New issue

Jump to bottom

A stack-buffer-overflow occurs while parsing movie details #32

Closed tank0123 opened this issue on Jul 13, 2021 · 1 comment

tank0123 commented on Jul 13, 2021

System Configuration

- AtomicParsley version: atomicparsley `020176f`
- Used arguments: -T 1 -t +
- Environment (Operating system, version and so on): Ubuntu 20.04.2 64bit
- Additional information: compilation with asan

Description
Buffer overflow occurs while 64-bit fread(util.cpp/APar_read64() 299line) because the size of the buffer(extracts.cpp/APar_ExtractDetails() 1591line) is small (5 bytes)

==53286==ERROR: AddressSanitizer: stack-buffer-overflow on address 0x7fffffffd8c5 at pc 0x7ffff75e858d bp 0x7fffffffd5d0 sp 0x7fffffffcd78
WRITE of size 8 at 0x7fffffffd8c5 thread T0
#0 0x7ffff75e858c (/lib/x86_64-linux-gnu/libasan.so.5+0x6b58c)
#1 0x5555555fd468 in fread /usr/include/x86_64-linux-gnu/bits/stdio2.h:297
#2 0x5555555fd468 in APar_read64(char*, _IO_FILE*, unsigned long) /home/ubuntu/tmp/atomicparsley/src/util.cpp:299
#3 0x5555555a05a0 in APar_ExtractTrackDetails(char*, _IO_FILE*, Trackage*, TrackInfo*) /home/ubuntu/tmp/atomicparsley/src/extracts.cpp:1247
#4 0x5555555a2883 in APar_ExtractDetails(_IO_FILE*, unsigned char) /home/ubuntu/tmp/atomicparsley/src/extracts.cpp:1635
#5 0x5555555c07e7 in real_main(int, char**) /home/ubuntu/tmp/atomicparsley/src/main.cpp:1637
#6 0x7ffff70650b2 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x270b2)
#7 0x55555559921d in _start (/home/ubuntu/tmp/atomicparsley/AtomicParsley+0x4521d)

Address 0x7fffffffd8c5 is located in stack of thread T0 at offset 453 in frame
#0 0x5555555a23df in APar_ExtractDetails(_IO_FILE*, unsigned char) /home/ubuntu/tmp/atomicparsley/src/extracts.cpp:1590

This frame has 3 object(s):
[32, 36) 'track' (line 1592)
[48, 384) 'track_info' (line 1632)
[448, 453) 'uint32_buffer' (line 1591) <== Memory access at offset 453 overflows this variable
HINT: this may be a false positive if your program uses some custom stack unwind mechanism, swapcontext or vfork
(longjmp and C++ exceptions *are* supported)
SUMMARY: AddressSanitizer: stack-buffer-overflow (/lib/x86_64-linux-gnu/libasan.so.5+0x6b58c)
Shadow bytes around the buggy address:
0x10007fff7ac0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x10007fff7ad0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x10007fff7ae0: f1 f1 f1 f1 04 f2 00 00 00 00 00 00 00 00 00 00
0x10007fff7af0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x10007fff7b00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x10007fff7b10: f2 f2 f2 f2 f2 f2 f2 f2[05]f3 f3 f3 00 00 00 00
0x10007fff7b20: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x10007fff7b30: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x10007fff7b40: f1 f1 f1 f1 f1 f1 01 f2 01 f2 01 f2 01 f2 01 f2
0x10007fff7b50: 01 f2 01 f2 01 f2 01 f2 01 f2 01 f2 01 f2 01 f2
0x10007fff7b60: 01 f2 01 f2 01 f2 01 f2 01 f2 01 f2 01 f2 01 f2
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
==53286==ABORTING

I've attached the file. Please download and check the file.
2021-05-04-09_21_45_0xf6b390a1_0xb1c1261c.zip

wez added a commit that referenced this issue on Jul 13, 2021

Avoid stack overflow ··· d72ccf0

wez commented on Jul 13, 2021 Owner

Thanks; I've pushed a fix along with a basic integration test for this using the data file you supplied!

wez closed this as completed on Jul 13, 2021

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

2 participants