adamstark / AudioFile Public

Code Issues 6 Pull requests 6 Actions Projects Security ...

New issue

Jump to bottom

[Bug]heap-buffer-overflow in function fouBytesToInt():AudioFile.h:1196 #58

Closed Asteriska8 opened this issue on Feb 8 · 2 comments

Asteriska8 commented on Feb 8

Description

A heap-buffer-overflow was discovered in function fouBytesToInt():AudioFile.h:1196
The issue is being triggered in function getIndexOfChunk()

Version

Version 004065d (Lastest commit)

Environment

Ubuntu 18.04, 64bit

Reproduce

Command

```
git clone the Lastest Version firstly.
mkdir build
cd build && cmake ..
g++ -g -fsanitize=address -o valibin a.cpp AudioFile.h
./ poc
```

program

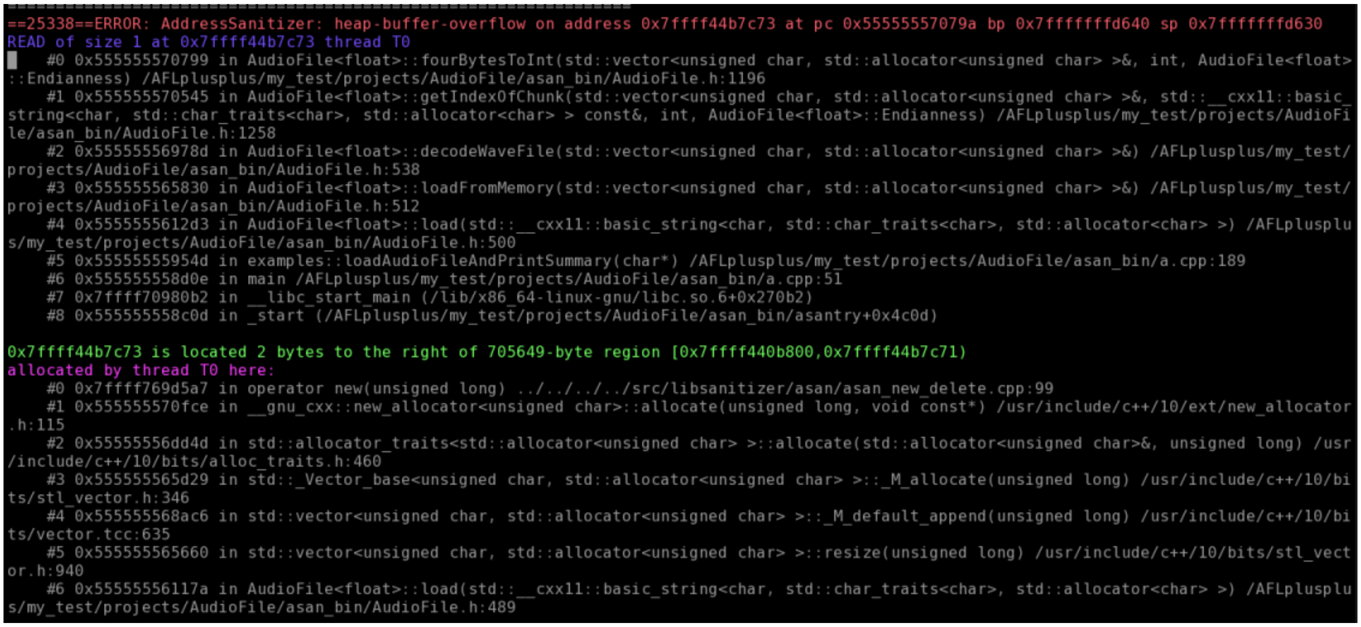
```
#include <iostream>
#define _USE_MATH_DEFINES
#include <cmath>
#include "AudioFile.h"
```

```
namespace examples

{

    void writeSineWaveToAudioFile();

    void loadAudioFileAndPrintSummary(char *);

    void loadAudioFileAndProcessSamples(char *);

} // namespace examples

int main(int argc, char **argv)

{
        examples::loadAudioFileAndPrintSummary(argv[1]);
        examples::loadAudioFileAndProcessSamples(argv[1]);
}




namespace examples

{

    void writeSineWaveToAudioFile()

    {


        AudioFile<float> a;

        a.setNumChannels(2);

        a.setNumSamplesPerChannel(44100);



        //---------------------------------------------------------------

        // 2. Create some variables to help us generate a sine wave



        const float sampleRate = 44100.f;

        const float frequencyInHz = 440.f;



        //---------------------------------------------------------------

        // 3. Write the samples to the AudioFile sample buffer
        for (int i = 0; i < a.getNumSamplesPerChannel(); i++)

        {

            for (int channel = 0; channel < a.getNumChannels(); channel++)

            {

                a.samples[channel][i] = sin((static_cast<float>(i) / sampleRate) * frequencyInHz * 
2.f * M_PI);

            }

        }



        //---------------------------------------------------------------

        // 4. Save the AudioFile



        std::string filePath = "sine-wave.wav"; // change this to somewhere useful for you

        a.save("sine-wave.wav", AudioFileFormat::Wave);

    }



    //=======================================================================

    void loadAudioFileAndPrintSummary(char *file)

    {
        const std::string filePath = std::string(file);

        AudioFile<float> a;

        bool loadedOK = a.load(filePath);



        /** If you hit this assert then the file path above

         probably doesn't refer to a valid audio file */

        assert(loadedOK);



        //---------------------------------------------------------------
    // 3. Let's print out some key details

    std::cout << "Bit Depth: " << a.getBitDepth() << std::endl;
    std::cout << "Sample Rate: " << a.getSampleRate() << std::endl;
    std::cout << "Num Channels: " << a.getNumChannels() << std::endl;
    std::cout << "Length in Seconds: " << a.getLengthInSeconds() << std::endl;
    std::cout << std::endl;
}

//=======================================================================
void loadAudioFileAndProcessSamples(char *file)
{
    //---------------------------------------------------------------
    std::cout << "**********************" << std::endl;
    std::cout << "Running Example: Load Audio File and Process Samples" << std::endl;
    std::cout << "**********************" << std::endl
              << std::endl;

    //---------------------------------------------------------------
    // 1. Set a file path to an audio file on your machine
    const std::string inputFilePath = std::string(file);

    //---------------------------------------------------------------
    // 2. Create an AudioFile object and load the audio file

    AudioFile<float> a;
    bool loadedOK = a.load(inputFilePath);
```

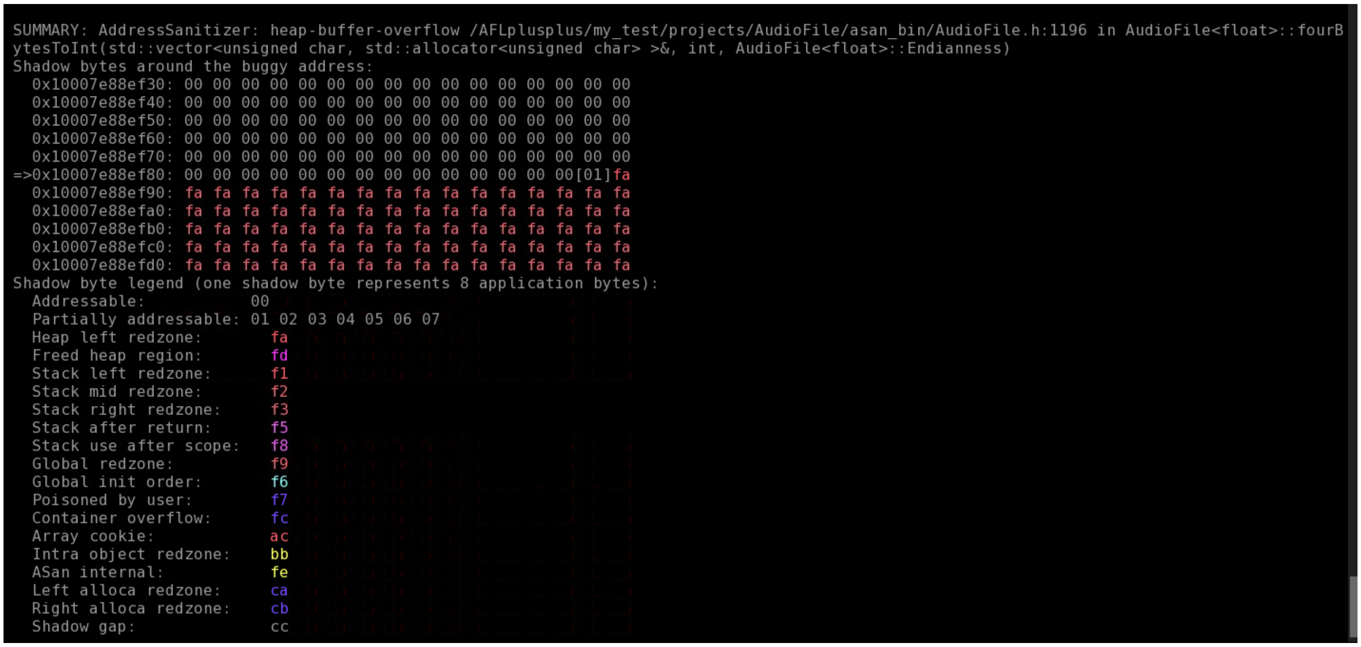
```
        /** If you hit this assert then the file path above
         probably doesn't refer to a valid audio file */
        assert(loadedOK);

        //---------------------------------------------------------------
        // 3. Let's apply a gain to every audio sample

        float gain = 0.5f;

        for (int i = 0; i < a.getNumSamplesPerChannel(); i++)
        {
            for (int channel = 0; channel < a.getNumChannels(); channel++)
            {
                a.samples[channel][i] = a.samples[channel][i] * gain;
            }
        }

        //---------------------------------------------------------------
        // 4. Write audio file to disk

        //std::string outputFilePath = "quieter-audio-filer.wav"; // change this to somewhere useful for you
        //a.save(outputFilePath, AudioFileFormat::Aiff);
    }
} // namespace examples
```

POC file at the bottom of this report.

ASAN Report



```
=================================================================
==25338==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x7ffff44b7c73 at pc 0x55555557079a bp 0x7fffffffd640 sp 0x7fffffffd630
READ of size 1 at 0x7ffff44b7c73 thread T0
    #0 0x555555570799 in AudioFile<float>::fourBytesToInt(std::vector<unsigned char, std::allocator<unsigned char> >&, int, AudioFile<float>
::Endianness) /AFLplusplus/my_test/projects/AudioFile/asan_bin/AudioFile.h:1196
    #1 0x555555570545 in AudioFile<float>::getIndexOfChunk(std::vector<unsigned char, std::allocator<unsigned char> >&, std::__cxx11::basic_
string<char, std::char_traits<char>, std::allocator<char> > const&, int, AudioFile<float>::Endianness) /AFLplusplus/my_test/projects/AudioFi
le/asan_bin/AudioFile.h:1258
    #2 0x55555556978d in AudioFile<float>::decodeWaveFile(std::vector<unsigned char, std::allocator<unsigned char> >&) /AFLplusplus/my_test/
projects/AudioFile/asan_bin/AudioFile.h:538
    #3 0x555555565830 in AudioFile<float>::loadFromMemory(std::vector<unsigned char, std::allocator<unsigned char> >&) /AFLplusplus/my_test/
projects/AudioFile/asan_bin/AudioFile.h:512
    #4 0x5555555612d3 in AudioFile<float>::load(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >) /AFLplusplu
s/my_test/projects/AudioFile/asan_bin/AudioFile.h:500
    #5 0x55555555954d in examples::loadAudioFileAndPrintSummary(char*) /AFLplusplus/my_test/projects/AudioFile/asan_bin/a.cpp:189
    #6 0x5555555558d0e in main /AFLplusplus/my_test/projects/AudioFile/asan_bin/a.cpp:51
    #7 0x7ffff70980b2 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x270b2)
    #8 0x555555558c0d in _start (/AFLplusplus/my_test/projects/AudioFile/asan_bin/asantry+0x4c0d)

0x7ffff44b7c73 is located 2 bytes to the right of 705649-byte region [0x7ffff440b800,0x7ffff44b7c71)
allocated by thread T0 here:
    #0 0x7ffff769d5a7 in operator new(unsigned long) ../../../../src/libsanitizer/asan/asan_new_delete.cpp:99
    #1 0x555555570fce in __gnu_cxx::new_allocator<unsigned char>::allocate(unsigned long, void const*) /usr/include/c++/10/ext/new_allocator
.h:115
    #2 0x55555556dd4d in std::allocator_traits<std::allocator<unsigned char> >::allocate(std::allocator<unsigned char>&, unsigned long) /usr
/include/c++/10/bits/alloc_traits.h:460
    #3 0x555555565d29 in std::_Vector_base<unsigned char, std::allocator<unsigned char> >::_M_allocate(unsigned long) /usr/include/c++/10/bi
ts/stl_vector.h:346
    #4 0x555555568ac6 in std::vector<unsigned char, std::allocator<unsigned char> >::_M_default_append(unsigned long) /usr/include/c++/10/bi
ts/vector.tcc:635
    #5 0x555555565660 in std::vector<unsigned char, std::allocator<unsigned char> >::resize(unsigned long) /usr/include/c++/10/bits/stl_vect
or.h:940
    #6 0x55555556117a in AudioFile<float>::load(std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> >) /AFLplusplu
s/my_test/projects/AudioFile/asan_bin/AudioFile.h:489
```



```
SUMMARY: AddressSanitizer: heap-buffer-overflow /AFLplusplus/my_test/projects/AudioFile/asan_bin/AudioFile.h:1196 in AudioFile<float>::fourB
ytesToInt(std::vector<unsigned char, std::allocator<unsigned char> >&, int, AudioFile<float>::Endianness)
Shadow bytes around the buggy address:
  0x10007e88ef30: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10007e88ef40: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10007e88ef50: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10007e88ef60: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x10007e88ef70: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x10007e88ef80: 00 00 00 00 00 00 00 00 00 00 00 00 00 00[01]fa
  0x10007e88ef90: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x10007e88efa0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x10007e88efb0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x10007e88efc0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x10007e88efd0: 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
```

POC

POC

Any issue plz contact with me:
asteriska001@gmail.com
OR:
twitter: **@Asteriska8**

adamstark commented on Aug 1

Owner

Hi there, thanks for this. What format is the file you are trying to load in?

adamstark commented on Aug 1 Owner

Nevermind - i think I understand now. I've made some changes that stop this kind of thing from happening. Those changes should be on develop now :) If you had time to verify I'd appreciate it!

😄 1

adamstark closed this as completed on Aug 1

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

2 participants