SEGV in function dwarf::cursor::skip_form at dwarf/cursor.cc:181 #47

Open xiaoxiongwang opened this issue on Aug 15, 2020 · 1 comment

xiaoxiongwang commented on Aug 15, 2020 • edited

Tested in Ubuntu 16.04, 64bit.

The tested program is the example program dump-lines.

The testcase is dump_line_segv2.

I use the following command:

```
/path-to-libelfin/examples/dump-lines dump_line_segv2
```

and got:

```
Segmentation fault (core dumped)
```

I use **valgrind** to analysis the bug and get the below information (absolute path information omitted):

```
valgrind /path-to-libelfin/examples/dump-lines dump_line_segv2
==11807== Memcheck, a memory error detector
==11807== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.
==11807== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info
==11807== Command: /path-to-libelfin/examples/dump-lines dump_line_segv2
==11807==
==11807== Invalid read of size 1
==11807==    at 0x42A39C: dwarf::cursor::skip_form(dwarf::DW_FORM) (cursor.cc:181)
==11807==    by 0x431FAC: dwarf::die::read(unsigned long) (die.cc:51)
==11807==    by 0x412EFC: dwarf::unit::root() const (dwarf.cc:195)
==11807==    by 0x413177: dwarf::compilation_unit::get_line_table() const (dwarf.cc:291)
==11807==    by 0x402CB7: main (dump-lines.cc:41)
==11807==  Address 0x10cd80af is not stack'd, malloc'd or (recently) free'd
==11807==
==11807==
==11807== Process terminating with default action of signal 11 (SIGSEGV)
==11807==  Access not within mapped region at address 0x10CD80AF
==11807==    at 0x42A39C: dwarf::cursor::skip_form(dwarf::DW_FORM) (cursor.cc:181)
==11807==    by 0x431FAC: dwarf::die::read(unsigned long) (die.cc:51)
==11807==    by 0x412EFC: dwarf::unit::root() const (dwarf.cc:195)
==11807==    by 0x413177: dwarf::compilation_unit::get_line_table() const (dwarf.cc:291)
==11807==    by 0x402CB7: main (dump-lines.cc:41)
==11807==  If you believe this happened as a result of a stack
==11807==  overflow in your program's main thread (unlikely but
==11807==  possible), you can try to increase the size of the
==11807==  main thread stack using the --main-stacksize= flag.
==11807==  The main thread stack size used in this run was 8388608.
--- <0>
==11807==
==11807== HEAP SUMMARY:
==11807==     in use at exit: 80,832 bytes in 64 blocks
==11807==   total heap usage: 122 allocs, 58 frees, 88,640 bytes allocated
==11807==
==11807== LEAK SUMMARY:
==11807==    definitely lost: 0 bytes in 0 blocks
==11807==    indirectly lost: 0 bytes in 0 blocks
==11807==      possibly lost: 0 bytes in 0 blocks
==11807==    still reachable: 80,832 bytes in 64 blocks
==11807==         suppressed: 0 bytes in 0 blocks
==11807== Rerun with --leak-check=full to see details of leaked memory
==11807==
==11807== For counts of detected and suppressed errors, rerun with: -v
==11807== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)
Segmentation fault (core dumped)
```

I use **AddressSanitizer** to build ffjpeg and running it with the following command:

```
/path-to-libelfin/examples/dump-lines dump_line_segv2
```

This is the ASAN information (absolute path information omitted):

```
/path-to-libelfin-address/examples/dump-lines dump_line_segv2
ASAN:SIGSEGV
=================================================================
==11879==ERROR: AddressSanitizer: SEGV on unknown address 0x7fc5c7ec50af (pc 0x000000416720 bp 0x7fffc67fa700 sp 0x7fffc67fa600 T0)
    #0 0x41671f in dwarf::cursor::skip_form(dwarf::DW_FORM) /path-to-libelfin-address/dwarf/cursor.cc:181
    #1 0x418023 in dwarf::die::read(unsigned long) /path-to-libelfin-address/dwarf/die.cc:51
    #2 0x40f158 in dwarf::unit::root() const /path-to-libelfin-address/dwarf/dwarf.cc:195
    #3 0x40f41e in dwarf::compilation_unit::get_line_table() const /path-to-libelfin-address/dwarf/dwarf.cc:291
    #4 0x403356 in main /path-to-libelfin-address/examples/dump-lines.cc:41
    #5 0x7fc5b96f682f in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x2082f)
    #6 0x403888 in _start (/path-to-libelfin-address/examples/dump-lines+0x403888)

AddressSanitizer can not provide additional info.
SUMMARY: AddressSanitizer: SEGV /path-to-libelfin-address/dwarf/cursor.cc:181 dwarf::cursor::skip_form(dwarf::DW_FORM)
==11879==ABORTING
```

An attacker can exploit this vulnerability by submitting a malicious elf file that exploits this bug which will result in a Denial of Service (DoS).

👍 1

fgeek commented on Aug 6, 2021

CVE-2020-24827 has been assigned for this issue.

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

2 participants