

COMSM0049 Lab 6

1. Download and compile AFL:

git clone https://github.com/google/AFL

cd afl

make && sudo make install

2. check if afl-fuzz and afl-gcc are working:

```
manos@ubuntu:~/Desktop/AFL$ ./afl-fuzz --help
afl-fuzz 2.57b by <lcantuf@google.com>
./afl-fuzz: invalid option -- '-'

./afl-fuzz [ options ] -- /path/to/fuzzed_app [ ... ]

Required parameters:

-i dir      - input directory with test cases
-o dir      - output directory for fuzzer findings

Execution control settings:

-f file     - location read by the fuzzed program (stdin)
-t msec     - timeout for each run (auto-scaled, 50-1000 ms)
-m megs     - memory limit for child process (50 MB)
-Q          - use binary-only instrumentation (QEMU mode)

Fuzzing behavior settings:

-d          - quick & dirty mode (skips deterministic steps)
-n          - fuzz without instrumentation (dumb mode)
-x dir      - optional fuzzer dictionary (see README)

Other stuff:

-T text     - text banner to show on the screen
-M / -S id  - distributed mode (see parallel_fuzzing.txt)
-C          - crash exploration mode (the peruvian rabbit thing)
-V          - show version number and exit

-b cpu_id   - bind the fuzzing process to the specified CPU core

For additional tips, please consult /usr/local/share/doc/afl/README.
```

```
manos@ubuntu:~/Desktop/AFL$ ./afl-gcc --version
afl-gcc 2.57b by <lcantuf@google.com>
gcc (Ubuntu 5.4.0-6ubuntu1~16.04.12) 5.4.0 20160609
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

3. create these subdirectories in the root directory:

\$ mkdir temp-in

\$ mkdir temp-out

\$ mkdir test-bins

3. Move heartbeat_normal.bin into temp-in:

4. Compile heartbeat.c

```
/home/manos/Desktop/AFL$ ./afl-gcc /home/manos/Desktop/AFL/heartbeat.c -o /home/manos/Desktop/AFL/test-bins/heartbeat
afl-gcc 2.57b by <lcantuf@google.com>
afl-as 2.57b by <lcantuf@google.com>
[+] Instrumented 24 locations (64-bit, non-hardened mode, ratio 100%).
```

5. Run AFL fuzzer: ./afl-fuzz -i /home/manos/Desktop/AFL/temp-in/ -o

/home/manos/Desktop/AFL/temp-out/ /home/manos/Desktop/AFL/test-bins/heartbleed @@

```
manos@ubuntu:~/Desktop/AFL$ ./afl-fuzz -i /home/manos/Desktop/AFL/temp-in/ -o /home/manos/Desktop/AFL/temp-out/ /home/manos/Desktop/AFL/test-bins/heartbleed @@
afl-fuzz 2.57b by <lcantuf@google.com>
[+] You have 2 CPU cores and 2 runnable tasks (utilization: 100%).
[*] Checking CPU core loadout...
[*] Found a free CPU core, binding to #0.
[*] Checking core_pattern...

[-] Hmm, your system is configured to send core dump notifications to an external utility. This will cause issues: there will be an extended delay between stumbling upon a crash and having this information relayed to the fuzzer via the standard waitpid() API.

To avoid having crashes misinterpreted as timeouts, please log in as root and temporarily modify /proc/sys/kernel/core_pattern, like so:

echo core >/proc/sys/kernel/core_pattern

[-] PROGRAM ABORT : Pipe at the beginning of 'core_pattern'
    Location : check_crash_handling(), afl-fuzz.c:7347
```

Fix:

echo core | sudo tee /proc/sys/kernel/core_pattern

./afl-fuzz -i /home/manos/Desktop/AFL/temp-in/ -o /home/manos/Desktop/AFL/temp-out/ /home/manos/Desktop/AFL/test-bins/heartbleed

```
manos@ubuntu:~/Desktop/AFL$ ./afl-fuzz -i /home/manos/Desktop/AFL/temp-in/ -o /home/manos/Desktop/AFL/temp-out/ /home/manos/Desktop/AFL/test-bins/heartbleed @@
afl-fuzz 2.57b by <lcantuf@google.com>
[+] You have 2 CPU cores and 2 runnable tasks (utilization: 100%).
[*] Checking CPU core loadout...
[*] Found a free CPU core, binding to #0.
[*] Checking core_pattern...

[-] Hmm, your system is configured to send core dump notifications to an external utility. This will cause issues: there will be an extended delay between stumbling upon a crash and having this information relayed to the fuzzer via the standard waitpid() API.

To avoid having crashes misinterpreted as timeouts, please log in as root and temporarily modify /proc/sys/kernel/core_pattern, like so:

echo core >/proc/sys/kernel/core_pattern

[-] PROGRAM ABORT : Pipe at the beginning of 'core_pattern'
    Location : check_crash_handling(), afl-fuzz.c:7347
```

export AFL_SKIP_CPUFREQ=1

sudo su (root) :

echo core >/proc/sys/kernel/core_pattern

./afl-fuzz -i /home/manos/Desktop/AFL/temp-in/ -o /home/manos/Desktop/AFL/temp-out/ /home/manos/Desktop/AFL/test-bins/heartbleed @@

```

+++ Testing aborted by user +++
[+] We're done here. Have a nice day!

root@ubuntu:/home/manos/Desktop/AFL# ./afl-fuzz -i /home/manos/Desktop/
/home/manos/Desktop/AFL/test-bins/heartbleed @@
afl-fuzz 2.57b by <lcantuf@google.com>
[+] You have 2 CPU cores and 2 runnable tasks (utilization: 100%).
[*] Checking CPU core loadout...
[+] Found a free CPU core, binding to #0.
[*] Checking core_pattern...
[*] Setting up output directories...
[+] Output directory exists but deemed OK to reuse.
[*] Deleting old session data...
[+] Output dir cleanup successful.
[*] Scanning '/home/manos/Desktop/AFL/temp-in/'...
[+] No auto-generated dictionary tokens to reuse.
[*] Creating hard links for all input files...
[*] Validating target binary...
[*] Attempting dry run with 'id:000000,orig:heartbea_normal.bin'...
[*] Spinning up the fork server...
[+] All right - fork server is up.
len = 9, map size = 11, exec speed = 1365 us
[+] All test cases processed.

[+] Here are some useful stats:

Test case count : 1 favored, 0 variable, 1 total
Bitmap range : 11 to 11 bits (average: 11.00 bits)
Exec timing : 1365 to 1365 us (average: 1365 us)

[*] No -t option specified, so I'll use exec timeout of 20 ms.
[+] All set and ready to roll!

```

The screenshot shows the AFL 2.57b interface with the title "american fuzzy lop 2.57b (heartbleed)". It displays a sidebar with icons for settings, file explorer, terminal, and other utilities. The main area is divided into several sections:

- process timing**: run time : 0 days, 0 hrs, 0 min, 22 sec; last new path : 0 days, 0 hrs, 0 min, 18 sec; last uniq crash : none seen yet; last uniq hang : none seen yet.
- cycle progress**: now processing : 0 (0.00%); paths timed out : 0 (0.00%).
- stage progress**: now trying : havoc; stage execs : 7308/32.8k (22.30%); total execs : 8699; exec speed : 366.8/sec.
- fuzzing strategy yields**: bit flips : 9/64, 2/63, 1/61; byte flips : 0/8, 0/7, 0/5; arithmetics : 1/447, 0/126, 0/0; known ints : 0/41, 0/175, 0/220; dictionary : 0/0, 0/0, 0/0; havoc : 0/0, 0/0; trim : 11.11%/2, 0.00%.
- map coverage**: map density : 0.02% / 0.05%; count coverage : 1.44 bits/tuple.
- findings in depth**: favored paths : 1 (4.76%); new edges on : 8 (38.10%); total crashes : 0 (0 unique); total tmouts : 6 (3 unique).
- overall results**: cycles done : 0; total paths : 21; uniq crashes : 0; uniq hangs : 0.
- path geometry**: levels : 2; pending : 21; pend fav : 1; own finds : 20; imported : n/a; stability : 100.00%.

At the bottom right, it shows "[cpu000:102%]".

6. Stop the fuzzer (ctrl C)

7. Compile the heartbleed.c again:

```
./afl-gcc -m32 -fsanitize=address /home/manos/Desktop/AFL/heartbleed.c -o
/home/manos/Desktop/AFL/test-bins/heartbleed-asan
```

(if this error: usr/include/features.h:367:25: fatal error: sys/cdefs.h: No such file or directory try and install apt install gcc-multilib)

Do again: ./afl-gcc -m32 -fsanitize=address heartbleed.c -o test-bins/heartbleed-asan

8. Run fuzzer:

```
./afl-fuzz -i /home/manos/Desktop/AFL/temp-in/ -o tesmp-out
/home/manos/Desktop/AFL/test-bins/heartbleed-asan @@
```

```

manos@ubuntu:~/Desktop/AFL$ ./afl-fuzz -i /home/manos/Desktop/AFL/temp-in/ -o tesmp-out /home/manos/Desktop/AFL/test-bins/heartbleed-asan @@
afl-fuzz 2.57b by <lcantuf@google.com>
[*] You have 2 CPU cores and 1 runnable tasks (utilization: 50%).
[*] Try parallel jobs - see /usr/local/share/doc/afl/parallel_fuzzing.txt.
[*] Checking CPU core loadout...
[*] Found a free CPU core, binding to #0.
[*] Checking core_pattern...
[*] Setting up output directories...
[*] Scanning /home/manos/Desktop/AFL/temp-in/'...
[*] No auto-generated dictionary tokens to reuse.
[*] Creating hard links for all input files...
[*] Validating target binary...
[*] Attempting dry run with 'id:000000,orig:heartbea_normal.bin'...
[*] Spinning up the fork server...

[-] Whoops, the target binary crashed suddenly, before receiving any input
    from the fuzzer! Since it seems to be built with ASAN and you have a
    restrictive memory limit configured, this is expected; please read
    /usr/local/share/doc/afl/notes_for_asan.txt for help.

[-] PROGRAM ABORT : Fork server crashed with signal 6
    Location : init_forkserver(), afl-fuzz.c:2230

```

Run again with -m:

```

./afl-fuzz -m 600 -i /home/manos/Desktop/AFL/temp-in/ -o tesmp-out
/home/manos/Desktop/AFL/test-bins/heartbleed-asan @@

```

If you get an error saying 600 is too low a memory to run the program, Please try:

```

./afl-fuzz -m none -i temp-in/ -o temp-out test-bins/heartbleed-asan @@

```

american fuzzy lop 2.57b (heartbleed-asan)			
process timing		overall results	
run time	: 0 days, 0 hrs, 0 min, 13 sec	cycles done	: 0
last new path	: 0 days, 0 hrs, 0 min, 10 sec	total paths	: 11
last uniq crash	: 0 days, 0 hrs, 0 min, 10 sec	uniq crashes	: 3
last uniq hang	: none seen yet	uniq hangs	: 0
cycle progress		map coverage	
now processing	: 0 (0.00%)	map density	: 0.03% / 0.05%
paths timed out	: 0 (0.00%)	count coverage	: 1.37 bits/tuple
stage progress		findings in depth	
now trying	: havoc	favored paths	: 1 (9.09%)
stage execs	: 5470/32.8k (16.69%)	new edges on	: 4 (36.36%)
total execs	: 6778	total crashes	: 5173 (3 unique)
exec speed	: 502.9/sec	total tmouts	: 0 (0 unique)
fuzzing strategy yields		path geometry	
bit flips	: 3/64, 0/63, 1/61	levels	: 2
byte flips	: 0/8, 0/7, 0/5	pending	: 11
arithmetics	: 1/447, 0/126, 0/0	pend fav	: 1
known ints	: 0/41, 0/175, 0/220	own finds	: 10
dictionary	: 0/0, 0/0, 0/0	imported	: n/a
havoc	: 0/0, 0/0	stability	: 100.00%
trim	: 11.11%/2, 0.00%		
[cpu000:100%]			

9. check the temp-out/crashes directory:

```

./test-bins/heartbleed-asan <input>

```

Three unique crashes found:

american fuzzy lop 2.57b (heartbleed-asan)		
process timing		overall results
run time :	0 days, 5 hrs, 31 min, 17 sec	cycles done : 2802
last new path :	0 days, 5 hrs, 18 min, 17 sec	total paths : 19
last uniq crash :	0 days, 5 hrs, 31 min, 14 sec	uniq crashes : 3
last uniq hang :	none seen yet	uniq hangs : 0
cycle progress	map coverage	
now processing : 14* (73.68%)	map density : 0.04% / 0.05%	
paths timed out : 0 (0.00%)	count coverage : 1.94 bits/tuple	
stage progress	findings in depth	
now trying : splice 2	favorable paths : 4 (21.05%)	
stage execs : 12/32 (37.50%)	new edges on : 4 (21.05%)	
total execs : 11.1M	total crashes : 6.81M (3 unique)	
exec speed : 605.3/sec	total tmouts : 18 (4 unique)	
fuzzing strategy yields	path geometry	
bit flips : 5/411k, 0/411k, 1/411k	levels : 4	
byte flips : 0/51.5k, 0/327, 0/303	pending : 0	
arithmetics : 1/18.9k, 0/13.6k, 0/6028	pend fav : 0	
known ints : 0/1408, 0/6484, 0/11.2k	own finds : 18	
dictionary : 0/0, 0/0, 0/0	imported : n/a	
havoc : 10/3.55M, 4/6.24M	stability : 100.00%	
trim : 37.06%/7171, 99.32%		
[cpu000:101%]		

./test-bins/heartbleed-asan

/home/manos/Desktop/AFL/tesmp-out/crashes/id:000000,sig:06,src:000000,op:flip1,pos:1

```
root@ubuntu:/home/manos/Desktop/AFL# ./test-bins/heartbleed-asan /home/manos/Desktop/AFL/tesmp-out/crashes/id:000000,sig:06,src:000000,op:flip1,pos:1
1
/home/manos/Desktop/AFL/tesmp-out/crashes/id:000000,sig:06,src:000000,op:flip1,pos:1
=====
==65944==ERROR: AddressSanitizer: heap-buffer-overflow on address 0xf5c007f9 at pc 0xf7a8ca15 bp 0xff8c6d68 sp 0xff8c693c
READ of size 32774 at 0xf5c007f9 thread T0
#0 0xf7a8ca14 in _asan_memcpy (/usr/lib32/libasan.so.2+0x8aa14)
#1 0xf7a8cbcf in memcpy (/usr/lib32/libasan.so.2+0x8abcf)
#2 0x804927f in memcpy /usr/include/bits/string3.h:53
#3 0x804927f in process_heartbeet /home/manos/Desktop/AFL/heartbleed.c:55
#4 0x8048de2 in main /home/manos/Desktop/AFL/heartbleed.c:103
#5 0xf7866646 in __libc_start_main (/lib32/libc.so.6+0x18646)
#6 0x8048fbb (/home/manos/Desktop/AFL/test-bins/heartbleed-asan+0x8048fbb)

0xf5c007f9 is located 0 bytes to the right of 9-byte region [0xf5c007f0,0xf5c007f9)
allocated by thread T0 here:
#0 0xf7a98d8e in malloc (/usr/lib32/libasan.so.2+0x96d8e)
#1 0x8048c10 in main /home/manos/Desktop/AFL/heartbleed.c:87
#2 0xf7866646 in __libc_start_main (/lib32/libc.so.6+0x18646)
```