author	Szymon Heidrich <szymon.heidrich@gmail.com>	2023-03-18 10:25:52 +0100
committer	Greg Kroah-Hartman <gregkh@linuxfoundation.org>	2023-03-30 12:49:06 +0200
commit	83de34967473ed31d276381373713cc2869a42e5 (patch)	
tree	00e8a6eeeea7fc5d3b397744e0de7bef14ca1cd7	
parent	5fc2c4e311a9341a2b0e044ab5f33afa37b56226 (diff)	
download	linux-83de34967473ed31d276381373713cc2869a42e5.tar.gz	

diff options

context: 3

space: include

mode: unified

net: usb: lan78xx: Limit packet length to skb->len

```
[ Upstream commit 7f247f5a2c18b3f21206cdd51193df4f38e1b9f5 ]

Packet length retrieved from descriptor may be larger than
the actual socket buffer length. In such case the cloned
skb passed up the network stack will leak kernel memory contents.

Additionally prevent integer underflow when size is less than
ETH_FCS_LEN.

Fixes: 55d7de9de6c3 ("Microchip's LAN7800 family USB 2/3 to 10/100/1000 Ethernet device driver")
Signed-off-by: Szymon Heidrich <szymon.heidrich@gmail.com>
Signed-off-by: David S. Miller <davem@davemloft.net>
Signed-off-by: Sasha Levin <sashal@kernel.org>
```

Diffstat

-rw-r--r-- drivers/net/usb/lan78xx.c 18

1 files changed, 17 insertions, 1 deletions

```
diff --git a/drivers/net/usb/lan78xx.c b/drivers/net/usb/lan78xx.c
index 068488890d57be..c458c030fadf6c 100644
--- a/drivers/net/usb/lan78xx.c
+++ b/drivers/net/usb/lan78xx.c
@@ -3579,13 +3579,29 @@ static int lan78xx_rx(struct lan78xx_net *dev, struct sk_buff *skb,
                size = (rx_cmd_a & RX_CMD_A_LEN_MASK_);
                align_count = (4 - ((size + RXW_PADDING) % 4)) % 4;

+               if (unlikely(size > skb->len)) {
+                       netif_dbg(dev, rx_err, dev->net,
+                                 "size err rx_cmd_a=0x%08x\n",
+                                 rx_cmd_a);
+                       return 0;
+               }
+
                if (unlikely(rx_cmd_a & RX_CMD_A_RED_)) {
                        netif_dbg(dev, rx_err, dev->net,
                                  "Error rx_cmd_a=0x%08x", rx_cmd_a);
                } else {
-                       u32 frame_len = size - ETH_FCS_LEN;
+                       u32 frame_len;
                        struct sk_buff *skb2;
```

```
+                       if (unlikely(size < ETH_FCS_LEN)) {
+                               netif_dbg(dev, rx_err, dev->net,
+                                         "size err rx_cmd_a=0x%08x\n",
+                                         rx_cmd_a);
+                               return 0;
+                       }
+
+                       frame_len = size - ETH_FCS_LEN;
+
                        skb2 = napi_alloc_skb(&dev->napi, frame_len);
                        if (!skb2)
                                return 0;
```