New issue

Jump to bottom

Null Pointer Dereference when dealing with XtraBox #2081

Closed **0xdd96** opened this issue on Jan 28 · 0 comments

0xdd96 commented on Jan 28

version info:

```
MP4Box - GPAC version 1.1.0-DEV-rev1678-g92faba3-master
  GPAC: https://doi.org/10.1145/1291233.1291452

GPAC Configuration: --prefix=/path_to_gpac/build --enable-debug --enable-sanitizer
Features: GPAC_CONFIG_LINUX GPAC_64_BITS GPAC_HAS_IPV6 GPAC_HAS_SSL GPAC_HAS_SOCK_UN
GPAC_MINIMAL_ODF GPAC_HAS_QJS GPAC_HAS_FAAD GPAC_HAS_MAD GPAC_HAS_LIBA52 GPAC_HAS_JPEG
GPAC_HAS_PNG GPAC_HAS_FFMPEG GPAC_HAS_JP2 GPAC_HAS_THEORA GPAC_HAS_VORBIS GPAC_HAS_XVID
GPAC_HAS_LINUX_DVB
```

poc: poc
command: MP4Box -hint -out /dev/null *poc*
crash:

```
root@d8a714203f6e:/path_to_gpac/build/bin# ./MP4Box -hint -out /dev/null poc
[iso file] Unknown box type t8Aak in parent moov
[iso file] Box "UNKN" is larger than container box
[iso file] Box "moov" size 211 (start 20) invalid (read 2209)
[iso file] Box "nmhd" (start 359) has 8 extra bytes
[iso file] Unknown box type dreu in parent dinf
[iso file] Box "UNKN" is larger than container box
[iso file] Missing dref box in dinf
[iso file] Box "dinf" size 36 (start 379) invalid (read 64)
[iso file] Unknown box type url  in parent srpp
[iso file] Unknown box type srpp in parent srpp
[iso file] Box "UNKN" is larger than container box
[iso file] Box "srpp" size 1814 (start 415) invalid (read 1854)
[iso file] Unknown box type dre- in parent dinf
[iso file] Box "UNKN" is larger than container box
[iso file] Missing dref box in dinf
[iso file] Box "dinf" size 36 (start 2229) invalid (read 64)
[isom] invalid tag size in Xtra !
[isom] invalid tag size in Xtra !
```

```
[isom] not enough bytes in box Xtra: 46 left, reading 1836070003 (file isomedia/box_code_base.c,
line 12754), skipping box
[iso file] Box "Xtra" (start 2265) has 60 extra bytes
[iso file] Unknown top-level box type 00000001
0.500 secs Interleaving
utils/bitstream.c:1053:6: runtime error: null pointer passed as argument 2, which is declared to
never be null
```

Here is the trace reported by debugging. We can see that the `memcpy` function is called on line 1053 of `utils/bitstream.c`, which will copy the contents of the second parameter `data` to the buffer pointed to by the first parameter. Unfortunately, in this trace the `data` is 0 (NULL), causing the program to crash.

```
In file: /path_to_gpac/src/utils/bitstream.c
   1048                 case GF_BITSTREAM_FILE_READ:
   1049                 case GF_BITSTREAM_FILE_WRITE:
   1050                         if (bs->cache_write) {
   1051                                 //if block fits in our write cache, write it
   1052                                 if (bs->buffer_written + nbBytes < bs->cache_write_size) {
 ► 1053                                       memcpy(bs->cache_write+bs->buffer_written, data,
nbBytes);
   1054                                         bs->buffer_written+=nbBytes;
   1055                                         return nbBytes;
   1056                                 }
   1057                                 //otherwise flush cache and use file write
   1058                                 bs_flush_write_cache(bs);

pwndbg> backtrace
#0  gf_bs_write_data (bs=0x60f00000dc90, data=0x0, nbBytes=1) at utils/bitstream.c:1053
#1  0x00007ff9797a8f82 in xtra_box_write (s=0x60400000d590, bs=0x60f00000dc90) at
isomedia/box_code_base.c:12814
#2  0x00007ff979816fb8 in gf_isom_box_write_listing (a=0x60400000d590, bs=0x60f00000dc90) at
isomedia/box_funcs.c:1834
#3  0x00007ff979817737 in gf_isom_box_write (a=0x60400000d590, bs=0x60f00000dc90) at
isomedia/box_funcs.c:1883
#4  0x00007ff9798b432c in WriteInterleaved (mw=0x7ffd2b3ab870, bs=0x60f00000dc90,
drift_inter=GF_TRUE) at isomedia/isom_store.c:1963
#5  0x00007ff9798bb1ca in WriteToFile (movie=0x616000009c80, for_fragments=GF_FALSE) at
isomedia/isom_store.c:2549
#6  0x00007ff9798574d1 in gf_isom_write (movie=0x616000009c80) at isomedia/isom_read.c:600
#7  0x00007ff979857a3f in gf_isom_close (movie=0x616000009c80) at isomedia/isom_read.c:624
#8  0x00000000004413cc in mp4boxMain (argc=5, argv=0x7ffd2b3b0478) at main.c:6547
#9  0x00000000004416f2 in main (argc=5, argv=0x7ffd2b3b0478) at main.c:6601
#10 0x00007ff975d2e840 in __libc_start_main (main=0x4416d2 <main>, argc=5, argv=0x7ffd2b3b0478,
init=<optimized out>, fini=<optimized out>, rtld_fini=<optimized out>, stack_end=0x7ffd2b3b0468)
at ../csu/libc-start.c:291
#11 0x000000000040fd09 in _start ()
```

I tracked the null assignment of `data` in `isomedia/box_code_base.c`. `data2` is initialized to NULL in line 12743. When the value of `prop_size` is greater than 4 (line 12764), the program will allocate a memory chunk to `data2` (line 12769). Otherwise, `data2` will remain NULL and will be assigned to `tag->prop_value` in line 12777.

In my crash, `prop_size` was set to 1 causing `tag->prop_value` to be NULL. The `tag` is then added to `ptr->tags` for subsequent access (line 12779).

gpac/src/isomedia/box_code_base.c
Lines 12736 to 12786 in `5d68ccd`

```
12736      GF_Err xtra_box_read(GF_Box *s, GF_BitStream *bs)
12737      {
12738              GF_XtraBox *ptr = (GF_XtraBox *)s;
12739              while (ptr->size) {
12740                      GF_XtraTag *tag;
12741                      u32 prop_type = 0;
12742
12743                      char *data=NULL, *data2=NULL;
12744                      ISOM_DECREASE_SIZE_NO_ERR(ptr, 18)
12745                      s32 tag_size = gf_bs_read_u32(bs);
```

When the program executes to `xtra_box_write` , it will get a `tag` from `ptr->tags` (line 12801), and pass `tag->prop_value` to the second parameter of `gf_bs_write_data` (line 12814), which eventually results in `data` being NULL.

Although the program judges whether `tag->prop_value` is 0 in line 12805, it does not change the execution flow of the program and the value of `tag->prop_value` .

gpac/src/isomedia/box_code_base.c
Lines 12791 to 12817 in `5d68ccd`

```
12791      GF_Err xtra_box_write(GF_Box *s, GF_BitStream *bs)
12792      {
12793              GF_Err e;
12794              GF_XtraBox *ptr = (GF_XtraBox *)s;
12795              u32 i, count = gf_list_count(ptr->tags);
12796
12797              e = gf_isom_box_write_header(s, bs);
12798              if (e) return e;
12799
12800              for (i=0; i<count; i++) {
12801                      GF_XtraTag *tag = gf_list_get(ptr->tags, i);
12802                      u32 tag_size = 16;
```

Hope my analysis will help.

jeanlf closed this as completed in `71f9871` on Jan 28

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

1 participant