

```

yung@yung-MS-7816:~/Lab-8$ valgrind ./a.out crash_0.bmp ./out.bmp
==2293273== Memcheck, a memory error detector
==2293273== Copyright (c) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==2293273== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==2293273== Command: ./a.out crash_0.bmp ./out.bmp
==2293273==
==2293273== Invalid write of size 1
==2293273== at 0x4842B63: memmove (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==2293273== by 0x48F15DA: _IO_file_xsgetn (fileops.c:1296)
==2293273== by 0x48E5062: fread (iofread.c:38)
==2293273== by 0x109740: bmp_transform (bmp_lib.c:158)
==2293273== by 0x10990E: main (bmpgrayscale.c:18)
==2293273== Address 0x4a552a0 is 0 bytes after a block of size 0 alloc'd
==2293273== at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==2293273== by 0x109670: bmp_transform (bmp_lib.c:132)
==2293273== by 0x10990E: main (bmpgrayscale.c:18)
==2293273==
--2293273-- VALGRIND INTERNAL ERROR: Valgrind received a signal 7 (SIGBUS) - exiting
--2293273-- si_code=128; Faulting address: 0x0; sp: 0x1002ca9e40

valgrind: the 'impossible' happened:
  Killed by fatal signal

host stacktrace:
==2293273== at 0x5805160F: ??? (in /usr/lib/x86_64-linux-gnu/valgrind/memcheck-amd64-linux)
==2293273== by 0x580058F7: ??? (in /usr/lib/x86_64-linux-gnu/valgrind/memcheck-amd64-linux)
==2293273== by 0x580A7204: ??? (in /usr/lib/x86_64-linux-gnu/valgrind/memcheck-amd64-linux)
==2293273== by 0x580F5FD4: ??? (in /usr/lib/x86_64-linux-gnu/valgrind/memcheck-amd64-linux)

sched status:
  running_tid=1

```

```

151 // read all file to data until EOF
152 int idx = 0;
153 while (1)
154 {
155     if (feof(bmpfile))
156         break;
157
158     fread((char *)&img->data[idx], sizeof(char), sizeof(char), bmpfile);
159     idx++;
160 }
161

```

原因：由上兩張圖可看到，透過 valgrind 找出錯誤為 158 行。

下面為解法：

```

151 // read all file to data until EOF
152 int idx = 0;
153 while (1)
154 {
155     // Fix
156     if (idx >= image_size) {
157         break;
158     }
159
160     if (feof(bmpfile))
161         break;
162
163     fread((char *)&img->data[idx], sizeof(char), sizeof(char), bmpfile);
164     idx++;
165 }
166

```

如上圖，加入 156~158 行即可解決錯誤。

推測根本原因應該是 fuzz 把 Input 檔案把 eof 字元改掉了，造成 160 行沒有正確讀到 EOF 產生 Crash

```

yung@yung-MS-1816:~/Lab-8$ ~/aflplusplus/afl-fuzz -i ./test_input -o ./m none -- ./bmgpayscale @@ @@
afl-fuzz++3.13a based on afl by Michal Zalewski and a large online community
[*] afl++ is maintained by Marc "van Hauser" Heuse, Heiko "hexcoder" Eißfeldt, Andrea Fioraldi and Dominik Maier
[*] afl++ is open source, get it at https://github.com/AFLplusplus/AFLplusplus
[*] NOTE: This is v3.x which changes defaults and behaviours - see README.md
[*] No -M/-S set, autoconfiguring for "-S default"
[*] Getting to work...
[*] Using exponential power schedule (FAST)
[*] Enabled testcache with 50 MB
[*] Checking core_pattern...
[*] Checking CPU scaling governor...
[*] You have 8 CPU cores and 3 runnable tasks (utilization: 38%).
[*] Try parallel jobs - see /usr/local/share/doc/afl/parallel_fuzzing.md.
[*] Setting up output directories...

      american fuzzy lop ++3.13a (default) [fast] {}

process timing                               overall results
  run time : 0 days, 3 hrs, 55 min, 11 sec    cycles done : 343
  last new path : 0 days, 3 hrs, 52 min, 48 sec
  last uniq crash : none seen yet             uniq crashes : 0
  last uniq hang : none seen yet              uniq hangs : 0

cycle progress                               map coverage
now processing : 3.926 (15.0%)                 map density : 7.81% / 31.25%
paths timed out : 0 (0.00%)                   count coverage : 2.05 bits/tuple

stage progress                               findings in depth
now trying : havoc                             favored paths : 7 (35.00%)
stage execs : 108/660 (16.36%)                new edges on : 7 (35.00%)
total execs : 16.1M                           total crashes : 0 (0 unique)
exec speed : 1794/sec                         total tmouts : 1 (1 unique)

fuzzing strategy yields                      path geometry
bit flips : disabled (default, enable with -D) levels : 3
byte flips : disabled (default, enable with -D) pending : 4
arithmetics : disabled (default, enable with -D) pend fav : 0
known ints : disabled (default, enable with -D) own finds : 18
dictionary : n/a                             imported : 0
havoc/splice : 18/6.77M, 0/9.32M              stability : 100.00%
py/custom/rq : unused, unused, unused, unused
trim/eff : 0.01%/6877, disabled

[cpu000:100%]
5

```