QEMU: Null Pointer Failure in fdctrl_read() in hw/block/fdc.c

Bug #1912780 reported by P J P on 2021-01-22

This bug affects 1 person 258

Affects	Status	Importance	Assigned to	Milestone
QEMU	Expired	High	Unassigned	

Bug Description

```
[via qemu-security list]

This is Gaoning Pan from Zhejiang University & Ant Security Light-Year
Lab.
I found a Null Pointer issue locates in fdctrl_read() in hw/block/fdc.c.
This flaw allows a malicious guest user or process in a denial of service
condition.

This issus was discovered in the latest Qemu-5.2.0. When using floppy
device, there are several
choices to get specific drive in get_drv(), depending on fdctrl->cur_drv.
But not all drives are
initialized properly, leaving fdctrl->drives[0]->blk as NULL. So when the
drive was used in
blk_pread(cur_drv->blk, fd_offset(cur_drv), fdctrl->fifo,
BDRV_SECTOR_SIZE) at line 1918,
null pointer access triggers, thus denial of service.My reproduced
environment is as follows:

    Host: ubuntu 18.04
    Guest: ubuntu 18.04

My boot command is as follows:

  qemu-system-x86_64 -enable-kvm -boot c -m 2G -drive format=qcow2,file=
./ubuntu.img \
   -nic user,hostfwd=tcp:0.0.0.0:5555-:22 -device floppy,unit=1,
drive=mydrive \
   -drive id=mydrive,file=null-co://,size=2M,format=raw,if=none -display
none

ASAN output is as follows:
=================================================================
==14688==ERROR: AddressSanitizer: SEGV on unknown address 0x00000000034c
(pc 0x5636eee9bbaf bp 0x7ff2a53fdea0 sp 0x7ff2a53fde90 T3)
==14688==The signal is caused by a WRITE memory access.
==14688==Hint: address points to the zero page.
    #0 0x5636eee9bbae in blk_inc_in_flight ../block/block-backend.c:1356
    #1 0x5636eee9b766 in blk_prw ../block/block-backend.c:1328
    #2 0x5636eee9cd76 in blk_pread ../block/block-backend.c:1491
    #3 0x5636ee1adf24 in fdctrl_read_data ../hw/block/fdc.c:1918
    #4 0x5636ee1a6654 in fdctrl_read ../hw/block/fdc.c:935
    #5 0x5636eebb84c8 in portio_read ../softmmu/ioport.c:179
    #6 0x5636ee9848c5 in memory_region_read_accessor ../softmmu/memory.
c:442
    #7 0x5636ee9855c2 in access_with_adjusted_size ../softmmu/memory.c:552
    #8 0x5636ee98f0b7 in memory_region_dispatch_read1 ../softmmu/memory.
c:1420
    #9 0x5636ee98f311 in memory_region_dispatch_read ../softmmu/memory.
c:1449
    #10 0x5636ee8ff64a in flatview_read_continue ../softmmu/physmem.c:2822
    #11 0x5636ee8ff9e5 in flatview_read ../softmmu/physmem.c:2862
    #12 0x5636ee8ffb83 in address_space_read_full ../softmmu/physmem.
c:2875
    #13 0x5636ee8ffdeb in address_space_rw ../softmmu/physmem.c:2903
    #14 0x5636eea6a924 in kvm_handle_io ../accel/kvm/kvm-all.c:2285
    #15 0x5636eea6c5e3 in kvm_cpu_exec ../accel/kvm/kvm-all.c:2531
    #16 0x5636eeca492b in kvm_vcpu_thread_fn ../accel/kvm/kvm-cpus.c:49
    #17 0x5636ef1bc296 in qemu_thread_start ../util/qemu-thread-posix.c:
521
    #18 0x7ff337c736da in start_thread (/lib/x86_64-linux-gnu/libpthread.
so.0+0x76da)
    #19 0x7ff33799ca3e in __clone (/lib/x86_64-linux-gnu/libc.
so.6+0x121a3e)

AddressSanitizer can not provide additional info.
SUMMARY: AddressSanitizer: SEGV ../block/block-backend.c:1356 in
blk_inc_in_flight
Thread T3 created by T0 here:
    #0 0x7ff33c580d2f in __interceptor_pthread_create (/usr/lib/x86_64-
linux-gnu/libasan.so.4+0x37d2f)
    #1 0x5636ef1bc673 in qemu_thread_create ../util/qemu-thread-posix.c:
558
    #2 0x5636eeca4ce7 in kvm_start_vcpu_thread ../accel/kvm/kvm-cpus.c:73
    #3 0x5636ee9aa965 in qemu_init_vcpu ../softmmu/cpus.c:622
    #4 0x5636ee82a9b4 in x86_cpu_realizefn ../target/i386/cpu.c:6731
    #5 0x5636eed002f4 in device_set_realized ../hw/core/qdev.c:886
    #6 0x5636eecc59bc in property_set_bool ../qom/object.c:2251
    #7 0x5636eecc0c28 in object_property_set ../qom/object.c:1398
    #8 0x5636eecb6fb9 in object_property_set_qobject ../qom/qom-qobject.
c:28
    #9 0x5636eecc1175 in object_property_set_bool ../qom/object.c:1465
    #10 0x5636eecfc286 in qdev_realize ../hw/core/qdev.c:399
    #11 0x5636ee739b34 in x86_cpu_new ../hw/i386/x86.c:111
    #12 0x5636ee739d6d in x86_cpus_init ../hw/i386/x86.c:138
    #13 0x5636ee6f843e in pc_init1 ../hw/i386/pc_piix.c:159
    #14 0x5636ee6fab1e in pc_init_v5_2 ../hw/i386/pc_piix.c:438
    #15 0x5636ee1cb4a7 in machine_run_board_init ../hw/core/machine.c:1134
    #16 0x5636ee9c323d in qemu_init ../softmmu/vl.c:4369
    #17 0x5636edd92c71 in main ../softmmu/main.c:49
    #18 0x7ff33789cb96 in __libc_start_main (/lib/x86_64-linux-gnu/libc.
so.6+0x21b96)
```

```
==14688==ABORTING

Reproducer is attached.

Best regards.
Gaoning Pan of Zhejiang University & Ant Security Light-Year Lab
```

Tags: cve security

CVE References

2021-20196

P J P (pjps) wrote on 2021-01-22: #1

Reproducer (523 bytes, text/x-csrc)

P J P (pjps) wrote on 2021-01-22: #2

```
The given reproducer does not seem to work as expected to trigger this
issue.
IIUC, issue occurs because a privileged guest user may change the selected
floppy drive via FD_REG_DOR:fdctrl_write_dor() ioport write command

  static void fdctrl_write_dor(FDCtrl *fdctrl, uint32_t value)
  {
      ...
      /* Selected drive */
      fdctrl->cur_drv = value & FD_DOR_SELMASK; <= selected drive changes
based on 'value'
      ...
  }

Little tweaking of parameters under gdb reproduces the crash

$ gdb --args ./bin/qemu-system-x86_64 -runas test -nographic -enable-kvm -
m 2048
   -drive file=fdc.img,format=qcow2,if=floppy,id=myfdc /var/lib/libvirt/
images/f27vm.qcow2
...
==541702==ERROR: AddressSanitizer: SEGV on unknown address 0x00000000034c
(pc 0x555559938377f bp 0x7fff6f3fdeb0 sp 0x7fff6f3fdea0 T3)
==541702==The signal is caused by a WRITE memory access.
==541702==Hint: address points to the zero page.
    #0 0x55559938377f in blk_inc_in_flight ../block/block-backend.c:1356
    #1 0x55559938325b in blk_prw ../block/block-backend.c:1328
    #2 0x555559384ec5 in blk_pread ../block/block-backend.c:1491
    #3 0x555557d7c798 in fdctrl_read_data ../hw/block/fdc.c:1919
    #4 0x555557d7207c in fdctrl_read ../hw/block/fdc.c:936
    #5 0x555558ee7c40 in portio_read ../softmmu/ioport.c:179
    #6 0x555558c9a0c1 in memory_region_read_accessor ../softmmu/memory.
c:442
    #7 0x555558c9af04 in access_with_adjusted_size ../softmmu/memory.c:552
    #8 0x555558ca7159 in memory_region_dispatch_read1 ../softmmu/memory.
c:1420
    #9 0x555558ca7433 in memory_region_dispatch_read ../softmmu/memory.
c:1449
    #10 0x555558f6214e in flatview_read_continue ../softmmu/physmem.c:2822
    #11 0x555558f62560 in flatview_read ../softmmu/physmem.c:2862
    #12 0x555558f62700 in address_space_read_full ../softmmu/physmem.
c:2875
    #13 0x555558f62977 in address_space_rw ../softmmu/physmem.c:2903
    #14 0x555558d037b9 in kvm_handle_io ../accel/kvm/kvm-all.c:2285
    #15 0x555558d05a4b in kvm_cpu_exec ../accel/kvm/kvm-all.c:2531
    #16 0x555558ee0efa in kvm_vcpu_thread_fn ../accel/kvm/kvm-cpus.c:49
    #17 0x55555977ec18 in qemu_thread_start ../util/qemu-thread-posix.c:
521
    #18 0x7ffff63323f8 in start_thread (/lib64/libpthread.so.0+0x93f8)
    #19 0x7ffff625f902 in __GI___clone (/lib64/libc.so.6+0x101902)
```

P J P (pjps) wrote on 2021-01-22: #3

```
Proposed patch:

$ git diff hw/block/
diff --git a/hw/block/fdc.c b/hw/block/fdc.c
index 3636874432..13a9470d19 100644
--- a/hw/block/fdc.c
+++ b/hw/block/fdc.c
@@ -1429,7 +1429,9 @@ static void fdctrl_write_dor(FDCtrl *fdctrl,
uint32_t value)
          }
      }
      /* Selected drive */
- fdctrl->cur_drv = value & FD_DOR_SELMASK;
+ if (fdctrl->drives[value & FD_DOR_SELMASK].blk) {
+ fdctrl->cur_drv = value & FD_DOR_SELMASK;
+ }

      fdctrl->dor = value;
 }
@@ -1894,6 +1896,10 @@ static uint32_t fdctrl_read_data(FDCtrl *fdctrl)
      uint32_t pos;

      cur_drv = get_cur_drv(fdctrl);
+ if (!cur_drv->blk) {
+ FLOPPY_DPRINTF("No drive connected\n");
+ return 0;
+ }
      fdctrl->dsr &= ~FD_DSR_PWRDOWN;
      if (!(fdctrl->msr & FD_MSR_RQM) || !(fdctrl->msr & FD_MSR_DIO)) {
          FLOPPY_DPRINTF("error: controller not ready for reading\n");
@@ -2420,7 +2426,8 @@ static void fdctrl_write_data(FDCtrl *fdctrl,
uint32_t value)
          if (pos == FD_SECTOR_LEN - 1 ||
              fdctrl->data_pos == fdctrl->data_len) {
              cur_drv = get_cur_drv(fdctrl);
- if (blk_pwrite(cur_drv->blk, fd_offset(cur_drv), fdctrl->fifo,
```

```
+ if (cur_drv->blk == NULL
+ || blk_pwrite(cur_drv->blk, fd_offset(cur_drv), fdctrl->fifo,
                            BDRV_SECTOR_SIZE, 0) < 0) {
                FLOPPY_DPRINTF("error writing sector %d\n",
                               fd_sector(cur_drv));
```

P J P (pjps) wrote on 2021-01-23: #4

On Friday, 22 January, 2021, 05:42:55 pm IST, 潘高宁 <email address hidden>
wrote:
> This patch seems to work now. I've re-compiled and tested the QEMU,
which showed the functional operation was working well.

CVE-2021-20196 assigned by Red Hat Inc.

P J P (pjps) wrote on 2021-01-23: #5

Upstream patch:
  -> https://lists.nongnu.org/archive/html/qemu-devel/2021-01/msg05986.
html

information type:Private Security → Public Security

Thomas Huth (th-huth) on 2021-05-14

Changed in qemu:
    status:New → In Progress
importance:Undecided → High

John Snow (jnsnow) wrote on 2021-05-17: #6

Took a look at the patch today, I think it might need a change or two but
it should be quick to do. I've asked Thomas to move this issue to gitlab
so I can keep a closer eye on it.

--js

Thomas Huth (th-huth) wrote on 2021-05-18: **Moved bug report** #7

This is an automated cleanup. This bug report has been moved to QEMU's
new bug tracker on gitlab.com and thus gets marked as 'expired' now.
Please continue with the discussion here:

 https://gitlab.com/qemu-project/qemu/-/issues/338

Changed in qemu:
status:In Progress → Expired

See full activity log

To post a comment you must log in.