New issue

Jump to bottom

heap-buffer-overflow in put_epel_16_fallback when decoding file #232

Open leonzhao7 opened this issue on Dec 24, 2019 · 3 comments

leonzhao7 commented on Dec 24, 2019

heap-buffer-overflow in put_epel_16_fallback when decoding file

I found some problems during fuzzing

Test Version

dev version, git clone https://github.com/strukturag/libde265

Test Environment

root@ubuntu:~# lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description: Ubuntu 16.04.6 LTS
Release: 16.04
Codename: xenial

Test Configure

./configure
configure: --
configure: Building dec265 example: yes
configure: Building sherlock265 example: no
configure: Building encoder: yes
configure: --

Test Program

```
dec265 [infile]
```

Asan Output

```
root@ubuntu:~# /opt/asan/bin/dec265 libde265-put_epel_16_fallback-heap_overflow.crash
WARNING: CTB outside of image area (concealing stream error...)
WARNING: pps header invalid
WARNING: faulty reference picture list
WARNING: pps header invalid
=================================================================
==39540==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x62b00001b510 at pc 0x000000433086 bp 0x7ffd99655f60 sp 0x7ffd99655f50
READ of size 2 at 0x62b00001b510 thread T0
    #0 0x433085 in put_epel_16_fallback(short*, long, unsigned short const*, long, int, int, int, int, short*, int) /root/src/libde265/libde265/fallback-motion.cc:289
    #1 0x52bfe0 in acceleration_functions::put_hevc_epel(short*, long, void const*, long, int, int, int, int, short*, int) const ../libde265/acceleration.h:298
    #2 0x52dc7a in void mc_chroma<unsigned short>(base_context const*, seq_parameter_set const*, int, int, int, int, short*, int, unsigned short const*, int, int, int, int)
/root/src/libde265/libde265/motion.cc:205
    #3 0x51f88a in generate_inter_prediction_samples(base_context*, slice_segment_header const*, de265_image*, int, int, int, int, int, int, int, PBMotion const*)
/root/src/libde265/libde265/motion.cc:382
    #4 0x52b8f9 in decode_prediction_unit(base_context*, slice_segment_header const*, de265_image*, PBMotionCoding const&, int, int, int, int, int, int, int, int)
/root/src/libde265/libde265/motion.cc:2107
    #5 0x47995d in read_coding_unit(thread_context*, int, int, int, int) /root/src/libde265/libde265/slice.cc:4310
    #6 0x47b6fe in read_coding_quadtree(thread_context*, int, int, int, int) /root/src/libde265/libde265/slice.cc:4647
    #7 0x47338a in read_coding_tree_unit(thread_context*) /root/src/libde265/libde265/slice.cc:2861
    #8 0x47beb1 in decode_substream(thread_context*, bool, bool) /root/src/libde265/libde265/slice.cc:4736
    #9 0x47db9f in read_slice_segment_data(thread_context*) /root/src/libde265/libde265/slice.cc:5049
    #10 0x40bf17 in decoder_context::decode_slice_unit_sequential(image_unit*, slice_unit*) /root/src/libde265/libde265/decctx.cc:843
    #11 0x40c6d7 in decoder_context::decode_slice_unit_parallel(image_unit*, slice_unit*) /root/src/libde265/libde265/decctx.cc:945
    #12 0x40b589 in decoder_context::decode_some(bool*) /root/src/libde265/libde265/decctx.cc:730
    #13 0x40e23e in decoder_context::decode(int*) /root/src/libde265/libde265/decctx.cc:1329
    #14 0x405a61 in de265_decode /root/src/libde265/libde265/de265.cc:346
    #15 0x404972 in main /root/src/libde265/dec265/dec265.cc:764
    #16 0x7fa1b901182f in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x2082f)
    #17 0x402b28 in _start (/opt/asan/bin/dec265+0x402b28)

0x62b00001b510 is located 0 bytes to the right of 25360-byte region [0x62b000015200,0x62b00001b510)
allocated by thread T0 here:
    #0 0x7fa1b9f12076 in __interceptor_posix_memalign (/usr/lib/x86_64-linux-gnu/libasan.so.2+0x99076)
    #1 0x43e00d in ALLOC_ALIGNED /root/src/libde265/libde265/image.cc:54
    #2 0x43e725 in de265_image_get_buffer /root/src/libde265/libde265/image.cc:132
    #3 0x440639 in de265_image::alloc_image(int, int, de265_chroma, std::shared_ptr<seq_parameter_set const>, bool, decoder_context*, long, void*, bool)
/root/src/libde265/libde265/image.cc:384
    #4 0x43afa4 in decoded_picture_buffer::new_image(std::shared_ptr<seq_parameter_set const>, decoder_context*, long, void*, bool) /root/src/libde265/libde265/dpb.cc:262
    #5 0x40ee8b in decoder_context::generate_unavailable_reference_picture(seq_parameter_set const*, int, bool) /root/src/libde265/libde265/decctx.cc:1418
    #6 0x411722 in decoder_context::process_reference_picture_set(slice_segment_header*) /root/src/libde265/libde265/decctx.cc:1648
    #7 0x414cc9 in decoder_context::process_slice_segment_header(slice_segment_header*, de265_error*, long, nal_header*, void*) /root/src/libde265/libde265/decctx.cc:2066
    #8 0x40acad in decoder_context::read_slice_NAL(bitreader&, NAL_unit*, nal_header&) /root/src/libde265/libde265/decctx.cc:639
    #9 0x40dbb3 in decoder_context::decode_NAL(NAL_unit*) /root/src/libde265/libde265/decctx.cc:1230
    #10 0x40e17b in decoder_context::decode(int*) /root/src/libde265/libde265/decctx.cc:1318
    #11 0x405a61 in de265_decode /root/src/libde265/libde265/de265.cc:346
    #12 0x404972 in main /root/src/libde265/dec265/dec265.cc:764
    #13 0x7fa1b901182f in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x2082f)

SUMMARY: AddressSanitizer: heap-buffer-overflow /root/src/libde265/libde265/fallback-motion.cc:289 put_epel_16_fallback(short*, long, unsigned short const*, long, int, int, int,
int, short*, int)
```

```
Shadow bytes around the buggy address:
  0x0c567fffb650: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c567fffb660: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c567fffb670: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c567fffb680: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c567fffb690: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x0c567fffb6a0: 00 00[fa]fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c567fffb6b0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c567fffb6c0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c567fffb6d0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c567fffb6e0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c567fffb6f0: 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
  Heap right redzone:      fb
  Freed heap region:       fd
  Stack left redzone:      f1
  Stack mid redzone:       f2
  Stack right redzone:     f3
  Stack partial redzone:   f4
  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
==39540==ABORTING
```

POC file

libde265-put_epel_16_fallback-heap_overflow.zip
password: leon.zhao.7

CREDIT

Zhao Liang, Huawei Weiran Labs

ist199099 commented on Oct 16

With the tip of the stable branch at the time of this bug report (`d065715`) on Ubuntu 20.04 (with gcc 9.4.0 and clang 10.0.0) on the aarch64 architecture, I do not get a heap out-of-bounds read, but a heap out-of-bounds write, that may lead to arbitrary code execution.

```
WARNING: CTB outside of image area (concealing stream error...)
WARNING: pps header invalid
SDL_Init() failed: Unable to open a console terminal
WARNING: faulty reference picture list
WARNING: pps header invalid
=================================================================
==781426==ERROR: AddressSanitizer: heap-buffer-overflow on address 0xffff96314500 at pc 0xffff9d21bf14 bp 0xfffffa35bba0 sp 0xfffffa35bbc0
WRITE of size 2 at 0xffff96314500 thread T0
    #0 0xffff9d21bf10 in put_weighted_bipred_16_fallback(unsigned short*, long, short const*, short const*, long, int, int, int, int, int, int, int) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x160f10)
    #1 0xffff9d260c20 in acceleration_functions::put_weighted_bipred(void*, long, short const*, short const*, long, int, int, int, int, int, int, int, int) const (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1a5c20)
    #2 0xffff9d253874 in generate_inter_prediction_samples(base_context*, slice_segment_header const*, de265_image*, int, int, int, int, int, int, int, PBMotion const*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x198874)
    #3 0xffff9d25f574 in decode_prediction_unit(base_context*, slice_segment_header const*, de265_image*, PBMotionCoding const&, int, int, int, int, int, int, int, int) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1a4574)
    #4 0xffff9d2a9380 in read_coding_unit(thread_context*, int, int, int, int) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1ee380)
    #5 0xffff9d2ab4f0 in read_coding_quadtree(thread_context*, int, int, int, int) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1f04f0)
    #6 0xffff9d2a18e4 in read_coding_tree_unit(thread_context*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1e68e4)
    #7 0xffff9d2abe38 in decode_substream(thread_context*, bool, bool) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1f0e38)
    #8 0xffff9d2adf84 in read_slice_segment_data(thread_context*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1f2f84)
    #9 0xffff9d1e32b4 in decoder_context::decode_slice_unit_sequential(image_unit*, slice_unit*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1282b4)
    #10 0xffff9d1e3bac in decoder_context::decode_slice_unit_parallel(image_unit*, slice_unit*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x128bac)
    #11 0xffff9d1e2650 in decoder_context::decode_some(bool*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x127650)
    #12 0xffff9d1e5c44 in decoder_context::decode(int*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x12ac44)
    #13 0xffff9d1c943c in de265_decode (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x10e43c)
    #14 0xaaaadfb974c0 in main (/home/azureuser/libde265/dec265/.libs/dec265+0x74c0)
    #15 0xffff9ccb2e0c in __libc_start_main ../csu/libc-start.c:308
    #16 0xaaaadfb94a90  (/home/azureuser/libde265/dec265/.libs/dec265+0x4a90)

0xffff96314500 is located 0 bytes to the right of 25344-byte region [0xffff9630e200,0xffff96314500)
allocated by thread T0 here:
    #0 0xffff9d59b2d8 in __interceptor_posix_memalign ../../../../src/libsanitizer/asan/asan_malloc_linux.cc:217
    #1 0xffff9d22b2fc in ALLOC_ALIGNED(unsigned long, unsigned long) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1702fc)
    #2 0xffff9d22bbf0 in de265_image_get_buffer(void*, de265_image_spec*, de265_image*, void*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x170bf0)
    #3 0xffff9d22e6a0 in de265_image::alloc_image(int, int, de265_chroma, std::shared_ptr<seq_parameter_set const>, bool, decoder_context*, encoder_context*, long, void*, bool) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1736a0)
    #4 0xffff9d226bec in decoded_picture_buffer::new_image(std::shared_ptr<seq_parameter_set const>, decoder_context*, long, void*, bool) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x16bbec)
    #5 0xffff9d1eccf4 in decoder_context::process_slice_segment_header(slice_segment_header*, de265_error*, long, nal_header*, void*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x131cf4)
    #6 0xffff9d1e1bdc in decoder_context::read_slice_NAL(bitreader&, NAL_unit*, nal_header&) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x126bdc)
    #7 0xffff9d1e54c8 in decoder_context::decode_NAL(NAL_unit*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x12a4c8)
    #8 0xffff9d1e5b24 in decoder_context::decode(int*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x12ab24)
    #9 0xffff9d1c943c in de265_decode (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x10e43c)
    #10 0xaaaadfb974c0 in main (/home/azureuser/libde265/dec265/.libs/dec265+0x74c0)
    #11 0xffff9ccb2e0c in __libc_start_main ../csu/libc-start.c:308
    #12 0xaaaadfb94a90  (/home/azureuser/libde265/dec265/.libs/dec265+0x4a90)

SUMMARY: AddressSanitizer: heap-buffer-overflow (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x160f10) in put_weighted_bipred_16_fallback(unsigned short*, long, short const*, short const*, long, int, int, int, int, int, int, int, int)
Shadow bytes around the buggy address:
  0x200ff2c62850: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x200ff2c62860: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x200ff2c62870: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x200ff2c62880: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x200ff2c62890: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
=>0x200ff2c628a0:[fa]fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x200ff2c628b0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x200ff2c628c0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x200ff2c628d0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x200ff2c628e0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x200ff2c628f0: 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
==781426==ABORTING
```

ist199099 commented on Oct 16

The same happens on the x86_64 architecture:

```
WARNING: CTB outside of image area (concealing stream error...)
WARNING: pps header invalid
SDL_Init() failed: Unable to open a console terminal
WARNING: faulty reference picture list
WARNING: pps header invalid
=================================================================
==791137==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x62b000014508 at pc 0x7fdad54ad4fb bp 0x7fff67eb7c80 sp 0x7fff67eb7c70
WRITE of size 2 at 0x62b000014508 thread T0
    #0 0x7fdad54ad4fa in put_weighted_bipred_16_fallback(unsigned short*, long, short const*, short const*, long, int, int, int, int, int, int, int)
(/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x16b4fa)
    #1 0x7fdad54e6469 in acceleration_functions::put_weighted_bipred(void*, long, short const*, short const*, long, int, int, int, int, int, int, int) const
(/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1a4469)
    #2 0x7fdad54dbcf0 in generate_inter_prediction_samples(base_context*, slice_segment_header const*, de265_image*, int, int, int, int, int, int, int, PBMotion const*)
(/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x199cf0)
    #3 0x7fdad54e5e7d in decode_prediction_unit(base_context*, slice_segment_header const*, de265_image*, PBMotionCoding const&, int, int, int, int, int, int, int, int)
(/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1a3e7d)
    #4 0x7fdad55253a6 in read_coding_unit(thread_context*, int, int, int, int) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1e33a6)
    #5 0x7fdad55272bd in read_coding_quadtree(thread_context*, int, int, int, int, int) [clone .localalias] (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1e52bd)
    #6 0x7fdad551e92d in read_coding_tree_unit(thread_context*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1dc92d)
    #7 0x7fdad5527a81 in decode_substream(thread_context*, bool, bool) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1e5a81)
    #8 0x7fdad55297ec in read_slice_segment_data(thread_context*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1e77ec)
    #9 0x7fdad547880f in decoder_context::decode_slice_unit_sequential(image_unit*, slice_unit*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x13680f)
    #10 0x7fdad5479011 in decoder_context::decode_slice_unit_parallel(image_unit*, slice_unit*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x137011)
    #11 0x7fdad5477ca3 in decoder_context::decode_some(bool*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x135ca3)
    #12 0x7fdad547ace2 in decoder_context::decode(int*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x138ce2)
    #13 0x7fdad54610e0 in de265_decode (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x11f0e0)
    #14 0x55a6ae05007f in main (/home/azureuser/libde265/dec265/.libs/dec265+0x807f)
    #15 0x7fdad4e26082 in __libc_start_main ../csu/libc-start.c:308
    #16 0x55a6ae04d9cd in _start (/home/azureuser/libde265/dec265/.libs/dec265+0x59cd)

0x62b000014508 is located 0 bytes to the right of 25352-byte region [0x62b00000e200,0x62b000014508)
allocated by thread T0 here:
    #0 0x7fdad58776e5 in __interceptor_posix_memalign ../../../../src/libsanitizer/asan/asan_malloc_linux.cc:217
    #1 0x7fdad54ba459 in ALLOC_ALIGNED(unsigned long, unsigned long) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x178459)
    #2 0x7fdad54bac50 in de265_image_get_buffer(void*, de265_image_spec*, de265_image*, void*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x178c50)
    #3 0x7fdad54bd2eb in de265_image::alloc_image(int, int, de265_chroma, std::shared_ptr<seq_parameter_set const>, bool, decoder_context*, encoder_context*, long, void*, bool)
(/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x17b2eb)
    #4 0x7fdad54b5da2 in decoded_picture_buffer::new_image(std::shared_ptr<seq_parameter_set const>, decoder_context*, long, void*, bool)
(/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x173da2)
    #5 0x7fdad5481332 in decoder_context::process_slice_segment_header(slice_segment_header*, de265_error*, long, nal_header*, void*)
(/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x13f332)
    #6 0x7fdad54772ea in decoder_context::read_slice_NAL(bitreader&, NAL_unit*, nal_header&) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x1352ea)
    #7 0x7fdad547a58c in decoder_context::decode_NAL(NAL_unit*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x13858c)
    #8 0x7fdad547abe9 in decoder_context::decode(int*) (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x138be9)
    #9 0x7fdad54610e0 in de265_decode (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x11f0e0)
    #10 0x55a6ae05007f in main (/home/azureuser/libde265/dec265/.libs/dec265+0x807f)
    #11 0x7fdad4e26082 in __libc_start_main ../csu/libc-start.c:308

SUMMARY: AddressSanitizer: heap-buffer-overflow (/home/azureuser/libde265/libde265/.libs/libde265.so.0+0x16b4fa) in put_weighted_bipred_16_fallback(unsigned short*, long, short
const*, short const*, long, int, int, int, int, int, int, int, int)
Shadow bytes around the buggy address:
  0x0c567fffa850: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c567fffa860: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c567fffa870: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c567fffa880: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c567fffa890: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x0c567fffa8a0: 00[fa]fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c567fffa8b0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c567fffa8c0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c567fffa8d0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c567fffa8e0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c567fffa8f0: 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
  ==791137==ABORTING
```

ist199099 commented on Oct 16

I cannot reproduce this in the tip of the stable branch (`b371427`) on Ubuntu 20.04 (with gcc 9.4.0 and clang 10.0.0) on the x86_64 and aarch64 architectures.

This has been assigned CVE-2020-21606.

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

2 participants