libjxl / **libjxl** Public

<> Code · Issues 139 · Pull requests 27 · Actions · Security 2 · Insights

New issue

Jump to bottom

Assertion failed in lib/jxl/image.cc jxl::PlaneBase::PlaneBase #422

Closed **aug5t7** opened this issue on Aug 8, 2021 · 2 comments

aug5t7 commented on Aug 8, 2021

Describe the bug
Assertion failed when compressing a gif with cjxl.

```
$ ./libjxl/build/tools/cjxl ./poc.gif /tmp/jxl.jxl
JPEG XL encoder v0.5.0 4122f3e [AVX2,SSE4,Scalar]
<path>/libjxl/lib/jxl/image.cc:94: JXL_CHECK: bytes_.get()
[1]    1073940 illegal hardware instruction  ./libjxl/build/tools/cjxl ./poc.gif /tmp/jxl.jxl
```

To Reproduce
Steps to reproduce the behavior:

```
$ CC=clang CXX=clang++ CFLAGS="-g" CXXFLAGS="-g" cmake -DCMAKE_BUILD_TYPE=Release -DBUILD_TESTING=OFF  ..
$ cmake --build . -- -j 8
$ tools/cjxl ./poc.gif /tmp/jxl.jxl
```

poc.gif

Expected behavior
No assertion failed.

Environment

- OS: `5.8.0-59-generic 20.04.1-Ubuntu`
- Compiler version: `clang version 7.0.1-12`
- CPU type: x86_64
- cjxl/djxl version string: cjxl v0.5.0 `4122f3e` [AVX2,SSE4,Scalar]

Additional context

It seems that the memory allocation size is too large causing the assertion failed.

libjxl/lib/jxl/image.cc
Lines 90 to 96 in `4122f3e`

```
90      // if nonzero, because "zero" bytes still have padding/bookkeeping overhead.
91      if (xsize != 0 && ysize != 0) {
92        bytes_per_row_ = BytesPerRow(xsize, sizeof_t);
93        bytes_ = AllocateArray(bytes_per_row_ * ysize);
94        JXL_CHECK(bytes_.get());
95        InitializePadding(sizeof_t, Padding::kRoundUp);
96      }
```

Some gdb information

```
gdb-peda$
[----------------------------------registers-----------------------------------]
RAX: 0x801
RBX: 0x3fffc
RCX: 0x40080
RDX: 0x0
RSI: 0x40080
RDI: 0x80
RBP: 0x7fffffffc490 --> 0x7fffffffc900 --> 0x7fffffffc940 --> 0x7fffffffca30 --> 0x7fffffffd350 --> 0x7fffffffe250 (--> ...)
RSP: 0x7fffffffc470 --> 0xffff
RIP: 0x55555573b858 (<jxl::PlaneBase::PlaneBase(unsigned long, unsigned long, unsigned long)+232>:      imul   r14,rcx)
R8 : 0xffffffffffffffe0
R9 : 0x6e4e ('Nn')
R10: 0x5555559fa8dc --> 0x0
R11: 0x5555559fa600 --> 0x206c786adc020000
R12: 0x4
R13: 0x5555559de970 --> 0x5555559daf10 --> 0x555555628e80 (<jxl::ColorEncoding::~ColorEncoding()>:      push   rbp)
R14: 0xffff
R15: 0x7fffffffc7a0 --> 0xffff0000ffff
EFLAGS: 0x246 (carry PARITY adjust ZERO sign trap INTERRUPT direction overflow)
[-------------------------------------code-------------------------------------]
   0x55555573b84b <jxl::PlaneBase::PlaneBase(unsigned long, unsigned long, unsigned long)+219>: test   rdx,rdx
   0x55555573b84e <jxl::PlaneBase::PlaneBase(unsigned long, unsigned long, unsigned long)+222>:
    jne    0x55555573b93c <jxl::PlaneBase::PlaneBase(unsigned long, unsigned long, unsigned long)+460>:     jne    0x55555573b93c <jxl::PlaneBase::PlaneBase(unsigned long, unsigned lo
   0x55555573b854 <jxl::PlaneBase::PlaneBase(unsigned long, unsigned long, unsigned long)+228>: mov    QWORD PTR [r15+0x10],rcx
=> 0x55555573b858 <jxl::PlaneBase::PlaneBase(unsigned long, unsigned long, unsigned long)+232>: imul   r14,rcx
   0x55555573b85c <jxl::PlaneBase::PlaneBase(unsigned long, unsigned long, unsigned long)+236>: call   0x5555556240b0 <jxl::CacheAligned::NextOffset()>
   0x55555573b861 <jxl::PlaneBase::PlaneBase(unsigned long, unsigned long, unsigned long)+241>: mov    rdi,r14
   0x55555573b864 <jxl::PlaneBase::PlaneBase(unsigned long, unsigned long, unsigned long)+244>: mov    rsi,rax
   0x55555573b867 <jxl::PlaneBase::PlaneBase(unsigned long, unsigned long, unsigned long)+247>:
    call   0x5555556240d0 <jxl::CacheAligned::Allocate(unsigned long, unsigned long)>:      call   0x5555556240d0 <jxl::CacheAligned::Allocate(unsigned long, unsigned long)>
[------------------------------------stack-------------------------------------]
0000| 0x7fffffffc470 --> 0xffff
0008| 0x7fffffffc478 --> 0x5555559f9580 --> 0xffff0000ffff
0016| 0x7fffffffc480 --> 0x7fffffffd368 --> 0xffffffffffffffff
0024| 0x7fffffffc488 --> 0xffff
0032| 0x7fffffffc490 --> 0x7fffffffc900 --> 0x7fffffffc940 --> 0x7fffffffca30 --> 0x7fffffffd350 --> 0x7fffffffe250 (--> ...)
0040| 0x7fffffffc498 --> 0x555555842c07 (<jxl::DecodeImageGIF(jxl::Span<unsigned char const>, jxl::ThreadPool*, jxl::CodecInOut*)+823>: lea    rdi,[rbp-0x140])
0048| 0x7fffffffc4a0 --> 0x0
0056| 0x7fffffffc4a8 --> 0x7fffffffc600 --> 0x0
[------------------------------------------------------------------------------]
Legend: code, data, rodata, value
93          bytes_ = AllocateArray(bytes_per_row_ * ysize);
gdb-peda$ p ysize
$1 = 0xffff
gdb-peda$ p bytes_per_row_
$2 = 0x40080
gdb-peda$ p  bytes_per_row_ * ysize
$3 = 0x4007bff80
```

deymo commented on Aug 9, 2021 Contributor

Hi, on that file I see that `gif->SWidth` and `gif->SHeight` are reported as 65535, and the only frame is reported as 10x10 at position 0,0.

I don't know what are we supposed to do differently. The way I understand this is that the virtual canvas is 65535 x 65535 white in color and there a single frame of 10x10 in the top left corner. `identify -verbose poc.gif` says `Page geometry: 65535x65535+0+0` but if I try to convert this to png it returns a 10x10 image.

Regarding the crash due to OOM, at the moment we have many places where we would crash if OOM instead of returning a failure.

jonsneyers commented on Nov 19, 2021 Member

Going to close this one, it's basically a special case of #762

jonsneyers closed this as completed on Nov 19, 2021

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

3 participants