author	Stanislav Fomichev <sdf@fomichev.me>	2025-03-13 03:06:57 -0700
committer	Greg Kroah-Hartman <gregkh@linuxfoundation.org>	2025-04-25 10:43:31 +0200
commit	53fb25e90c0a503a17c639341ba5e755cb2feb5c (patch)	
tree	f171c3fdb459f3d1a5affd1df2b28dffe3cf60cf	
parent	31d43652232118a754cb023e2e656ad00ace32fb (diff)	
download	linux-53fb25e90c0a503a17c639341ba5e755cb2feb5c.tar.gz	

diff options

context: 3

space: include

mode: unified

net: vlan: don't propagate flags on open

```
[ Upstream commit 27b918007d96402aba10ed52a6af8015230f1793 ]

With the device instance lock, there is now a possibility of a deadlock:

[    1.211455] ============================================
[    1.211571] WARNING: possible recursive locking detected
[    1.211687] 6.14.0-rc5-01215-g032756b4ca7a-dirty #5 Not tainted
[    1.211823] --------------------------------------------
[    1.211936] ip/184 is trying to acquire lock:
[    1.212032] ffff8881024a4c30 (&dev->lock){+.+.}-{4:4}, at: dev_set_allmulti+0x4e/0xb0
[    1.212207]
[    1.212207] but task is already holding lock:
[    1.212332] ffff8881024a4c30 (&dev->lock){+.+.}-{4:4}, at: dev_open+0x50/0xb0
[    1.212487]
[    1.212487] other info that might help us debug this:
[    1.212626]  Possible unsafe locking scenario:
[    1.212626]
[    1.212751]        CPU0
[    1.212815]        ----
[    1.212871]   lock(&dev->lock);
[    1.212944]   lock(&dev->lock);
[    1.213016]
[    1.213016]  *** DEADLOCK ***
[    1.213016]
[    1.213143]  May be due to missing lock nesting notation
[    1.213143]
[    1.213294] 3 locks held by ip/184:
[    1.213371]  #0: ffffffff838b53e0 (rtnl_mutex){+.+.}-{4:4}, at: rtnl_nets_lock+0x1b/0xa0
[    1.213543]  #1: ffffffff84e5fc70 (&net->rtnl_mutex){+.+.}-{4:4}, at: rtnl_nets_lock+0x37/0xa0
[    1.213727]  #2: ffff8881024a4c30 (&dev->lock){+.+.}-{4:4}, at: dev_open+0x50/0xb0
[    1.213895]
[    1.213895] stack backtrace:
[    1.213991] CPU: 0 UID: 0 PID: 184 Comm: ip Not tainted 6.14.0-rc5-01215-g032756b4ca7a-dirty #5
[    1.213993] Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS Arch Linux 1.16.3-1-1 04/01/2014
[    1.213994] Call Trace:
[    1.213995]  <TASK>
[    1.213996]  dump_stack_lvl+0x8e/0xd0
[    1.214000]  print_deadlock_bug+0x28b/0x2a0
[    1.214020]  lock_acquire+0xea/0x2a0
[    1.214027]  __mutex_lock+0xbf/0xd40
[    1.214038]  dev_set_allmulti+0x4e/0xb0 # real_dev->flags & IFF_ALLMULTI
[    1.214040]  vlan_dev_open+0xa5/0x170 # ndo_open on vlandev
[    1.214042]  __dev_open+0x145/0x270
[    1.214046]  __dev_change_flags+0xb0/0x1e0
[    1.214051]  netif_change_flags+0x22/0x60 # IFF_UP vlandev
[    1.214053]  dev_change_flags+0x61/0xb0 # for each device in group from dev->vlan_info
[    1.214055]  vlan_device_event+0x766/0x7c0 # on netdevsim0
[    1.214058]  notifier_call_chain+0x78/0x120
[    1.214062]  netif_open+0x6d/0x90
```

```
[    1.214064]  dev_open+0x5b/0xb0 # locks netdevsim0
[    1.214066]  bond_enslave+0x64c/0x1230
[    1.214075]  do_set_master+0x175/0x1e0 # on netdevsim0
[    1.214077]  do_setlink+0x516/0x13b0
[    1.214094]  rtnl_newlink+0xaba/0xb80
[    1.214132]  rtnetlink_rcv_msg+0x440/0x490
[    1.214144]  netlink_rcv_skb+0xeb/0x120
[    1.214150]  netlink_unicast+0x1f9/0x320
[    1.214153]  netlink_sendmsg+0x346/0x3f0
[    1.214157]  __sock_sendmsg+0x86/0xb0
[    1.214160]  ____sys_sendmsg+0x1c8/0x220
[    1.214164]  ___sys_sendmsg+0x28f/0x2d0
[    1.214179]  __x64_sys_sendmsg+0xef/0x140
[    1.214184]  do_syscall_64+0xec/0x1d0
[    1.214190]  entry_SYSCALL_64_after_hwframe+0x77/0x7f
[    1.214191] RIP: 0033:0x7f2d1b4a7e56

Device setup:

     netdevsim0 (down)
     ^         ^
  bond         netdevsim1.100@netdevsim1 allmulticast=on (down)

When we enslave the lower device (netdevsim0) which has a vlan, we
propagate vlan's allmuti/promisc flags during ndo_open. This causes
(re)locking on of the real_dev.

Propagate allmulti/promisc on flags change, not on the open. There
is a slight semantics change that vlans that are down now propagate
the flags, but this seems unlikely to result in the real issues.

Reproducer:

  echo 0 1 > /sys/bus/netdevsim/new_device

  dev_path=$(ls -d /sys/bus/netdevsim/devices/netdevsim0/net/*)
  dev=$(echo $dev_path | rev | cut -d/ -f1 | rev)

  ip link set dev $dev name netdevsim0
  ip link set dev netdevsim0 up

  ip link add link netdevsim0 name netdevsim0.100 type vlan id 100
  ip link set dev netdevsim0.100 allmulticast on down
  ip link add name bond1 type bond mode 802.3ad
  ip link set dev netdevsim0 down
  ip link set dev netdevsim0 master bond1
  ip link set dev bond1 up
  ip link show

Reported-by: syzbot+b0c03d76056ef6cd12a6@syzkaller.appspotmail.com
Closes: https://lore.kernel.org/netdev/Z9CfXjLMKn6VLG5d@mini-arch/T/#m15ba130f53227c883e79fb969687d69d670337a0
Signed-off-by: Stanislav Fomichev <sdf@fomichev.me>
Reviewed-by: Simon Horman <horms@kernel.org>
Link: https://patch.msgid.link/20250313100657.2287455-1-sdf@fomichev.me
Signed-off-by: Paolo Abeni <pabeni@redhat.com>
Signed-off-by: Sasha Levin <sashal@kernel.org>
```

Diffstat

-rw-r--r-- net/8021q/vlan_dev.c 31

1 files changed, 4 insertions, 27 deletions

```
diff --git a/net/8021q/vlan_dev.c b/net/8021q/vlan_dev.c
index d3e511e1eba8a3..c08228d4883460 100644
--- a/net/8021q/vlan_dev.c
+++ b/net/8021q/vlan_dev.c
@@ -272,17 +272,6 @@ static int vlan_dev_open(struct net_device *dev)
                        goto out;
        }
```

```
-       if (dev->flags & IFF_ALLMULTI) {
-               err = dev_set_allmulti(real_dev, 1);
-               if (err < 0)
-                       goto del_unicast;
-       }
-       if (dev->flags & IFF_PROMISC) {
-               err = dev_set_promiscuity(real_dev, 1);
-               if (err < 0)
-                       goto clear_allmulti;
-       }
-
        ether_addr_copy(vlan->real_dev_addr, real_dev->dev_addr);

        if (vlan->flags & VLAN_FLAG_GVRP)
@@ -296,12 +285,6 @@ static int vlan_dev_open(struct net_device *dev)
                netif_carrier_on(dev);
        return 0;

-clear_allmulti:
-       if (dev->flags & IFF_ALLMULTI)
-               dev_set_allmulti(real_dev, -1);
-del_unicast:
-       if (!ether_addr_equal(dev->dev_addr, real_dev->dev_addr))
-               dev_uc_del(real_dev, dev->dev_addr);
 out:
        netif_carrier_off(dev);
        return err;
@@ -314,10 +297,6 @@ static int vlan_dev_stop(struct net_device *dev)

        dev_mc_unsync(real_dev, dev);
        dev_uc_unsync(real_dev, dev);
-       if (dev->flags & IFF_ALLMULTI)
-               dev_set_allmulti(real_dev, -1);
-       if (dev->flags & IFF_PROMISC)
-               dev_set_promiscuity(real_dev, -1);

        if (!ether_addr_equal(dev->dev_addr, real_dev->dev_addr))
                dev_uc_del(real_dev, dev->dev_addr);
@@ -474,12 +453,10 @@ static void vlan_dev_change_rx_flags(struct net_device *dev, int change)
 {
        struct net_device *real_dev = vlan_dev_priv(dev)->real_dev;

-       if (dev->flags & IFF_UP) {
-               if (change & IFF_ALLMULTI)
-                       dev_set_allmulti(real_dev, dev->flags & IFF_ALLMULTI ? 1 : -1);
-               if (change & IFF_PROMISC)
-                       dev_set_promiscuity(real_dev, dev->flags & IFF_PROMISC ? 1 : -1);
-       }
+       if (change & IFF_ALLMULTI)
+               dev_set_allmulti(real_dev, dev->flags & IFF_ALLMULTI ? 1 : -1);
+       if (change & IFF_PROMISC)
+               dev_set_promiscuity(real_dev, dev->flags & IFF_PROMISC ? 1 : -1);
 }

 static void vlan_dev_set_rx_mode(struct net_device *vlan_dev)
```