



author Zhang Qilong <zhangqilong3@huawei.com> 2022-10-29 00:10:49 +0800
committer Greg Kroah-Hartman <gregkh@linuxfoundation.org> 2022-11-10 17:57:51 +0100
commit [b13be5e852b03f376058027e462fad4230240891](#) (patch)
tree [3d61d933fd647fbc0fa9ee477481e350d526aa4b](#)
parent [8457a00c981fe1a799ce34123908856b0f5973b8](#) (diff)
download [linux-b13be5e852b03f376058027e462fad4230240891.tar.gz](#)

diff options

context:
space:
mode:

rose: Fix NULL pointer dereference in rose_send_frame()

[Upstream commit [e97c089d7a49f67027395ddf70bf327eeac2611e](#)]

The syzkaller reported an issue:

```
KASAN: null-ptr-deref in range [0x0000000000000380-0x0000000000000387]
CPU: 0 PID: 4069 Comm: kworker/0:15 Not tainted 6.0.0-syzkaller-02734-g0326074ff465 #0
Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS Google 09/22/2022
Workqueue: rcu_gp srcu_invoke_callbacks
RIP: 0010:rose_send_frame+0x1dd/0x2f0 net/rose/rose_link.c:101
Call Trace:
<IRQ>
  rose_transmit_clear_request+0x1d5/0x290 net/rose/rose_link.c:255
  rose_rx_call_request+0x4c0/0x1bc0 net/rose/af_rose.c:1009
  rose_loopback_timer+0x19e/0x590 net/rose/rose_loopback.c:111
  call_timer_fn+0x1a0/0x6b0 kernel/time/timer.c:1474
  expire_timers kernel/time/timer.c:1519 [inline]
  __run_timers.part.0+0x674/0xa80 kernel/time/timer.c:1790
  __run_timers kernel/time/timer.c:1768 [inline]
  run_timer_softirq+0xb3/0x1d0 kernel/time/timer.c:1803
  __do_softirq+0x1d0/0x9c8 kernel/softirq.c:571
  [...]
</IRQ>
```

It triggers NULL pointer dereference when 'neigh->dev->dev_addr' is called in the rose_send_frame(). It's the first occurrence of the 'neigh' is in rose_loopback_timer() as 'rose_loopback_neigh', and the 'dev' in 'rose_loopback_neigh' is initialized sa nullptr.

It had been fixed by commit [3b3fd068c56e3fbea30090859216a368398e39bf](#) ("rose: Fix Null pointer dereference in rose_send_frame()") ever. But it's introduced by commit [3c53cd65dece47dd1f9d3a809f32e59d1d87b2b8](#) ("rose: check NULL rose_loopback_neigh->loopback") again.

We fix it by add NULL check in rose_transmit_clear_request(). When the 'dev' in 'neigh' is NULL, we don't reply the request and just clear it.

syzkaller don't provide repro, and I provide a syz repro like:

```
r0 = syz_init_net_socket$bt_sco(0x1f, 0x5, 0x2)
ioctl$sock_inet_SIOCSIFFLAGS(r0, 0x8914, &(0x7f00000000180)={'rose0\x00', 0x201})
r1 = syz_init_net_socket$rose(0xb, 0x5, 0x0)
bind$rose(r1, &(0x7f000000000c0)=@full={0xb, @dev, @null, 0x0, [@null, @null, @netrom, @netrom, @default, @null]}, 0x40)
connect$rose(r1, &(0x7f00000000240)=@short={0xb, @dev={0xbb, 0xbb, 0xbb, 0x1, 0x0}, @remote={0xcc, 0xcc, 0xcc, 0xcc, 0xcc, 0xcc, 0x1}, 0x1, @netrom=
```

Fixes: [3c53cd65dece](#) ("rose: check NULL rose_loopback_neigh->loopback")

Signed-off-by: Zhang Qilong <zhangqilong3@huawei.com>

Signed-off-by: David S. Miller <davem@davemloft.net>

Signed-off-by: Sasha Levin <sashal@kernel.org>

Diffstat

```
-rw-r--r-- net/rose/rose_link.c 3
```

1 files changed, 3 insertions, 0 deletions

diff --git a/net/rose/rose_link.c b/net/rose/rose_link.c

index [f6102e6f51617f..730d2205f1976a](#) 100644

--- a/net/rose/rose_link.c

+++ b/net/rose/rose_link.c

```
@@ -236,6 +236,9 @@ void rose_transmit_clear_request(struct rose_neigh *neigh, unsigned int lci, uns
    unsigned char *dptr;
    int len;

+    if (!neigh->dev)
+        return;
+
    len = AX25_BPQ_HEADER_LEN + AX25_MAX_HEADER_LEN + ROSE_MIN_LEN + 3;

    if ((skb = alloc_skb(len, GFP_ATOMIC)) == NULL)
```