fix: resolve use-after-free on png_ptr->trans_alpha #824

Closed **Oblivionsage** wants to merge 3 commits into `pnggroup:libpng16` from `Oblivionsage:fix/trans-alpha-uaf`

Oblivionsage commented 3 weeks ago — Contributor

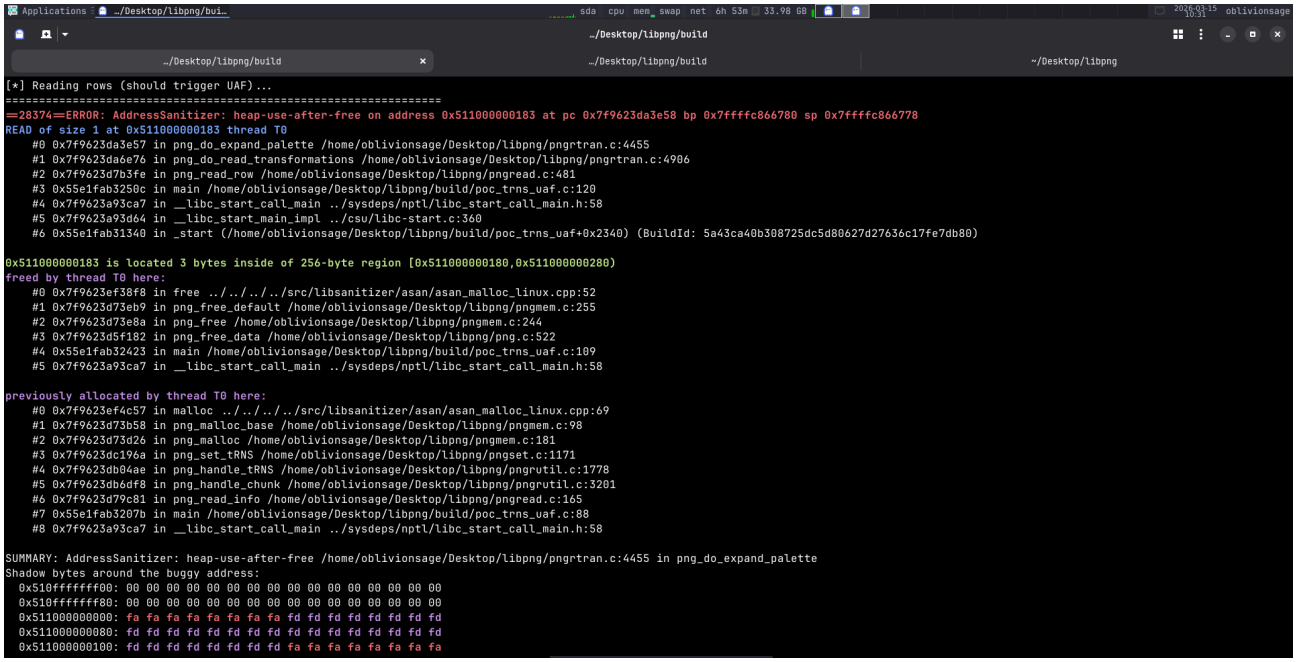
`png_set_tRNS()` aliases `png_ptr->trans_alpha` directly to `info_ptr->trans_alpha` same heap buffer, two owners. Calling `png_free_data(PNG_FREE_TRNS)` frees the buffer through `info_ptr` but leaves `png_ptr->trans_alpha` dangling. Next row read hits `png_do_expand_palette()` and dereferences freed memory.

There was already a TODO about this in `png_handle_tRNS`:

```
/* TODO: this is a horrible side effect in the palette case because the
 * png_struct ends up with a pointer to the tRNS buffer owned by the
 * png_info.  Fix this.
 */
```

The fix gives `png_struct` its own copy of the buffer. Both `libpng16` and `libpng18` are affected.

ASan trace and screenshot below



▶ ASan trace (click to expand)

`fix: resolve use-after-free on png_ptr->trans_alpha (CWE-416)` ··· ✓ `a178f46`

Oblivionsage commented 3 weeks ago — Contributor, Author

@**ctruta** just to clarify , the fix only decouples the pointer ownership between `png_struct` and `png_info`, nothing else changes in the read/write pipeline. Both structs get their own `trans_alpha` buffer now instead of sharing one. All 34 tests pass clean under ASan.

I have a standalone PoC that triggers the UAF if you want to reproduce it, happy to attach

ctruta commented 3 weeks ago — Member

Hello @**Oblivionsage**, and thank you very much.

I confirm both your discovery and your fix. The function `png_set_tRNS` aliases `png_ptr->trans_alpha` directly to `info_ptr->trans_alpha`. Calling `png_free_data(PNG_FREE_TRNS)` frees the buffer through `info_ptr`, leaving `png_ptr->trans_alpha` dangling. Subsequent row transforms (e.g. `png_do_expand_palette`) dereference freed memory. This is CWE-416 (use-after-free) indeed.

The fix is correct in principle: give `png_struct` its own independent copy of the `trans_alpha` buffer, breaking the aliasing. The TODO in `png_handle_tRNS` identified this problem explicitly. The write-side cleanup in `png_write_destroy` and the `else` branch that NULLs out `png_ptr->trans_alpha` when `num_trans` is out of range are both good additions.

I will apply the fix to the branch `libpng16`, then I'll merge that into `libpng18` after the release, as usual.

Before merging

Please add your full name to `AUTHORS` if it's not already there. (I think it isn't.)

Following up: uninitialized tail bytes

Both `info_ptr->trans_alpha` and `png_ptr->trans_alpha` are allocated as `PNG_MAX_PALETTE_LENGTH` (i.e. 256 bytes), but only `num_trans` bytes are copied. The remaining bytes are uninitialized. This is a pre-existing condition, not introduced by this PR, but worth fixing while we're here. Entries beyond `num_trans` are conventionally treated as fully opaque (i.e. 0xff), so `memset(buf, 0xff, PNG_MAX_PALETTE_LENGTH)` before `memcpy` would be the most defensive choice. This applies to both allocations.

We need to take care of this in a follow-up commit. I can do it, or you can do it. Whoever gets to it first ;-)

Attaching a PoC

Yes, please do attach your standalone reproducer. It will be useful for regression testing.

Requesting a CVE

This is clearly a security defect, reachable via a publicly-documented code path. If you wish to apply for a CVE ID, go for it. Otherwise, I'll do it, too.

Oblivionsage added 2 commits 3 weeks ago

`Add Halil Oktay to AUTHORS` `99d5330`

`Initialize uninitialized tail bytes in trans_alpha buffers` ··· ✓ `3659e3b`

Oblivionsage commented 3 weeks ago — Contributor · Author

Thanks **@ctruta** , really appreciate the thorough review

I've pushed two follow-up commits:

- Added my name to `AUTHORS`
- `memset(0xff)` before `memcpy` on both `trans_alpha` allocations, as you suggested

PoC is here:

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <png.h>

static void make_png(const char *path)
{
    FILE *fp = fopen(path, "wb");
    png_structp wp = png_create_write_struct(PNG_LIBPNG_VER_STRING, NULL, NULL, NULL);
    png_infop wi = png_create_info_struct(wp);
    setjmp(png_jmpbuf(wp));
    png_init_io(wp, fp);
    png_set_IHDR(wp, wi, 4, 2, 8, PNG_COLOR_TYPE_PALETTE,
                 PNG_INTERLACE_NONE, PNG_COMPRESSION_TYPE_DEFAULT, PNG_FILTER_TYPE_DEFAU
    png_color pal[4] = {{255,0,0},{0,255,0},{0,0,255},{128,128,128}};
    png_set_PLTE(wp, wi, pal, 4);
    png_byte ta[4] = {128, 255, 64, 255};
    png_set_tRNS(wp, wi, ta, 4, NULL);
    png_write_info(wp, wi);
    png_byte r0[4] = {0,1,2,3}, r1[4] = {2,0,3,1};
    png_write_row(wp, r0);
    png_write_row(wp, r1);
    png_write_end(wp, NULL);
    png_destroy_write_struct(&wp, &wi);
    fclose(fp);
}

int main(void)
{
    const char *f = "/tmp/poc_trns.png";
    make_png(f);

    FILE *fp = fopen(f, "rb");
    png_structp p = png_create_read_struct(PNG_LIBPNG_VER_STRING, NULL, NULL, NULL);
    png_infop i = png_create_info_struct(p);
    setjmp(png_jmpbuf(p));
    png_init_io(p, fp);
    png_read_info(p, i);
    png_set_tRNS_to_alpha(p);
    png_read_update_info(p, i);

    /* free trans_alpha through info_ptr, png_ptr->trans_alpha dangles */
    png_free_data(p, i, PNG_FREE_TRNS, 0);

    /* triggers heap-use-after-free in png_do_expand_palette */
    png_bytep row = malloc(png_get_rowbytes(p, i));
    png_read_row(p, row, NULL);

    free(row);
    png_destroy_read_struct(&p, &i, NULL);
    fclose(fp);
```

```
    return 0;
}
```

For the CVE , it would be great if you could handle it, MITRE tends to take a while from my end. A GitHub Security Advisory might be faster if that works for you. Happy to help with whatever's needed either way.

ctruta approved these changes 2 weeks ago

View reviewed changes

ctruta commented 2 weeks ago — Member

@Oblivionsage wrote:

> For the CVE , it would be great if you could handle it, MITRE tends to take a while from my end. A GitHub Security Advisory might be faster if that works for you. Happy to help with whatever's needed either way.

Ok, will do.

❤️ 1

Oblivionsage commented 2 weeks ago — Contributor, Author

> **@Oblivionsage** wrote:
>
> > For the CVE , it would be great if you could handle it, MITRE tends to take a while from my end. A GitHub Security Advisory might be faster if that works for you. Happy to help with whatever's needed either way.
>
> Ok, will do.

appreciate it! feel free to tag me as reporter on the advisory whenever you get to it

👍 1

ctruta commented 2 weeks ago — Member

Integrated in trunk. Thank you again, **@Oblivionsage** :-)

Oblivionsage commented 2 weeks ago — Contributor, Author

> Integrated in trunk. Thank you again, **@Oblivionsage** :-)

thanks for the quick turnaround! let me know when the advisory is up

ctruta commented 2 weeks ago — Member

@Oblivionsage, here's the update on the advisory and a related fix.

We received an independent report from **@shimarda** describing the same `trans_alpha` use-after-free you discovered. Their analysis confirms the vulnerability and includes exploitation PoCs. I will publish a single advisory covering the `trans_alpha` fix, crediting both of you.

During the review of your PR, we identified the same pointer-aliasing pattern in `png_set_PLTE`: `info_ptr->palette` and `png_ptr->palette` shared a single buffer, with the same use-after-free consequences. This will be fixed in a follow-up commit, using the same approach as your `trans_alpha` fix (independent allocation, unconditional free in `png_read_destroy`). The palette fix also addresses a latent issue in `png_set_quantize`, which aliased a caller-provided pointer into `png_ptr->palette`.

Both fixes will ship in libpng 1.6.56. Thank you again for finding and fixing this bug.

ctruta closed this 2 weeks ago

Oblivionsage commented 2 weeks ago — Contributor, Author

good to know, thanks for the update **@ctruta** nice catch on the png_set_PLTE variant too , makes sense it would have the same pattern. glad the approach generalized well. btw would be great to be included on the advisory as well since we reported independently , happy to help with any details you need for it

ctruta commented 2 weeks ago • edited — Member

> btw would be great to be included on the advisory as well since we reported independently

I did that already, **@Oblivionsage**, and please check your GitHub inbox because you should receive a notification with an invitation to accept your status as a reporter. I cannot paste it here because it's a secret URL (for now --- until I make it public, which will be at the time of the public release), but you should be able to access it from that invitation.

Oblivionsage commented 2 weeks ago — Contributor, Author

just checked, **@ctruta** nothing in my notifications or inbox yet. could you double check the invite went to **@Oblivionsage**? might have gone to the other reporter by mistake

ctruta commented 2 weeks ago — Member

Check your inbox again. If that still doesn't work, let's wait a few hours. We'll retry tomorrow.

❤️ 1

Oblivionsage commented 2 weeks ago — Contributor, Author

Hey **@ctruta**, still no luck unfortunately :(checked notifications, email, spam, even tried opening the advisory link directly but got a 404. Not sure what's going on on github's end. maybe try removing and re-adding me? Or if there's another way to sort this out happy to try whatever works

ctruta commented 2 weeks ago — Member

@Oblivionsage, I re-checked the reference to you in the draft advisory, and I updated it, too. Not only that, but also the libpng code. Two additional fixes were necessary, and you can see these fixes in commits `7ea9eea` and `c1b0318`. NOW the bug is fixed. (Or so I think.)

If you still don't see any invitation to accept your contribution as you co-reported a vulnerability (**and** you did the fixing, which should make you eligible for bounty programs such as the Google Patch Rewards), then the next thing to do is wait until I make the next libpng release v1.6.56, which should be early next week. At that time, I will make the security advisory public, and you should be able to access it, update it, and even file a ticket to the GitHub admins in the event that you are STILL unable to formally accept the acknowledgement.

👍 1

ctruta commented 2 weeks ago • edited Member

> maybe try removing and re-adding me? Or if there's another way to sort this out happy to try whatever works

I just tried this, too. If it still doesn't work, just give it a few more days until I make it public.

Alternatively, send an email directly to my email address (which you can see in the git log) and I will respond privately with links that I cannot publish here at this time.

👍 1

ctruta commented last week Member

The new libpng release 1.6.56 is published, **@Oblivionsage**, and so are the two CVEs fixed in this release -- one of which is the one that you discovered and fixed right here :-)

GHSA-m4pc-p4q3-4c7j

❤️ 1

Oblivionsage commented last week Contributor Author

> The new libpng release 1.6.56 is published, **@Oblivionsage**, and so are the two CVEs fixed in this release -- one of which is the one that you discovered and fixed right here :-)
>
> GHSA-m4pc-p4q3-4c7j

awesome, just saw it , thanks for sorting everything out. really appreciate it

LEEKIYOON-SEC mentioned this pull request last week

[Argus] CVE-2026-33416: LIBPNG의 `png_set_tRNS` 및 `png_set_PLTE` 함수에서 포인터 별칭으로 인한 use-after-free 취약점 LEEKIYOON-SEC/Argus-AI-Threat-Intelligence#784

Open

This was referenced last week

Migrate to php mkaraki/cbzViewer#52

Merged

Add pdf renderer mkaraki/cbzViewer#53

Draft

Bump the npm-all group in /frontend with 5 updates mkaraki/cbzViewer#48

Merged

Fix there are no current dir display on list view. mkaraki/cbzViewer#54

Merged

Fix cypress test mkaraki/cbzViewer#55

Merged

Update README with quality badges mkaraki/cbzViewer#56

Merged

Remove duplicated codes mkaraki/cbzViewer#57

Merged

63 hidden items

Load more…

This was referenced last week

Add double page view mkaraki/cbzViewer#26

Merged

Increase PHP memory limit to 512M via FrankenPHP Caddyfile mkaraki/cbzViewer#58

Merged

Bump cypress-io/github-action from 7.1.8 to 7.1.9 in the actions-all group mkaraki/cbzViewer#59

Merged

Reviewers

 **ctruta** ✓

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

None yet

2 participants