ezXML Bugs

Status: Beta
Brought to you by: voisine

#26 Out-of-bounds write in ezxml_internal_dtd()

Milestone: v1.0_(example) **Status:** open **Owner:** nobody **Labels:** None
Priority: 5
Updated: 2021-10-25 **Created:** 2021-04-15 **Creator:** rc0r **Private:** No

Description

Function `ezxml_internal_dtd()` performs incorrect memory handling while parsing crafted XML files which leads to an out-of-bounds write of a one byte constant.

This leads to a crash because the write attempts to write past the `mmap()` 'ed memory region used for reading the crafted XML file in case `EZXML_NOMMAP` is not set. Memory mapping occurs in `ezxml_parse_fd()` :

```
#ifndef EZXML_NOMMAP
    l = (st.st_size + sysconf(_SC_PAGESIZE) - 1) & ~(sysconf(_SC_PAGESIZE) -1);
    if ((m = mmap(NULL, l, PROT_READ | PROT_WRITE, MAP_PRIVATE, fd, 0)) !=
        MAP_FAILED) {
        madvise(m, l, MADV_SEQUENTIAL); // optimize for sequential access
        root = (ezxml_root_t)ezxml_parse_str(m, st.st_size);
        madvise(m, root->len = l, MADV_NORMAL); // put it back to normal
    }
    else { // mmap failed, read file into memory
#endif // EZXML_NOMMAP
```

The invalid write occurs in case the call to `strcspn(n, EZXML_WS)` in the code below does not find any whitespace characters (defined in `EZXML_WS`) in `n` . Then the length of the string `n` is returned. Note that `n` is just a pointer to the original string `s` . Thus the addition `n + strcspn(...)` points the first byte after `s` .

```
else if (! strncmp(s, "<!ENTITY", 8)) { // parse entity definitions
    c = s += strspn(s + 8, EZXML_WS) + 8; // skip white space separator
    n = s + strspn(s, EZXML_WS "%"); // find name
    *(s = n + strcspn(n, EZXML_WS)) = ';'; // append ; to name
```

MITRE assigned **CVE-2021-31229** for this issue.

Debugging Output

```
$ gdb ~/tmp/ezxml/ezxml_test CVE-2021-31229-OOBW-000.sample
>>> r
Program received signal SIGSEGV, Segmentation fault.
0x0000555555556698 in ezxml_internal_dtd (root=root@entry=0x55555555d2a0, s=0x7ffff7fc6000 
333                 *(s = n + strcspn(n, EZXML_WS)) = ';'; // append ; to name

Assembly
 0x0000555555556680  4c 8d 7c 05 00        ezxml_internal_dtd+192 lea    0x0(%rbp,%rax,1),%
 0x0000555555556685  4c 89 ff              ezxml_internal_dtd+197 mov    %r15,%rdi
 0x0000555555556688  e8 83 ea ff ff        ezxml_internal_dtd+200 call   0x555555555110 <st
 0x000055555555668d  48 8d 35 a6 39 00 00  ezxml_internal_dtd+205 lea    0x39a6(%rip),%rsi
 0x0000555555556694  4d 8d 34 07           ezxml_internal_dtd+212 lea    (%r15,%rax,1),%r14
>0x0000555555556698  41 c6 06 3b           ezxml_internal_dtd+216 movb   $0x3b,(%r14)
 0x000055555555669c  49 8d 7e 01           ezxml_internal_dtd+220 lea    0x1(%r14),%rdi
 0x00005555555566a0  e8 5b ea ff ff        ezxml_internal_dtd+224 call   0x555555555100 <st
 0x00005555555566a5  4d 8d 4c 06 01        ezxml_internal_dtd+229 lea    0x1(%r14,%rax,1),%
 0x00005555555566aa  45 0f b6 21           ezxml_internal_dtd+234 movzbl (%r9),%r12d

>>> i r
rax            0x1bd               445
rbx            0x0                 0
rcx            0x0                 0
rdx            0x0                 0
rsi            0x55555555a03a      93824992256058
rdi            0x7ffff7fc5e43      140737353899587
rbp            0x7ffff7fc5e43      0x7ffff7fc5e43
rsp            0x7fffffffd810      0x7fffffffd810
r8             0x55555555a030      93824992256048
r9             0x7ffff7fc6000      140737353900032
r10            0xfffffffffffff90a  -1782
r11            0x7ffff7e3ac40      140737352281152
r12            0x7ffff7fc5e3b      140737353899579
r13            0x7ffff7fc5def      140737353899503
r14            0x7ffff7fc6000      140737353900032
r15            0x7ffff7fc5e43      140737353899587
rip            0x555555556698      0x555555556698 <ezxml_internal_dtd+216>
eflags         0x10216             [ PF AF IF RF ]
cs             0x33                51
ss             0x2b                43
ds             0x0                 0
es             0x0                 0
fs             0x0                 0
gs             0x0                 0

>>> bt
#0  0x0000555555556698 in ezxml_internal_dtd (root=root@entry=0x55555555d2a0, s=0x7ffff7fc6
#1  0x00005555555588d1 in ezxml_parse_str (s=<optimized out>, s@entry=0x7ffff7fc3000 "<?xdl
#2  0x0000555555558b59 in ezxml_parse_fd (fd=fd@entry=3) at ezxml.c:641
#3  0x0000555555558bfb in ezxml_parse_file (file=<optimized out>) at ezxml.c:659
#4  0x000055555555526a in main (argc=<optimized out>, argv=<optimized out>) at ezxml.c:1008

>>> p s
$1 = 0x7ffff7fc6000 <error: Cannot access memory at address 0x7ffff7fc6000>

>>> p (int) strlen(s)
$2 = 0

>>> p c
$3 = 0x7ffff7fc5e43 "<!D>]PE\377V[ENT<!ATTLISTAY0]><!DO>N<!>N<!]><!]>N<7]><#]><ml?>N<!D~><!
>>> p n
$4 = 0x7ffff7fc5e43 "<!D>]PE\377V[ENT<!ATTLISTAY0]><!DO>N<!>N<!]><!]>N<7]><#]><ml?>N<!D~><!

>>> up
#1  0x00005555555588d1 in ezxml_parse_str (s=<optimized out>, s@entry=0x7ffff7fc3000 "<?xdl
576                 if (1 && ! ezxml_internal_dtd(root, d, s++ - d)) return &root->xml;

>>> up
#2  0x0000555555558b59 in ezxml_parse_fd (fd=fd@entry=3) at ezxml.c:641
641             root = (ezxml_root_t)ezxml_parse_str(m, st.st_size);

>>> p m
$5 = (void *) 0x7ffff7fc3000    // start of mmap()ed memory region

>>> p m + st.st_size
$6 = (void *) 0x7ffff7fc6000
```

Reproduction

```
$ cd ~/tmp/ezxml
$ gcc -Wall -O2 -DEZXML_TEST -g -ggdb -o ezxml_test ezxml.c
$ gdb ~/tmp/ezxml/ezxml_test CVE-2021-31229-OOBW-000.sample
```

Patch

```
--- ezxml.c 2006-06-08 04:33:38.000000000 +0200
+++ ezxml-fixed.c   2021-04-15 15:40:38.054755080 +0200
@@ -320,6 +320,7 @@
 {
     char q, *c, *t, *n = NULL, *v, **ent, **pe;
     int i, j;
+    size_t n_len, n_off;

     pe = memcpy(malloc(sizeof(EZXML_NIL)), EZXML_NIL, sizeof(EZXML_NIL));

@@ -330,7 +331,13 @@
         else if (! strncmp(s, "<!ENTITY", 8)) { // parse entity definitions
             c = s += strspn(s + 8, EZXML_WS) + 8; // skip white space separator
             n = s + strspn(s, EZXML_WS "%"); // find name
-            *(s = n + strcspn(n, EZXML_WS)) = ';'; // append ; to name
+            n_len = strlen(n);
+            n_off = strcspn(n, EZXML_WS);
+            if(n_off >= n_len) {
+                ezxml_err(root, t, "write past buffer (<!ENTITY)");
+                break;
+            }
+            *(s = n + n_off) = ';'; // append ; to name

             v = s + strspn(s + 1, EZXML_WS) + 1; // find value
             if ((q = *(v++)) != '"' && q != '\'') { // skip externals
```

Files

- CVE-2021-31229-OOBW-000.sample (Crash sample)
- CVE-2021-31229-OOBW-000.patch (Patch adding size check)

2 Attachments

CVE-2021-31229-OOBW-000.patch

CVE-2021-31229-OOBW-000.sample

Discussion

rc0r - *2021-04-26*

Please **do not** apply the originally proposed patch `CVE-2021-31229-OOBW-000.patch` as it calls `ezxml_err()` with a potentially corrupt `t` parameter leading to another crash (that does not occur without the patch applied). To avoid this newly introduced issue the offending call to `ezxml_err()` was changed to not include the potentially corrupt `t`:

```
--- ezxml.c 2006-06-08 04:33:38.000000000 +0200
+++ ezxml-fixed.c   2021-04-15 15:40:38.054755080 +0200
@@ -320,6 +320,7 @@
 {
     char q, *c, *t, *n = NULL, *v, **ent, **pe;
     int i, j;
+    size_t n_len, n_off;

     pe = memcpy(malloc(sizeof(EZXML_NIL)), EZXML_NIL, sizeof(EZXML_NIL));

@@ -330,7 +331,13 @@
         else if (! strncmp(s, "<!ENTITY", 8)) { // parse entity definitic
             c = s += strspn(s + 8, EZXML_WS) + 8; // skip white space sep
             n = s + strspn(s, EZXML_WS "%"); // find name
-            *(s = n + strcspn(n, EZXML_WS)) = ';'; // append ; to name
+            n_len = strlen(n);
+            n_off = strcspn(n, EZXML_WS);
+            if(n_off >= n_len) {
+                ezxml_err(root, NULL, "write past buffer (<!ENTITY)");
+                break;
+            }
+            *(s = n + n_off) = ';'; // append ; to name

             v = s + strspn(s + 1, EZXML_WS) + 1; // find value
             if ((q = *(v++)) != '"' && q != '\'') { // skip externals
```

CVE-2021-31229-OOBW-001.patch

Egbert Eich - *2021-10-25*

The proposed patch also resolves the issues reported in bug 16 and bug 20.
This exposes a general weakness of this parser: it assumes that the XML passed is well formed to a certain extent and doesn't have thorough checking. strcspn() (and strspn()) are used in many more places throughout the code and exhibit the same problem: if the requested condition cannot be met anywhere in the string, these function stop at the /0 terminating the string. This condition is never

checked and acted upon as this would not allow the deeply nested assignments used throughout the code.

Log in to post a comment.