SEGV in function line_table::line_table at dwarf/line.cc:104 #46

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_segv.

I use the following command:

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

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_segv
==4796== Memcheck, a memory error detector
==4796== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.
==4796== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info
==4796== Command: /path-to-libelfin/examples/dump-lines dump_line_segv
==4796==
==4796== Invalid write of size 1
==4796==    at 0x47DA88: dwarf::line_table::line_table(std::shared_ptr<dwarf::section> const&, unsigned long, unsigned int, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&) (line.cc:104)
==4796==    by 0x413558: dwarf::compilation_unit::get_line_table() const (dwarf.cc:304)
==4796==    by 0x402CB7: main (dump-lines.cc:41)
==4796==  Address 0x0 is not stack'd, malloc'd or (recently) free'd
==4796==
==4796==
==4796== Process terminating with default action of signal 11 (SIGSEGV)
==4796==  Access not within mapped region at address 0x0
==4796==    at 0x47DA88: dwarf::line_table::line_table(std::shared_ptr<dwarf::section> const&, unsigned long, unsigned int, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&) (line.cc:104)
==4796==    by 0x413558: dwarf::compilation_unit::get_line_table() const (dwarf.cc:304)
==4796==    by 0x402CB7: main (dump-lines.cc:41)
==4796==  If you believe this happened as a result of a stack
==4796==  overflow in your program's main thread (unlikely but
==4796==  possible), you can try to increase the size of the
==4796==  main thread stack using the --main-stacksize= flag.
==4796==  The main thread stack size used in this run was 8388608.
--- <0>
==4796==
==4796== HEAP SUMMARY:
==4796==     in use at exit: 81,475 bytes in 72 blocks
==4796==   total heap usage: 132 allocs, 60 frees, 89,399 bytes allocated
==4796==
==4796== LEAK SUMMARY:
==4796==    definitely lost: 0 bytes in 0 blocks
==4796==    indirectly lost: 0 bytes in 0 blocks
==4796==      possibly lost: 0 bytes in 0 blocks
==4796==    still reachable: 81,475 bytes in 72 blocks
==4796==         suppressed: 0 bytes in 0 blocks
==4796== Rerun with --leak-check=full to see details of leaked memory
==4796==
==4796== For counts of detected and suppressed errors, rerun with: -v
==4796== 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_segv
```

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

```
/path-to-libelfin-address/examples/dump-lines dump_line_segv
ASAN:SIGSEGV
=================================================================
==4850==ERROR: AddressSanitizer: SEGV on unknown address 0x000000000000 (pc 0x00000043b335 bp 0x7fff42876e40 sp 0x7fff428769e0 T0)
    #0 0x43b334 in dwarf::line_table::line_table(std::shared_ptr<dwarf::section> const&, unsigned long, unsigned int, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&) /path-to-libelfin-address/dwarf/line.cc:104
    #1 0x40f67b in dwarf::compilation_unit::get_line_table() const /path-to-libelfin-address/dwarf/dwarf.cc:304
    #2 0x403356 in main /path-to-libelfin-address/examples/dump-lines.cc:41
    #3 0x7f82f990682f in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x2082f)
    #4 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/line.cc:104 dwarf::line_table::line_table(std::shared_ptr<dwarf::section> const&, unsigned long, unsigned int,
    std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&)
    ==4850==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-2020-24825 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