New issue

Jump to bottom

Stack-buffer-overflow in fallback-motion.cc: void put_epel_hv_fallback<unsigned short> #344

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+0x148bb1) in void put_epel_hv_fallback(short*, long, unsigned short const*, long, int, int, int, int, short*, 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 poc10-1
./dec265/dec265 poc10-2
./dec265/dec265 poc10-3
```

ASAN

```
WARNING: end_of_sub_stream_one_bit not set to 1 when it should be
WARNING: CTB outside of image area (concealing stream error...)
=================================================================
==49284==ERROR: AddressSanitizer: stack-buffer-overflow on address 0x7ffd5d1376e1 at pc 0x7fc6e4cc7bb
READ of size 2 at 0x7ffd5d1376e1 thread T0
    #0 0x7fc6e4cc7bb1 in void put_epel_hv_fallback<unsigned short>(short*, long, unsigned short const
    #1 0x7fc6e4cf60de in acceleration_functions::put_hevc_epel_h(short*, long, void const*, long, int
    #2 0x7fc6e4cf8ca2 in void mc_chroma<unsigned char>(base_context const*, seq_parameter_set const*,
    #3 0x7fc6e4ce8e2e in generate_inter_prediction_samples(base_context*, slice_segment_header const*
    #4 0x7fc6e4cf590f in decode_prediction_unit(base_context*, slice_segment_header const*, de265_ima
    #5 0x7fc6e4d307e3 in read_prediction_unit(thread_context*, int, int, int, int, int, int, int, int
    #6 0x7fc6e4d32469 in read_coding_unit(thread_context*, int, int, int, int) (/libde265/build/libde
    #7 0x7fc6e4d33250 in read_coding_quadtree(thread_context*, int, int, int, int) (/libde265/build/l
    #8 0x7fc6e4d2a726 in read_coding_tree_unit(thread_context*) (/libde265/build/libde265/liblibde265
    #9 0x7fc6e4d339ea in decode_substream(thread_context*, bool, bool) (/libde265/build/libde265/libl
    #10 0x7fc6e4d3570f in read_slice_segment_data(thread_context*) (/libde265/build/libde265/liblibde
    #11 0x7fc6e4c946d2 in decoder_context::decode_slice_unit_sequential(image_unit*, slice_unit*) (/l
    #12 0x7fc6e4c94ec1 in decoder_context::decode_slice_unit_parallel(image_unit*, slice_unit*) (/lib
    #13 0x7fc6e4c93c0f in decoder_context::decode_some(bool*) (/libde265/build/libde265/liblibde265.s
    #14 0x7fc6e4c9393d in decoder_context::read_slice_NAL(bitreader&, NAL_unit*, nal_header&) (/libde
    #15 0x7fc6e4c9643e in decoder_context::decode_NAL(NAL_unit*) (/libde265/build/libde265/liblibde26
    #16 0x7fc6e4c96ab3 in decoder_context::decode(int*) (/libde265/build/libde265/liblibde265.so+0x11
    #17 0x7fc6e4c7de95 in de265_decode (/libde265/build/libde265/liblibde265.so+0xfee95)
    #18 0x56089bc03bc9 in main (/libde265/build/dec265/dec265+0x6bc9)
    #19 0x7fc6e47afc86 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x21c86)
    #20 0x56089bc019b9 in _start (/libde265/build/dec265/dec265+0x49b9)

Address 0x7ffd5d1376e1 is located in stack of thread T0 at offset 9121 in frame
    #0 0x7fc6e4cf83b8 in void mc_chroma<unsigned char>(base_context const*, seq_parameter_set const*,

  This frame has 2 object(s):
    [32, 9120) 'mcbuffer' <== Memory access at offset 9121 overflows this variable
    [9152, 14512) '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+0x148bb1) i
Shadow bytes around the buggy address:
  0x10002ba1ee80: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002ba1ee90: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002ba1eea0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002ba1eeb0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
  0x10002ba1eec0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x10002ba1eed0: 00 00 00 00 00 00 00 00 00 00 00 00[f2]f2 f2 f2
  0x10002ba1eee0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002ba1eef0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002ba1ef00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002ba1ef10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10002ba1ef20: 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
==49284==ABORTING
```

POC

https://github.com/FDU-Sec/poc/blob/main/libde265/poc10-1
https://github.com/FDU-Sec/poc/blob/main/libde265/poc10-2
https://github.com/FDU-Sec/poc/blob/main/libde265/poc10-3

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