Double Free in TCPServer #22

Open **Halcy0nic** opened this issue on Jul 14 · 1 comment

Halcy0nic commented on Jul 14

Hi there!

I was running my fuzzer in the background when I discovered a double free in the SimpleNetwork TCPServer.

Impact

Triggering the double free will allow client to crash any SimpleNetwork TCP server remotely. In other situations, double free vulnerabilities can cause undefined behavior and potentially code execution in the right circumstances.

Reproduction

Create a file with a large amount of random characters

Start a TCP server and send the large file to the server a few consecutive times

```
┌──(kali㉿kali)-[~/projects/fuzzing/fuzzotron]
└─$ for i in `seq 1 5`; do echo $i; ./replay -h 127.0.0.1 -p 2020 -P tcp testcases/test1 ; done
```

View the crash and gdb backtrace

```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

socket:  4
enable:  1
close client[ id:0 ip:127.0.0.1 socket:4 ]
exit thread: 139955393259072
accept client[ id:1 ip:127.0.0.1 handle:7 ]
0 ip:I socket:1533492534 send:1 ]
close client[ id:0 ip:I socket:1533492534 ]
close client[ id:0 ip:I socket:1533492534 ]
double free or corruption (out)
```

```
[ Legend: Modified register | Code | Heap | Stack | String ]
──────────────────────────────────────────────────────────────── registers ────
$rax   : 0×0
$rbx   : 0×007ffff722f640  →  0×007ffff722f640  →  [loop detected]
$rcx   : 0×007ffff7bb58a1  →  <raise+321> mov rax, QWORD PTR [rsp+0×108]
$rdx   : 0×0
$rsp   : 0×007ffff722ea20  →  0×0000000000000000
$rbp   : 0×007ffff722ed70  →  0×0000000000000009 ("\t"?)
$rsi   : 0×007ffff722ea20  →  0×0000000000000000
$rdi   : 0×2
$rip   : 0×007ffff7bb58a1  →  <raise+321> mov rax, QWORD PTR [rsp+0×108]
$r8    : 0×0
$r9    : 0×007ffff722ea20  →  0×0000000000000000
$r10   : 0×8
$r11   : 0×246
$r12   : 0×007ffff722ec90  →  0×00555555581690  →  0×00007fff00000006
$r13   : 0×1000
$r14   : 0×10
$r15   : 0×007ffff7fc5000  →  0×6565726600001000
$eflags: [ZERO carry PARITY adjust sign trap INTERRUPT direction overflow resume virtualx86 identification]
$cs: 0×33 $ss: 0×2b $ds: 0×00 $es: 0×00 $fs: 0×00 $gs: 0×00
──────────────────────────────────────────────────────────────────── stack ────
0×007ffff722ea20│+0×0000: 0×0000000000000000    ← $rsp, $rsi, $r9
0×007ffff722ea28│+0×0008: 0×0055555558afd0  →  "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA[...]"
0×007ffff722ea30│+0×0010: 0×0055555556e3a0  →  0×0000000000000001
0×007ffff722ea38│+0×0018: 0×0000000000000000
0×007ffff722ea40│+0×0020: 0×0000000000000000
0×007ffff722ea48│+0×0028: 0×0000000000000000
0×007ffff722ea50│+0×0030: 0×0000000000000000
0×007ffff722ea58│+0×0038: 0×0000ffff00001f80
────────────────────────────────────────────────────────────── code:x86:64 ────
   0×7ffff7bb5895 <raise+309>      mov    edi, 0×2
   0×7ffff7bb589a <raise+314>      mov    eax, 0×e
   0×7ffff7bb589f <raise+319>      syscall
 → 0×7ffff7bb58a1 <raise+321>      mov    rax, QWORD PTR [rsp+0×108]
   0×7ffff7bb58a9 <raise+329>      sub    rax, QWORD PTR fs:0×28
   0×7ffff7bb58b2 <raise+338>      jne    0×7ffff7bb58d4 <__GI_raise+372>
   0×7ffff7bb58b4 <raise+340>      mov    eax, r8d
   0×7ffff7bb58b7 <raise+343>      add    rsp, 0×118
   0×7ffff7bb58be <raise+350>      ret
────────────────────────────────────────────────────────────────── threads ────
[#0] Id 1, Name: "server", stopped 0×7ffff7bfd116 in futex_wait (), reason: SIGABRT
```

Extra Resources

- https://owasp.org/www-community/vulnerabilities/Doubly_freeing_memory
- https://cwe.mitre.org/data/definitions/415.html

Halcy0nic commented on Jul 22 Author

Here is the valgrind output showing the invalid double free:

```
==2704737== Invalid free() / delete / delete[] / realloc()
==2704737==    at 0x484271B: operator delete(void*) (vg_replace_malloc.c:923)
==2704737==    by 0x10CB7B: TCPServer::Task(void*) (in /home/kali/projects/fuzzing/f
==2704737==    by 0x486DD7F: start_thread (pthread_create.c:481)
==2704737==    by 0x4BBF76E: clone (clone.S:95)
==2704737==  Address 0x4df4e80 is 0 bytes inside a block of size 88 free'd
==2704737==    at 0x484271B: operator delete(void*) (vg_replace_malloc.c:923)
==2704737==    by 0x10CB7B: TCPServer::Task(void*) (in /home/kali/projects/fuzzing/f
==2704737==    by 0x486DD7F: start_thread (pthread_create.c:481)
==2704737==    by 0x4BBF76E: clone (clone.S:95)
==2704737==  Block was alloc'd at
==2704737==    at 0x483FF2F: operator new(unsigned long) (vg_replace_malloc.c:422)
==2704737==    by 0x10CE01: TCPServer::accepted() (in /home/kali/projects/fuzzing/fu
==2704737==    by 0x10B048: main (in /home/kali/projects/fuzzing/fuzz_targets/Simple
==2704737==
exit thread: 161396288
```

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

1 participant