Closed Issue created 6 months ago by **Adam Doupe**

avidemux: heap buffer overwrite in gst_avi_demux_invert/swap_line

Describe the vulnerability

heap-based buffer overflow in avidemux element, specifically in the functions `gst_avi_demux_invert` / `swap_line` .

The root cause vulnerability is that these values come from the `.avi` file:

```
h = stream->strf.vids->height;
w = stream->strf.vids->width;
bpp = stream->strf.vids->bit_cnt ? stream->strf.vids->bit_cnt : 8;
stride = GST_ROUND_UP_4 (w * (bpp / 8));
```

https://gitlab.freedesktop.org/gstreamer/gstreamer/-/blob/main/subprojects/gst-plugins-good/gst/avi/gstavidemux.c#L5004

And the size of the buffer is `malloc` d based on that:

```
tmp = g_malloc (stride);
```

https://gitlab.freedesktop.org/gstreamer/gstreamer/-/blob/main/subprojects/gst-plugins-good/gst/avi/gstavidemux.c#L5015

There is a size check, however the vulnerability is that by choosing `stride` and `h` correctly then `stride * h` will overflow and wrap around, bypassing the size check

```
if (map.size < (stride * h)) {
  GST_WARNING ("Buffer is smaller than reported Width x Height x Depth");
  gst_buffer_unmap (buf, &map);
  return buf;
}
```

https://gitlab.freedesktop.org/gstreamer/gstreamer/-/blob/main/subprojects/gst-plugins-good/gst/avi/gstavidemux.c#L5009

Thus causing a heap overwrite here:

```
for (y = 0; y < h / 2; y++) {
  swap_line (map.data + stride * y, map.data + stride * (h - 1 - y), tmp,
      stride);
}
```

https://gitlab.freedesktop.org/gstreamer/gstreamer/-/blob/main/subprojects/gst-plugins-good/gst/avi/gstavidemux.c#L5017

Expected Behavior

Not segfault.

Observed Behavior

segfault.

Setup

- **Operating System:** Ubuntu 20.04.4 LTS
- **Device:** Computer
- **GStreamer Version:** tested on 1.16.2, but vulnerability present on `main`

Steps to reproduce the bug

1. Download the attached file: crash-gst.avi

2. Use gat-play-1.0 to run the file:

```
gst-play-1.0 ./crash-gst.avi
```

How reproducible is the bug?

Always

Impact

Likely code execution through heap manipulation, although I only have this crashing POC.

Additional Information

I'd like to request a CVE as part of this process.

Thank you!

Drag your designs here or click to upload.

Tasks 0

No tasks are currently assigned. Use tasks to break down this issue into smaller parts.

Linked items 0

Related merge requests 1

avidemux: Fix integer overflow resulting in heap corruption in DIB buffer inversion code

!2608 1.21.1

When this merge request is accepted, this issue will be closed automatically.

Activity

Sebastian Dröge @slomo · 6 months ago Owner

0001-avidemux-Fix-integer-overflow-resulting-in-heap-corr.patch

This should fix it but I'm not 100% certain whether the multiplications/additions in the various places are actually doing the right things with the C integer promotion rules.

Thanks for analyzing and reporting this one too.

Edited by Sebastian Dröge 6 months ago

Tim-Philipp Müller added Security label 6 months ago

Tim-Philipp Müller @tpm · 6 months ago Owner

We'll probably merge these patches closer to the next stable bug-fix release %1.20.3 in ca. 2 weeks time or so.

It would be great if you could give us some indication whether you expect there to be more issues forthcoming in the near future (e.g. if you're running a lab at the moment that's still actively identifying problems), so we don't do a new bug-fix release and then 10 new issues come in the day after :)

Adam Doupe @adamdoupe · 6 months ago Author

@slomo I verified that this patch fixes the POC.

I also ran a simplified version of these checks through angr's symbolic execution engine, and the only case that could potentially cause a problem is if `map.size == 0`.

But I don't think it's possible for a GstBuffer to have size 0.

Sebastian Dröge @slomo · 6 months ago Owner

It actually is possible, and `map.data` would then be `NULL`. How would that cause problems? To not get caught by the old check, either `stride` or `h` would have to be 0. If `h` is 0 then the loop would do nothing, if `stride` is 0 then `swap_line()` would call `memcpy()` with `NULL` pointers and 0 counts a couple of times. And that's technically UB.

Is that what you mean? I guess adding some checks for 0 `stride`, `w`, `h` and buffer size would be good to add in any case.

Adam Doupe @adamdoupe · 6 months ago Author

Ah, I forgot about `map.data` being `NULL`, and yes `stride` (or rather `bpp`) would also need to be `0` for that to occur.

But like you said this might be a `NULL` pointer dereference if those conditions are met.

Sebastian Dröge @slomo · 6 months ago Owner

Updated (also in the original comment): 0001-avidemux-Fix-integer-overflow-resulting-in-heap-corr.patch

It's not going to dereference `NULL` anywhere. There's always going to be one `NULL` pointer in the `memcpy()` together with the count being 0, but that's nonetheless UB according to C99 at least. It's probably not going to cause a problem in practice though but better safe than sorry :)

Adam Doupe @adamdoupe · 6 months ago Author

Red Hat assigned "heap overwrite in avi demuxing" CVE-2022-1921 to this issue.

Tim-Philipp Müller @tpm · 5 months ago Owner

Patch looks OK to me.

Tim-Philipp Müller changed milestone to %1.20.3 5 months ago

Sebastian Dröge mentioned in commit `tpm/gstreamer@7dfd1ecf` 5 months ago

Tim-Philipp Müller mentioned in merge request !2608 (merged) 5 months ago

Sebastian Dröge mentioned in commit `tpm/gstreamer@934ba27f` 5 months ago

Sebastian Dröge mentioned in commit `tpm/gstreamer@0d9ce6c9` 5 months ago

Sebastian Dröge mentioned in commit `tpm/gstreamer@f503caad` 5 months ago

Sebastian Dröge closed via commit `f503caad` 5 months ago

Tim-Philipp Müller made the issue visible to everyone 5 months ago

Sebastian Dröge mentioned in commit `wtaymans/gstreamer@448f6e3b` 2 weeks ago

Please register or sign in to reply