New issue

Jump to bottom

splines: segfault due to out of bounds access of segment array #735

Closed lovell opened this issue on Oct 14, 2021 · 7 comments · Fixed by #757

Assignees

Labels fuzzerbug

lovell commented on Oct 14, 2021 — Contributor

Hello, this 161 byte JPEG-XL image, found via fuzz testing, causes a segfault during decoding (using the latest commit on the `main` branch).

https://github.com/libjxl/libjxl/files/7348994/fuzz39533.jxl.txt

```
==1086695==ERROR: UndefinedBehaviorSanitizer: SEGV on unknown address 0x7fb932913530 (pc 0x00000153da11 bp 0x7fb9343f5bb0 sp 0x7fb9343f5b50 T1086698)
==1086695==The signal is caused by a READ memory access.
#0 0x153da11 in jxl::N_AVX2::(anonymous namespace)::DrawSegment(jxl::SplineSegment const&, bool, unsigned long, long, long, float* restrict*) libjxl/lib/jxl/splines.cc:86:37
#1 0x153c987 in jxl::N_AVX2::(anonymous namespace)::DrawSegments(jxl::Image3<float>*, jxl::Rect const&, jxl::Rect const&, bool, jxl::SplineSegment const*, unsigned long const*,
unsigned long const*) libjxl/lib/jxl/splines.cc:151:5
#2 0x15353f5 in void jxl::Splines::Apply<true>(jxl::Image3<float>*, jxl::Rect const&, jxl::Rect const&) const libjxl/lib/jxl/splines.cc:556:5
#3 0x14de591 in jxl::FinalizeImageRect(jxl::Image3<float>*, jxl::Rect const&, std::__1::vector<std::__1::pair<jxl::Plane<float>*, jxl::Rect>,
std::__1::allocator<std::__1::pair<jxl::Plane<float>*, jxl::Rect> > > const&, jxl::PassesDecoderState*, unsigned long, jxl::ImageBundle*, jxl::Rect const&)
libjxl/lib/jxl/dec_reconstruct.cc:912:28
#4 0x1711d27 in jxl::PassesDecoderState::FinalizeGroup(unsigned long, unsigned long, jxl::Image3<float>*, jxl::ImageBundle*) libjxl/lib/jxl/dec_cache.cc:164:5
#5 0x141bed3 in jxl::N_AVX2::DecodeGroupImpl(jxl::GetBlock*, jxl::GroupDecCache*, jxl::PassesDecoderState*, unsigned long, unsigned long, jxl::ImageBundle*, jxl::DrawMode)
libjxl/lib/jxl/dec_group.cc:441:5
#6 0x142fc1e in jxl::DecodeGroup(jxl::BitReader* restrict*, unsigned long, unsigned long, jxl::PassesDecoderState*, jxl::GroupDecCache*, unsigned long, jxl::ImageBundle*, unsigned
long, bool, bool) libjxl/lib/jxl/dec_group.cc:754:3
#7 0x14048da in jxl::FrameDecoder::ProcessACGroup(unsigned long, jxl::BitReader* restrict*, unsigned long, unsigned long, bool, bool) libjxl/lib/jxl/dec_frame.cc:579:5
#8 0x140b6ef in operator() libjxl/lib/jxl/dec_frame.cc:744:16
```

It looks like, when drawing spline segments, `segment_y_start` for this image contains 253 entries but `image_rect.y0()` can return higher values for `y` that result in `DrawSegment()` reading beyond the end of this.

libjxl/lib/jxl/splines.cc
Lines 149 to 153 in 795ba9c

```
149     size_t y = image_rect.y0();
150     for (size_t i = segment_y_start[y]; i < segment_y_start[y + 1]; i++) {
151       DrawSegment(segments[segment_indices[i]], add, y, image_rect.x0(),
152                   image_rect.x0() + image_rect.xsize(), rows);
153     }
```

The following patch to `Apply()` demonstrates a possible guard to prevent the segfault, but there's almost certainly a better way to fix this.

```
--- a/lib/jxl/splines.cc
+++ b/lib/jxl/splines.cc
@@ -552,6 +552,7 @@ template <bool add>
 void Splines::Apply(Image3F* const opsin, const Rect& opsin_rect,
                     const Rect& image_rect) const {
   if (segments_.empty()) return;
+  if (image_rect.y0() >= segment_y_start_.size()) return;
   for (size_t iy = 0; iy < image_rect.ysize(); iy++) {
     HWY_DYNAMIC_DISPATCH(DrawSegments)
     (opsin, opsin_rect.Line(iy), image_rect.Line(iy), add, segments_.data(),
```

jonsneyers added the fuzzerbug label on Oct 14, 2021

jonsneyers commented on Oct 14, 2021 — Member

Good catch! Spline rendering was recently optimized, so this is probably a result of that.

@veluca93 is there an earlier point where this can be prevented? Otherwise the suggested fix looks good enough, at least as a quick patch. Should probably backport it to the 0.6 branch too, decoder segfault is a rather severe bug after all.

veluca93 commented on Oct 14, 2021 — Member

image_rect.y0() really ought to be smaller than segment_y_start_ in all cases, it is probably better to fix that... **@sboukortt** too

veluca93 assigned **veluca93** and **sboukortt** on Oct 14, 2021

sboukortt commented on Oct 19, 2021 — Member

Not yet fully sure what is happening but `Splines::InitializeDrawCache` is first called with an image size of 46×252, and then `AddTo` is called on 48×1 rects (`x0` = 8 for `opsin_rect` , 0 for `image_rect`) of varying `y0` on a 304×292 opsin image. For what it's worth, `jxlinfo` indicates that it's a 37×37 JXL file.

sboukortt added a commit to sboukortt/libjxl that referenced this issue on Oct 20, 2021

`Splines: initialize the draw cache with the correct dimensions` ··· `bb65999`

sboukortt mentioned this issue on Oct 20, 2021

Splines: initialize the draw cache with the correct dimensions #757

Merged

sboukortt closed this as completed in #757 on Oct 20, 2021

---

sboukortt added a commit that referenced this issue on Oct 20, 2021

`Splines: initialize the draw cache with the correct dimensions` ··· ✓ `0eff04c`

deymo pushed a commit to deymo/libjxl that referenced this issue on Oct 27, 2021

`Splines: initialize the draw cache with the correct dimensions` ··· `d39d1f2`

deymo pushed a commit that referenced this issue on Oct 27, 2021

`Splines: initialize the draw cache with the correct dimensions` ··· `b0b3969`

deymo commented on Oct 29, 2021 Contributor

Note: This bug got assigned CVE-2021-22563

deymo commented on Nov 1, 2021 Contributor

@lovell Please let me know if you would like to be credited in the CVE description and how (name, company affiliation, etc).

lovell commented on Nov 1, 2021 Contributor Author

@deymo I'm happy for my name to appear but please ensure libvips is credited too as its fuzzers found this. If there's a bounty, please donate this to https://opencollective.com/libvips

deymo commented on Nov 1, 2021 Contributor

Thanks. We don't have bug bounty for libjxl.

👍 1

Assignees

veluca93

sboukortt

Labels

fuzzerbug

Projects

None yet

Milestone

No milestone

Development

Successfully merging a pull request may close this issue.

Splines: initialize the draw cache with the correct dimensions
sboukortt/libjxl

5 participants