Xfig Tickets

Xfig is a diagramming tool
Brought to you by: tklxfiguser

#59 stack-buffer-overflow in read_textobject() function

Milestone: xfig **Status:** closed **Owner:** nobody **Labels:** None
Updated: 2020-12-21 **Created:** 2019-12-12 **Creator:** Suhwan Song **Private:** No

Hi,
I found a stack-buffer-overflow in read_textobject() function at read.c:1378
Please run following command to reproduce it,

```
fig2dev -L box $PoC
```

Here's log

```
==13709==ERROR: AddressSanitizer: stack-buffer-overflow on address 0x7ffe53f28e0e at pc 0x7
READ of size 1 at 0x7ffe53f28e0e thread T0
    #0 0x7f9abe98da68  (/usr/lib/x86_64-linux-gnu/libasan.so.4+0x5aa68)
    #1 0x558cfdd2cc46 in read_textobject fig2dev-3.2.7b/fig2dev/read.c:1378
    #2 0x558cfdd25d9d in read_compoundobject fig2dev-3.2.7b/fig2dev/read.c:711
    #3 0x558cfdd235f0 in read_objects fig2dev-3.2.7b/fig2dev/read.c:440
    #4 0x558cfdd221d3 in readfp_fig fig2dev-3.2.7b/fig2dev/read.c:172
    #5 0x558cfdd220a9 in read_fig fig2dev-3.2.7b/fig2dev/read.c:142
    #6 0x558cfdd19ef3 in main fig2dev-3.2.7b/fig2dev/fig2dev.c:422
    #7 0x7f9abdd81b96 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x21b96)
    #8 0x558cfdd0a979 in _start (fig2dev-3.2.7b+0x6e979)

Address 0x7ffe53f28e0e is located in stack of thread T0 at offset 8446 in frame
    #0 0x558cfdd2c280 in read_textobject fig2dev-3.2.7b/fig2dev/read.c:1304

  This frame has 5 object(s):
    [32, 36) 'num'
    [96, 104) 't'
    [160, 162) 'junk'
    [224, 8416) 's'
    [8448, 16640) 's_temp' <== Memory access at offset 8446 underflows this variable
HINT: this may be a false positive if your program uses some custom stack unwind mechanism (
      (longjmp and C++ exceptions *are* supported)
SUMMARY: AddressSanitizer: stack-buffer-overflow (/usr/lib/x86_64-linux-gnu/libasan.so.4+0x
Shadow bytes around the buggy address:
  0x10004a7dd170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10004a7dd180: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10004a7dd190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10004a7dd1a0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10004a7dd1b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 f2 f2
=>0x10004a7dd1c0: f2[f2]00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10004a7dd1d0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10004a7dd1e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10004a7dd1f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10004a7dd200: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10004a7dd210: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
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
==13709==ABORTING
```

fig2dev Version 3.2.7b
I also tested this in git master and can reproduce it.

1 Attachments

id:000004,sig:06,src:000000,op:havoc,rep:4

Discussion

tkl - *2020-01-06*

- **status**: open --> pending

tkl - *2020-01-06*

Fixed with commit [41b9bb].

Related

Commit: [41b9bb]

tkl - *2020-12-21*

- **status**: pending --> closed
- **xfig / fig2dev**: fig2dev --> xfig

Log in to post a comment.