#2407 closed defect (fixed)

Opened 3 months ago
Closed 3 months ago

A heap memory corruption occurred in function free_mp_image() of libmpcodecs/mp_image.c

Reported by:	ylzs	Owned by:	beastd
Priority:	normal	Component:	undetermined
Version:	HEAD	Severity:	major
Keywords:		Cc:	
Blocked By:		Blocking:	
Reproduced by developer:	no	Analyzed by developer:	no

Description

Version: SVN-r38374-13.0.1

Build command: ../configure --disable-ffmpeg_a && make (compiling with asan)

Summary of the bug: I found a heap memory corruption crash when I tried to fuzz the mencoder.

```
[ … ]
1 duplicate frame(s)!
Movie-Aspect is undefined - no prescaling applied.
Writing header...
ODML: Aspect information not (yet?) available or unspecified, not writing vprp
ODML: Aspect information not (yet?) available or unspecified, not writing vprp

Skipping frame!

Writing index...

Video stream: 743743.500 kbit/s  (92967937 B/s)  size: 16892 bytes  0.000 secs
Aborted
```

But when I try to debug this crash to figure out the reason I find the free function's argument is not a heap address. The pointer points to a block of memory which is full of 0x80.

```
Breakpoint 1, free_mp_image (mpi=0x60e000000120) at libmpcodecs/mp_image.c:271
271             av_free(mpi->planes[0]);
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
─────────────────────────────────────────────────────────────────────────────────
 RAX  0x2b00
 RBX  0xc1c00000024 ◂— 0x0
 RCX  0x7fffed1ff800 ◂— 0xbfbebfbebebebebebe
 RDX  0x1
 RDI  0x60e000000120 —▸ 0x40000c30f ◂— 0x0
 RSI  0x7fffee7f90e0 ◂— 0x0
 R8   0xd8
 R9   0x7fffee489708 —▸ 0x555555857598 (uninit_video+216) ◂— mov    qword ptr
 R10  0x7fffffffcd20 —▸ 0x555555857598 (uninit_video+216) ◂— mov    qword ptr
 R11  0x20
 R12  0x0
 R13  0xffffffffad5 ◂— 0x0
 R14  0x555555ec7fa0 (__afl_area_ptr) —▸ 0x7fffed1ff800 ◂— 0xbfbebfbebebebebebe
 R15  0x60e000000120 —▸ 0x40000c30f ◂— 0x0
```

```
 RBP  0x7fffffffdda0 ◂— 0x0
 RSP  0x7fffffffd5e0 —▸ 0x616000000380 —▸ 0x555555e1c380 (vf_info_expand) —▸ 0
──────────────────────────────────────────────────────────────────────────────
   0x55555585ec3f <free_mp_image+111>    mov    rax, rdi
   0x55555585ec42 <free_mp_image+114>    shr    rax, 3
   0x55555585ec46 <free_mp_image+118>    cmp    byte ptr [rax + 0x7fff8000], 0
   0x55555585ec4d <free_mp_image+125>    jne    free_mp_image+294 <free_mp_imag

   0x55555585ec53 <free_mp_image+131>    mov    rdi, qword ptr [r15 + 0x30]
   0x55555585ec57 <free_mp_image+135>    call   av_free@plt <av_free@plt>

   0x55555585ec5c <free_mp_image+140>    mov    al, byte ptr [rbx + 0x7fff8000]
   0x55555585ec62 <free_mp_image+146>    test   al, al
   0x55555585ec64 <free_mp_image+148>    jne    free_mp_image+269 <free_mp_imag

   267 void free_mp_image(mp_image_t* mpi){
   268     if(!mpi) return;
   269     if(mpi->flags&MP_IMGFLAG_ALLOCATED){
   270         /* because we allocate the whole image at once */
 ► 271         av_free(mpi->planes[0]);
   272         if (mpi->flags & MP_IMGFLAG_RGB_PALETTE)
   273             av_free(mpi->planes[1]);
   274     }
   275     free(mpi);
   276 }
──────────────────────────────────────────────────────────────────────────────
02:0010│       0x7fffffffd5f0 —▸ 0x616000000080 —▸ 0x555555e0b5c0 (ve_info_lavc
04:0020│       0x7fffffffd600 —▸ 0x61a000001130 —▸ 0x616000000380 —▸ 0x555555e1
06:0030│       0x7fffffffd610 —▸ 0xc3400000192 ◂— 0x0
07:0038│       0x7fffffffd618 —▸ 0x5555558575c6 (uninit_video+262) ◂— call   0x
──────────────────────────────────────────────────────────────────────────────
   f 1     55555587b998 vf_uninit_filter_chain+200
   f 2     55555587b998 vf_uninit_filter_chain+200
   f 3     5555558575c6 uninit_video+262
   f 4     555555737d1b main+47819
   f 5     7ffff55070b3 __libc_start_main+243
──────────────────────────────────────────────────────────────────────────────
$4 = (unsigned char *) 0x7fffeb6cb040 '\200' <repeats 200 times>...
pwndbg> vmmap 0x7fffeb6cb040

LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
    0x7fffeb490000     0x7fffebf19000 rw-p   a89000 0       +0x23b040
```

How to reproduce:

1.Command: ./mencoder -ovc lavc -oac lavc -o /dev/null ./testcase

Attachments (2)

- testcase (642 bytes) - added by ylzs 3 months ago.
- valgrind_output (20.1 KB) - added by ylzs 3 months ago.

Change History (7)

by ylzs, 3 months ago

Attachment: *testcase* added

comment:1 by reimar, 3 months ago

Cannot reproduce unfortunately

comment:2 by ylzs, 3 months ago

I get all these testcase on an amd64 vitural machine with ubuntu 20.04 as OS.And I compile the mencoder and mplayer with clang version 13.0.1-++20220120110924+75e33f71c2da-1~exp1~20220120231001.58.
I'm not familiar with the internal of the mpalyer and mencoder so I don't kown why t some testcase can't reproduce the bug. I 'm sorry about this.

comment:3 by reimar, 3 months ago

It's suspicious that it's all the mencoder ones I cannot reproduce, not sure what the difference might be.
Can you run the problematic testcase with valgrind instead of ASAN? It might provide more useful information, which might be enough to fix even though I cannot reproduce it.

comment:4 by ylzs, 3 months ago

I've run this testcase with the valgrind and put the result into another attached file.I hope this can help you.

by ylzs, 3 months ago

Attachment: *valgrind_output* added

comment:5 by reimar, 3 months ago

Resolution: → fixed
Status: new → closed

If the valgrind output is accurate this should be fixed by r38402

Note: See TracTickets for help on using tickets.