Bug 206361 - Linux Kernel 5.4.7 - n_tty_receive_buf_common use-after-free

Status: NEW

Alias: None

Product: Drivers
Component: Console/Framebuffers (show other bugs)
Hardware: All Linux

Importance: P1 normal
Assignee: James Simmons

URL:
Keywords:

Depends on:
Blocks:

Reported: 2020-01-30 16:58 UTC by Tristan Madani
Modified: 2020-03-16 18:58 UTC (History)
CC List: 3 users (show)

See Also:
Kernel Version: 5.4.7
Tree: Mainline
Regression: No

Attachments

Add an attachment (proposed patch, testcase, etc.)

Tristan Madani 2020-01-30 16:58:10 UTC **Description**

```
Linux Kernel 5.4.7 - n_tty_receive_buf_common use-after-free

0x01 - Introduction
===

# Product: Linux Kernel
# Version: 5.4.7 (stable) and probably other versions
# Bug: UAF (Read)
# Tested on: GNU/Linux Debian 9 x86_64


0x02 - Details
===

There is a UAF read in "n_tty_receive_buf_common" function from the Linux tty
driver.

Code analysis (drivers/tty/n_tty.c):

static int
n_tty_receive_buf_common(struct tty_struct *tty, const unsigned char *cp,
                         char *fp, int count, int flow)
{
        struct n_tty_data *ldata = tty->disc_data;
        int room, n, rcvd = 0, overflow;

        down_read(&tty->termios_rwsem);

        do {

                size_t tail = smp_load_acquire(&ldata->read_tail);

                room = N_TTY_BUF_SIZE - (ldata->read_head - tail);
                if (I_PARMRK(tty))
                        room = (room + 2) / 3;
                room--;
                if (room <= 0) {
                        overflow = ldata->icanon && ldata->canon_head == tail;
                        if (overflow && room < 0)
                                ldata->read_head--;
                        room = overflow;
                        ldata->no_room = flow && !room;
                } else
                        overflow = 0;

                n = min(count, room);
                if (!n)
                        break;

                /* ignore parity errors if handling overflow */
                if (!overflow || !fp || *fp != TTY_PARITY)                      //
<-- UAF occurs here
                        __receive_buf(tty, cp, fp, n);

                cp += n;
                if (fp)
                        fp += n;
                count -= n;
                rcvd += n;
        } while (!test_bit(TTY_LDISC_CHANGING, &tty->flags));

        tty->receive_room = room;


0x03 - Crash report
===

BUG: KASAN: use-after-free in n_tty_receive_buf_common+0x2481/0x2940
drivers/tty/n_tty.c:1741
Read of size 1 at addr ffff8880089e40e9 by task syz-executor.1/13184

CPU: 0 PID: 13184 Comm: syz-executor.1 Not tainted 5.4.7 #1
Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.12.0-1 04/01/2014
Call Trace:
 __dump_stack lib/dump_stack.c:77 [inline]
 dump_stack+0xee/0x16e lib/dump_stack.c:118
 print_address_description.constprop.8+0x36/0x50 mm/kasan/report.c:374
 __kasan_report.cold.11+0x1a/0x3a mm/kasan/report.c:506
 kasan_report+0xe/0x20 mm/kasan/common.c:634
 n_tty_receive_buf_common+0x2481/0x2940 drivers/tty/n_tty.c:1741
 tty_ldisc_receive_buf+0xac/0x190 drivers/tty/tty_buffer.c:461
 paste_selection+0x297/0x400 drivers/tty/vt/selection.c:372
 tioclinux+0x20d/0x4e0 drivers/tty/vt/vt.c:3044
 vt_ioctl+0x1bcf/0x28d0 drivers/tty/vt/vt_ioctl.c:364
 tty_ioctl+0x525/0x15a0 drivers/tty/tty_io.c:2657
 vfs_ioctl fs/ioctl.c:47 [inline]
 file_ioctl fs/ioctl.c:510 [inline]
 do_vfs_ioctl+0x1c5/0x1310 fs/ioctl.c:697
 ksys_ioctl+0x9b/0xc0 fs/ioctl.c:714
 __do_sys_ioctl fs/ioctl.c:721 [inline]
 __se_sys_ioctl fs/ioctl.c:719 [inline]
 __x64_sys_ioctl+0x6f/0xb0 fs/ioctl.c:719
 do_syscall_64+0xbc/0x560 arch/x86/entry/common.c:290
 entry_SYSCALL_64_after_hwframe+0x49/0xbe
RIP: 0033:0x4662e9
Code: ff ff c3 66 2e 0f 1f 84 00 00 00 00 00 0f 1f 40 00 48 89 f8 48 89 f7 48 89 d6
48 89 ca 4d 89 c2 4d 89 c8 4c 8b 4c 24 08 0f 05 <48> 3d 01 f0 ff ff 73 01 c3 48 c7
c1 bc ff ff ff f7 d8 64 89 01 48
RSP: 002b:00007fb274648c68 EFLAGS: 00000246 ORIG_RAX: 0000000000000010
RAX: ffffffffffffffda RBX: 000000000052bf00 RCX: 00000000004662e9
RDX: 0000000020000000 RSI: 000000000000541c RDI: 0000000000000004
RBP: 00000000ffffffff R08: 0000000000000000 R09: 0000000000000000
R10: 0000000000000000 R11: 0000000000000246 R12: 00000000004a744b
```

```
R13: 00000000004edd18 R14: 00000000004ac84e R15: 00007fb2746496bc

Allocated by task 10830:
 save_stack+0x19/0x80 mm/kasan/common.c:69
 set_track mm/kasan/common.c:77 [inline]
 __kasan_kmalloc mm/kasan/common.c:510 [inline]
 __kasan_kmalloc.constprop.7+0xc1/0xd0 mm/kasan/common.c:483
 kmalloc_array include/linux/slab.h:618 [inline]
 set_selection_kernel+0x877/0x13b0 drivers/tty/vt/selection.c:305
 set_selection_user+0x94/0xe0 drivers/tty/vt/selection.c:177
 tioclinux+0x331/0x4e0 drivers/tty/vt/vt.c:3039
 vt_ioctl+0x1bcf/0x28d0 drivers/tty/vt/vt_ioctl.c:364
 tty_ioctl+0x525/0x15a0 drivers/tty/tty_io.c:2657
 vfs_ioctl fs/ioctl.c:47 [inline]
 file_ioctl fs/ioctl.c:510 [inline]
 do_vfs_ioctl+0x1c5/0x1310 fs/ioctl.c:697
 ksys_ioctl+0x9b/0xc0 fs/ioctl.c:714
 __do_sys_ioctl fs/ioctl.c:721 [inline]
 __se_sys_ioctl fs/ioctl.c:719 [inline]
 __x64_sys_ioctl+0x6f/0xb0 fs/ioctl.c:719
 do_syscall_64+0xbc/0x560 arch/x86/entry/common.c:290
 entry_SYSCALL_64_after_hwframe+0x49/0xbe

Freed by task 13213:
 save_stack+0x19/0x80 mm/kasan/common.c:69
 set_track mm/kasan/common.c:77 [inline]
 kasan_set_free_info mm/kasan/common.c:332 [inline]
 __kasan_slab_free+0x12e/0x180 mm/kasan/common.c:471
 slab_free_hook mm/slub.c:1424 [inline]
 slab_free_freelist_hook mm/slub.c:1457 [inline]
 slab_free mm/slub.c:3004 [inline]
 kfree+0x101/0x3b0 mm/slub.c:3956
 set_selection_kernel+0x8a1/0x13b0 drivers/tty/vt/selection.c:312
 set_selection_user+0x94/0xe0 drivers/tty/vt/selection.c:177
 tioclinux+0x331/0x4e0 drivers/tty/vt/vt.c:3039
 vt_ioctl+0x1bcf/0x28d0 drivers/tty/vt/vt_ioctl.c:364
 tty_ioctl+0x525/0x15a0 drivers/tty/tty_io.c:2657
 vfs_ioctl fs/ioctl.c:47 [inline]
 file_ioctl fs/ioctl.c:510 [inline]
 do_vfs_ioctl+0x1c5/0x1310 fs/ioctl.c:697
 ksys_ioctl+0x9b/0xc0 fs/ioctl.c:714
 __do_sys_ioctl fs/ioctl.c:721 [inline]
 __se_sys_ioctl fs/ioctl.c:719 [inline]
 __x64_sys_ioctl+0x6f/0xb0 fs/ioctl.c:719
 do_syscall_64+0xbc/0x560 arch/x86/entry/common.c:290
 entry_SYSCALL_64_after_hwframe+0x49/0xbe

The buggy address belongs to the object at ffff8880089e4000
 which belongs to the cache kmalloc-4k of size 4096
The buggy address is located 233 bytes inside of
 4096-byte region [ffff8880089e4000, ffff8880089e5000)
The buggy address belongs to the page:
page:ffffea0000227800 refcount:1 mapcount:0 mapping:ffff88803580de00 index:0x0
compound_mapcount: 0
raw: 0100000000010200 0000000000000000 0000000100000001 ffff88803580de00
raw: 0000000000000000 0000000000040004 00000001ffffffff 0000000000000000
page dumped because: kasan: bad access detected

Memory state around the buggy address:
 ffff8880089e3f80: fc fc fc fc fc fc fc fc fc fc fc fc fc fc fc fc
 ffff8880089e4000: fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb
>ffff8880089e4080: fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb
                                                 ^
 ffff8880089e4100: fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb
 ffff8880089e4180: fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb fb
==================================================================
```

Thadeu Lima de Souza Cascardo **2020-02-10 14:13:04 UTC** **Comment 1**

This could be caused by a race around set_selection_kernel and paste_selection.

Though the call to set_selection_kernel is protected with console_lock,
paste_selection isn't. So, the use of sel_buffer in paste_selection is unprotected,
and it could be freed with a parallel call to set_selection.

Cascardo.

Jiri Slaby **2020-02-17 07:40:59 UTC** **Comment 2**

Likely fixed by:
commit 07e6124a1a46b4b5a9b3cacc0c306b50da87abf5
Author: Jiri Slaby <jslaby@suse.cz>
Date: Mon Feb 10 09:11:31 2020 +0100

    vt: selection, close sel_buffer race

Note

You need to log in before you can comment on or make changes to this bug.