Stack overflow via recursive loop in Gawk

Description

GAWK is a widely used domain-specific language designed for text processing. The "calc_eclosure_iter" function in "support/regcomp.c" may trigger stack overflow via recursive loop. The vulnerability exists in latest stable release (gawk-5.1.1) and latest master branch (12f0f4f27872cd4df6c63977fd0c6ff63d29e424, updated on July 29, 2022).The vulnerable code (located at support/regcomp.c) and the bug's basic introduction are highlighted as follows:

```
// support/regcomp.c
if (dfa->eclosures[edest].nelem == 0)
    {
    // line 1701
    // under this poc, the function has recursive loop which leads to stack
overflow
    err = calc_eclosure_iter (&eclosure_elem, dfa, edest, false);
    if (__glibc_unlikely (err != REG_NOERROR))
        return err;
    }
else
    eclosure_elem = dfa->eclosures[edest];
```

Proof of Concept

Build the gawk (5.1.1 or latest commit 12f0f4f27872cd4df6c63977fd0c6ff63d29e424) and run it using the input POC.

```
# build the gawk with address sanitizer
export CFLAGS=" -fsanitize=address "; export CXXFLAGS=" -fsanitize=address ";
export LDFLAGS=" -fsanitize=address ";
CFGARG=" --enable-shared=no "; configure

AFL_USE_ASAN=1 LD=afl-gcc--disable-shared make

# the INPUT_FILE is not decesive, which can be any valid input.
./gawk -f POC_FILE INPUT_FILE
```

The stack dump is:

```
Disassembly:
Stack Head (1000 entries):
   None                    @ 0x00007ffff7a18a49: in (BL)
   malloc                  @ 0x00007ffff7a1a2e2: in (BL)
   re_node_set_alloc       @ 0x00005555556df1b6: in /src/AWKs/gawk-5.1.1/gawk
   calc_eclosure_iter      @ 0x00005555556e8af7: in /src/AWKs/gawk-5.1.1/gawk
   calc_eclosure_iter      @ 0x00005555556e8da7: in /src/AWKs/gawk-5.1.1/gawk
   calc_eclosure_iter      @ 0x00005555556e8da7: in /src/AWKs/gawk-5.1.1/gawk
```

```
    calc_eclosure_iter        @ 0x00005555556e8da7: in /src/AWKs/gawk-5.1.1/gawk
    calc_eclosure_iter        @ 0x00005555556e8da7: in /src/AWKs/gawk-5.1.1/gawk
    calc_eclosure_iter        @ 0x00005555556e8da7: in /src/AWKs/gawk-5.1.1/gawk
    calc_eclosure_iter        @ 0x00005555556e8da7: in /src/AWKs/gawk-5.1.1/gawk
    calc_eclosure_iter        @ 0x00005555556e8da7: in /src/AWKs/gawk-5.1.1/gawk
    calc_eclosure_iter        @ 0x00005555556e8da7: in /src/AWKs/gawk-5.1.1/gawk
    calc_eclosure_iter        @ 0x00005555556e8da7: in /src/AWKs/gawk-5.1.1/gawk
    calc_eclosure_iter        @ 0x00005555556e8da7: in /src/AWKs/gawk-5.1.1/gawk
    calc_eclosure_iter        @ 0x00005555556e8da7: in /src/AWKs/gawk-5.1.1/gawk
    calc_eclosure_iter        @ 0x00005555556e8da7: in /src/AWKs/gawk-5.1.1/gawk
Registers:
rax=0x000000000000001b rbx=0x00007ffff7b8ec80 rcx=0x0000000000000000
rdx=0x000055555594a010
rsi=0x0000000000000004 rdi=0x00007ffff7b8ec80 rbp=0x0000000000000004
rsp=0x00007fffff7ff000
 r8=0x0000000000000000  r9=0x0000000000199430 r10=0x000000000000006f
r11=0x00007ffff6da5010
r12=0x0000000000000000 r13=0x00007fffff7ff160 r14=0x0000000000019943
r15=0x000055555595e9d0
rip=0x00007ffff7a18a49 efl=0x0000000000010202  cs=0x0000000000000033
 ss=0x000000000000002b
 ds=0x0000000000000000  es=0x0000000000000000  fs=0x0000000000000000
 gs=0x0000000000000000
 k0=0x0000000020820810  k1=0x00000000000000fc  k2=0x00000000dffbf7bf
 k3=0x0000000000000000
 k4=0x0000000000000000  k5=0x0000000000000000  k6=0x0000000000000000
 k7=0x0000000000000000

#0  0x000055555565fc41 in __sanitizer::BufferedStackTrace::UnwindImpl(unsigned
long, unsigned long, void*, bool, unsigned int) ()
#1  0x0000555555655cbf in malloc ()
#2  0x000055555592eaf5 in pma_malloc (size=4) at pma.c:519
#3  0x00005555558ec193 in re_node_set_alloc (set=0x7fffff7ff940,
    size=1) at ./regex_internal.c:939
#4  0x00005555558eaf52 in calc_eclosure_iter (new_set=0x7fffff7ffc80,
    dfa=0x611000000a40, node=20899, root=false) at ./regcomp.c:1661
#5  0x00005555558eba7c in calc_eclosure_iter (new_set=0x7fffff7fffa0,
    dfa=0x611000000a40, node=20900, root=false) at ./regcomp.c:1701
......
#10452 0x00005555558eba7c in calc_eclosure_iter (new_set=0x7fffffff8680,
dfa=0x611000000a40, node=6, root=false) at ./regcomp.c:1701
#10453 0x00005555558eba7c in calc_eclosure_iter (new_set=0x7fffffff89a0,
dfa=0x611000000a40, node=4, root=false) at ./regcomp.c:1701
#10454 0x00005555558eba7c in calc_eclosure_iter (new_set=0x7fffffff8ca0,
dfa=0x611000000a40, node=2, root=true) at ./regcomp.c:1701
#10455 0x00005555558e823c in calc_eclosure (dfa=0x611000000a40) at
./regcomp.c:1639
#10456 0x00005555558c1257 in analyze (preg=0x60b0000007d0) at ./regcomp.c:1166
#10457 0x00005555558b18e6 in re_compile_internal (preg=0x60b0000007d0,
pattern=0x602000002810 "5#?{28}{2288}\220{", length=15, syntax=2339405) at
./regcomp.c:769
#10458 0x00005555558b0a7e in re_compile_pattern (pattern=0x602000002810 "5#?{28}
{2288}\220{", length=15, bufp=0x60b0000007d0) at ./regcomp.c:217
#10459 0x000055555585fc4d in make_regexp (s=0x6020000027f0 "5#?{28}{2288}\220{",
len=15, ignorecase=false, dfa=true, canfatal=false) at re.c:257
```

```
#10460 0x00005555556d2159 in make_regnode (type=Node_regex, exp=0x6260000062c0)
at /src/Source/gawk_newest/gawk/awkgram.y:5297
#10461 0x00005555556a4257 in yyparse () at
/src/Source/gawk_newest/gawk/awkgram.y:572
#10462 0x00005555556e1ea1 in parse_program (pcode=0x555556351c60 <code_block>,
from_eval=false) at /src/Source/gawk_newest/gawk/awkgram.y:2803
#10463 0x000055555581c88e in main (argc=4, argv=0x7fffffffe3b8) at main.c:504
```

Impact

This vulnerability can be used for causing the crash or long-term loop of the software which would leads to denial of service(DoS). Besides, the attacker can exploit weak points to launch remote code execution.

Reference

POC_FILE