
  total execs : 571k
  exec speed : 332.0/sec

fuzzing strategy yields
  bit flips : disabled (default, enable with -D)
  byte flips : disabled (default, enable with -D)
  arithmetics : disabled (default, enable with -D)
  known ints : disabled (default, enable with -D)
  dictionary : n/a
  havoc/splice : 8/527k, 0/42.5k
  py/custom/rq : unused, unused, unused
  trim/eff : 99.87%/40, disabled

overall results
  cycles done : 141
  corpus count : 8
  saved crashes : 1
  saved hangs : 0

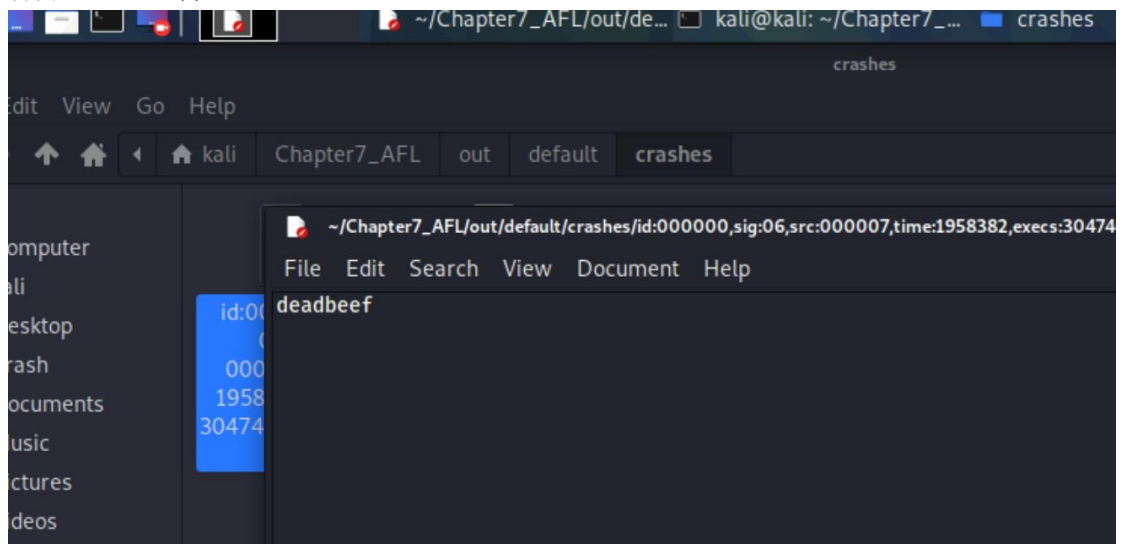
map coverage
  map density : 0.00% / 0.00%
  count coverage : 1.00 bits/tuple

findings in depth
  favored items : 8 (100.00%)
  new edges on : 8 (100.00%)
  total crashes : 1 (1 saved)
  total tmouts : 2 (2 saved)

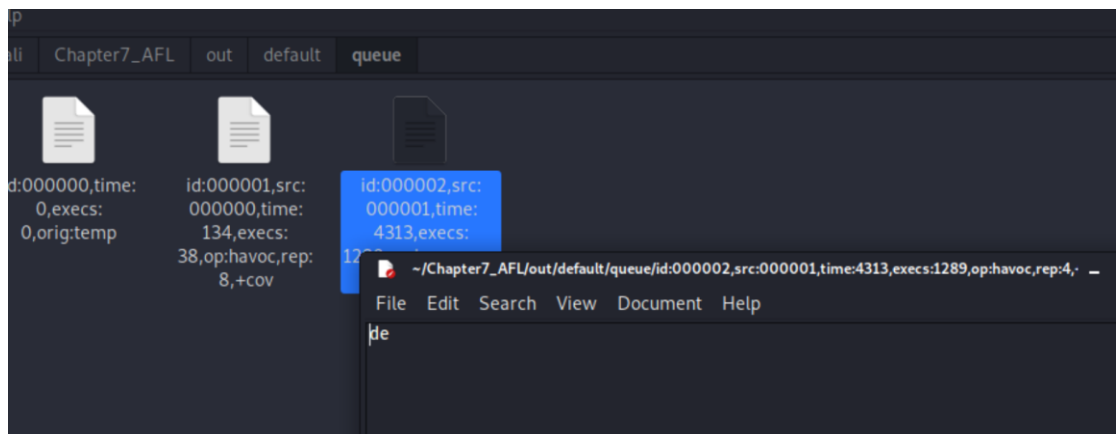
item geometry
  levels : 7
  pending : 0
  pend fav : 0
  own finds : 7
  imported : 0
  stability : 100.00%

[cpu000: 75%]
```

打开 crashes 得:



- 覆盖引导和文件变异的概念以及含义  
变异字符存放在/out/default/queue 文件夹下的文档中。



覆盖引导：将文件按照一定策略进行“突变”，如果这些突变了的文件能够更新覆盖

范围，则保留在队列中。

文件变异：1. 按位翻转；2. 从 8bit 级别开始进行加减操作；3. 使用 1 过程中产生的重要的额外重要数据信息替换文件内容；4. 使用 token 替换要进行变异的文件内容；5. 对源文件进行大量变异：随机选择 bit 翻转；随机选择 byte 构成随机的重要信息；6. 拼接两个文件，如果两个文件差别不大则重新选择，否则就随机选取位置将两个文件切割，将当前文件的头与随机文件的尾拼接得到新文件；7. 下一轮继续变异。

#### **心得体会：**

对 kali 系统下的 AFL 模糊测试进行初步了解；

在老师的引导下复现了文件变异实验；

查阅资料了解了更多 AFL 在 kali 下的应用；

了解了覆盖引导和文件变异的概念以及含义。