crash replay

```
sec@ubuntu160432:~/tmp/mp3gain-1_5_2_r2$ gdb ./mp3gain
GNU gdb (Ubuntu 7.11.1-0ubuntu1~16.5) 7.11.1
Copyright (C) 2016 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "i686-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./mp3gain...done.
(gdb) set args ./poc
(gdb) r
Starting program: /home/sec/tmp/mp3gain-1_5_2_r2/mp3gain ./poc
./poc

Program received signal SIGSEGV, Segmentation fault.
0x0805aecb in III_dequantize_sample (xr=0x8095940 <hybridIn.5353+4608>,
scf=0xbfff287c, gr_infos=0x845ebd8 <sideinfo+216>,
    sfreq=6, part2bits=17) at mpglibDBL/layer3.c:904
904           v = gr_infos->pow2gain[((*scf++) + (*pretab++)) << shift];

(gdb) list
899       for(;lp;lp--,mc--) {
900         int x,y;
901
902         if(!mc) {
903           mc = *m++;
904           v = gr_infos->pow2gain[((*scf++) + (*pretab++)) << shift];
905           cb = *m++;
906         }
907         {
908           register short *val = (short *)h->table;
(gdb) p lp
$5 = -285
(gdb)
```

Vuln analysis

Read access violation in III_dequantize_sample function in mpglibDBL/lyaer3.c in mp3gain. Analysis about the vuln in the comments.

```
/*The related source code in layer3.c*/
...
static const int pretab1 [22] = {0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,2,2,3,3,3,2,0};
/* char enough ? */
```

```
static const int pretab2 [22] = {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

/*
 *  ---> the vuln function: III_dequantize_sample
 */
static int III_dequantize_sample(real xr[SBLIMIT][SSLIMIT],int *scf,
   struct gr_info_s *gr_infos,int sfreq,int part2bits)
{
  int shift = 1 + gr_infos->scalefac_scale;
  real *xrpnt = (real *) xr;
  int l[3],l3;
  int part2remain = gr_infos->part2_3_length - part2bits;
  int *me;
...
/* The elements of l is used for loop control( as variable lp) in the following
code*/
    if(bv <= region1) {
      l[0] = bv; l[1] = 0; l[2] = 0;
    }
    else {
      l[0] = region1;
      if(bv <= region2) {
        l[1] = bv - l[0];  l[2] = 0;
      }
      else {
        l[1] = region2 - l[0]; l[2] = bv - region2;
      }
    }
  }
...
    // in line 899
    for(i=0;i<3;i++) {
      int lp = l[i];
      struct newhuff *h = (struct newhuff *)(ht+gr_infos->table_select[i]);

      for(;lp;lp--,mc--) { //---> root cause: Ignore that if lp is a negative
number, the loop won't stop until crash(in gdb info)
        int x,y;

        if(!mc) {
          mc = *m++;
          v = gr_infos->pow2gain[((*scf++) + (*pretab++)) << shift];// --->
crash point, pretab is a pointer to pretab2, the non-stop loop will lead a read
access violation and get an exceptionally large index after ((*scf++) +
(*pretab++)) << shift, then the following gr_infos->pow2gain[index] will read an
invalid address which lead to the crash.
          cb = *m++;
        }
        {
 ...
```