★ 70602836 Security Report - [Stack overflow in external/aac/libMpegTPDec/tpdec_asc.cpp]

0 people have starred this issue.

Public Trackers Android External Security Reports

xiangxiaobo1994@gmail.com <xiangxiaobo1994@gmail.com> #1 Dec 13, 2017 04:01PM

Created issue.

[summary] 04:01PM

There is a stack-based buffer overflow vulnerability in external/aac , which is part of Android Media Framework.
I've tested this on an ASAN-enabled device and it crashed due to stack overflow. According to my further analysis, this vulnerability will cause out of bound write and will affect the program's control flow. which may probably result in remote code execution.

[details]
In /external/aac/libMpegTPDec/src/tpdec_asc.cpp:563:

```
563int CProgramConfig_LookupElement(
564           CProgramConfig *pPce,
565           UINT                channelConfig,
566           const UINT         tag,
567           const UINT         channelIdx,
568           UCHAR               chMapping[],
569           AUDIO_CHANNEL_TYPE chType[],
570           UCHAR               chIndex[],
571           UCHAR              *elMapping,
572           MP4_ELEMENT_ID   elList[],
573           MP4_ELEMENT_ID   elType
574         )
575{
--------------------------------
623     else {
624        /* Accept the additional channel(s), only if the tag is in the lists */
625        int isCpe = 0, i;
626        /* Element counter */
627        int ec[PC_NUM_HEIGHT_LAYER] = {0};            // <======== PC_NUM_HEIGHT_LAYER
is 3
628        /* Channel counters */
629        int cc[PC_NUM_HEIGHT_LAYER] = {0};
630        int fc[PC_NUM_HEIGHT_LAYER] = {0};
631        int sc[PC_NUM_HEIGHT_LAYER] = {0};
632        int bc[PC_NUM_HEIGHT_LAYER] = {0};
633        int lc = 0;;
634
------------------------------------------------------
645        switch (elType)
646        {
647        case ID_CPE:
648           isCpe = 1;
649        case ID_SCE:
650           /* search in front channels */
651           for (i = 0; i < pPce->NumFrontChannelElements; i++) {
652              int heightLayer = pPce->FrontElementHeightInfo[i];       //
<================ if heightLayer is 3, buffer overflow
653              if (isCpe == pPce->FrontElementIsCpe[i] && pPce->FrontElementTagSelect[i]
== tag) {
654                 int h, elIdx = ec[heightLayer], chIdx = cc[heightLayer];
655                 AUDIO_CHANNEL_TYPE aChType = (AUDIO_CHANNEL_TYPE)((heightLayer<<4) |
ACT_FRONT);
656                 for (h = heightLayer-1; h >= 0; h-=1) {
657                    int el;
658                    /* Count front channels/elements */
659                    for (el = 0; el < pPce->NumFrontChannelElements; el+=1) {
660                       if (pPce->FrontElementHeightInfo[el] == h) {
661                          elIdx += 1;
662                          chIdx += (pPce->FrontElementIsCpe[el]) ? 2 : 1;
663                       }
664                    }
665                    /* Count side channels/elements */
666                    for (el = 0; el < pPce->NumSideChannelElements; el+=1) {
667                       if (pPce->SideElementHeightInfo[el] == h) {
668                          elIdx += 1;
669                          chIdx += (pPce->SideElementIsCpe[el]) ? 2 : 1;
670                       }
671                    }
672                    /* Count back channels/elements */
673                    for (el = 0; el < pPce->NumBackChannelElements; el+=1) {
674                       if (pPce->BackElementHeightInfo[el] == h) {
```

Reporter
xiangxiaobo1994@gmail.com

Type
Bug

Priority
P2

Severity
S2

Status
Fixed

Assignee
as...@google.com

Verifier
--

CC
an...@google.com
xiangxiaobo1994@gmail.com
Un-CC me

Android ID
70637599

ASR Eligible
Yes

ASR Payment
6000

ASR Severity
Critical

CVE
CVE-2017-13276

Resolution Notes
--

SPL
--

Status
01

Found In
--

Targeted To
--

Verified In
--

In Prod
OFF

```
675                         elIdx += 1;
676                         chIdx += (pPce->BackElementIsCpe[el]) ? 2 : 1;
677                      }
678                   }
679                   if (h == 0) {   /* normal height */
680                      elIdx += pPce->NumLfeChannelElements;
681                      chIdx += pPce->NumLfeChannelElements;
682                   }
683                }
684                chMapping[chIdx] = channelIdx;    // <========== the size of chMapping
is 8, chIdx can be larger than 8
685                chType[chIdx] = aChType;
686                chIndex[chIdx] = fc[heightLayer];
687                if (isCpe) {
688                   chMapping[chIdx+1] = channelIdx+1;
689                   chType[chIdx+1] = aChType;
690                   chIndex[chIdx+1] = fc[heightLayer]+1;
691                }
692                *elMapping = elIdx;
693                return 1;
694             }
695             ec[heightLayer] += 1;
696             if (pPce->FrontElementIsCpe[i]) {
697                cc[heightLayer] += 2;
698                fc[heightLayer] += 2;
699             } else {
700                cc[heightLayer] += 1;
701                fc[heightLayer] += 1;
702             }
703          }
-------------------------------------------------------
878   return 1;
879}
```

In the else branch above, there are five counter lists defined with size 3 in the stack. There logical addresses in memory are next to each other.

```
627         int ec[PC_NUM_HEIGHT_LAYER] = {0};
628         /* Channel counters */
629         int cc[PC_NUM_HEIGHT_LAYER] = {0};
630         int fc[PC_NUM_HEIGHT_LAYER] = {0};
631         int sc[PC_NUM_HEIGHT_LAYER] = {0};
632         int bc[PC_NUM_HEIGHT_LAYER] = {0};
633         int lc = 0;;
```

The heightLayer used to index the counters can be 3 If we well contruct the ASC data, because it is read from the MediaFuffer via FDKreadBits(bs,2), and two bits ranges from 0 to 3. This will result in four bytes(dword) overflow.

```
int heightLayer = pPce->FrontElementHeightInfo[i];
```

The Element Height Infos are assigned in /external/aac/libMpegTPDec/src/tpdec_asc.cpp:124, in CProgramConfig_ReadHeightExt function:

```
123static
124int CProgramConfig_ReadHeightExt(
125                                                CProgramConfig *pPce,
126                                                HANDLE_FDK_BITSTREAM bs,
127                                                int * const bytesAvailable,
128                                                const UINT alignmentAnchor
129                                              )
130{
------------------------------
141
142   if ( (startAnchor >= 24) && (*bytesAvailable >= 3)
143      && (FDKreadBits(bs,8) == PCE_HEIGHT_EXT_SYNC) )
144   {
145      int i;
146
147      for (i=0; i < pPce->NumFrontChannelElements; i++)
148      {
149         pPce->FrontElementHeightInfo[i] = (UCHAR) FDKreadBits(bs,2);      // <=======
range from 0..3
150      }
151      for (i=0; i < pPce->NumSideChannelElements; i++)
152      {
153         pPce->SideElementHeightInfo[i] = (UCHAR) FDKreadBits(bs,2);      // <=======
range from 0..3
154      }
155      for (i=0; i < pPce->NumBackChannelElements; i++)
156      {
157         pPce->BackElementHeightInfo[i] = (UCHAR) FDKreadBits(bs,2);      // <=======
range from 0..3
158      }
159      FDKbyteAlign(bs, alignmentAnchor);
------------------------------
```

176}

The address of ec[3] is exactly the same as cc[0], the address of cc[3] is actually fc[0], and so on.
As the counter is incremental, and will not get out of bound when used to index other arrays, but If we try to increase the ec[3] and cc[0] at the same time, cc[0] will be exceed its max value, and this will overflow other arrays in the function, and cause unexpected behavious.

[crash log]

I've tested this bug on the latest AOSP-7.1.2, My build fingerprint is as below:

```
generic_x86:/ # getprop ro.build.fingerprint
Android/aosp_x86/generic_x86:7.1.2/NZH54D/root12021126:eng/test-keys
```

In My asan-enabled devices, I've got the crash log below:

```
12-13 15:24:29.121   2080   2087 I             :
=================================================================
12-13 15:24:29.121   2080   2087 I             :
12-13 15:24:29.121   2080   2087 I             :
12-13 15:24:29.121   2080   2087 I             : ==2080==ERROR: AddressSanitizer: stack-
buffer-overflow on address 0xb05ed79c at pc 0xae7731dd bp 0xb05ed768 sp 0xb05ed74c
12-13 15:24:29.121   2080   2087 I             :
12-13 15:24:29.121   2080   2087 I             :
12-13 15:24:29.121   2080   2087 I             : READ of size 4 at 0xb05ed79c thread T2
(gle.aac.decoder)
12-13 15:24:29.121   2080   2087 I             :
12-13 15:24:29.122   2080   2087 I             :       #0 0xae7731dc
  (/system/lib/libstagefright_soft_aacdec.so+0xd91dc)
12-13 15:24:29.122   2080   2087 I             :
12-13 15:24:29.122   2080   2087 I             :       #1 0xae71cd1b
  (/system/lib/libstagefright_soft_aacdec.so+0x82d1b)
12-13 15:24:29.122   2080   2087 I             :
12-13 15:24:29.122   2080   2087 I             :       #2 0xae6b82ef
  (/system/lib/libstagefright_soft_aacdec.so+0x1e2ef)
12-13 15:24:29.122   2080   2087 I             :
12-13 15:24:29.122   2080   2087 I             :       #3 0xae6ad36a
  (/system/lib/libstagefright_soft_aacdec.so+0x1336a)
12-13 15:24:29.122   2080   2087 I             :
12-13 15:24:29.122   2080   2087 I             :       #4 0xb2d60503
  (/system/lib/libstagefright_omx.so+0x5a503)
12-13 15:24:29.122   2080   2087 I             :
12-13 15:24:29.122   2080   2087 I             :       #5 0xb2d65e30
  (/system/lib/libstagefright_omx.so+0x5fe30)
12-13 15:24:29.122   2080   2087 I             :
12-13 15:24:29.122   2080   2087 I             :       #6 0xb26bd093
  (/system/lib/libstagefright_foundation.so+0x10093)
12-13 15:24:29.122   2080   2087 I             :
12-13 15:24:29.122   2080   2087 I             :       #7 0xb26c1b8c
  (/system/lib/libstagefright_foundation.so+0x14b8c)
12-13 15:24:29.122   2080   2087 I             :
12-13 15:24:29.122   2080   2087 I             :       #8 0xb26be5fc
  (/system/lib/libstagefright_foundation.so+0x115fc)
12-13 15:24:29.122   2080   2087 I             :
12-13 15:24:29.122   2080   2087 I             :       #9 0xb26bf0b3
  (/system/lib/libstagefright_foundation.so+0x120b3)
12-13 15:24:29.122   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :       #10 0xb29aa02f
  (/system/lib/libutils.so+0x1202f)
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :       #11 0xb29a9883
  (/system/lib/libutils.so+0x11883)
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :       #12 0xb21f6b43
  (/system/lib/libclang_rt.asan-i686-android.so+0x88b43)
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :       #13 0xb21d375a
  (/system/lib/libclang_rt.asan-i686-android.so+0x6575a)
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :       #14 0xb2b3dfe2
  (/system/lib/libc.so+0x74fe2)
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :       #15 0xb2ae929e
  (/system/lib/libc.so+0x2029e)
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :       #16 0xb2ae7076
  (/system/lib/libc.so+0x1e076)
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             : Address 0xb05ed79c is located in stack of
thread T2 (gle.aac.decoder)
12-13 15:24:29.123   2080   2087 I             :   at offset 28 in frame
```

```
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :       #0 0xae77025f
  (/system/lib/libstagefright_soft_aacdec.so+0xd625f)
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :    This frame has 5 object(s):
12-13 15:24:29.123   2080   2087 I             :
12-13 15:24:29.123   2080   2087 I             :       [16, 28) 'ec' <== Memory access at
offset 28 overflows this variable
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             :       [48, 60) 'cc'
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             :       [80, 92) 'fc'
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             :       [112, 124) 'sc'
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             :       [144, 156) 'bc'
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             : HINT: this may be a false positive if your
program uses some custom stack unwind mechanism or swapcontext
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             :          (longjmp and C++ exceptions *are*
supported)
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             : Thread T2 (gle.aac.decoder) created by T1
(Binder:2080_1) here:
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             :       #0 0xb21d35d5
  (/system/lib/libclang_rt.asan-i686-android.so+0x655d5)
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             :       #1 0xb29a96f4
  (/system/lib/libutils.so+0x116f4)
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             :       #2 0xb29a9ed5
  (/system/lib/libutils.so+0x11ed5)
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             :       #3 0xb26be35f
  (/system/lib/libstagefright_foundation.so+0x1135f)
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             :       #4 0xb2d5cb41
  (/system/lib/libstagefright_omx.so+0x56b41)
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             :       #5 0xae6a9086
  (/system/lib/libstagefright_soft_aacdec.so+0xf086)
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             :       #6 0xae6b24e9
  (/system/lib/libstagefright_soft_aacdec.so+0x184e9)
12-13 15:24:29.124   2080   2087 I             :
12-13 15:24:29.124   2080   2087 I             :       #7 0xb2d67755
  (/system/lib/libstagefright_omx.so+0x61755)
12-13 15:24:29.125   2080   2087 I             :
12-13 15:24:29.125   2080   2087 I             :       #8 0xb2d3df8c
  (/system/lib/libstagefright_omx.so+0x37f8c)
12-13 15:24:29.125   2080   2087 I             :
12-13 15:24:29.125   2080   2087 I             :       #9 0xb2d32d4d
  (/system/lib/libstagefright_omx.so+0x2cd4d)
12-13 15:24:29.125   2080   2087 I             :
12-13 15:24:29.125   2080   2087 I             :       #10 0xb20ddb3e
  (/system/lib/libmedia.so+0xefb3e)
12-13 15:24:29.125   2080   2087 I             :
12-13 15:24:29.125   2080   2087 I             :       #11 0xb20def21
  (/system/lib/libmedia.so+0xf0f21)
12-13 15:24:29.125   2080   2087 I             :
12-13 15:24:29.125   2080   2087 I             :       #12 0xb2a47fd6
  (/system/lib/libbinder.so+0x38fd6)
12-13 15:24:29.125   2080   2087 I             :
12-13 15:24:29.125   2080   2087 I             :       #13 0xb2a56dc9
  (/system/lib/libbinder.so+0x47dc9)
12-13 15:24:29.125   2080   2087 I             :
12-13 15:24:29.125   2080   2087 I             :       #14 0xb2a568cb
  (/system/lib/libbinder.so+0x478cb)
12-13 15:24:29.125   2080   2087 I             :
12-13 15:24:29.125   2080   2087 I             :       #15 0xb2a5707f
  (/system/lib/libbinder.so+0x4807f)
12-13 15:24:29.125   2080   2087 I             :
12-13 15:24:29.125   2080   2087 I             :       #16 0xb2a7de5e
  (/system/lib/libbinder.so+0x6ee5e)
12-13 15:24:29.125   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :       #17 0xb29aa095
  (/system/lib/libutils.so+0x12095)
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :       #18 0xb29a9883
  (/system/lib/libutils.so+0x11883)
```

```
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :       #19 0xb21f6b43
  (/system/lib/libclang_rt.asan-i686-android.so+0x88b43)
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :       #20 0xb21d375a
  (/system/lib/libclang_rt.asan-i686-android.so+0x6575a)
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :       #21 0xb2b3dfe2
  (/system/lib/libc.so+0x74fe2)
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :       #22 0xb2ae929e
  (/system/lib/libc.so+0x2029e)
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :       #23 0xb2ae7076
  (/system/lib/libc.so+0x1e076)
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             : Thread T1 (Binder:2080_1) created by T0
here:
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :       #0 0xb21d35d5
  (/system/lib/libclang_rt.asan-i686-android.so+0x655d5)
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :       #1 0xb29a96f4
  (/system/lib/libutils.so+0x116f4)
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :       #2 0xb29a9eaa
  (/system/lib/libutils.so+0x11eaa)
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :       #3 0xb2a7d08f
  (/system/lib/libbinder.so+0x6e08f)
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :       #4 0xb2a7cf83
  (/system/lib/libbinder.so+0x6df83)
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :       #5 0xb30a1e56
  (/system/bin/mediacodec+0xe56)
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.126   2080   2087 I             :       #6 0xb2ade32c
  (/system/lib/libc.so+0x1532c)
12-13 15:24:29.126   2080   2087 I             :
12-13 15:24:29.127   2080   2087 I             :       #7 0xb30a1c62
  (/system/bin/mediacodec+0xc62)
12-13 15:24:29.127   2080   2087 I             :
12-13 15:24:29.127   2080   2087 I             :       #8 0x0   (<unknown module>)
12-13 15:24:29.127   2080   2087 I             :
12-13 15:24:29.127   2080   2087 I             :
12-13 15:24:29.127   2080   2087 I             :
12-13 15:24:29.127   2080   2087 I             : SUMMARY: AddressSanitizer: stack-buffer-
overflow (/system/lib/libstagefright_soft_aacdec.so+0xd91dc)
12-13 15:24:29.127   2080   2087 I             :
12-13 15:24:29.127   2080   2087 I             : Shadow bytes around the buggy address:
12-13 15:24:29.127   2080   2087 I             :    0x160bdaa0: 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00
12-13 15:24:29.127   2080   2087 I             :    0x160bdab0: 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00
12-13 15:24:29.127   2080   2087 I             :    0x160bdac0: 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00
12-13 15:24:29.127   2080   2087 I             :    0x160bdad0: 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00
12-13 15:24:29.127   2080   2087 I             :    0x160bdae0: 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00
12-13 15:24:29.127   2080   2087 I             : =>0x160bdaf0: f1 f1 00[04]f2 f2 00 04 f2 f2
00 04 f2 f2 00 04
12-13 15:24:29.127   2080   2087 I             :    0x160bdb00: f2 f2 00 04 f3 f3 00 00 00 00
00 00 00 00 00 00
12-13 15:24:29.127   2080   2087 I             :    0x160bdb10: 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00
12-13 15:24:29.127   2080   2087 I             :    0x160bdb20: 00 00 00 00 00 00 00 00 f1 f1
00 00 00 00 00 00
12-13 15:24:29.127   2080   2087 I             :    0x160bdb30: 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00
12-13 15:24:29.127   2080   2087 I             :    0x160bdb40: 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00
12-13 15:24:29.127   2080   2087 I             : Shadow byte legend (one shadow byte
represents 8 application bytes):
12-13 15:24:29.127   2080   2087 I             :    Addressable:               00
12-13 15:24:29.127   2080   2087 I             :    Partially addressable: 01 02 03 04 05 06
07
12-13 15:24:29.127   2080   2087 I             :    Heap left redzone:         fa
12-13 15:24:29.127   2080   2087 I             :    Heap right redzone:        fb
12-13 15:24:29.127   2080   2087 I             :    Freed heap region:         fd
12-13 15:24:29.127   2080   2087 I             :    Stack left redzone:        f1
12-13 15:24:29.127   2080   2087 I             :    Stack mid redzone:         f2
```

```
12-13 15:24:29.127   2080   2087 I             :    Stack right redzone:       f3
12-13 15:24:29.127   2080   2087 I             :    Stack partial redzone:    f4
12-13 15:24:29.127   2080   2087 I             :    Stack after return:         f5
12-13 15:24:29.127   2080   2087 I             :    Stack use after scope:    f8
12-13 15:24:29.127   2080   2087 I             :    Global redzone:              f9
12-13 15:24:29.127   2080   2087 I             :    Global init order:          f6
12-13 15:24:29.127   2080   2087 I             :    Poisoned by user:           f7
12-13 15:24:29.127   2080   2087 I             :    Container overflow:         fc
12-13 15:24:29.127   2080   2087 I             :    Array cookie:                  ac
12-13 15:24:29.127   2080   2087 I             :    Intra object redzone:      bb
12-13 15:24:29.127   2080   2087 I             :    ASan internal:                fe
12-13 15:24:29.127   2080   2087 I             :    Left alloca redzone:       ca
12-13 15:24:29.127   2080   2087 I             :    Right alloca redzone:      cb
12-13 15:24:29.127   2080   2087 I             :
12-13 15:24:29.127   2080   2087 I             : ==2080==ABORTING
12-13 15:24:29.128   2080   2087 I             :
12-13 15:24:29.128    976    976 I ServiceManager: service 'media.codec' died
```

[poc]
I've write a script to generate a m4a file that can trigger this vulnerability via stagefright cmd tool. I've attached both the m4a file and the script.

There are two points in my scripts:

In gen_esds() function , we can assign how many channel elements are there , In my script, I simply made one front channel element.

```
369             # this three number should not be zero, to trigger the vulnerability
370             num_front_channel_ele = to_bin_str(1, 4)
371             num_side_channel_ele = to_bin_str(0, 4)
372             num_back_channel_ele = to_bin_str(0, 4)
```

right under the num_front_channel_ele, we can set the height_infos of each elements to 3, this will eventually results in the stack overflow we mentioned in this report.

```
414             for i in range( int(num_front_channel_ele,2)) :
415                   front_element_height_info_i = to_bin_str(3,2)
```

If you want to modify the script to make more overflows, you should attention to the tag_size , sub_tag_size and sub_sub_tag_size in gen_esds() function, they should be adjusted according to the size of esds chunk. Also, the CRC should be fixed too.

[how to fix]
Set PC_NUM_HEIGHT_LAYER to 4 will simply fix this vulnerability, because no other function uses these height infos. And the definition of macros are all like 4,8,16 to avoid one element overflow, this will satisfy the code style:

```
#define PC_FSB_CHANNELS_MAX 16 /* Front/Side/Back channels */
#define PC_LFE_CHANNELS_MAX 4
#define PC_ASSOCDATA_MAX      8
#define PC_CCEL_MAX             16 /* CC elements */
#define PC_COMMENTLENGTH      256
#define PC_NUM_HEIGHT_LAYER 4   // update this value from 3 to 4
```

I've attached a patch for this issue, hope this will help.

Best Regards,
Elphet and Gong Guang of Alpha Team, Qihoo 360 Technology Co. Ltd.

Component:	Public Trackers > Android External Security Reports	04:01PM
Status:	New	04:01PM
Reporter:	xiangxiaobo1994@gmail.com	04:01PM
+CC:	an...@google.com , xiangxiaobo1994@gmail.com	04:01PM
Type:	Bug	04:01PM
Priority:	P2	04:01PM
Severity:	S2	04:01PM
Title:	Security Report - [Stack overflow in external/aac/libMpegTPDec/tpdec_asc.cpp]	04:01PM
Status:	01	04:01PM

output.m4a
544 B Download

m4a-maker.py
19 KB View Download

tpdec_lib.h.patch
848 B View Download

mi...@google.com <mi...@google.com> #2 Dec 14, 2017 08:08AM

Assigned to as...@google.com.

+Blocked by:	70637599	07:58AM

Thank you for submitting this report. We've filed an internal report for the Android engineering team to investigate further (specified by the Android ID label). Please follow coordinated disclosure practices, such as keeping this report confidential until we have had time to assess your issue, and if necessary, release a update for Android devices. 08:08AM

The typical lifecycle for a confirmed security vulnerability is as follows:

1. Initial severity rating assessment (subject to change after review by component owners) (1)
2. Development of an update
3. Assignment of CVE
4. Shared under NDA, as part of coordinated disclosure, to Android partners for remediation
5. Release in a public Android security bulletin
6. Android Security Rewards payment (if applicable)

Note: Most of these steps will be communicated through the labels found on the sidebar.

Taking the following actions up front will assist us in timely processing of your report:

1. Sign the Google Contributor License Agreement (2)
2. Let us know how you would like attribution to be noted (if necessary)
3. Provide a PoC (Required for complete rewards submissions)

Thank you,
Android Security Team

(1) Android severity guidelines: https://source.android.com/security/overview/updates-resources.html
(2) Contributor license agreement: https://cla.developers.google.com/clas

Status:	New	Assigned	08:08AM
Assignee:	<none>	as...@google.com	
Android ID:	<none>	70637599	

---

tk...@google.com <tk...@google.com> Dec 21, 2017 07:00AM

+Hotlist:	468630		07:00AM
ASR Severity:	<none>	Critical	

---

mj...@google.com <mj...@google.com> #3 Feb 22, 2018 07:23AM

Hello,

We will be releasing a patch for this issue in an upcoming bulletin. It will first be released to partners, then in the Android Security bulletin the following month.

If you haven't already, please complete the Google Contributor License Agreement for Individuals, so we can use your patch and test code:
https://cla.developers.google.com/clas

We'd also like to recognize your contribution at https://source.android.com/security/overview/acknowledgements and in the security bulletin, please let us know how you would like your name and information to appear.

We may also make this bug publicly accessible when the fix is submitted to AOSP. Please let us know if you would like to keep the bug private instead.

Thanks,

Android Security Team

---

mr...@google.com <mr...@google.com> #4 Apr 3, 2018 06:44AM

Congratulations! The rewards committee decided to reward you USD $6000 for reporting this Critical severity vulnerability. We are paying for the bug report and PoC.

To collect the reward, if you haven't already, please complete the Android Contributor License Agreement for Individuals, so we can use your test code:
https://cla.developers.google.com/clas

You will receive an email with details on the next steps to collect the reward. As a reminder, please continue to keep this vulnerability confidential until it is officially disclosed (we will update you when disclosure takes place).

Thanks,
Android Security Team

---

br...@google.com <br...@google.com> May 24, 2018 03:22AM

Marked as fixed.

CVE:	<none>	CVE-2017-13276	03:22AM
Status:	Assigned	Fixed	03:22AM

---

mi...@google.com <mi...@google.com> Jul 24, 2018 08:06AM

-Hotlist:	468630		08:06AM

br...@google.com <br...@google.com> Oct 16, 2018 03:29AM

ASR Eligible:	<none>	Yes	03:29AM
ASR Payment:	<none>	6000	03:29AM