
Subject: [PATCH net-2.6.25 1/4][NETNS][RAW]: Make ipv[46] raw sockets lookup namespaces aware.

Posted by [Pavel Emelianov](#) on Mon, 14 Jan 2008 13:03:55 GMT

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

This requires just to pass the appropriate struct net pointer into __raw_v[46]_lookup and skip sockets that do not belong to a needed namespace.

The proper net is get from skb->dev in all the cases.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

```
---
net/ipv4/raw.c | 21 ++++++-----
net/ipv6/raw.c | 18 ++++++-----
2 files changed, 26 insertions(+), 13 deletions(-)

diff --git a/net/ipv4/raw.c b/net/ipv4/raw.c
index 747911a..a490a9d 100644
--- a/net/ipv4/raw.c
+++ b/net/ipv4/raw.c
@@ -116,16 +116,15 @@ static void raw_v4_unhash(struct sock *sk)
     raw_unhash_sk(sk, &raw_v4_hashinfo);
 }

-static struct sock *__raw_v4_lookup(struct sock *sk, unsigned short num,
-    __be32 raddr, __be32 laddr,
-    int dif)
+static struct sock *__raw_v4_lookup(struct net *net, struct sock *sk,
+    unsigned short num, __be32 raddr, __be32 laddr, int dif)
 {
     struct hlist_node *node;

     sk_for_each_from(sk, node) {
         struct inet_sock *inet = inet_sk(sk);

-        if (inet->num == num    &&
+        if (sk->sk_net == net && inet->num == num    &&
             !(inet->daddr && inet->daddr != raddr) &&
             !(inet->rcv_saddr && inet->rcv_saddr != laddr) &&
             !(sk->sk_bound_dev_if && sk->sk_bound_dev_if != dif))
@@ -169,12 +168,15 @@ static int raw_v4_input(struct sk_buff *skb, struct iphdr *iph, int hash)
     struct sock *sk;
     struct hlist_head *head;
     int delivered = 0;
+    struct net *net;

```

```

    read_lock(&raw_v4_hashinfo.lock);
    head = &raw_v4_hashinfo.ht[hash];
    if (hlist_empty(head))
        goto out;
- sk = __raw_v4_lookup(__sk_head(head), iph->protocol,
+
+ net = skb->dev->nd_net;
+ sk = __raw_v4_lookup(net, __sk_head(head), iph->protocol,
    iph->saddr, iph->daddr,
    skb->dev->ifindex);

@@ -187,7 +189,7 @@ static int raw_v4_input(struct sk_buff *skb, struct iphdr *iph, int hash)
    if (clone)
        raw_rcv(sk, clone);
}
- sk = __raw_v4_lookup(sk_next(sk), iph->protocol,
+ sk = __raw_v4_lookup(net, sk_next(sk), iph->protocol,
    iph->saddr, iph->daddr,
    skb->dev->ifindex);
}
@@ -273,6 +275,7 @@ void raw_icmp_error(struct sk_buff *skb, int protocol, u32 info)
    int hash;
    struct sock *raw_sk;
    struct iphdr *iph;
+ struct net *net;

    hash = protocol & (RAW_HTABLE_SIZE - 1);

@@ -280,8 +283,10 @@ void raw_icmp_error(struct sk_buff *skb, int protocol, u32 info)
    raw_sk = sk_head(&raw_v4_hashinfo.ht[hash]);
    if (raw_sk != NULL) {
        iph = (struct iphdr *)skb->data;
- while ((raw_sk = __raw_v4_lookup(raw_sk, protocol, iph->daddr,
-     iph->saddr,
+ net = skb->dev->nd_net;
+
+ while ((raw_sk = __raw_v4_lookup(net, raw_sk, protocol,
+     iph->daddr, iph->saddr,
        skb->dev->ifindex)) != NULL) {
        raw_err(raw_sk, skb, info);
        raw_sk = sk_next(raw_sk);
diff --git a/net/ipv6/raw.c b/net/ipv6/raw.c
index cb0b110..6f20086 100644
--- a/net/ipv6/raw.c
+++ b/net/ipv6/raw.c
@@ -76,8 +76,9 @@ static void raw_v6_unhash(struct sock *sk)
}

```

```

-static struct sock *__raw_v6_lookup(struct sock *sk, unsigned short num,
- struct in6_addr *loc_addr, struct in6_addr *rmt_addr, int dif)
+static struct sock *__raw_v6_lookup(struct net *net, struct sock *sk,
+ unsigned short num, struct in6_addr *loc_addr,
+ struct in6_addr *rmt_addr, int dif)
{
    struct hlist_node *node;
    int is_multicast = ipv6_addr_is_multicast(loc_addr);
@@ -86,6 +87,9 @@ static struct sock *__raw_v6_lookup(struct sock *sk, unsigned short num,
    if (inet_sk(sk)->num == num) {
        struct ipv6_pinfo *np = inet6_sk(sk);

+    if (sk->sk_net != net)
+        continue;
+
        if (!ipv6_addr_any(&np->daddr) &&
            !ipv6_addr_equal(&np->daddr, rmt_addr))
            continue;
@@ -165,6 +169,7 @@ static int ipv6_raw_deliver(struct sk_buff *skb, int nexthdr)
    struct sock *sk;
    int delivered = 0;
    __u8 hash;
+ struct net *net;

    saddr = &ipv6_hdr(skb)->saddr;
    daddr = saddr + 1;
@@ -182,7 +187,8 @@ static int ipv6_raw_deliver(struct sk_buff *skb, int nexthdr)
    if (sk == NULL)
        goto out;

-    sk = __raw_v6_lookup(sk, nexthdr, daddr, saddr, IP6CB(skb)->iif);
+    sk = __raw_v6_lookup(net, sk, nexthdr, daddr, saddr, IP6CB(skb)->iif);
+    net = skb->dev->nd_net;

    while (sk) {
        int filtered;
@@ -225,7 +231,7 @@ static int ipv6_raw_deliver(struct sk_buff *skb, int nexthdr)
        rawv6_rcv(sk, clone);
    }
}
-    sk = __raw_v6_lookup(sk_next(sk), nexthdr, daddr, saddr,
+    sk = __raw_v6_lookup(net, sk_next(sk), nexthdr, daddr, saddr,
        IP6CB(skb)->iif);
}
out:
@@ -359,6 +365,7 @@ void raw6_icmp_error(struct sk_buff *skb, int nexthdr,
    struct sock *sk;

```

```

int hash;
struct in6_addr *saddr, *daddr;
+ struct net *net;

hash = nexthdr & (RAW_HTABLE_SIZE - 1);

@@ -367,8 +374,9 @@ void raw6_icmp_error(struct sk_buff *skb, int nexthdr,
if (sk != NULL) {
saddr = &ipv6_hdr(skb)->saddr;
daddr = &ipv6_hdr(skb)->daddr;
+ net = skb->dev->nd_net;

- while ((sk = __raw_v6_lookup(sk, nexthdr, saddr, daddr,
+ while ((sk = __raw_v6_lookup(net, sk, nexthdr, saddr, daddr,
IP6CB(skb)->iif))) {
rawv6_err(sk, skb, NULL, type, code,
inner_offset, info);
--
1.5.3.4

```

Subject: Re: [PATCH net-2.6.25 1/4][NETNS][RAW]: Make ipv[46] raw sockets lookup namespaces aware.

Posted by [davem](#) on Mon, 14 Jan 2008 13:36:58 GMT

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

From: Pavel Emelyanov <xemul@openvz.org>

Date: Mon, 14 Jan 2008 16:03:55 +0300

> This requires just to pass the appropriate struct net pointer
> into __raw_v[46]_lookup and skip sockets that do not belong
> to a needed namespace.

>

> The proper net is get from skb->dev in all the cases.

>

> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.
