
Subject: [PATCH] OOPS with NETLINK_FIB_LOOKUP netlink socket

Posted by [den](#) on Fri, 21 Dec 2007 08:58:57 GMT

[View Forum Message](#) <> [Reply to Message](#)

nl_fib_input re-reuses incoming skb to send the reply. This means that this packet will be freed twice, namely in:

- netlink_unicast_kernel
- on receive path

Use clone to send as a cure, the caller is responsible for kfree_skb on error.

Thanks to Alexey Dobryan, who originally found the problem.

Signed-off-by: Denis V. Lunev <den@openvz.org>

diff --git a/net/ipv4/fib_frontend.c b/net/ipv4/fib_frontend.c

index b03c4c6..ac6238a 100644

--- a/net/ipv4/fib_frontend.c

+++ b/net/ipv4/fib_frontend.c

@@ -832,10 +832,13 @@ static void nl_fib_input(struct sk_buff *skb)

```
    nlh = nlmsg_hdr(skb);
    if (skb->len < NLMSG_SPACE(0) || skb->len < nlh->nlmsg_len ||
-    nlh->nlmsg_len < NLMSG_LENGTH(sizeof(*frn))) {
-    kfree_skb(skb);
+    nlh->nlmsg_len < NLMSG_LENGTH(sizeof(*frn)))
    return;
- }
+
+ skb = skb_clone(skb, GFP_KERNEL);
+ if (skb == NULL)
+ return;
+ nlh = nlmsg_hdr(skb);

    frn = (struct fib_result_nl *) NLMSG_DATA(nlh);
    tb = fib_get_table(frn->tb_id_in);
```
