Heap buffer overflow in libr/bin/format/mach0/mach0.c in radareorg/radare2

0

✓ Valid Reported on Apr 4th 2022

This vulnerability is of type heap-buffer-overflow. And after quick investigation I think it is very likely to be successfully exploited to remote code execution. The bug exists in latest stable release (radare2-5.6.6) and lastest master branch (8317a34b7e4ab731e230dcdd81adc9323c5b518b, updated in April 03, 2022). Specifically, the vulnerable code (located at `libr/bin/format/mach0/mach0.c`) and the bug's basic explanation are highlighted as follows:

```
3177            size_t i;
3178            for (i = 0; i < num; i++) {
3179                    struct relocation_info a_info = info[I];
3180                    ut32 sym_num = a_info.r_symbolnum;
3181                    if (sym_num > bin->nsymtab) {
3182                            continue;
3183                    }
3184
// heap-buffer-overflow here.
3185                    ut32 stridx = bin->symtab[sym_num].n_strx;
3186                    char *sym_name = get_name (bin, stridx, false);
3187                    if (!sym_name) {
3188                            continue;
3189                    }
```

Proof of Concept

Build the radare2 (8317a34b7e4ab731e230dcdd81adc9323c5b518b, updated in April 03, 2022) and run it using the input POC.

```
# build the radare2 with address sanitizer
export CFLAGS=" -fsanitize=address "; export CXXFLAGS=" -fsa
CFGARG=" --enable-shared=no " PREFIX=`realpath install` bash sys/build.sh
```

```
# disable some features of address sanitizer to avoid false positives
export ASAN_OPTIONS=detect_leaks=0:abort_on_error=1:symbolize=0:allocator_m

# trigger the crash
./radare2 -A -q POC_FILE
```

The crash stack is:

```
=================================================================
==25752==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x6060000
READ of size 4 at 0x6060000151e0 thread T0
    #0 0x7ffff29fcb2b  (/src/cmdline-fuzz/exprs/radare2-5.5.4/src/install/l
    #1 0x7ffff29cc2e5  (/src/cmdline-fuzz/exprs/radare2-5.5.4/src/install/l
    #2 0x7ffff26477f9  (/src/cmdline-fuzz/exprs/radare2-5.5.4/src/install/l
    #3 0x7ffff2645004  (/src/cmdline-fuzz/exprs/radare2-5.5.4/src/install/l
    #4 0x7ffff262a1fe  (/src/cmdline-fuzz/exprs/radare2-5.5.4/src/install/l
    #5 0x7ffff25cd9fb  (/src/cmdline-fuzz/exprs/radare2-5.5.4/src/install/l
    #6 0x7ffff25ccad6  (/src/cmdline-fuzz/exprs/radare2-5.5.4/src/install/l
    #7 0x7ffff384136c  (/src/cmdline-fuzz/exprs/radare2-5.5.4/src/install/l
    #8 0x7ffff7548697  (/src/cmdline-fuzz/exprs/radare2-5.5.4/src/install/l
    #9 0x7ffff72bc0b2  (/lib/x86_64-linux-gnu/libc.so.6+0x270b2)
    #10 0x55555557239d  (/src/cmdline-fuzz/exprs/radare2-5.5.4/radare2+0x1e

0x6060000151e0 is located 0 bytes to the right of 64-byte region [0x6060000
allocated by thread T0 here:
    #0 0x5555555ed772  (/src/cmdline-fuzz/exprs/radare2-5.5.4/radare2+0x997
    #1 0x7ffff2a24ab2  (/src/cmdline-fuzz/exprs/radare2-5.5.4/src/install/l
    #2 0x7ffff29d7a58  (/src/cmdline-fuzz/exprs/radare2-5.5.4/src/install/l
    #3 0x7ffff29d9417  (/src/cmdline-fuzz/exprs/radare2-5.5.4/src/install/l

SUMMARY: AddressSanitizer: heap-buffer-overflow (/src/cmdline-fuzz/exprs/ra
Shadow bytes around the buggy address:
  0x0c0c7fffa9e0: 00 00 00 00 00 00 04 fa fa fa fa fa 00 00 00 00
  0x0c0c7fffa9f0: 00 00 00 01 fa fa fa fa 00 00 00 00 00 00 00 01
  0x0c0c7fffaa00: fa fa fa fa 00 00 00 00 00 00 00 06 fa fa fa fa
  0x0c0c7fffaa10: 00 00 00 00 00 00 00 01 fa fa fa fa 00 00 00 00
  0x0c0c7fffaa20: 00 00 00 02 fa fa fa fa 00 00 00 00 00 00
=>0x0c0c7fffaa30: fa fa fa fa 00 00 00 00 00 00 00 00[fa]fa fa fa
```

```
  0x0c0c7fffaa40: fd fd fd fd fd fd fd fa fa fa fa fa fd fd fd fd
  0x0c0c7fffaa50: fd fd fd fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c0c7fffaa60: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa

  0x0c0c7fffaa70: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c0c7fffaa80: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
Shadow byte legend (one shadow byte represents 8 application bytes):
  Addressable:           00
  Partially addressable: 01 02 03 04 05 06 07
  Heap left redzone:       fa
  Freed heap region:       fd
  Stack left redzone:      f1
  Stack mid redzone:       f2
  Stack right redzone:     f3
  Stack after return:      f5
  Stack use after scope:   f8
  Global redzone:          f9
  Global init order:       f6
  Poisoned by user:        f7
  Container overflow:      fc
  Array cookie:            ac
  Intra object redzone:    bb
  ASan internal:           fe
  Left alloca redzone:     ca
  Right alloca redzone:    cb
  Shadow gap:              cc
==25752==ABORTING

Program received signal SIGABRT, Aborted.
0x00007ffff72db18b in raise () from /lib/x86_64-linux-gnu/libc.so.6
(gdb) bt
#0  0x00007ffff72db18b in raise () from /lib/x86_64-linux-gnu/libc.so.6
#1  0x00007ffff72ba859 in abort () from /lib/x86_64-linux-gnu/libc.so.6
#2  0x000055555560ba77 in __sanitizer::Abort() ()
#3  0x0000555555609fa1 in __sanitizer::Die() ()
#4  0x00005555555f14e4 in __asan::ScopedInErrorReport::~ScopedInErrorReport
#5  0x00005555555f30aa in __asan::ReportGenericError(unsigned long, unsigne
#6  0x00005555555f38b8 in __asan_report_load4 ()
#7  0x00007ffff29fcb2c in parse_relocation_info (bin=0x61800
#8  get_relocs_64 (bin=0x618000004880) at /src/cmdline-fuzz
#9  0x00007ffff29cc2e6 in relocs (bf=0x60d000000ad0) at /src/cmdline-fuzz/
```

```
#10 0x00007ffff26477fa in r_bin_object_set_items (bf=<optimized out>, bo=<o
#11 0x00007ffff2645005 in r_bin_object_new (bf=<optimized out>, plugin=<opt
#12 0x00007ffff262a1ff in r_bin_file_new_from_buffer (bin=0x616000000680, f

    pluginname=<optimized out>) at bfile.c:585
#13 0x00007ffff25cd9fc in r_bin_open_buf (bin=<optimized out>, buf=<optimiz
#14 0x00007ffff25ccad7 in r_bin_open_io (bin=0x616000000680, opt=<optimized
#15 0x00007ffff384136d in r_core_file_do_load_for_io_plugin (r=0x7fffec2d38
#16 r_core_bin_load (r=0x7fffec2d3800, filenameuri=<optimized out>, baddr=<
#17 0x00007ffff7548698 in r_main_radare2 (argc=<optimized out>, argv=<optim
#18 0x00007ffff72bc0b3 in __libc_start_main () from /lib/x86_64-linux-gnu/l
#19 0x000055555557239e in _start ()
(gdb) frame 7
#7  0x00007ffff29fcb2c in parse_relocation_info (bin=0x618000004880, relocs
3185                            ut32 stridx = bin->symtab[sym_num].n_strx;
(gdb) p bin->symtab
$2 = (struct nlist_64 *) 0x6060000151a0
(gdb) p bin->symtab[4]
$3 = {n_strx = 3429799609, n_type = 185 '\271', n_sect = 150 '\226', n_desc
(gdb) p &(bin->symtab[4])
$4 = (struct nlist_64 *) 0x6060000151e0
```

Impact

If address sanitizer is disabled during the compiling, the program should executes into the `r_str_ncpy` function. Therefore I think it is very likely to be exploitable. For more general description of heap buffer overflow, see CWE.

References

- POC File

CVE
CVE-2022-1240
(Published)

Vulnerability Type
CWE-122: Heap-based Buffer Overflow

Severity
High (7.6)

High (7.8)

Registry
Other

Affected Version
5.6.6

Visibility
Public

Status
Fixed

Found by
Han0nly
@han0nly
legend

Fixed by

pancake
@trufae
maintainer

This report was seen 711 times.

We are processing your report and will contact the **radareorg/radare2** team within 24 hours.
8 months ago

We have contacted a member of the **radareorg/radare2** team and are waiting to hear back
8 months ago

pancake validated this vulnerability 8 months ago

Han0nly has been awarded the disclosure bounty ✓

The fix bounty is now up for grabs

pancake marked this as fixed in **5.8.6** with commit **ca8d8b** 8 months ago

pancake has been awarded the fix bounty ✓

This vulnerability will not receive a CVE ✗

Sign in to join this conversation

huntr

home

hacktivity

leaderboard

FAQ

contact us

terms

privacy policy

part of 418sec

company

about

team

Chat with us