heap-buffer-overflow Read in Exiv2::Internal::CrwMap::encode #1530

Closed **henices** opened this issue on Apr 8, 2021 · 15 comments · Fixed by #1539

Labels security (crash)

Milestone v0.27.4

henices commented on Apr 8, 2021

VERSION
exiv 2 0.27.4.1
https://github.com/Exiv2/exiv2/tree/0.27-maintenance

REPRODUCE

Compile exiv2 with asan:

```
CC=clang CXX=clang++ cmake .. -DCMAKE_BUILD_TYPE=Release -DCMAKE_CXX_FLAGS="-fsanitize=address" \
-DCMAKE_C_FLAGS="-fsanitize=address" -DCMAKE_EXE_LINKER_FLAGS="-fsanitize=address" \
-DCMAKE_MODULE_LINKER_FLAGS="-fsanitize=address"
```

Dowload testcases:

https://github.com/henices/pocs/raw/master/tests_1bd0a5f4935b053f33ac00f931dde1f47a043487
https://github.com/henices/pocs/raw/master/tests_1bd0a5f4935b053f33ac00f931dde1f47a043487.exv

Run command: `exiv2 in tests_1bd0a5f4935b053f33ac00f931dde1f47a043487`

```
=================================================================
==119384==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x62600000585e at pc 0x0000004c4d0a bp 0x7ffef1036370 sp 0x7ffef1035b20
READ of size 4294967293 at 0x62600000585e thread T0
    #0 0x4c4d09 in __asan_memcpy (/home/henices/tests/exiv2/build_asan/bin/exiv2+0x4c4d09)
    #1 0x7f40c9907d88 in Exiv2::Internal::CrwMap::encode0x1810(Exiv2::Image const&, Exiv2::Internal::CrwMapping const*, Exiv2::Internal::CiffHeader*)
(/home/henices/tests/exiv2/build_asan/lib/libexiv2.so.27+0x4c8d88)
    #2 0x7f40c9911007 in Exiv2::Internal::CrwMap::encode(Exiv2::Internal::CiffHeader*, Exiv2::Image const&) (/home/henices/tests/exiv2/build_asan/lib/libexiv2.so.27+0x4d2007)
    #3 0x7f40c9769376 in Exiv2::CrwImage::writeMetadata() (/home/henices/tests/exiv2/build_asan/lib/libexiv2.so.27+0x32a376)
    #4 0x541653 in (anonymous namespace)::metacopy(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&, std::__cxx11::basic_string<char,
std::char_traits<char>, std::allocator<char> > const&, int, bool) (/home/henices/tests/exiv2/build_asan/bin/exiv2+0x541653)
    #5 0x545049 in Action::Insert::run(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&)
(/home/henices/tests/exiv2/build_asan/bin/exiv2+0x545049)
    #6 0x4fddf3 in main (/home/henices/tests/exiv2/build_asan/bin/exiv2+0x4fddf3)
    #7 0x7f40c8ede1e1 in __libc_start_main /usr/src/debug/glibc-2.32-37-g760e1d2878/csu/../csu/libc-start.c:314:16
    #8 0x4224cd in _start (/home/henices/tests/exiv2/build_asan/bin/exiv2+0x4224cd)

0x62600000585e is located 0 bytes to the right of 10078-byte region [0x626000003100,0x62600000585e)
allocated by thread T0 here:
    #0 0x4fad47 in operator new[](unsigned long) (/home/henices/tests/exiv2/build_asan/bin/exiv2+0x4fad47)
    #1 0x7f40c98688b1 in Exiv2::DataBuf::alloc(long) (/home/henices/tests/exiv2/build_asan/lib/libexiv2.so.27+0x4298b1)
    #2 0x541653 in (anonymous namespace)::metacopy(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&, std::__cxx11::basic_string<char,
std::char_traits<char>, std::allocator<char> > const&, int, bool) (/home/henices/tests/exiv2/build_asan/bin/exiv2+0x541653)
    #3 0x545049 in Action::Insert::run(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&)
(/home/henices/tests/exiv2/build_asan/bin/exiv2+0x545049)
    #4 0x4fddf3 in main (/home/henices/tests/exiv2/build_asan/bin/exiv2+0x4fddf3)
    #5 0x7f40c8ede1e1 in __libc_start_main /usr/src/debug/glibc-2.32-37-g760e1d2878/csu/../csu/libc-start.c:314:16

SUMMARY: AddressSanitizer: heap-buffer-overflow (/home/henices/tests/exiv2/build_asan/bin/exiv2+0x4c4d09) in __asan_memcpy
Shadow bytes around the buggy address:
  0x0c4c7fff8ab0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c4c7fff8ac0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c4c7fff8ad0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c4c7fff8ae0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c4c7fff8af0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x0c4c7fff8b00: 00 00 00 00 00 00 00 00 00 00 00 00[06]fa fa fa
  0x0c4c7fff8b10: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c4c7fff8b20: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c4c7fff8b30: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c4c7fff8b40: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c4c7fff8b50: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
Shadow byte legend (one shadow byte represents 8 application bytes):
  Addressable:           00
  Partially addressable: 01 02 03 04 05 06 07
  Heap left redzone:       fa
  Freed heap region:       fd
  Stack left redzone:      f1
  Stack mid redzone:       f2
  Stack right redzone:     f3
  Stack after return:      f5
  Stack use after scope:   f8
  Global redzone:          f9
  Global init order:       f6
  Poisoned by user:        f7
  Container overflow:      fc
  Array cookie:            ac
  Intra object redzone:    bb
  ASan internal:           fe
  Left alloca redzone:     ca
  Right alloca redzone:    cb
  Shadow gap:              cc
==119384==ABORTING
```

Credit: Zhen Zhou of NSFOCUS Security Team

clanmills commented on Apr 8, 2021 Collaborator

What is your plan here? This is the third similar CVE in three days. Exiv2 v0.27.4 is scheduled to ship on 2021-05-22. Are you planning to continuously bombard us with CVEs for weeks and months?

Is it possible to have a Zoom meeting to discuss your intention and how we can cooperate?

hassec added help wanted labels on Apr 8, 2021

henices commented on Apr 8, 2021 • edited — Author

@clanmills Thanks for your hard work to make exiv2 better. I indeed have several other exiv2 security bugs, but I don't submit all the bugs at the same time, I can't agree with the strong word bombard. Security testing for exiv2 also takes a lot time, if your guys don't like to see these kind of bugs, feel free to them, I will never submit them again.

I don't know is there a deadline for exiv2 release schedule, sorry for the inconvience.

clanmills commented on Apr 9, 2021 • edited — Collaborator

Thank You @henices for the courtesy of your reply. And thank you for opening issues on GitHub about these matters. That's very helpful. The sooner Team Exiv2 knows about these matters, the sooner they can be fixed.

Team Exiv2 agrees that knowing about those issues and fixing them is better that having in the code and unknown to us.

The Exiv2 development plan is to create a new branch called 'main' and to release Exiv2 v1.00 from that branch on 2021-12-15. We would like to ask you to focus your attention on 'main'. We will fix the issues you have opened on 0.27-maintenance and ship that as v0.27.4 on/before 2021-05-22. If we ever make another release from the 0.27-maintenance branch, we will back-port security fixes from 'main'.

I appreciate the effort that you and your co-workers are putting into the important matter of security. I apologise for saying 'bombardment'. My hope this week was to finish my 13 years of working on Exiv2. I was distressed to see those CVEs arrive on day on which I intended to retire!

kevinbackhouse commented on Apr 9, 2021 — Collaborator

I am unable to reproduce this. I tested it on Ubuntu 20.04, using the latest version of 0-27-maintenance (commit 05ec05342e17dc94670db1818447c06d0da8f41a). These are the exact steps that I tried:

```
git checkout 0.27-maintenance
mkdir build
cd build
CC=clang CXX=clang++ cmake .. -DCMAKE_BUILD_TYPE=Release -DCMAKE_CXX_FLAGS="-fsanitize=address" -DCMAKE_C_FLAGS="-fsanitize=address" -DCMAKE_EXE_LINKER_FLAGS="-fsanitize=address" -
DCMAKE_MODULE_LINKER_FLAGS="-fsanitize=address"
make -j8
./bin/exiv2 ~/Downloads/tests_1bd0a5f4935b053f33ac00f931dde1f47a043487
```

I do not see any ASAN failures.

kevinbackhouse commented on Apr 9, 2021 — Collaborator

Oh, I see. I missed the `in` parameter.

clanmills commented on Apr 9, 2021 — Collaborator

@kevinbackhouse I also missed that on #1529 (comment)

I reproduced #1529 as follows:

```
.../foo $ ls -l
total 88
-rw-r--r--@ 1 rmills  staff  40609  8 Apr 08:01 tests_83a94b3337206caa6803f625eb63db061395cf14
-rw-r--r--@ 1 rmills  staff      9  8 Apr 08:09 tests_83a94b3337206caa6803f625eb63db061395cf14.exv
.../foo $ exiv2 in tests_83a94b3337206caa6803f625eb63db061395cf14
=================================================================
==52084==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x6020000001b7 at pc 0x00010525f7f4 bp 0x7ffeeab2ed10 sp 0x7ffeeab2ed08
WRITE of size 8 at 0x6020000001b7 thread T0
    #0 0x10525f7f3 in Exiv2::Jp2Image::doWriteMetadata(Exiv2::BasicIo&)+0x2143 (libexiv2.0.27.4.2.dylib:x86_64+0xf27f3)
```

I believe similar medicine is needed for this issue.

The 'in' command is 'insert'. It reads metadata from tests_xxxxx.exv and updates tests_xxxxx.

kevinbackhouse mentioned this issue on Apr 9, 2021

Fix integer overflow #1536

Closed

clanmills mentioned this issue on Apr 9, 2021

Fix out of buffer access in #1529 #1534

Merged

kevinbackhouse commented on Apr 9, 2021 — Collaborator

@clanmills: when I have fuzzed exiv2 in the past, I did not try any of these extra command line options. So my testing probably didn't hit any of these "encode" methods. So it seems quite plausible that there are several more of these bugs lurking there.

From a security perspective, these command line arguments seem much less interesting to me than vanilla exiv2. I can imagine somebody downloading a untrusted image off the internet and using exiv2 to look at it's metadata. I have a much harder time imagining somebody downloading a pair of untrusted files like this and running `exiv2 in ...` on them.

pydera commented on Apr 9, 2021 — Collaborator

I would still suggest to change the type of 'size' from uint32_t to size_t.

clanmills commented on Apr 9, 2021 — Collaborator

As always, Kevin, you are saying smart things. I also missed the unusual/obscure 'in' command.

You will be aware that I was in a state of distress yesterday about those CVEs. However, I've had a nice conversation with **@henices**. The security folks in China are on our side. Their work will make Exiv2 stronger.

My brain isn't up to thinking about the merits of size_t and uint32_t. I believe the CRW format is 32 bit, so either will work. I would change the one that minimises casts.

kevinbackhouse commented on Apr 9, 2021 Collaborator

@pydera: I think `uint32_t` is a better choice than `size_t` for this code. The reason is that the type of `CiffComponent::size()` is `uint32_t`, which in turn is because we are parsing a `uint32_t` from the input file. Introducing `size_t` here would just add a risk of the code behaving differently on a 64 bit platform compared to 32 bit, for no good reason.

pydera commented on Apr 9, 2021 • edited Collaborator

@kevinbackhouse as far as I can see DataBuf.size() returns 'long' (int64_t on LP64). I did not look deeper into this, but was afraid that it might be possible to handcraft files where a size of >uint32_t-max could be returned and then overflow the uint32_t size.

kevinbackhouse commented on Apr 9, 2021 Collaborator

@pydera: Yes, I agree that `size_t` would probably be a better choice for DataBuf.size(), rather than `long`. Unfortunately `long` is very widely used in this codebase so I think it would be quite a lot of work to switch everything over to `size_t`. My biggest concern is that `long` is 32 bits on Windows, so there is a higher risk of an integer overflow on Windows. The good news is that the `new` in `DataBuf`'s constructor would throw an exception (and terminate the program) if you managed to overflow the size.

pydera commented on Apr 9, 2021 • edited Collaborator

@kevinbackhouse Agreed. My point of view was "just looking at THIS function" without deeper research on CiffComponent, I saw that DataBuf::size() could potentially overflow 'size'. Not looking at CiffComponent I concluded that changing 'size' to size_t would always be a safe choice while uint32_t needs further investigation.
As CiffComponent::size() returns indeed uint32_t we are safe here, but looking at 'this' you could only know by also looking at CiffComponent, whereas size_t would make it clear at first sight.

henices commented on Apr 9, 2021 Author

> **@clanmills**: when I have fuzzed exiv2 in the past, I did not try any of these extra command line options. So my testing probably didn't hit any of these "encode" methods. So it seems quite plausible that there are several more of these bugs lurking there.
>
> From a security perspective, these command line arguments seem much less interesting to me than vanilla exiv2. I can imagine somebody downloading a untrusted image off the internet and using exiv2 to look at it's metadata. I have a much harder time imagining somebody downloading a pair of untrusted files like this and running `exiv2 in ...` on them.

there is another way to exploit these bugs, a single image file is enough.

👍 2

henices closed this as completed on Apr 9, 2021

---

kevinbackhouse added a commit to kevinbackhouse/exiv2 that referenced this issue on Apr 9, 2021

`Regression test for Exiv2#1530` `c92ac88`

clanmills added this to the **v0.27.4** milestone on Apr 9, 2021

clanmills linked a pull request on Apr 9, 2021 that will close this issue

Fix out of buffer access in #1530 #1532 Closed

clanmills modified the milestone: **v0.27.4** on Apr 9, 2021

clanmills added security (crash) and removed good first issue labels on Apr 9, 2021

clanmills reopened this on Apr 9, 2021

kevinbackhouse mentioned this issue on Apr 9, 2021

Fix integer overflow #1539
Merged

clanmills closed this as completed in #1539 on Apr 9, 2021

---

clanmills mentioned this issue on Apr 9, 2021

Exiv2 RoadMap #1018
Open

mergify bot pushed a commit that referenced this issue on May 10, 2021

`Regression test for #1530` ··· `48587f4`

clrpackages pushed a commit to clearlinux-pkgs/exiv2 that referenced this issue on Jun 17, 2021

`exiv2: Autospec creation for update from version 0.27.3 to version 0.…` ··· `bcea9b0`

fgeek commented on Aug 6, 2021

CVE-2021-31292 has been assigned for this issue.

Assignees

No one assigned

Labels

security (crash)

Projects

None yet

Milestone

v0.27.4

Development

Successfully merging a pull request may close this issue.

- **Fix integer overflow**
  Exiv2/exiv2
- **Fix integer overflow**
  kevinbackhouse/exiv2
- **Fix out of buffer access in #1530**
  Exiv2/exiv2

6 participants