10 High memory usage for generating preview of broken image

Share:

TIMELINE

fancycode submitted a report to Nextcloud. Jul 14th (about 1 year ago)

When the attached file is uploaded and a preview is generated (e.g. in the folder overview of the files app), the PHP process allocates a very large amount of memory (on my machine it was shortly around 5 GByte) and CPU.

Tested with latest master (1366b35081f1d92429787696f4175c19a602858a) on Ubuntu 20.04 (php7.4-fpm). Option "memory_limit" is set to 512M.

Impact

An attacker can cause a denial of service by uploading lots of such files which will cause the server to allocate too much memory / CPU.

1 attachment:
F1376155: bad-header.jpg

OT: posted a comment. Jul 14th (about 1 year ago)

Thanks a lot for reporting this potential issue back to us!

Our security team will take a look at this issue as soon as possible. We will reply to your report within 72 hours, usually much faster. For obvious reasons we'd like to ask you to not disclose this issue to any other party.

lukasreschkenc posted a comment. Jul 14th (about 1 year ago)

@fancycode Is there any more details on how you generated the image? - Was it via fuzzing etc?

Also do you know if this is a bug inside our code or a PHP module? (if not sure, I can also try to debug this :))

Sorry for the late reply. I had that image on my machine from some other project / tests and figured I try and see what happens if I upload it to Nextcloud. It's basically an "empty" jpeg file with header information that don't match the actual image. Here is a rundown of the contents:

Code 1.38 KiB Wrap lines Copy Download

```
00000000  ff d8 ff db 00 43 00 41  41 41 41 41 41 41 41 41  |.....C.AAAAAAAAA|
00000010  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41  |AAAAAAAAAAAAAAAA|
00000020  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41  |AAAAAAAAAAAAAAAA|
00000030  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41  |AAAAAAAAAAAAAAAA|
00000040  41 41 41 41 41 41 41 ff  c0 00 11 08 63 9c 63 9c  |AAAAAAA.....c.c.|
00000050  03 00 22 00 01 22 01 02  22 00 ff da 00 08 01 00  |..".."..".......|
00000060  00 00 3f 00 ff d9 0a                              |..?....|
00000067

0xff 0xd8 -> soi (start of image)
0xff 0xdb -> dqt (define quantization table)
2 bytes length (0x43)
1 byte index (0x00)
64 bytes table (0x41 ... 0x41)

0xff 0xc0 -> sof0 (baseline dct)
2 bytes length (0x11)
1 byte data precision (0x08)
2 bytes height (0x63 0x9c -> 25500)
2 bytes width (0x63 0x9c -> 25500)
1 byte number of components (0x03)
1 byte id first component (0x00)
1 byte sample factor (0x22)
1 byte number quant. table (0x00)
1 byte id second component (0x01)
1 byte sample factor (0x22)
1 byte number quant. table (0x01)
1 byte id third component (0x02)
1 byte sample factor (0x22)
1 byte number quant. table (0x00)

```

```
1 byte id of component (0x00)
1 byte table index (0x00)
3 byte ignore data (0x00 0x3f 0x00)

0xff 0xd9 -> eoi (end of image)

0x0a (line break)
```

Unfortunately I don't know how to debug such lowlevel issues with PHP, so any guidance would be great (or if you could take a look yourself).

As an intermediate fix, you could parse the JPEG header and reject decoding it if the size from the header is too large (through configurable limits). Opening a 25500 x 25500 pixels image on the server is probably never a good idea (`25500 * 25500 * 4 = 2.6 GByte`, assuming an internal 32 bits per pixel).

fancycode posted a comment. Aug 10th (about 1 year ago)

@lukasreschkenc Friendly ping ;-) Anything I can do to help getting this fixed?

fancycode posted a comment. Sep 20th (about 1 year ago)

Hey, what's the status on this?

lukasreschkenc posted a comment. Sep 22nd (about 1 year ago)

Apologies in the delay following up here, there were a few other tickets that we had to triage first :-)

I took a look into the PHP function that causes the issue and it seems to be `imagecreatefromjpeg`, a simple test file could look like the following:

Code 57 Bytes Wrap lines Copy Download

```php
<?php
imagecreatefromjpeg(__DIR__ . '/bad-header.jpg');
```

And looking at the memory in the activity monitor seems to consume quite some memory:

Image F1457254: Screenshot_2021-09-22_at_12.27.04.png 81.61 KiB

Zoom in Zoom out Copy Download

My assumption is that this is an issue in PHP-GD as this is what is being invoked at https://github.com/php/php-src/blob/411da30031feba8775606a156d43f8ac0801a041/ext/gd/gd.c#L1765-L1768 and https://github.com/php/php-src/blob/411da30031feba8775606a156d43f8ac0801a041/ext/gd/gd.c#L1632.

I am still not sure if this is expected by PHP-GD or not, such as per https://bugs.php.net/bug.php?id=71093, https://bugs.php.net/bug.php?id=43675&edit=1 and other issues. If yes, I wonder what would be the best pragmatic way in PHP to get the size considering EXIF information may not be there.

1 attachment:

F1457254: Screenshot_2021-09-22_at_12.27.04.png

fancycode posted a comment. Sep 22nd (about 1 year ago)

> I am still not sure if this is expected by PHP-GD or not [...].

Well, the file specifies a size of 25500 x 25500 pixels, so it's expected to allocate lots of memory to hold the image data.

However the configured memory limit of PHP is not evaluated for this (was set to 512MB in my tests). I would suggest to parse the image header to find out the dimensions without decoding the image. Then, estimate the required memory and don't decode the image if more than `memory_limit` would be allocated to hold the image data. Another option would be to define maximum width/height of images for which to generate previews. The important thing is to get the width/height of the source image without fully decoding it.

EXIF data is optional / unreliable and most likely will not be used during decoding by PHP anyway.

Btw, the same most likely also applies to other image formats Nextcloud can generate previews for.

fancycode posted a comment. Sep 22nd (about 1 year ago)

Just read about the handy function `getimagesize` (or `getimagesizefromstring`).

The check could be something like this:

```
$width = $size[0];
$height = $size[1];
if ($width * $height * 4 > $memory_limit) {
        // Decoding image would require too much memory.
        return;
}

// Create preview.
$image = imagecreatefromjpeg(__DIR__ . '/bad-header.jpg');
...
```

fancycode posted a comment. Oct 21st (about 1 year ago)

Just checking back - any progress?

fancycode posted a comment. Dec 16th (12 months ago)

I finally implemented the check myself due to the lack of feedback here: https://github.com/nextcloud/server/pull/30291

nickvergessen Nextcloud staff posted a comment. Jan 3rd (11 months ago)

Thanks for the patch, I will ping some people to review it

fancycode posted a comment. Updated Mar 7th (9 months ago)

Thanks for getting releases out that contain the fix! Will this get a CVE assigned and be disclosed?

nickvergessen Nextcloud staff closed the report and changed the status to **Resolved**. Mar 7th (9 months ago)

Thanks a lot for your report again. This has been resolved in our latest maintenance releases and we're working on the advisories at the moment.

If you have a GitHub account please let us know the username, and we will associate it with the advisory.

fancycode posted a comment. Mar 8th (9 months ago)

Great, feel free to reference my @fancycode account (same username everywhere).

Nextcloud rewarded fancycode with a **$100** bounty. Mar 8th (9 months ago)

nickvergessen Nextcloud staff posted a comment. Mar 8th (9 months ago)
Advisory will be published at https://github.com/nextcloud/security-advisories/security/advisories/GHSA-jf3h-xf4q-mh89

nickvergessen Nextcloud staff updated CVE reference to **CVE-2022-24741**. Mar 9th (9 months ago)

nickvergessen Nextcloud staff requested to disclose this report. Mar 9th (9 months ago)

fancycode agreed to disclose this report. Mar 9th (9 months ago)

This report has been disclosed. Mar 9th (9 months ago)