Normalize vul discovery

0x1 Introduction

0x1.1 background

Normalize is a tool for adjusting the volume of audio files to a standard level.
https://github.com/kklobe/normalize

0x1.2 Description

An issue discovered in Normalize 0.7.7. There is an FPE (divide-by-zero error) in wiener_af.c:afGetTrackBytes

0x1.3 Impact

A remote attacker, via a crafted .wav file, could cause a crash of the Normalize.

0x2 Fuzzing results

By using the modified AFL tools in 24hs, we found crashes with the cmd:

```
-i /fuzz_test/in -o /fuzz_test/out ../normalize/src/normalize @@ /dev/null
```

Some pocs are produced in the output/crashes

id:000000,sig:08,src:000000,op:MOpt-havoc,rep:128

id:000001,sig:08,src:000007+000465,op:MOpt-core-splice,rep:4

id:000002,sig:08,src:000634,op:MOpt-havoc,rep:32

A signal SIGFPE occured, resulting the program gets crash after executing these pocs.

```
Program received signal SIGFPE, Arithmetic exception.
0x000000000040d2f7 in afGetFrameCount (fh=0x60c00000bc80, track=0x3e9) at wiener_af.c:559
559         return tracklen / framesize;
[ Legend: Modified register | Code | Heap | Stack | String ]

$rax   : 0x701
$rbx   : 0x00000fffffffffa62  → 0x0000000000000000
$rcx   : 0x0
$rdx   : 0x0
$rsp   : 0x00007fffffffd230  → 0x000003e9ffffd370  → 0x0000000000000000
$rbp   : 0x00007fffffffd250  → 0x00007fffffffd420  → 0x00007fffffffd520  → 0x00007fffffffda90  → 0x000000000041ed00
$rsi   : 0x7
$rdi   : 0x000060c00000bc80  → 0x000060300000efb0  → 0x000061600000f980  → 0xbebebebefbad2488
$rip   : 0x000000000040d2f7  → <afGetFrameCount+71> div ecx
$r8    : 0x00007ffff7fd3780  → 0x00007ffff7fd3780  → [loop detected]
$r9    : 0x000060c00000bcdc  → 0xbebebebe00000701
$r10   : 0x00007ffff7fd3780  → 0x00007ffff7fd3780  → [loop detected]
$r11   : 0x246
$r12   : 0x00007fffffffd310  → 0x0000000041b58ab3
$r13   : 0x00007fffffffd3f0  → 0x0000000000000000
$r14   : 0x00007fffffffd5a0  → 0x0000000041b58ab3
$r15   : 0x00007fffffffd310  → 0x0000000041b58ab3
$eflags: [carry PARITY adjust ZERO sign trap INTERRUPT direction overflow RESUME virtualx86 identification]
$cs: 0x0033 $ss: 0x002b $ds: 0x0000 $es: 0x0000 $fs: 0x0000 $gs: 0x0000
```

0x3 Cause of vulnerability

By following the stack trace in gdb, we can find the framesize calculted to zero lead a devide error in the function **afGetTrackBytes** at wiener_af.c+559.

```
   0x40d2ec <afGetFrameCount+60> mov    ecx, DWORD PTR [rbp-0x8]
   0x40d2ef <afGetFrameCount+63> mov    eax, DWORD PTR [rbp-0x4]
   0x40d2f2 <afGetFrameCount+66> mov    edx, 0x0
→  0x40d2f7 <afGetFrameCount+71> div    ecx
   0x40d2f9 <afGetFrameCount+73> mov    eax, eax
   0x40d2fb <afGetFrameCount+75> leave
   0x40d2fc <afGetFrameCount+76> ret
   0x40d2fd <afSetErrorHandler+0> push   rbp
   0x40d2fe <afSetErrorHandler+1> mov    rbp, rsp

   554     uint32_t tracklen;
   555
   556     framesize = _afGetFrameSize(fh, track, 0);
   557     tracklen = afGetTrackBytes(fh, track);
   558
           // framesize=0x0
→  559     return tracklen / framesize;
   560   }
   561
   562   static AFerrfunc _af_err_func = NULL;
   563
   564   AFerrfunc

[#0] Id 1, Name: "normalize", stopped, reason: SIGFPE

[#0] 0x40d2f7 →afGetFrameCount(fh=0x60c00000bc80, track=0x3e9)
[#1] 0x406bec →signal_max_power(filename=0x7fffffffdf89 "id:000000,sig:08,src:000000
[#2] 0x404cea →compute_levels(sis=0x60600000efc0, fnames=0x60200000edd0, nfiles=0x1)
[#3] 0x403736 →main(argc=0x2, argv=0x7fffffffdb78)
```

```
AFframecount
afGetFrameCount(AFfilehandle fh, int track)
{
  int framesize;
  uint32_t tracklen;

  framesize = _afGetFrameSize(fh, track, 0);
  tracklen = afGetTrackBytes(fh, track);

  return tracklen / framesize;
}
```

The framesize is calculted in the function **_afGetFrameSize**.

Thus, by controlling a crafted .wav file which channels equals to zero could cause the vulnerability.

```
static inline int
_afGetFrameSize(AFfilehandle fh, int track, int expand3to4)
{
  int bytes_per_sample;

  bytes_per_sample = (fh->fmt.bits_per_sample - 1) / 8 + 1;
  if (bytes_per_sample == 3 && expand3to4)
    bytes_per_sample = 4;

  return bytes_per_sample * fh->fmt.channels;
}
```

0x4 Reproduce

Run the test cases under the poc folder to reproduce.