huntr

Use of Uninitialized Function Pointer in radareorg/radare2

0

✓ Valid Reported on May 21st 2022

Description

When providing a crafted input binary to radare2, the context->read_addr function pointer is never initialized before use. This is due to the switch statement responsible for the assignment not finding a matching value for its switch cases.

Calling function

```
static bool vtable_is_value_in_text_section(RVTableContext *context, ut64 c
    //value at the current address
    ut64 curAddressValue;
    if (!context->read_addr (context->anal, curAddress, &curAddressValue))
        return false;
    }
    //if the value is in text section
    bool ret = vtable_addr_in_text_section (context, curAddressValue);
    if (value) {
        *value = curAddressValue;
    }
    return ret;
}
```

Setting Logic

```
R_API bool r_anal_vtable_begin(RAnal *anal, RVTableContext *context) {
    context->anal = anal;
    context->abi = anal->cxxabi;
    context->word_size = (ut8) (anal->config->bits / 8);
```

Chat with us

```
    context->word_size = (ut8) (anal->config->bits / 8);
    const bool is_arm = anal->cur->arch && r_str_startswith (anal->cur->arc
    if (is_arm && context->word_size < 4) {
        context->word_size = 4;
    }
    const bool be = anal->config->big_endian;
    switch (context->word_size) { //*** THIS IS THE CAUSE ***
    case 1:
        context->read_addr = be? vtable_read_addr_be8 : vtable_read_addr_le
        break;
    case 2:
        context->read_addr = be? vtable_read_addr_be16 : vtable_read_addr_l
        break;
    case 4:
        context->read_addr = be? vtable_read_addr_be32 : vtable_read_addr_l
        break;
    case 8:
        context->read_addr = be? vtable_read_addr_be64 : vtable_read_addr_l
        break;
    default:
        return false;
    }
    return true;
}
```

```
#define VTABLE_READ_ADDR_FUNC(fname, read_fname, sz) \
    static bool fname(RAnal *anal, ut64 addr, ut64 *buf) {\
        ut8 tmp[sz];\
        if (!anal->iob.read_at (anal->iob.io, addr, tmp, sz)) {\
            return false;\
        }\
        *buf = read_fname (tmp);\
        return true;\
    }
VTABLE_READ_ADDR_FUNC (vtable_read_addr_le8, r_read_le8, 1)
VTABLE_READ_ADDR_FUNC (vtable_read_addr_le16, r_read_le16, 2)
VTABLE_READ_ADDR_FUNC (vtable_read_addr_le32, r_read_le32,
VTABLE_READ_ADDR_FUNC (vtable_read_addr_le64, r_read_le64,
VTABLE_READ_ADDR_FUNC (vtable_read_addr_be8, r_read_be8, 1)
VTABLE_READ_ADDR_FUNC (vtable_read_addr_be16, r_read_be16, 2)
VTABLE_READ_ADDR_FUNC (vtable_read_addr_be32, r_read_be32, 4)
VTABLE_READ_ADDR_FUNC (vtable_read_addr_be64, r_read_be64, 8)
```

Backtrace

```
#0  0x00007ffff08ff800 in ?? ()
#1  0x00007ffff461e745 in vtable_is_value_in_text_section (context=context@
    curAddress=curAddress@entry=1616928864, value=value@entry=0x0) at vtabl
#2  0x00007ffff461e934 in vtable_is_addr_vtable_start_msvc (context=context
    curAddress=curAddress@entry=1616928864) at vtable.c:141
#3  0x00007ffff462008c in vtable_is_addr_vtable_start (section=<optimized c
    at vtable.c:175
#4  r_anal_vtable_search (context=context@entry=0x7fffffffd500) at vtable.c
#5  0x00007ffff4621cc0 in r_anal_rtti_recover_all (anal=<optimized out>) at
#6  0x00007ffff5829bee in cmd_anal_rtti (input=<optimized out>, core=0x7fff
#7  cmd_anal_virtual_functions (input=<optimized out>, core=0x7ffff119d800)
#8  cmd_anal (data=0x7ffff119d800, input=<optimized out>) at cmd_anal.c:124
#9  0x00007ffff5950bca in r_cmd_call (cmd=0x620000000080, input=input@entry
#10 0x00007ffff5771229 in r_core_cmd_subst_i (tmpseek=<optimized out>, colc
    core=<optimized out>) at cmd.c:4528
#11 r_core_cmd_subst (core=core@entry=0x7ffff119d800, cmd=<optimized out>,
#12 0x00007ffff577781f in run_cmd_depth (cmd=<optimized out>, core=0x7ffff1
#13 r_core_cmd (core=core@entry=0x7ffff119d800, cstr=<optimized out>, cstr@
    at cmd.c:5513
#14 0x00007ffff5791087 in r_core_cmd0 (core=core@entry=0x7ffff119d800, cmd=
#15 0x00007ffff57df095 in cmd_anal_all (core=core@entry=0x7ffff119d800, inp
    at cmd_anal.c:11302
#16 0x00007ffff5828df8 in cmd_anal (data=0x7ffff119d800, input=0x602000366
#17 0x00007ffff5950bca in r_cmd_call (cmd=0x620000000080, input=input@entry
#18 0x00007ffff5771229 in r_core_cmd_subst_i (tmpseek=<optimized out>, colc
    core=<optimized out>) at cmd.c:4528
#19 r_core_cmd_subst (core=core@entry=0x7ffff119d800, cmd=<optimized out>,
#20 0x00007ffff577781f in run_cmd_depth (cmd=<optimized out>, core=0x7ffff1
#21 r_core_cmd (core=core@entry=0x7ffff119d800, cstr=<optimized out>, cstr@
    at cmd.c:5513
#22 0x00007ffff5791087 in r_core_cmd0 (core=core@entry=0x7ffff119d800, cmd=
#23 0x00007ffff3ab1ef4 in r_main_radare2 (argc=<optimized o
#24 0x00007ffff38cf09b in __libc_start_main (main=0x5555555
    fini=<optimized out>, rtld_fini=<optimized out>, stack_end=0x7fffffffde
```

Chat with us

```
#25 0x0000555555556afa in _start ()
```

ASAN

```
AddressSanitizer:DEADLYSIGNAL
=================================================================
==100644==ERROR: AddressSanitizer: SEGV on unknown address 0x7f4f322ff800 (
==100644==The signal is caused by a READ memory access.
==100644==Hint: PC is at a non-executable region. Maybe a wild jump?
    #0 0x7f4f322ff7ff  (<unknown module>)

AddressSanitizer can not provide additional info.
SUMMARY: AddressSanitizer: SEGV (<unknown module>)
==100644==ABORTING
```

Valgrind

```
==58932== Conditional jump or move depends on uninitialised value(s)
==58932==    at 0x716ECB9: r_anal_get_reg_profile (anal.c:250)
==58932==    by 0x716EE8F: r_anal_set_reg_profile (anal.c:262)
==58932==    by 0x716F2FC: r_anal_set_bits (anal.c:325)
==58932==    by 0x5FC31AA: r_core_init (core.c:3057)
==58932==    by 0x5FC3EB6: r_core_new (core.c:919)
==58932==    by 0x7FBF79A: r_main_radare2 (radare2.c:489)
==58932==    by 0x800509A: (below main) (libc-start.c:308)
==58932==
==58932== Conditional jump or move depends on uninitialised value(s)
==58932==    at 0x716EEDE: r_anal_set_reg_profile (anal.c:258)
==58932==    by 0x716F2FC: r_anal_set_bits (anal.c:325)
==58932==    by 0x5FC31AA: r_core_init (core.c:3057)
==58932==    by 0x5FC3EB6: r_core_new (core.c:919)
==58932==    by 0x7FBF79A: r_main_radare2 (radare2.c:489)
==58932==    by 0x800509A: (below main) (libc-start.c:308)
==58932==
==58932== Conditional jump or move depends on uninitialised
==58932==    at 0x716F2A9: r_anal_set_bits (anal.c:324)
```

Chat with us

```
==58932==    by 0x5FC31AA: r_core_init (core.c:3057)
==58932==    by 0x5FC3EB6: r_core_new (core.c:919)
==58932==    by 0x7FBF79A: r_main_radare2 (radare2.c:489)

==58932==    by 0x800509A: (below main) (libc-start.c:308)
==58932==
==58932==
==58932== More than 1000 different errors detected.  I'm not reporting any
==58932== Final error counts will be inaccurate.  Go fix your program!
==58932== Rerun with --error-limit=no to disable this cutoff.  Note
==58932== that errors may occur in your program without prior warning from
==58932== Valgrind, because errors are no longer being displayed.
```

Proof of Concept

```
radare2 -AA -q minified_crash
```

https://github.com/GreaterGoodest/pocs/blob/master/minified_crash

Impact

DoS, with potential for code execution or read/write if the random address pointed to by the function pointer can be coerced to point at something useful.

CVE
CVE-2022-1809
(Published)

Vulnerability Type
CWE-824: Access of Uninitialized Pointer

Severity
High (7.4)

Registry
Other

Affected Version
5.6.9

Visibility
Public

Chat with us

Status
Fixed

Found by

Ryan Good
@greatergoodest
legend

Fixed by

pancake
@trufae
maintainer

This report was seen 701 times.

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

Ryan Good 6 months ago Researcher

I feel like this is more severe than a typical untrusted pointer dereference, as it's a function pointer. Not sure what to call it though...

Ryan Good modified the report 6 months ago

Ryan Good modified the report 6 months ago

Ryan Good modified the report 6 months ago

Ryan Good modified the report 6 months ago

Ryan Good 6 months ago Researcher

After further analysis, this appears to be the use of an uninitialized pointer, as th
statement does not find a match.

Chat with us

Ryan Good modified the report 6 months ago

pancake validated this vulnerability 6 months ago

Ryan Good has been awarded the disclosure bounty ✓

The fix bounty is now up for grabs

The researcher's credibility has increased: +7

pancake marked this as fixed in **5.7.0** with commit **919e3a** 6 months ago

pancake has been awarded the fix bounty ✓

This vulnerability will not receive a CVE ✗

ajakk 6 months ago

```
$ curl -I https://github.com/GreaterGoodest/pocs/blob/master/minified_crash
HTTP/2 404
```

Where did the reproducer go?

Ryan Good 6 months ago Researcher

I renamed it to patched_minified_crash

Sign in to join this conversation

Chat with us