New issue

Jump to bottom

# Stack-buffer-overflow in fallback-motion.cc: void put_qpel_fallback<unsigned short> #343

Open FDU-Sec opened this issue on Oct 10 · 0 comments

FDU-Sec commented on Oct 10

## Description

Stack-buffer-overflow (/libde265/build/libde265/liblibde265.so+0x14bef2) in void put_qpel_fallback(short*, long, unsigned short const*, long, int, int, short*, int, int, int)

## Version

```
$ ./dec265 -h
 dec265  v1.0.8
--------------
usage: dec265 [options] videofile.bin
The video file must be a raw bitstream, or a stream with NAL units (option -n).

options:
  -q, --quiet       do not show decoded image
  -t, --threads N   set number of worker threads (0 - no threading)
  -c, --check-hash  perform hash check
  -n, --nal         input is a stream with 4-byte length prefixed NAL units
  -f, --frames N    set number of frames to process
  -o, --output      write YUV reconstruction
  -d, --dump        dump headers
  -0, --noaccel     do not use any accelerated code (SSE)
  -v, --verbose     increase verbosity level (up to 3 times)
  -L, --no-logging  disable logging
  -B, --write-bytestream FILENAME  write raw bytestream (from NAL input)
  -m, --measure YUV compute PSNRs relative to reference YUV
  -T, --highest-TID select highest temporal sublayer to decode
      --disable-deblocking   disable deblocking filter
      --disable-sao          disable sample-adaptive offset filter
  -h, --help        show help
```

## Replay

```
git clone https://github.com/strukturag/libde265.git
cd libde265
mkdir build
cd build
cmake ../ -DCMAKE_CXX_FLAGS="-fsanitize=address"
make -j$(nproc)
./dec265/dec265 poc9-1
./dec265/dec265 poc9-2
./dec265/dec265 poc9-3
./dec265/dec265 poc9-4
```

## ASAN

```
WARNING: pps header invalid
WARNING: CTB outside of image area (concealing stream error...)
WARNING: maximum number of reference pictures exceeded
WARNING: faulty reference picture list
WARNING: non-existing PPS referenced
WARNING: faulty reference picture list
=================================================================
==18325==ERROR: AddressSanitizer: stack-buffer-overflow on address 0x7ffd5f83a761 at pc 0x7f031a7b3ef
READ of size 2 at 0x7ffd5f83a761 thread T0
    #0 0x7f031a7b3ef2 in void put_qpel_fallback<unsigned short>(short*, long, unsigned short const*,
    #1 0x7f031a7af248 in put_qpel_2_1_fallback_16(short*, long, unsigned short const*, long, int, int
    #2 0x7f031a7df40d in acceleration_functions::put_hevc_qpel(short*, long, void const*, long, int,
    #3 0x7f031a7e08ab in void mc_luma<unsigned char>(base_context const*, seq_parameter_set const*, i
    #4 0x7f031a7d1995 in generate_inter_prediction_samples(base_context*, slice_segment_header const*
    #5 0x7f031a7de90f in decode_prediction_unit(base_context*, slice_segment_header const*, de265_ima
    #6 0x7f031a8197e3 in read_prediction_unit(thread_context*, int, int, int, int, int, int, int, int
    #7 0x7f031a81b264 in read_coding_unit(thread_context*, int, int, int, int) (/libde265/build/libde
    #8 0x7f031a81c250 in read_coding_quadtree(thread_context*, int, int, int, int) (/libde265/build/l
    #9 0x7f031a813726 in read_coding_tree_unit(thread_context*) (/libde265/build/libde265/liblibde265
    #10 0x7f031a81c9ea in decode_substream(thread_context*, bool, bool) (/libde265/build/libde265/lib
    #11 0x7f031a81e70f in read_slice_segment_data(thread_context*) (/libde265/build/libde265/liblibde
    #12 0x7f031a77d6d2 in decoder_context::decode_slice_unit_sequential(image_unit*, slice_unit*) (/l
    #13 0x7f031a77dec1 in decoder_context::decode_slice_unit_parallel(image_unit*, slice_unit*) (/lib
    #14 0x7f031a77cc0f in decoder_context::decode_some(bool*) (/libde265/build/libde265/liblibde265.s
    #15 0x7f031a77c93d in decoder_context::read_slice_NAL(bitreader&, NAL_unit*, nal_header&) (/libde
    #16 0x7f031a77f43e in decoder_context::decode_NAL(NAL_unit*) (/libde265/build/libde265/liblibde26
    #17 0x7f031a77fab3 in decoder_context::decode(int*) (/libde265/build/libde265/liblibde265.so+0x11
    #18 0x7f031a766e95 in de265_decode (/libde265/build/libde265/liblibde265.so+0xfee95)
    #19 0x5564657f6bc9 in main (/libde265/build/dec265/dec265+0x6bc9)
    #20 0x7f031a298c86 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x21c86)
    #21 0x5564657f49b9 in _start (/libde265/build/dec265/dec265+0x49b9)

Address 0x7ffd5f83a761 is located in stack of thread T0 at offset 9121 in frame
    #0 0x7f031a7dffb7 in void mc_luma<unsigned char>(base_context const*, seq_parameter_set const*, i

  This frame has 2 object(s):
    [32, 9120) 'mcbuffer' <== Memory access at offset 9121 overflows this variable
    [9152, 14832) 'padbuf'
HINT: this may be a false positive if your program uses some custom stack unwind mechanism or swapcon
      (longjmp and C++ exceptions *are* supported)
```

```
SUMMARY: AddressSanitizer: stack-buffer-overflow (/libde265/build/libde265/liblibde265.so+0x14bef2) i
Shadow bytes around the buggy address:
  0x10002beff490: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002beff4a0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002beff4b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002beff4c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002beff4d0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x10002beff4e0: 00 00 00 00 00 00 00 00 00 00 00 00[f2]f2 f2 f2
  0x10002beff4f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002beff500: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002beff510: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002beff520: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002beff530: 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
==18325==ABORTING
```

## POC

https://github.com/FDU-Sec/poc/blob/main/libde265/poc9-1
https://github.com/FDU-Sec/poc/blob/main/libde265/poc9-2
https://github.com/FDU-Sec/poc/blob/main/libde265/poc9-3
https://github.com/FDU-Sec/poc/blob/main/libde265/poc9-4

## Environment

```
Ubuntu 16.04
Clang 10.0.1
gcc 5.5
```

## Credit

Peng Deng (Fudan University)

**Assignees**

No one assigned

**Labels**

None yet

**Projects**

None yet

**Milestone**

No milestone

**Development**

No branches or pull requests

**1 participant**