axiomatic-systems / Bento4 Public

Code | Issues 422 | Pull requests 28 | Actions | Projects | Wiki

New issue

Jump to bottom

Exhaustive memory usage #413

Open 5hadowblad3 opened this issue on Aug 9, 2019 · 0 comments

Assignees

Labels fuzzing

5hadowblad3 commented on Aug 9, 2019

There is a buffer overflow inside AP4_IkmsAtom of AP4IkmsAtom.cpp.
It is similar to #412 and #396.
./mp42aac input_file /dev/null

In file Source/C++/Core/AP4IkmsAtom.cpp
AP4_RtpAtom allocates a new buffer to parse the atom in the stream.
The unhandled memory allocation failure causes the read content memcpy to a null pointer.
This is the start points.

```
/*----------------------------------------------------------------------
|   AP4_IkmsAtom::AP4_IkmsAtom
+----------------------------------------------------------------------*/
AP4_IkmsAtom::AP4_IkmsAtom(AP4_UI32        size,
                           AP4_UI08        version,
                           AP4_UI32        flags,
                           AP4_ByteStream& stream) :
    AP4_Atom(AP4_ATOM_TYPE_IKMS, size, version, flags)
{
    AP4_Size string_size = size-AP4_FULL_ATOM_HEADER_SIZE;
    if (m_Version == 1 && string_size >= 8) {
        string_size -= 8;
        stream.ReadUI32(m_KmsId);
        stream.ReadUI32(m_KmsVersion);
    } else {
        m_KmsId      = 0;
        m_KmsVersion = 0;
    }
    if (string_size) {
        char* str = new char[string_size];
        stream.Read(str, string_size);
        str[string_size-1] = '\0'; // force null-termination
        m_KmsUri = str;
        delete[] str;
    }
}

```

In file In file Source/C++/Core/AP4IkmsAtom.cpp

```
AP4_ByteStream::Read(void* buffer, AP4_Size bytes_to_read)
{
    // shortcut
    if (bytes_to_read == 0) return AP4_SUCCESS;

    // read until failure
    AP4_Size bytes_read;
    while (bytes_to_read) {
        AP4_Result result = ReadPartial(buffer, bytes_to_read, bytes_read);
        if (AP4_FAILED(result)) return result;
        if (bytes_read == 0) return AP4_ERROR_INTERNAL;
        AP4_ASSERT(bytes_read <= bytes_to_read);
        bytes_to_read -= bytes_read;
        buffer = (void*)(((AP4_Byte*)buffer)+bytes_read);
    }

    return AP4_SUCCESS;
}
```

No buffer size check

```
AP4_Result
AP4_MemoryByteStream::ReadPartial(void*     buffer,
                                  AP4_Size  bytes_to_read,
                                  AP4_Size& bytes_read)
{
    // default values
    bytes_read = 0;

    // shortcut
    if (bytes_to_read == 0) {
        return AP4_SUCCESS;
    }

    // clamp to range
    if (m_Position+bytes_to_read > m_Buffer->GetDataSize()) {
        bytes_to_read = (AP4_Size)(m_Buffer->GetDataSize() - m_Position);
    }

    // check for end of stream
    if (bytes_to_read == 0) {
        return AP4_ERROR_EOS;
    }

    // read from the memory
    AP4_CopyMemory(buffer, m_Buffer->GetData()+m_Position, bytes_to_read);
    m_Position += bytes_to_read;
```

No buffer check

AP4_CopyMemory is the macro define of memcpy and the path formed.

Asan trace report:

```
==149039==WARNING: AddressSanitizer failed to allocate 0xff7efffd bytes
==149039==AddressSanitizer's allocator is terminating the process instead of returning 0
==149039==If you don't like this behavior set allocator_may_return_null=1
==149039==AddressSanitizer CHECK failed: ../../../../../../src/libsanitizer/sanitizer_common/sanitizer_allocator.cc:147 "((0)) != (0)" (0x0, 0x0)
#0 0xf724a797 (/usr/lib32/libasan.so.2+0x9f797)
#1 0xf724fa69 in __sanitizer::CheckFailed(char const*, int, char const*, unsigned long long, unsigned long long) (/usr/lib32/libasan.so.2+0xa4a69)
#2 0xf71c107b (/usr/lib32/libasan.so.2+0x1607b)
#3 0xf724de80 (/usr/lib32/libasan.so.2+0xa2e80)
#4 0xf71c6229 (/usr/lib32/libasan.so.2+0x1b229)
#5 0xf7242e16 in operator new[](unsigned int) (/usr/lib32/libasan.so.2+0x97e16)
#6 0x90075ba in AP4_IkmsAtom::AP4_IkmsAtom(unsigned int, unsigned char, unsigned int, AP4_ByteStream&) /mnt/data/playground/mp42-a/Source/C++/Core/Ap4IkmsAtom.cpp:87
#7 0x9008e85 in AP4_IkmsAtom::Create(unsigned int, AP4_ByteStream&) /mnt/data/playground/mp42-a/Source/C++/Core/Ap4IkmsAtom.cpp:51
#8 0x82db1ec in AP4_AtomFactory::CreateAtomFromStream(AP4_ByteStream&, unsigned int, unsigned int, unsigned long long, AP4_Atom*&) /mnt/data/playground/mp42-
a/Source/C++/Core/Ap4AtomFactory.cpp:604
#9 0x8301ca3 in AP4_AtomFactory::CreateAtomFromStream(AP4_ByteStream&, unsigned long long&, AP4_Atom*&) /mnt/data/playground/mp42-a/Source/C++/Core/Ap4AtomFactory.cpp:225
#10 0x82b6bae in AP4_ContainerAtom::ReadChildren(AP4_AtomFactory&, AP4_ByteStream&, unsigned long long) /mnt/data/playground/mp42-a/Source/C++/Core/Ap4ContainerAtom.cpp:194
#11 0x82b6bae in AP4_ContainerAtom::AP4_ContainerAtom(unsigned int, unsigned long long, bool, AP4_ByteStream&, AP4_AtomFactory&) /mnt/data/playground/mp42-
a/Source/C++/Core/Ap4ContainerAtom.cpp:139
#12 0x841a898 in AP4_MoovAtom::AP4_MoovAtom(unsigned int, AP4_ByteStream&, AP4_AtomFactory&) /mnt/data/playground/mp42-a/Source/C++/Core/Ap4MoovAtom.cpp:80
#13 0x82e2631 in AP4_MoovAtom::Create(unsigned int, AP4_ByteStream&, AP4_AtomFactory&) /mnt/data/playground/mp42-a/Source/C++/Core/Ap4MoovAtom.h:56
#14 0x82e2631 in AP4_AtomFactory::CreateAtomFromStream(AP4_ByteStream&, unsigned int, unsigned int, unsigned long long, AP4_Atom*&) /mnt/data/playground/mp42-
a/Source/C++/Core/Ap4AtomFactory.cpp:363
#15 0x82fa1f7 in AP4_AtomFactory::CreateAtomFromStream(AP4_ByteStream&, unsigned long long&, AP4_Atom*&) /mnt/data/playground/mp42-a/Source/C++/Core/Ap4AtomFactory.cpp:225
#16 0x82fa1f7 in AP4_AtomFactory::CreateAtomFromStream(AP4_ByteStream&, AP4_Atom*&) /mnt/data/playground/mp42-a/Source/C++/Core/Ap4AtomFactory.cpp:151
#17 0x809a044 in AP4_File::ParseStream(AP4_ByteStream&, AP4_AtomFactory&, bool) /mnt/data/playground/mp42-a/Source/C++/Core/Ap4File.cpp:104
#18 0x809a044 in AP4_File::AP4_File(AP4_ByteStream&, bool) /mnt/data/playground/mp42-a/Source/C++/Core/Ap4File.cpp:78
#19 0x8082ce7 in main /mnt/data/playground/mp42-a/Source/C++/Apps/Mp42Aac/Mp42Aac.cpp:250
#20 0xf69cb636 in __libc_start_main (/lib/i386-linux-gnu/libc.so.6+0x18636)
#21 0x808df1b (/mnt/data/playground/mp42-patch/Build/mp42aac+0x808df1b)
```

The attachment is the poc file.
poc_input4.zip

5hadowblad3 mentioned this issue on Aug 9, 2019

Null pointer dereference bug #417

Open

barbibulle self-assigned this on Aug 25, 2019

barbibulle added the fuzzing label on Aug 25, 2019

Assignees

barbibulle

Labels

fuzzing

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

2 participants