

[New issue](#)[Jump to bottom](#)

heap overflow in dwarf_global_formref_b #119

✓ Closed sleicasper opened this issue on Jun 15 · 1 comment

sleicasper commented on Jun 15

asan output:

```
=====
==3946410==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x61b000000697 at pc
0x0000005d9f2d bp 0x7fffffff94f0 sp 0x7fffffff94e8
READ of size 8 at 0x61b000000697 thread T0
#0 0x5d9f2c in dwarf_global_formref_b /libdwarf/SRC/src/lib/libdwarf/dwarf_form.c:840:9
#1 0x5d87b4 in dwarf_global_formref /libdwarf/SRC/src/lib/libdwarf/dwarf_form.c:671:11
#2 0x54c86e in dd_get_integer_and_name /libdwarf/SRC/src/bin/dwarfdump/print_die.c:2400:20
#3 0x546d08 in print_attribute /libdwarf/SRC/src/bin/dwarfdump/print_die.c:3981:15
#4 0x54464a in print_one_die /libdwarf/SRC/src/bin/dwarfdump/print_die.c:2290:25
#5 0x55a11d in print_die_and_children_internal
/libdwarf/SRC/src/bin/dwarfdump/print_die.c:1632:21
#6 0x540f23 in print_die_and_children /libdwarf/SRC/src/bin/dwarfdump/print_die.c:954:11
#7 0x540f23 in print_one_die_section /libdwarf/SRC/src/bin/dwarfdump/print_die.c:1283:28
#8 0x540f23 in print_infos /libdwarf/SRC/src/bin/dwarfdump/print_die.c:648:12
#9 0x50b925 in process_one_file /libdwarf/SRC/src/bin/dwarfdump/dwarfdump.c:1001:15
#10 0x5099ec in main /libdwarf/SRC/src/bin/dwarfdump/dwarfdump.c:503:9
#11 0x7ffff7bfb082 in __libc_start_main /build/glibc-SzIz7B/glibc-2.31/csu/../csu/libc-
start.c:308:16
#12 0x42848d in _start (/libdwarf/fuzzrun/dwarfdump+0x42848d)

0x61b000000697 is located 5 bytes to the right of 1554-byte region [0x61b000000080,0x61b000000692)
allocated by thread T0 here:
#0 0x4cd59f in malloc /fuzz/fuzzdeps/llvm-project-11.0.0/compiler-
rt/lib/asan/asan_malloc_linux.cpp:145:3
#1 0x6a3554 in elf_load_nolibelf_section /libdwarf/SRC/src/lib/libdwarf/dwarf_elfread.c:229:26

SUMMARY: AddressSanitizer: heap-buffer-overflow /libdwarf/SRC/src/lib/libdwarf/dwarf_form.c:840:9
in dwarf_global_formref_b
Shadow bytes around the buggy address:
 0x0c367fff8080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0x0c367fff8090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0x0c367fff80a0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0x0c367fff80b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 0x0c367fff80c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x0c367fff80d0: 00 00[02]fa fa fa fa fa fa fa fa fa fa fa fa fa fa
```

```
0x0c367fff80e0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
0x0c367fff80f0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
0x0c367fff8100: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
0x0c367fff8110: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
0x0c367fff8120: 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
==3946410==ABORTING
```

poc:
[poc.zip](#)

repro:

```
dwarfdump -vv -a ./poc
```

davea42 commented on Jun 15

Owner

The bug was code failing to check if reading a DW_FORM_ref_sig8 would run off the end of the section being read.

This was a long-standing bug introduced with DWARF4 support.

Given dwarf vulnerability id: DW202206-001

The fix committed moments ago.



davea42 closed this as completed on Jun 15

✓

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

2 participants

