AddressSanitizer: heap-use-after-free in stbi__jpeg_huff_decode #1289

Open pietroborrello opened this issue on Feb 17 · 1 comment

Labels 1 stb_image

pietroborrello commented on Feb 17

Describe the bug

UndefinedBehaviorSanitizer: undefined-behavior: index out of bounds + AddressSanitizer: heap-use-after-free in stbi__jpeg_huff_decode.

To Reproduce

Built stb according to the oss-fuzz script with `CXXFLAGS='-O1 -fsanitize=address -fsanitize=array-bounds,bool,builtin,enum,float-divide-by-zero,function,integer-divide-by-zero,null,object-size,return,returns-nonnull-attribute,shift,signed-integer-overflow,unreachable,vla-bound,vptr'`

ASAN Output

```
$ ./stbi_read_fuzzer ./id:000233,sig:06,src:005305,time:31715435,op:havoc,rep:4,trial:1493419.jpeg

INFO: Seed: 1219869484
INFO: Loaded 1 modules   (6883 inline 8-bit counters): 6883 [0x5e1b33, 0x5e3616),
INFO: Loaded 1 PC tables (6883 PCs): 6883 [0x573228,0x58e058),
./stbi_read_fuzzer: Running 1 inputs 1 time(s) each.
Running: id:000233,sig:06,src:005305,time:31715435,op:havoc,rep:4,trial:1493419
src/stb/tests/../stb_image.h:1990:10: runtime error: index 257 out of bounds for type 'stbi_uc
[257]'
SUMMARY: UndefinedBehaviorSanitizer: undefined-behavior src/stb/tests/../stb_image.h:1990:10 in
src/stb/tests/../stb_image.h:1991:4: runtime error: index 654 out of bounds for type 'stbi_uc
[257]'
SUMMARY: UndefinedBehaviorSanitizer: undefined-behavior src/stb/tests/../stb_image.h:1991:4 in
src/stb/tests/../stb_image.h:2001:13: runtime error: index 256 out of bounds for type
'stbi__uint16 [256]'
SUMMARY: UndefinedBehaviorSanitizer: undefined-behavior src/stb/tests/../stb_image.h:2001:13 in
src/stb/tests/../stb_image.h:2000:17: runtime error: index 257 out of bounds for type 'stbi_uc
[257]'
SUMMARY: UndefinedBehaviorSanitizer: undefined-behavior src/stb/tests/../stb_image.h:2000:17 in
```

```
src/stb/tests/../stb_image.h:1999:11: runtime error: index 264 out of bounds for type 'stbi_uc
[257]'
SUMMARY: UndefinedBehaviorSanitizer: undefined-behavior src/stb/tests/../stb_image.h:1999:11 in
src/stb/tests/../stb_image.h:2013:15: runtime error: index 257 out of bounds for type 'stbi_uc
[257]'
SUMMARY: UndefinedBehaviorSanitizer: undefined-behavior src/stb/tests/../stb_image.h:2013:15 in
src/stb/tests/../stb_image.h:2115:4: runtime error: index -19895 out of bounds for type 'stbi_uc
[257]'
SUMMARY: UndefinedBehaviorSanitizer: undefined-behavior src/stb/tests/../stb_image.h:2115:4 in
=================================================================
==1324041==ERROR: AddressSanitizer: heap-use-after-free on address 0x629000007a21 at pc
0x00000051110c bp 0x7fffffffcf40 sp 0x7fffffffcf38
READ of size 1 at 0x629000007a21 thread T0
    #0 0x51110b in stbi__jpeg_huff_decode(stbi__jpeg*, stbi__huffman*) (stbi_read_fuzzer+0x51110b)
    #1 0x50fb8e in stbi__jpeg_decode_block_prog_ac(stbi__jpeg*, short*, stbi__huffman*, short*)
(stbi_read_fuzzer+0x50fb8e)
    #2 0x508bee in stbi__parse_entropy_coded_data(stbi__jpeg*) (stbi_read_fuzzer+0x508bee)
    #3 0x505f3a in stbi__decode_jpeg_image(stbi__jpeg*) (stbi_read_fuzzer+0x505f3a)
    #4 0x500cd3 in load_jpeg_image(stbi__jpeg*, int*, int*, int*, int) (stbi_read_fuzzer+0x500cd3)
    #5 0x4d5ae3 in stbi__jpeg_load(stbi__context*, int*, int*, int*, int, stbi__result_info*)
(stbi_read_fuzzer+0x4d5ae3)
    #6 0x4cf171 in stbi__load_main(stbi__context*, int*, int*, int*, int, stbi__result_info*, int)
(stbi_read_fuzzer+0x4cf171)
    #7 0x4c9642 in stbi__load_and_postprocess_8bit(stbi__context*, int*, int*, int*, int)
(stbi_read_fuzzer+0x4c9642)
    #8 0x4cadbc in stbi_load_from_memory (stbi_read_fuzzer+0x4cadbc)
    #9 0x4ced22 in LLVMFuzzerTestOneInput (stbi_read_fuzzer+0x4ced22)
    #10 0x531329 in fuzzer::Fuzzer::ExecuteCallback(unsigned char const*, unsigned long)
(stbi_read_fuzzer+0x531329)
    #11 0x51c239 in fuzzer::RunOneTest(fuzzer::Fuzzer*, char const*, unsigned long)
(stbi_read_fuzzer+0x51c239)
    #12 0x521142 in fuzzer::FuzzerDriver(int*, char***, int (*)(unsigned char const*, unsigned
long)) (stbi_read_fuzzer+0x521142)
    #13 0x51bfc2 in main (stbi_read_fuzzer+0x51bfc2)
    #14 0x7ffff7a6b0b2 in __libc_start_main /build/glibc-eX1tMB/glibc-2.31/csu/../csu/libc-
start.c:308:16
    #15 0x41e98d in _start (stbi_read_fuzzer+0x41e98d)

0x629000007a21 is located 10273 bytes inside of 18568-byte region [0x629000005200,0x629000009a88)
freed by thread T0 here:
    #0 0x496e4d in free (stbi_read_fuzzer+0x496e4d)
    #1 0x4d5a2d in stbi__jpeg_test(stbi__context*) (stbi_read_fuzzer+0x4d5a2d)
    #2 0x4cf142 in stbi__load_main(stbi__context*, int*, int*, int*, int, stbi__result_info*, int)
(stbi_read_fuzzer+0x4cf142)
    #3 0x4c9642 in stbi__load_and_postprocess_8bit(stbi__context*, int*, int*, int*, int)
(stbi_read_fuzzer+0x4c9642)
    #4 0x4cadbc in stbi_load_from_memory (stbi_read_fuzzer+0x4cadbc)
    #5 0x4ced22 in LLVMFuzzerTestOneInput (stbi_read_fuzzer+0x4ced22)
    #6 0x531329 in fuzzer::Fuzzer::ExecuteCallback(unsigned char const*, unsigned long)
(stbi_read_fuzzer+0x531329)

previously allocated by thread T0 here:
    #0 0x4970cd in malloc (stbi_read_fuzzer+0x4970cd)
    #1 0x4cd258 in stbi__malloc(unsigned long) (stbi_read_fuzzer+0x4cd258)
    #2 0x4d59e6 in stbi__jpeg_test(stbi__context*) (stbi_read_fuzzer+0x4d59e6)
    #3 0x4cf142 in stbi__load_main(stbi__context*, int*, int*, int*, int, stbi__result_info*, int)
```

```
(stbi_read_fuzzer+0x4cf142)
    #4 0x4c9642 in stbi__load_and_postprocess_8bit(stbi__context*, int*, int*, int*, int)
(stbi_read_fuzzer+0x4c9642)
    #5 0x4cadbc in stbi_load_from_memory (stbi_read_fuzzer+0x4cadbc)
    #6 0x4ced22 in LLVMFuzzerTestOneInput (stbi_read_fuzzer+0x4ced22)
    #7 0x531329 in fuzzer::Fuzzer::ExecuteCallback(unsigned char const*, unsigned long)
(stbi_read_fuzzer+0x531329)

SUMMARY: AddressSanitizer: heap-use-after-free (stbi_read_fuzzer+0x51110b) in
stbi__jpeg_huff_decode(stbi__jpeg*, stbi__huffman*)
Shadow bytes around the buggy address:
  0x0c527fff8ef0: fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd
  0x0c527fff8f00: fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd
  0x0c527fff8f10: fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd
  0x0c527fff8f20: fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd
  0x0c527fff8f30: fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd
=>0x0c527fff8f40: fd fd fd fd[fd]fd fd fd fd fd fd fd fd fd fd fd
  0x0c527fff8f50: fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd
  0x0c527fff8f60: fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd
  0x0c527fff8f70: fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd
  0x0c527fff8f80: fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd
  0x0c527fff8f90: fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd fd
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

==1324041==ABORTING
```

Crashing file

id:000233,sig:06,src:005305,time:31715435,op:havoc,rep:4,trial:1493419

pietroborrello mentioned this issue on Feb 17

UBSAN: index out of bounds #1291

Open

nothings added the 1 stb_image label on Feb 17

NeilBickford-NV mentioned this issue on Feb 23

Additional stb_image fixes for bugs from ossfuzz and issues 1289, 1291, 1292, and 1293 #1297

Open

NeilBickford-NV commented on Feb 23

I did a quick analysis of how this file seems to cause an invalid memory access, and I think I may have a fix in PR #1297.

Loading this file results in a couple of invalid memory accesses in `stbi__jpeg_huff_decode` -- in the following two lines, `code` and `values` are both arrays of length 256, but `c` has been set to -19895:

```
    STBI_ASSERT((((j->code_buffer) >> (32 - h->size[c])) & stbi__bmask[h->size[c]]) == h->code[c]);
...
    return h->values[c];
```

Digging a bit deeper reveals that this file results in running over the bounds of the arrays in stbi__huffman in two other places. The first is in the "DHT - define huffman table" block in `stbi__process_marker` : the file encodes a `sizes` array of `{0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 99, 99, 99, 99, 255}` , making `n` , the sum, 654 (larger than 256). When control reaches this loop later in the block:

```
            for (i=0; i < n; ++i)
               v[i] = stbi__get8(z->s);
```

v (either `z->huff_dc[th].values` or `z->huff_ac[th].values`) is overrun.

The second place is in `stbi__build_huffman` :

```
    int i,j,k=0;
    ...
    for (i=0; i < 16; ++i)
       for (j=0; j < count[i]; ++j)
          h->size[k++] = (stbi_uc) (i+1);
```

This is more or less the same as the first case: if the sum of `count` (which turns out to be the same as `sizes`) is greater than 256, k writes past the end of `h->size` .

slouken pushed a commit to libsdl-org/SDL_image that referenced this issue on May 28

`stb_image.h: imported three fuzz fixes by Neil Bickford from mainstream` ··· `04562ed`

Assignees

No one assigned

Labels

1 stb_image

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

3 participants