huntr

Out-of-bound read in function msg_outtrans_attr in vim/vim

0

✓ Valid Reported on Jun 23rd 2022

Description

Out-of-bound read in function `msg_outtrans_attr` at message.c:1551

Version

```
commit 8eba2bd291b347e3008aa9e565652d51ad638cfa (HEAD -> master, tag: v8.2.
```

Proof of Concept

```
./vim/src/vim -u NONE -i NONE -n -m -X -Z -e -s -S poc_vim01 -c :qa!
=================================================================
==21232==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x6190000
READ of size 620 at 0x619000000981 thread T0
    #0 0x7faf43c4f66d  (/usr/lib/x86_64-linux-gnu/libasan.so.4+0x5166d)
    #1 0x562f077dacc5 in msg_outtrans_attr /home/guest/trung/vim/src/messag
    #2 0x562f077d4777 in msg_attr_keep /home/guest/trung/vim/src/message.c:
    #3 0x562f077d4550 in msg_attr /home/guest/trung/vim/src/message.c:123
    #4 0x562f077d7aec in msg_trunc_attr /home/guest/trung/vim/src/message.c
    #5 0x562f072d8239 in process_next_cpt_value /home/guest/trung/vim/src/i
    #6 0x562f072daed2 in ins_compl_get_exp /home/guest/trung/vim/src/insexp
    #7 0x562f072dbba5 in find_next_completion_match /home/guest/trung/vim/s
    #8 0x562f072dbf6e in ins_compl_next /home/guest/trung/vim/src/insexpand
    #9 0x562f072df01d in ins_complete /home/guest/trung/vim/src/insexpand.c
    #10 0x562f0713b591 in edit /home/guest/trung/vim/src/edi
    #11 0x562f073ac613 in invoke_edit /home/guest/trung/vim
    #12 0x562f073a7135 in n_opencmd /home/guest/trung/vim/src/normal.c:6279
```

Chat with us

```
    #13 0x562f073aeb89 in nv_open /home/guest/trung/vim/src/normal.c:7416
    #14 0x562f073843de in normal_cmd /home/guest/trung/vim/src/normal.c:939
    #15 0x562f0720b759 in exec_normal /home/guest/trung/vim/src/ex_docmd.c:

    #16 0x562f0720b535 in exec_normal_cmd /home/guest/trung/vim/src/ex_docm
    #17 0x562f0720add6 in ex_normal /home/guest/trung/vim/src/ex_docmd.c:86
    #18 0x562f071e76d6 in do_one_cmd /home/guest/trung/vim/src/ex_docmd.c:2
    #19 0x562f071dea5c in do_cmdline /home/guest/trung/vim/src/ex_docmd.c:9
    #20 0x562f074ff9cb in do_source_ext /home/guest/trung/vim/src/scriptfil
    #21 0x562f07500b0d in do_source /home/guest/trung/vim/src/scriptfile.c:
    #22 0x562f074fd646 in cmd_source /home/guest/trung/vim/src/scriptfile.c
    #23 0x562f074fd6a7 in ex_source /home/guest/trung/vim/src/scriptfile.c:
    #24 0x562f071e76d6 in do_one_cmd /home/guest/trung/vim/src/ex_docmd.c:2
    #25 0x562f071dea5c in do_cmdline /home/guest/trung/vim/src/ex_docmd.c:9
    #26 0x562f071dce5b in do_cmdline_cmd /home/guest/trung/vim/src/ex_docmd
    #27 0x562f077cda63 in exe_commands /home/guest/trung/vim/src/main.c:313
    #28 0x562f077c6c58 in vim_main2 /home/guest/trung/vim/src/main.c:780
    #29 0x562f077c6514 in main /home/guest/trung/vim/src/main.c:432
    #30 0x7faf42e5ac86 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.
    #31 0x562f07068a99 in _start (/home/guest/trung/vim/src/vim+0x137a99)

0x619000000981 is located 0 bytes to the right of 1025-byte region [0x61900
allocated by thread T0 here:
    #0 0x7faf43cdcb40 in __interceptor_malloc (/usr/lib/x86_64-linux-gnu/li
    #1 0x562f07068edf in lalloc /home/guest/trung/vim/src/alloc.c:246
    #2 0x562f07068ce4 in alloc /home/guest/trung/vim/src/alloc.c:151
    #3 0x562f077c6e7c in common_init /home/guest/trung/vim/src/main.c:914
    #4 0x562f077c6222 in main /home/guest/trung/vim/src/main.c:185
    #5 0x7faf42e5ac86 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6

SUMMARY: AddressSanitizer: heap-buffer-overflow (/usr/lib/x86_64-linux-gnu/
Shadow bytes around the buggy address:
  0x0c327fff80e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c327fff80f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c327fff8100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c327fff8110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c327fff8120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x0c327fff8130:[01]fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c327fff8140: fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c327fff8150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c327fff8160: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c327fff8170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Chat with us

```
  0x0c327fff8170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c327fff8180: 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
==21232==ABORTING
```

Attachment

poc_vim01

Impact

This may result in corruption of sensitive information, a crash, or code execution among other things.

CVE
CVE-2022-2206
(Published)

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

Chat with us

Severity
High (7.8)

Registry
Other

Affected Version
8.2.5151

Visibility
Public

Status
Fixed

Found by

xikhud
@acquykhud
legend

Fixed by

Bram Moolenaar
@brammool
maintainer

This report was seen 826 times.

We are processing your report and will contact the **vim** team within 24 hours. 5 months ago

We have contacted a member of the **vim** team and are waiting to hear back 5 months ago

Bram Moolenaar 5 months ago Maintainer

I cannot reproduce the problem, valgrind output is empty.

xikhud 5 months ago Researcher

My compile commands are below:

Chat with us

```
git checkout 8eba2bd291b347e3008aa9e565652d51ad638cfa
CFLAGS="-fsanitize=address -g" CXXFLAGS="-fsanitize=address -g" LDFLAGS="-fsanitize=ac

make -j2
```

Then I run

```
./vim2/src/vim  -u NONE -i NONE -n -m -X -Z -e -s -S poc_vim01 -c :qa!
```

I can still reproduce the problem with the latest version, which is `8.2.5155` at the moment.

xikhud 5 months ago   Researcher

I tried to compile vim without ASAN:

```
./configure --prefix=/home/guest/trung/install_vim2
make -j2
```

My `valgrind` is showing the result

```
$ valgrind ./vim2/src/vim  -u NONE -i NONE -n -m -X -Z -e -s -S poc_vim01 -c :qa!
==19219== Memcheck, a memory error detector
==19219== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==19219== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==19219== Command: ./vim2/src/vim -u NONE -i NONE -n -m -X -Z -e -s -S poc_vim01 -c :q
==19219==
==19219== Conditional jump or move depends on uninitialised value(s)
==19219==    at 0x4C34D08: strlen (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux
==19219==    by 0x38774F: msg_outtrans_attr (message.c:1551)
==19219==    by 0x387B03: msg_attr_keep (message.c:176)
==19219==    by 0x387F73: msg_attr (message.c:123)
==19219==    by 0x387F73: msg_trunc_attr (message.c:933)
==19219==    by 0x1FE1B4: process_next_cpt_value (insexpand.c:3271)
==19219==    by 0x1FE1B4: ins_compl_get_exp (insexpand.c:3743)
==19219==    by 0x1FE1B4: find_next_completion_match (insexpand.c:3992)
```

Chat with us

```
==19219==    by 0x1FE1B4: ins_compl_next (insexpand.c:4093)
==19219==    by 0x1FE72A: ins_complete (insexpand.c:4944)
==19219==    by 0x180426: edit (edit.c:1281)
==19219==    by 0x22F599: invoke_edit.isra.1 (normal.c:7035)
==19219==    by 0x2317D1: n_opencmd (normal.c:6279)
==19219==    by 0x2317D1: nv_open (normal.c:7416)
==19219==    by 0x2385B4: normal_cmd (normal.c:939)
==19219==    by 0x1B671C: exec_normal (ex_docmd.c:8807)
==19219==    by 0x1B697F: ex_normal (ex_docmd.c:8693)
==19219==
==19219==
==19219== HEAP SUMMARY:
==19219==     in use at exit: 73,436 bytes in 382 blocks
==19219==   total heap usage: 1,227 allocs, 845 frees, 416,677 bytes allocated
==19219==
==19219== LEAK SUMMARY:
==19219==    definitely lost: 0 bytes in 0 blocks
==19219==    indirectly lost: 0 bytes in 0 blocks
==19219==      possibly lost: 139 bytes in 8 blocks
==19219==    still reachable: 73,297 bytes in 374 blocks
==19219==         suppressed: 0 bytes in 0 blocks
==19219== Rerun with --leak-check=full to see details of leaked memory
==19219==
==19219== For counts of detected and suppressed errors, rerun with: -v
==19219== Use --track-origins=yes to see where uninitialised values come from
==19219== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)
```

`valgrind` version:

```
$ valgrind --version
valgrind-3.13.0
```

`vim` version

```
$ ./vim2/src/vim --version
VIM - Vi IMproved 8.2 (2019 Dec 12, compiled Jun 25 2022 00:35:46)
Included patches: 1-5151
Compiled by guest@elk
Huge version without GUI.  Features included (+) or not (-):
+acl                +file_in_path       +mouse_urxvt        -tag_any_white
+arabic             +find_in_path       +mouse_xterm        -tcl
```

Chat with us

```
+autocmd           +float             +multi_byte        +termguicolors
+autochdir         +folding           +multi_lang        +terminal
-autoservername    -footer            -mzscheme          +terminfo

-balloon_eval      +fork()            +netbeans_intg     +termresponse
+balloon_eval_term -gettext           +num64             +textobjects
-browse            -hangul_input      +packages          +textprop
++builtin_terms    +iconv             +path_extra        +timers
+byte_offset       +insert_expand     -perl              +title
+channel           +ipv6              +persistent_undo   -toolbar
+cindent           +job               +popupwin          +user_commands
-clientserver      +jumplist          +postscript        +vartabs
-clipboard         +keymap            +printer           +vertsplit
+cmdline_compl     +lambda            +profile           +vim9script
+cmdline_hist      +langmap           -python            +viminfo
+cmdline_info      +libcall           -python3           +virtualedit
+comments          +linebreak         +quickfix          +visual
+conceal           +lispindent        +reltime           +visualextra
+cryptv            +listcmds          +rightleft         +vreplace
+cscope            +localmap          -ruby              +wildignore
+cursorbind        -lua               +scrollbind        +wildmenu
+cursorshape       +menu              +signs             +windows
+dialog_con        +mksession         +smartindent       +writebackup
+diff              +modify_fname      -sodium            -X11
+digraphs          +mouse             -sound             -xfontset
-dnd               -mouseshape        +spell             -xim
-ebcdic            +mouse_dec         +startuptime       -xpm
+emacs_tags        -mouse_gpm         +statusline        -xsmp
+eval              -mouse_jsbterm     -sun_workshop      -xterm_clipboard
+ex_extra          +mouse_netterm     +syntax            -xterm_save
+extra_search      +mouse_sgr         +tag_binary
-farsi             -mouse_sysmouse    -tag_old_static
   system vimrc file: "$VIM/vimrc"
     user vimrc file: "$HOME/.vimrc"
 2nd user vimrc file: "~/.vim/vimrc"
      user exrc file: "$HOME/.exrc"
       defaults file: "$VIMRUNTIME/defaults.vim"
  fall-back for $VIM: "/home/guest/trung/install_vim2/share/vim"
Compilation: gcc -c -I. -Iproto -DHAVE_CONFIG_H -g -O2 -D_REENTRANT -U_FORTIFY_SOURCE
Linking: gcc -L/usr/local/lib -Wl,--as-needed -o vim -lm -ltinfo -lrt -ldl
```

Bram Moolenaar 5 months ago

Chat with us

In the POC the CTRL-N command is used in Insert mode. Perhaps for how you start Vim (in

which directory) makes a difference.
What if you make the "src" directory the current directory?
You can also see if changing the 'complete' option makes a difference.

Bram Moolenaar 5 months ago Maintainer

Note that at insexpand.c:3271 the messag is "Scanning tags.", which is very short. Do you perhaps the environment set to translated messages? What is $LANG or $LC_ALL set to?

xikhud 5 months ago Researcher

I try to run `vim` with the current directory is set to `vim2/src` :

```
guest@elk:~/trung/vim2/src$ valgrind ./vim  -u NONE -i NONE -n -m -X -Z -e -s -S ../..
==956== Memcheck, a memory error detector
==956== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==956== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==956== Command: ./vim -u NONE -i NONE -n -m -X -Z -e -s -S ../../poc_vim01 -c :qa!
==956==
==956== Conditional jump or move depends on uninitialised value(s)
==956==    at 0x4C34D08: strlen (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.s
==956==    by 0x38774F: msg_outtrans_attr (message.c:1551)
==956==    by 0x387B03: msg_attr_keep (message.c:176)
==956==    by 0x387F73: msg_attr (message.c:123)
==956==    by 0x387F73: msg_trunc_attr (message.c:933)
==956==    by 0x1FE1B4: process_next_cpt_value (insexpand.c:3271)
==956==    by 0x1FE1B4: ins_compl_get_exp (insexpand.c:3743)
==956==    by 0x1FE1B4: find_next_completion_match (insexpand.c:3992)
==956==    by 0x1FE1B4: ins_compl_next (insexpand.c:4093)
==956==    by 0x1FE72A: ins_complete (insexpand.c:4944)
==956==    by 0x180426: edit (edit.c:1281)
==956==    by 0x22F599: invoke_edit.isra.1 (normal.c:7035)
==956==    by 0x2317D1: n_opencmd (normal.c:6279)
==956==    by 0x2317D1: nv_open (normal.c:7416)
==956==    by 0x2385B4: normal_cmd (normal.c:939)
==956==    by 0x1B671C: exec_normal (ex_docmd.c:8807)
==956==    by 0x1B697F: ex_normal (ex_docmd.c:8693)
==956==
==956==
==956== HEAP SUMMARY:
==956==     in use at exit: 73,433 bytes in 382 blocks
==956==   total heap usage: 1,227 allocs, 845 frees, 416,842 bytes al
==956==
==956== LEAK SUMMARY:
==956==    definitely lost: 0 bytes in 0 blocks
```

Chat with us

```
==956==    indirectly lost: 0 bytes in 0 blocks
==956==      possibly lost: 139 bytes in 8 blocks
==956==    still reachable: 73,294 bytes in 374 blocks
==956==         suppressed: 0 bytes in 0 blocks
==956== Rerun with --leak-check=full to see details of leaked memory
==956==
==956== For counts of detected and suppressed errors, rerun with: -v
==956== Use --track-origins=yes to see where uninitialised values come from
==956== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)
```

Here is the output of my `locale`:

```
guest@elk:~/trung/vim2/src$ locale
LANG=en_US.UTF-8
LANGUAGE=
LC_CTYPE="en_US.UTF-8"
LC_NUMERIC="en_US.UTF-8"
LC_TIME="en_US.UTF-8"
LC_COLLATE="en_US.UTF-8"
LC_MONETARY="en_US.UTF-8"
LC_MESSAGES="en_US.UTF-8"
LC_PAPER="en_US.UTF-8"
LC_NAME="en_US.UTF-8"
LC_ADDRESS="en_US.UTF-8"
LC_TELEPHONE="en_US.UTF-8"
LC_MEASUREMENT="en_US.UTF-8"
LC_IDENTIFICATION="en_US.UTF-8"
LC_ALL=
```

The stack trace looks much like from here
I also try changing the whole environment using these commands:

```
export LC_ALL=en_US.UTF-8
export LANGUAGE=en_US.UTF-8
```

But still get the error

Chat with us

Bram Moolenaar 5 months ago Maintainer

Perhaps the width of the terminal matters? A smaller width may cause the truncation to kick in. I tried a few sizes but still cannot reproduce the problem.

xikhud 5 months ago Researcher

Yes, I think the height and the width of the terminal matters.
You can use this command to print out the size of the terminal

```
stty size
```

With width = 209, height = 52, it crashes

```
guest@elk:~/trung$ stty size
52 209
guest@elk:~/trung$ valgrind ./vim2/src/vim -u NONE -i NONE -n -m -X -Z -e -s -S ./poc_
==27959== Memcheck, a memory error detector
==27959== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==27959== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==27959== Command: ./vim2/src/vim -u NONE -i NONE -n -m -X -Z -e -s -S ./poc_vim01 -c
==27959==
==27959== Conditional jump or move depends on uninitialised value(s)
==27959==    at 0x4C34D08: strlen (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux
==27959==    by 0x38774F: msg_outtrans_attr (message.c:1551)
==27959==    by 0x387B03: msg_attr_keep (message.c:176)
==27959==    by 0x387F73: msg_attr (message.c:123)
==27959==    by 0x387F73: msg_trunc_attr (message.c:933)
==27959==    by 0x1FE1B4: process_next_cpt_value (insexpand.c:3271)
==27959==    by 0x1FE1B4: ins_compl_get_exp (insexpand.c:3743)
==27959==    by 0x1FE1B4: find_next_completion_match (insexpand.c:3992)
==27959==    by 0x1FE1B4: ins_compl_next (insexpand.c:4093)
==27959==    by 0x1FE72A: ins_complete (insexpand.c:4944)
==27959==    by 0x180426: edit (edit.c:1281)
==27959==    by 0x22F599: invoke_edit.isra.1 (normal.c:7035)
==27959==    by 0x2317D1: n_opencmd (normal.c:6279)
==27959==    by 0x2317D1: nv_open (normal.c:7416)
==27959==    by 0x2385B4: normal_cmd (normal.c:939)
==27959==    by 0x1B671C: exec_normal (ex_docmd.c:8807)
==27959==    by 0x1B697F: ex_normal (ex_docmd.c:8693)
==27959==
==27959==
==27959== HEAP SUMMARY:
```

Chat with us

```
==27959== HEAP SUMMARY:
==27959==     in use at exit: 73,438 bytes in 382 blocks
==27959==   total heap usage: 1,227 allocs, 845 frees, 416,543 bytes allocated
==27959==
==27959== LEAK SUMMARY:
==27959==    definitely lost: 0 bytes in 0 blocks
==27959==    indirectly lost: 0 bytes in 0 blocks
==27959==      possibly lost: 139 bytes in 8 blocks
==27959==    still reachable: 73,299 bytes in 374 blocks
==27959==         suppressed: 0 bytes in 0 blocks
==27959== Rerun with --leak-check=full to see details of leaked memory
==27959==
==27959== For counts of detected and suppressed errors, rerun with: -v
==27959== Use --track-origins=yes to see where uninitialised values come from
==27959== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)
```

But if width = 201, and height = 10, it doesn't:

```
guest@elk:~/trung$ stty size
10 201
guest@elk:~/trung$ valgrind ./vim2/src/vim -u NONE -i NONE -n -m -X -Z -e -s -S ./poc_
==28333== Memcheck, a memory error detector
==28333== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==28333== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==28333== Command: ./vim2/src/vim -u NONE -i NONE -n -m -X -Z -e -s -S ./poc_vim01 -c
==28333==
==28333==
==28333== HEAP SUMMARY:
==28333==     in use at exit: 73,438 bytes in 382 blocks
==28333==   total heap usage: 1,227 allocs, 845 frees, 416,543 bytes allocated
==28333==
==28333== LEAK SUMMARY:
==28333==    definitely lost: 0 bytes in 0 blocks
==28333==    indirectly lost: 0 bytes in 0 blocks
==28333==      possibly lost: 139 bytes in 8 blocks
==28333==    still reachable: 73,299 bytes in 374 blocks
==28333==         suppressed: 0 bytes in 0 blocks
==28333== Rerun with --leak-check=full to see details of leaked memory
==28333==
==28333== For counts of detected and suppressed errors, rerun with: -v
==28333== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from
```

Chat with us

You can use this command to print out all the variables at this line:

```
gdb -ex "b msg_may_trunc" -ex "r" -ex "printf \"Rows=%d %c\", Rows,0xA" -ex "printf \"
```

xikhud 5 months ago Researcher

My output which causes crash:

```
Rows=2
Columns=12
cmdline_row=51
sc_col=209
```

With this, `room` will be `0xfffffe78`

Bram Moolenaar validated this vulnerability 5 months ago

finally I can reproduce it. Thanks for being persistant.

xikhud has been awarded the disclosure bounty ✓

The fix bounty is now up for grabs

The researcher's credibility has increased: +7

Bram Moolenaar 5 months ago Maintainer

Fixed in patch 8.2.5160

Bram Moolenaar marked this as fixed in 8.2 with commit e178af 5 months a

Chat with us

Bram Moolenaar has been awarded the fix bounty ✓

This vulnerability will not receive a CVE ×

Sign in to join this conversation