

## CVE-2020-24659: read-heap-buffer-overflow found by fuzz

### Description of problem:

I got a heap-buffer-overflow while fuzzing gnuts-master

```
==8==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x602000000000 at pc 0x000000ba4514 bp 0x7ffe4831ba00 sp 0x7f4
READ of size 4 at 0x602000000000 thread T0
SCARINESS: 17 (4-byte-read-heap-buffer-overflow)
#0 0xb04513 in __gmp2_clear /src/gmp/mpz/clear.c:38:7
#1 0x7be127 in wrap_nettle_mpi_release /src/gnutls/lib/nettle/mpi.c:212:2
#2 0x80a21f in _gnutls_mpi_release /src/gnutls/lib/.mpi.h:71:2
#3 0x80dea3 in gnutls_pk_params_release /src/gnutls/lib/pk.c:536:3
#4 0x673445 in deinit_keys /src/gnutls/lib/state.c:380:3
#5 0x672b86 in _gnutls_handshake_internal_state_clear /src/gnutls/lib/state.c:444:2
#6 0x676a57 in gnutls_deinit /src/gnutls/lib/state.c:669:2
#7 0x55475e in LLVMFuzzerTestOneInput /src/gnutls/fuzz/gnutls_psk_client_fuzzer.c:86:2
#8 0x45a1c1 in fuzzer::Fuzzer::ExecuteCallback(unsigned char const*, unsigned long) /src/llvm/projects/compiler-rt/lib/
#9 0x44da4d in fuzzer::RunOneTest(fuzzer::Fuzzer*, char const*, unsigned long) /src/llvm/projects/compiler-rt/lib/fuzz
#10 0x44a9be in fuzzer::FuzzerDriver(int*, char**, int (*)(unsigned char const*, unsigned long)) /src/llvm/projects/c
#11 0x474c12 in main /src/llvm/projects/compiler-rt/lib/fuzzer/FuzzerMain.cpp:19:10
#12 0x7f1470da82f in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x2802f)
#13 0x41e198 in _start (/out/gnutls_psk_client_fuzzer+0x41e198)

0x602000000000 is located 16 bytes to the left of 16-byte region [0x602000000010,0x602000000020)
freed by thread T0 here:
#0 0x52176d in __interceptor_free /src/llvm/projects/compiler-rt/lib/asan/asan_malloc_linux.cpp:123:3
#1 0xb0bde31 in _asn1_delete_list /src/libtasn1/lib/parser_aux.c:590:7
#2 0xb947c8 in asn1_array2tree /src/libtasn1/lib/structure.c:278:5
#3 0x64b073 in _gnutls_global_init /src/gnutls/lib/global.c:293:8
#4 0x64a936 in gnutls_global_init /src/gnutls/lib/global.c:224:9
#5 0x553da4 in init /src/gnutls/fuzz/.fuzzer.h:36:2
#6 0xcdfa1c in __libc_csu_init (/out/gnutls_psk_client_fuzzer+0xcdfa1c)

previously allocated by thread T0 here:
#0 0x5219ed in malloc /src/llvm/projects/compiler-rt/lib/asan/asan_malloc_linux.cpp:145:3
#1 0xb0ba993 in _asn1_add_static_node /src/libtasn1/lib/parser_aux.c:76:7
#2 0xb93d83 in asn1_array2tree /src/libtasn1/lib/structure.c:199:11
#3 0x64b073 in _gnutls_global_init /src/gnutls/lib/global.c:293:8
#4 0x64a936 in gnutls_global_init /src/gnutls/lib/global.c:224:9
#5 0x553da4 in init /src/gnutls/fuzz/.fuzzer.h:36:2
#6 0xcdfa1c in __libc_csu_init (/out/gnutls_psk_client_fuzzer+0xcdfa1c)

SUMMARY: AddressSanitizer: heap-buffer-overflow /src/gmp/mpz/clear.c:38:7 in __gmp2_clear
```

### Version of gnuts used:

master

### Distributor of gnuts (e.g., Ubuntu, Fedora, RHEL)

Ubuntu 16.04

### How reproducible:

run oss-fuzz locally

Steps to Reproduce: use attach file as the corpus to reproduce, like: python infra/helper.py reproduce gnuts gnuts\_psk\_client\_fuzzer  
gnuts\_psk\_client\_fuzzer-heap-buffer-overflow [8\\_gnuts\\_psk\\_client\\_fuzzer-heap-buffer-overflow](#)

### Actual results:


as description, ASAN report a heap-buffer-overflow bug

### Expected results:


no error report

Edited 2 years ago by [Daki User](#)


📎 Drag your designs here or [click to upload](#).


Tasks 

No tasks are currently assigned. Use tasks to break down this issue into smaller parts.


Linked items 

Link issues together to show that they're related or that one is blocking others. [Learn more](#).

Related merge requests 

 handshake: reject no renegotiation alert if handshake is incomplete

11320



When this merge request is accepted, this issue will be closed automatically.

### Activity

 [lutanxiong](#) @ltx2018 · 2 years ago

Author

I found heap-memory address stored in session->key.proto.tls12 was overwriten by read\_server\_hello(handshake.c:line 1947)

```
if (!vers->tls13_sem) {
    gnutls_memset(&session->key.proto.tls13, 0,
                sizeof(session->key.proto.tls13));
    reset_binders(session);
}
```

following is debuginfo:

```
1947     if (!vers->tls13_sem) {
(gdb) p session->key.proto.tls12.dh
$7 = {params = {params = {0x602000004d10, 0x0, 0x602000004cd0, 0x0 <repeats 13 times>}, params_nr = 3, pkflags =
      size = 0}, raw_priv = {data = 0x0, size = 0}, seed_size = 0, seed = '\000' <repeats 255 times>}, palgo = GN
      algo = GNUTLS_PK_DH}, client_Y = 0x602000004cb0}
(gdb) n
1948         gnutls_memset(&session->key.proto.tls13, 0,
(gdb) n
1950         reset_binders(session);
(gdb) n
1956         if (!vers->tls13_sem &&
(gdb) p session->key.proto.tls12.dh
$8 = {params = {params = {0x602000000000, 0x0, 0x602000004cd0, 0x0 <repeats 13 times>}, params_nr = 3, pkflags =
      size = 0}, raw_priv = {data = 0x0, size = 0}, seed_size = 0, seed = '\000' <repeats 255 times>}, palgo = GN
      algo = GNUTLS_PK_DH}, client_Y = 0x602000004cb0}
(gdb)
```

later, while gnutls\_deinit the session: gnutls\_pk\_params\_release(&session->key.proto.tls12.dh.params) trigger the read-heap-buffer-overflow

 [lutanxiong](#) @ltx2018 · 2 years ago


Author

I also get some other ASAN/MSAN error reports while fuzzing gnutls\_psk\_client\_fuzzer, like memleak, use-of-uninitialized-value, same reason as this issue.

session->key.proto is a union, clear session->key.proto.tls13 in read\_server\_hello will lead to overwrite of session->key.proto.tls12

[lutianxiong](#) changed title from **memleak found by fuzz** to **read-heap-buffer-overflow found by fuzz** 2 years ago


[Daiki Ueno](#) made the issue confidential 2 years ago

 [lutianxiong](#) @ltx2018 · 2 years ago

Author


A simple way to fix this is just change key.proto to a 'struct', but that may not be a good idea. [1cx2018/gnutls@b0c0113](#)

[@dueno](#), do you have any ideas?

 [Daiki Ueno](#) @dueno · 2 years ago


Owner

Thank you for the report and analysis. That part in `read_server_hello` is trying to clear the early traffic secret (used for 0-RTT) if TLS 1.3 is not negotiated (i.e. the server doesn't accept early data). Therefore, although I agree that the usage of union is a bit too much micro-optimization, we should nevertheless wipe the secret if it is not used. I suppose we should tighten the check also taking into account of the `HSK_EARLY_DATA_IN_FLIGHT` handshake flag.

 [Daiki Ueno](#) @dueno · 2 years ago


Owner

OK, I think I know this issue (discussed with [@tomato42](#) a while ago). The problem is that the server sends a `no_renegotiation` alert after receiving a Client Hello, right after sending Server Key Exchange (i.e., before finishing the initial handshake). I guess the correct client response would be to close the connection, so the necessary security parameters should be released at this point.

 [Daiki Ueno](#) @dueno · 2 years ago

Owner

Minimal fix: [#0001-handshake-reject-no\\_renegotiation-alert-if-handshake.patch](#)

 [lutianxiong](#) @ltx2018 · 2 years ago

Author

looks good

Please [register](#) or [sign in](#) to reply

[Daiki Ueno](#) added [label](#) 2 years ago

[Daiki Ueno](#) changed title from **read-heap-buffer-overflow found by fuzz** to **CVE-2020-24659: read-heap-buffer-overflow found by fuzz** 2 years ago

[Daiki Ueno](#) mentioned in merge request [1320 \(merged\)](#) 2 years ago

[Daiki Ueno](#) mentioned in commit [1efeb7b7](#) 2 years ago

[Daiki Ueno](#) closed via merge request [1320 \(merged\)](#) 2 years ago

[Daiki Ueno](#) made the issue visible to everyone 2 years ago

[Daiki Ueno](#) changed milestone to [%Release of GnuTLS 3.7.0](#) 2 years ago

Please [register](#) or [sign in](#) to reply