[bug report] Program crash due to uncontrolled memory allocation on function DataBuf data(subBox.length-sizeof(box)) #742

Closed wcventure opened this issue on Mar 13, 2019 · 3 comments · Fixed by #753

Assignees

Labels bug

wcventure commented on Mar 13, 2019

Hi there,

An issue was discovered in DataBuf data(subBox.length-sizeof(box)) function in image.cpp, as distributed in master and version 0.27. There is an uncontrolled memory allocation problem, leading to a program crash. I have also confirmed this issue by using addressSanitizer.

Here is the POC file. Please use the "./exiv2 -pX $POC" to reproduce the bug.
POC.zip

```
subBox.length = getLong((byte*)&subBox.length, bigEndian);
subBox.type   = getLong((byte*)&subBox.type, bigEndian);

// subBox.length makes no sense if it is larger than the rest of the file
if (subBox.length > io_->size() - io_->tell()) {
     throw Error(kerCorruptedMetadata);
}
DataBuf data(subBox.length-sizeof(box));
io_->read(data.pData_,data.size_);
```

The ASAN dumps the stack trace as follows:

```
==9819==WARNING: AddressSanitizer failed to allocate 0xffffffffffffffff bytes
=================================================================
==9819==ERROR: AddressSanitizer: unknown-crash on address 0xffffffffffffffff at pc 0x0000004a9325 bp 0x7fffe470cec0 sp 0x7fffe470c670
WRITE of size 18446744073709551615 at 0xffffffffffffffff thread T0
    #0 0x4a9324 in __asan_memset (/home/wencheng/Documents/FuzzingObject/exiv2/build/bin/exiv2+0x4a9324)
    #1 0x7f3986594f9c in Exiv2::DataBuf::DataBuf(long) /home/wencheng/Documents/FuzzingObject/exiv2/src/types.cpp:141:42
    #2 0x7f39864b354c in Exiv2::Jp2Image::printStructure(std::ostream&, Exiv2::PrintStructureOption, int) /home/wencheng/Documents/FuzzingObject/exiv2/src/jp2image.cpp:506:37
    #3 0x53fa0d in (anonymous namespace)::printStructure(std::ostream&, Exiv2::PrintStructureOption, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >
const&) /home/wencheng/Documents/FuzzingObject/exiv2/src/actions.cpp:2368:9
    #4 0x5400f2 in Action::setModeAndPrintStructure(Exiv2::PrintStructureOption, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&)
/home/wencheng/Documents/FuzzingObject/exiv2/src/actions.cpp:237:16
    #5 0x5400f2 in Action::Print::run(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&)
/home/wencheng/Documents/FuzzingObject/exiv2/src/actions.cpp:256
    #6 0x4f42f5 in main /home/wencheng/Documents/FuzzingObject/exiv2/src/exiv2.cpp:172:23
    #7 0x7f3984e1982f in __libc_start_main /build/glibc-C15G7W/glibc-2.23/csu/../csu/libc-start.c:291
    #8 0x41f0a8 in _start (/home/wencheng/Documents/FuzzingObject/exiv2/build/bin/exiv2+0x41f0a8)

==9819==AddressSanitizer CHECK failed: /build/llvm-toolchain-3.8-_PD09B/llvm-toolchain-3.8-3.8/projects/compiler-rt/lib/asan/asan_report.cc:354 "((0 && "Address is not in memory
and not in shadow?")) != (0)" (0x0, 0x0)
    #0 0x4c87dd in __asan::AsanCheckFailed(char const*, int, char const*, unsigned long long, unsigned long long)
(/home/wencheng/Documents/FuzzingObject/exiv2/build/bin/exiv2+0x4c87dd)
    #1 0x4cf403 in __sanitizer::CheckFailed(char const*, int, char const*, unsigned long long, unsigned long long)
(/home/wencheng/Documents/FuzzingObject/exiv2/build/bin/exiv2+0x4cf403)
    #2 0x4c3f9b in __asan::DescribeAddress(unsigned long, unsigned long, char const*) (/home/wencheng/Documents/FuzzingObject/exiv2/build/bin/exiv2+0x4c3f9b)
    #3 0x4c4480 in __asan::ReportGenericError(unsigned long, unsigned long, unsigned long, unsigned long, bool, unsigned long, unsigned int, bool) [clone .part.18]
(/home/wencheng/Documents/FuzzingObject/exiv2/build/bin/exiv2+0x4c4480)
    #4 0x4a9346 in __asan_memset (/home/wencheng/Documents/FuzzingObject/exiv2/build/bin/exiv2+0x4a9346)
    #5 0x7f3986594f9c in Exiv2::DataBuf::DataBuf(long) /home/wencheng/Documents/FuzzingObject/exiv2/src/types.cpp:141:42
    #6 0x7f39864b354c in Exiv2::Jp2Image::printStructure(std::ostream&, Exiv2::PrintStructureOption, int) /home/wencheng/Documents/FuzzingObject/exiv2/src/jp2image.cpp:506:37
    #7 0x53fa0d in (anonymous namespace)::printStructure(std::ostream&, Exiv2::PrintStructureOption, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >
const&) /home/wencheng/Documents/FuzzingObject/exiv2/src/actions.cpp:2368:9
    #8 0x5400f2 in Action::setModeAndPrintStructure(Exiv2::PrintStructureOption, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&)
/home/wencheng/Documents/FuzzingObject/exiv2/src/actions.cpp:237:16
    #9 0x5400f2 in Action::Print::run(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&)
/home/wencheng/Documents/FuzzingObject/exiv2/src/actions.cpp:256
    #10 0x4f42f5 in main /home/wencheng/Documents/FuzzingObject/exiv2/src/exiv2.cpp:172:23
    #11 0x7f3984e1982f in __libc_start_main /build/glibc-C15G7W/glibc-2.23/csu/../csu/libc-start.c:291
    #12 0x41f0a8 in _start (/home/wencheng/Documents/FuzzingObject/exiv2/build/bin/exiv2+0x41f0a8)
```

wcventure commented on Mar 13, 2019 Author

I also use gdb to debug the program, I will show you the process.

```
$ gdb --args ./exiv2 -pX ./POC
(gdb) b jp2image.cpp:499
Breakpoint 1 (jp2image.cpp:499) pending.
(gdb) start
Temporary breakpoint 2 at 0x413463: file /exiv2/src/exiv2.cpp, line 132.
Starting program: /exiv2/build/bin/exiv2 -pX ../../../exiv2/Fuzzing/out_pX/crashes/id:000001,sig:06,src:000013,op:int32,pos:20,val:+0
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Temporary breakpoint 2, main (argc=3, argv=0x7fffffffdf68) at /exiv2/src/exiv2.cpp:132
```

```
132     {
(gdb) c
Continuing.

Breakpoint 1, Exiv2::Jp2Image::printStructure (this=0x65eea0, out=..., option=Exiv2::kpsXMP, depth=0)
    at /exiv2/src/jp2image.cpp:499
499                                 subBox.length = getLong((byte*)&subBox.length, bigEndian);
(gdb) p subBox.length
$1 = 0
(gdb) n
500                                 subBox.type   = getLong((byte*)&subBox.type, bigEndian);
(gdb) p subBox.type
$2 = 4286578432
(gdb) n
503                                 if (subBox.length > io_->size() - io_->tell()) {
(gdb) p  io_->size() - io_->tell()
$3 = 56
(gdb) p subBox.length
$4 = 0
(gdb) n
506                                 DataBuf data(subBox.length-sizeof(box));
(gdb) p subBox.length-sizeof(box)
$5 = 4294967288
```

you can see that in `p subBox.length-sizeof(box)` ,

```
$5 = 4294967288
```

The DataBuf data() try to consume too much memory, leading to program crash.
The normal output is

```
$ ./exiv2 -pX ./POC
Uncaught exception: std::bad_alloc
```

wcventure changed the title ~~Program crash due to uncontrolled memory allocation on function DataBuf data(subBox.length-sizeof(box))~~ [bug report] Program crash due to uncontrolled memory allocation on function DataBuf data(subBox.length-sizeof(box)) on Mar 13, 2019

D4N commented on Mar 13, 2019 — Member

Thanks for the report, that is indeed a nasty bug.

Looks like an integer overflow as the input is not scrubbed. It's unlikely though that I'll be able to tackle this before #740 is done & merged, as the code where you found the issue is not really covered by tests.

D4N added the bug label on Mar 13, 2019

piponazo self-assigned this on Mar 20, 2019

piponazo commented on Mar 20, 2019 — Collaborator

I will investigate the issue. It is also reproducible on Windows+MSVC

piponazo mentioned this issue on Mar 27, 2019

Fix #742 - master #753

Merged

piponazo closed this as completed in #753 on Mar 31, 2019

---

piponazo added a commit that referenced this issue on Mar 31, 2019

`Merge pull request #753 from piponazo/fix742-master` ··· `e93c372`

mergify bot pushed a commit that referenced this issue on Mar 31, 2019

`Fix #742 by detecting incorrect subBox size` ··· `c0bedb9`

mergify bot pushed a commit that referenced this issue on Mar 31, 2019

`Add regression test for #742` ··· `2afb748`

piponazo added a commit that referenced this issue on Apr 7, 2019

`Fix #742 by detecting incorrect subBox size` ··· `b9cd1d8`

piponazo added a commit that referenced this issue on Apr 7, 2019

`Add regression test for #742` ··· `87863b8`

piponazo added a commit that referenced this issue on Apr 7, 2019

`Fix #742 by detecting incorrect subBox size` ··· `051b5d9`

piponazo added a commit that referenced this issue on Apr 7, 2019

`Add regression test for #742` ··· `f33d8da`

Assignees

piponazo

Labels

bug

Projects

None yet

Milestone

No milestone

Development

Successfully merging a pull request may close this issue.

Fix #742 - master

3 participants