Access of Memory Location After End of Buffer in radareorg/radare2

0

Valid Reported on Jan 22nd 2022

Description

This vulnerability is of out-of-bound read which accesses the address beyond/past the buffer. The bug exists in latest stable release (radare2-5.5.4) and lastest master branch (ed2030b79e68986bf04f3a6279463ab989fe400f, updated in Jan 22, 2022). Specifically, the vulnerable code and the bug's basic explanation is highlighted as follows:

```
// shlr/java/class.c
R_API RBinJavaAttrInfo *r_bin_java_signature_attr_new(RBinJavaObj *bin, ut8
...
// line 3612
// the buffer[offset] can access memory beyond the buffer's size
// in our poc, it is the case that the second byte of the USHORT is out of
    attr->info.signature_attr.signature_idx = R_BIN_JAVA_USHORT (buffer, of
...
```

Proof of Concept

Build the radare2 (5.5.4 or latest commit ed2030b79e68986bf04f3a6279463ab989fe400f) and run it using the input POC.

```
# build the radare2 with address sanitizer
export CFLAGS=" -fsanitize=address "; export CXXFLAGS=" -fsanitize=address
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:symboli
# trigger the crash
./radare2 -A -q POC_FILE
```

The crash stack is:

```
    #0 0x7ffff56ceba8  (/src/projects/radare2-5.5.4/lastest-radare2/install
    #1 0x7ffff56af317  (/src/projects/radare2-5.5.4/lastest-radare2/install
    #2 0x7ffff569f5a2  (/src/projects/radare2-5.5.4/lastest-radare2/install
    #3 0x7ffff56b3203  (/src/projects/radare2-5.5.4/lastest-radare2/install
    #4 0x7ffff56b5d33  (/src/projects/radare2-5.5.4/lastest-radare2/install
    #5 0x7ffff56c694f  (/src/projects/radare2-5.5.4/lastest-radare2/install
    #6 0x7ffff282d06a  (/src/projects/radare2-5.5.4/lastest-radare2/install
    #7 0x7ffff2597fea  (/src/projects/radare2-5.5.4/lastest-radare2/install
    #8 0x7ffff257df9e  (/src/projects/radare2-5.5.4/lastest-radare2/install
    #9 0x7ffff252179b  (/src/projects/radare2-5.5.4/lastest-radare2/install
    #10 0x7ffff2520876  (/src/projects/radare2-5.5.4/lastest-radare2/instal
    #11 0x7ffff386facc  (/src/projects/radare2-5.5.4/lastest-radare2/instal
    #12 0x7ffff76312ae  (/src/projects/radare2-5.5.4/lastest-radare2/instal
    #13 0x7ffff73a50b2  (/lib/x86_64-linux-gnu/libc.so.6+0x270b2)
    #14 0x55555557239d  (/src/projects/radare2-5.5.4/lastest-radare2/instal

0x602000067cb7 is located 0 bytes to the right of 7-byte region [0x60200006
allocated by thread T0 here:
    #0 0x5555555ed772  (/src/projects/radare2-5.5.4/lastest-radare2/install
    #1 0x7ffff569f50f  (/src/projects/radare2-5.5.4/lastest-radare2/install

SUMMARY: AddressSanitizer: heap-buffer-overflow (/src/projects/radare2-5.5.
Shadow bytes around the buggy address:
  0x0c0480004f40: fa fa fd fa fa fa 05 fa fa fa 05 fa fa fa 01 fa
  0x0c0480004f50: fa fa 05 fa fa fa 01 fa fa fa 05 fa fa fa 01 fa
  0x0c0480004f60: fa fa 05 fa fa fa 01 fa fa fa 05 fa fa fa 01 fa
  0x0c0480004f70: fa fa 05 fa fa fa 01 fa fa fa 05 fa fa fa 01 fa
  0x0c0480004f80: fa fa 05 fa fa fa 01 fa fa fa 05 fa fa fa 00 fa
=>0x0c0480004f90: fa fa 05 fa fa fa[07]fa fa fa fa fa fa fa fa fa
  0x0c0480004fa0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c0480004fb0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c0480004fc0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c0480004fd0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c0480004fe0: 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
==17996==ABORTING

Program received signal SIGABRT, Aborted.
0x00007ffff73c418b in raise () from /lib/x86_64-linux-gnu/libc.so.6
(gdb) bt
#0  0x00007ffff73c418b in raise () from /lib/x86_64-linux-gnu/libc.so.6
#1  0x00007ffff73a3859 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  0x00005555555f3798 in __asan_report_load1 ()
#7  0x00007ffff56ceba9 in r_bin_java_signature_attr_new (bin=<optimized out
#8  0x00007ffff56af318 in r_bin_java_read_next_attr_from_buffer (bin=<optim
#9  0x00007ffff569f5a3 in r_bin_java_read_next_attr (bin=<optimized out>, c
#10 0x00007ffff56b3204 in r_bin_java_parse_attrs (bin=<optimized out>, offs
#11 0x00007ffff56b5d34 in r_bin_java_load_bin (bin=0x614000001240, buf=<opt
#12 0x00007ffff56b5954 in r_bin_java_new_bin (bin=<optimized out>, loadaddr
#13 0x00007ffff56c6950 in r_bin_java_new_buf (buf=<optimized
#14 0x00007ffff282d06b in load_buffer (bf=<optimized out>,
    at /src/projects/radare2-5.5.4/lastest-radare2/libr/../libr/bin/p/bin_
```

```
#15 0x00007ffff2597feb in r_bin_object_new (bf=0x60d000000bc0, plugin=<opti
    sz=<optimized out>) at bobj.c:147
#16 0x00007ffff257df9f in r_bin_file_new_from_buffer (bin=0x616000000980, f

    loadaddr=<optimized out>, fd=<optimized out>, pluginname=<optimized out
#17 0x00007ffff252179c in r_bin_open_buf (bin=<optimized out>, buf=<optimiz
#18 0x00007ffff2520877 in r_bin_open_io (bin=0x616000000980, opt=<optimized
#19 0x00007ffff386facd in r_core_file_do_load_for_io_plugin (r=0x7fffec3328
#20 r_core_bin_load (r=0x7fffec332800, filenameuri=<optimized out>, baddr=<
#21 0x00007ffff76312af in r_main_radare2 (argc=<optimized out>, argv=<optim
#22 0x00007ffff73a50b3 in __libc_start_main () from /lib/x86_64-linux-gnu/l
#23 0x000055555557239e in _start ()
```

Impact

The bug causes the program reads data past the end of the intented buffer. Typically, this can allow attackers to read sensitive information from other memory locations or cause a crash. More details see CWE-125: Out-of-bounds read.

References

- PoC file

CVE
CVE-2022-0521
(Published)

Vulnerability Type
CWE-788: Access of Memory Location After End of Buffer

Severity
Medium (6.3)

Visibility
Public

Status
Fixed

Found by

Cen Zhang
@occia

unranked

Fixed by

pancake

@trufae

maintainer

This report was seen 443 times.

We are processing your report and will contact the **radareorg/radare2** team within 24 hours. 10 months ago

Cen Zhang modified the report 10 months ago

We have contacted a member of the **radareorg/radare2** team and are waiting to hear back 10 months ago

We have sent a follow up to the **radareorg/radare2** team. We will try again in 7 days. 10 months ago

We have sent a second follow up to the **radareorg/radare2** team. We will try again in 10 days. 10 months ago

pancake validated this vulnerability 10 months ago

Cen Zhang has been awarded the disclosure bounty ✔

The fix bounty is now up for grabs

pancake 10 months ago Maintainer

Fixed in https://github.com/radareorg/radare2/pull/19667/commits/aef756dcaec364a54a0f58407e4fd874fdb47d5e

pancake marked this as fixed in **5.6.2** with commit **6c4428** 10 months ago

pancake has been awarded the fix bounty ✔

This vulnerability will not receive a CVE ✖

pancake 10 months ago Maintainer

As long as huntr seems to be a friendly and clean way to report and track vulnerabilities to opensource projects. it would be good to have a way to also automate the check for the fix. I am adding all those samples to the fuzz suite of r2, so my CI is verifying that no future commits will reintroduce those bugs, but when there are lot of tickets to track it's sometimes hard to match the right commit (sometimes one commit fixes multiple crash files).

Clusterfuzz already verifies from time to time that their samples arent regressing and detect when the fix has been commited upstream. Sumarizing my thoughs:

huntr should recommend/provide a reference ID (can we we the boundy id/url) for the commits

how to automate checking if the sample has been fixed and in which commit

is there a way to assign the CVE before the commit hits master? that would force me to not merge fixes in master, unless we have a different ID to associate the commit with the bounty, the reproducer sample

Thanks for such an awesome bugtracking/bounty platform!

Cen Zhang 10 months ago Researcher

lol, really great and practical feedback from developer's perspective for improving huntr @admin

Jamie Slome 10 months ago Admin

Amazing feedback @pancake - we really value your honesty and depth into your workflow and how our platform can help serve you better ♥

I'd love to invite you to create a feature request ticket on our public roadmap HERE.

We can give you better responses on your ideas and also keep you updated there with any progress we have made in helping to achieve your visions! 🥰

Let me know if you would not like to create the issue, and I will do it on your behalf!

pancake 10 months ago Maintainer

Thanks will do

Sign in to join this conversation