**3GPP TSG-SA4 Meeting #122 *S4-230354***

 **Athens, Greece, February 20 - 24, 2023 *(Rev 1 S4-230053)***

|  |
| --- |
| *CR-Form-v12.2* |
| **CHANGE REQUEST** |
|  |
|  | **26.204** | **CR** | **0021** | **rev** | **-** | **Current version:** | **17.0.0** |  |
|  |
| *For* [***HELP***](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form: comprehensive instructions can be found at* [*http://www.3gpp.org/Change-Requests*](http://www.3gpp.org/Change-Requests)*.* |
|  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Proposed change affects:*** | UICC apps |  | ME | **X** | Radio Access Network |  | Core Network | **X** |

|  |
| --- |
|  |
| ***Title:***  | Correction to Undefined Behaviour caused by out-of-bounds pointer arithmetic |
|  |  |
| ***Source to WG:*** | Google |
| ***Source to TSG:*** | SA4 |
|  |  |
| ***Work item code:*** | 34004 |  | ***Date:*** | 2023-02-10 |
|  |  |  |  |  |
| ***Category:*** | **F** |  | ***Release:*** | Rel-17 |
|  | *Use one of the following categories:****F*** *(correction)****A*** *(mirror corresponding to a change in an earlier release)****B*** *(addition of feature),* ***C*** *(functional modification of feature)****D*** *(editorial modification)*Detailed explanations of the above categories canbe found in 3GPP [TR 21.900](http://www.3gpp.org/ftp/Specs/html-info/21900.htm). | *Use one of the following releases:Rel-8 (Release 8)Rel-9 (Release 9)Rel-10 (Release 10)Rel-11 (Release 11)…Rel-16 (Release 16)Rel-17 (Release 17)Rel-18 (Release 18)Rel-19 (Release 19)* |
|  |  |
| ***Reason for change:*** | A common pattern is used in the code to obtain offset pointers on arrays, which is valid and legal as long as it points to an element in the array or one past the last one. Our UBSAN-enabled tests caught a few cases when the offset is negative, causing an UB. |
|  |  |
| ***Summary of change:*** | The CR corrects the UB by obtaining the offset and passing it around alongside the original array to avoid out-of-bounds pointer arithmetic. |
|  |  |
| ***Consequences if not approved:*** | An Undefined Behaviour in the code, UBSAN failures. |
|  |  |
| ***Clauses affected:*** | c-code/enc\_acelp.cc-code/enc\_gain.c |
|  |  |
|  | **Y** | **N** |  |  |
| ***Other specs*** |  | **X** |  Other core specifications  | TS/TR ... CR ...  |
| ***affected:*** |  | **X** |  Test specifications | TS/TR ... CR ...  |
| ***(show related CRs)*** |  | **X** |  O&M Specifications | TS/TR ... CR ...  |
|  |  |
| ***Other comments:*** |  |
|  |  |
| ***This CR's revision history:*** |  |

**\*\*\* Start change 1 \*\*\***

--- enc\_acelp.c 2009-11-25 18:54:04.000000000 -0800

+++ enc\_acelp.c 2022-07-22 20:43:45.000000000 -0700

@@ -1145,6 +1145,7 @@

 Float32 val;

 Float32 s, cor;

 Float32 \*p0, \*p1, \*p2, \*p3, \*psign;

+ Word32 p0\_offset;

 Float32 \*h, \*h\_inv, \*ptr\_h1, \*ptr\_h2, \*ptr\_hf;

 switch (nbbits)

@@ -1369,6 +1370,7 @@

 pos = 256 - 1;

 ptr\_hf = h + 1;

+ p0\_offset = -16;

 for(k = 0; k < 16; k++)

 {

@@ -1376,7 +1378,7 @@

 p3 = &rrixiy[2][pos];

 p2 = &rrixiy[1][pos];

 p1 = &rrixiy[0][pos];

- p0 = &rrixiy[3][pos - 16];

+ p0 = &rrixiy[3][pos];

 cor = 0.0F;

 ptr\_h1 = h;

@@ -1399,7 +1401,7 @@

 cor += (\*ptr\_h1) \* (\*ptr\_h2);

 ptr\_h1++;

 ptr\_h2++;

- \*p0 = cor;

+ \*(p0 + p0\_offset) = cor;

 p3 -= (16 + 1);

 p2 -= (16 + 1);

--- enc\_gain.c 2018-01-19 15:20:47.000000000 -0800

+++ enc\_gain.c 2022-07-22 20:43:45.000000000 -0700

@@ -448,8 +448,8 @@

 \* void

 \*/

 static void E\_GAIN\_norm\_corr(Float32 exc[], Float32 xn[], Float32 h[],

- Word32 t\_min, Word32 t\_max, Float32 corr\_norm[])

-{

+ Word32 t\_min, Word32 t\_max, Float32 corr\_norm[],

+ Word32 corr\_offset) {

 Float32 excf[L\_SUBFR]; /\* filtered past excitation \*/

 Float32 alp, ps, norm;

 Word32 t, j, k;

@@ -482,7 +482,7 @@

 /\* Normalize correlation = correlation \* (1/sqrt(energy)) \*/

- corr\_norm[t] = ps \* norm;

+ corr\_norm[corr\_offset + t] = ps \* norm;

 /\* update the filtered excitation excf[] for the next iteration \*/

@@ -566,7 +566,7 @@

 {

 Float32 corr\_v[15 + 2 \* L\_INTERPOL1 + 1];

 Float32 cor\_max, max, temp;

- Float32 \*corr;

+ Word32 corr\_v\_offset;

 Word32 i, fraction, step;

 Word32 t0, t\_min, t\_max;

@@ -575,21 +575,20 @@

 t\_min = t0\_min - L\_INTERPOL1;

 t\_max = t0\_max + L\_INTERPOL1;

- /\* allocate memory to normalized correlation vector \*/

- corr = &corr\_v[-t\_min]; /\* corr[t\_min..t\_max] \*/

+ corr\_v\_offset = -t\_min;

 /\* Compute normalized correlation between target and filtered excitation \*/

- E\_GAIN\_norm\_corr(exc, xn, h, t\_min, t\_max, corr);

+ E\_GAIN\_norm\_corr(exc, xn, h, t\_min, t\_max, corr\_v, corr\_v\_offset);

 /\* find integer pitch \*/

- max = corr[t0\_min];

+ max = corr\_v[corr\_v\_offset + t0\_min];

 t0 = t0\_min;

 for(i = t0\_min + 1; i <= t0\_max; i++)

 {

- if( corr[i] > max)

+ if( corr\_v[corr\_v\_offset + i] > max)

 {

- max = corr[i];

+ max = corr\_v[corr\_v\_offset + i];

 t0 = i;

 }

 }

@@ -621,11 +620,12 @@

 fraction = 0;

 }

- cor\_max = E\_GAIN\_norm\_corr\_interpolate(&corr[t0], fraction);

+ cor\_max =

+ E\_GAIN\_norm\_corr\_interpolate(&corr\_v[corr\_v\_offset + t0], fraction);

 for (i = (fraction + step); i <= 3; i += step)

 {

- temp = E\_GAIN\_norm\_corr\_interpolate(&corr[t0], i);

+ temp = E\_GAIN\_norm\_corr\_interpolate(&corr\_v[corr\_v\_offset + t0], i);

 if (temp > cor\_max)

 {

**\*\*\* End change 1 \*\*\***