
Subject: [PATCH 2.6.25 4/6] net: Make AF_PACKET handle multiple network namespaces

Posted by [den](#) on Thu, 15 Nov 2007 15:59:54 GMT

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

This is done by making packet_sklist_lock and packet_sklist per network namespace and adding an additional filter condition on received packets to ensure they came from the proper network namespace.

Changes from v1:

- prohibit to call inet_dgram_ops.ioctl in other than init_net

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

Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

```
include/net/net_namespace.h | 4 +
net/packet/af_packet.c      | 131 ++++++
2 files changed, 89 insertions(+), 46 deletions(-)
```

diff --git a/include/net/net_namespace.h b/include/net/net_namespace.h

index 90802a6..4d0d634 100644

--- a/include/net/net_namespace.h

+++ b/include/net/net_namespace.h

```
@ @ -32,6 +32,10 @ @ struct net {
    struct hlist_head *dev_index_head;
```

```
    struct sock *rtnl; /* rtnetlink socket */
```

```
+
```

```
+ /* List of all packet sockets. */
```

```
+ rwlock_t packet_sklist_lock;
```

```
+ struct hlist_head packet_sklist;
```

```
};
```

```
#ifdef CONFIG_NET
```

diff --git a/net/packet/af_packet.c b/net/packet/af_packet.c

index 8a7807d..45e3cbc 100644

--- a/net/packet/af_packet.c

+++ b/net/packet/af_packet.c

```
@ @ -135,10 +135,6 @ @ dev->hard_header == NULL (If header is added by device, we cannot control it)
```

```
    packet classifier depends on it.
```

```
*/
```

```
/* List of all packet sockets. */
```

```
-static HLIST_HEAD(packet_sklist);
```

```
-static DEFINE_RWLOCK(packet_sklist_lock);
```

```
-
```

```
/* Private packet socket structures. */
```

```
struct packet_mclist
```

```
@@ -246,9 +242,6 @@ static int packet_rcv_spkt(struct sk_buff *skb, struct net_device *dev, struct
```

```
    struct sock *sk;  
    struct sockaddr_pkt *spkt;
```

```
- if (dev->nd_net != &init_net)
```

```
- goto out;
```

```
-
```

```
/*
```

```
 * When we registered the protocol we saved the socket in the data
```

```
 * field for just this event.
```

```
@@ -270,6 +263,9 @@ static int packet_rcv_spkt(struct sk_buff *skb, struct net_device *dev, struct
```

```
    if (skb->pkt_type == PACKET_LOOPBACK)  
        goto out;
```

```
+ if (dev->nd_net != sk->sk_net)
```

```
+ goto out;
```

```
+
```

```
    if ((skb = skb_share_check(skb, GFP_ATOMIC)) == NULL)  
        goto oom;
```

```
@@ -341,7 +337,7 @@ static int packet_sendmsg_spkt(struct kiocb *iocb, struct socket *sock,  
    */
```

```
    saddr->spkt_device[13] = 0;
```

```
- dev = dev_get_by_name(&init_net, saddr->spkt_device);
```

```
+ dev = dev_get_by_name(sk->sk_net, saddr->spkt_device);
```

```
    err = -ENODEV;
```

```
    if (dev == NULL)
```

```
        goto out_unlock;
```

```
@@ -449,15 +445,15 @@ static int packet_rcv(struct sk_buff *skb, struct net_device *dev, struct packet
```

```
    int skb_len = skb->len;
```

```
    unsigned int snaplen, res;
```

```
- if (dev->nd_net != &init_net)
```

```
- goto drop;
```

```
-
```

```
    if (skb->pkt_type == PACKET_LOOPBACK)  
        goto drop;
```

```
    sk = pt->af_packet_priv;
```

```
    po = pkt_sk(sk);
```

```

+ if (dev->nd_net != sk->sk_net)
+ goto drop;
+
+ skb->dev = dev;

+ if (dev->header_ops) {
@@ -566,15 +562,15 @@ static int tpacket_rcv(struct sk_buff *skb, struct net_device *dev, struct
packe
+ struct sk_buff *copy_skb = NULL;
+ struct timeval tv;

- if (dev->nd_net != &init_net)
- goto drop;
-
+ if (skb->pkt_type == PACKET_LOOPBACK)
+ goto drop;

+ sk = pt->af_packet_priv;
+ po = pkt_sk(sk);

+ if (dev->nd_net != sk->sk_net)
+ goto drop;
+
+ if (dev->header_ops) {
+ if (sk->sk_type != SOCK_DGRAM)
+ skb_push(skb, skb->data - skb->mac_header(skb));
@@ -732,7 +728,7 @@ static int packet_sendmsg(struct kiocb *iocb, struct socket *sock,
+ }

- dev = dev_get_by_index(&init_net, ifindex);
+ dev = dev_get_by_index(sk->sk_net, ifindex);
+ err = -ENXIO;
+ if (dev == NULL)
+ goto out_unlock;
@@ -799,15 +795,17 @@ static int packet_release(struct socket *sock)
+ {
+ struct sock *sk = sock->sk;
+ struct packet_sock *po;
+ struct net *net;

+ if (!sk)
+ return 0;

+ net = sk->sk_net;
+ po = pkt_sk(sk);

- write_lock_bh(&packet_sklist_lock);

```

```

+ write_lock_bh(&net->packet_sklist_lock);
  sk_del_node_init(sk);
- write_unlock_bh(&packet_sklist_lock);
+ write_unlock_bh(&net->packet_sklist_lock);

/*
 * Unhook packet receive handler.
@@ -916,7 +914,7 @@ static int packet_bind_spkt(struct socket *sock, struct sockaddr *uaddr,
int add
  return -EINVAL;
  strcpy(name,uaddr->sa_data,sizeof(name));

- dev = dev_get_by_name(&init_net, name);
+ dev = dev_get_by_name(sk->sk_net, name);
  if (dev) {
    err = packet_do_bind(sk, dev, pkt_sk(sk)->num);
    dev_put(dev);
@@ -943,7 +941,7 @@ static int packet_bind(struct socket *sock, struct sockaddr *uaddr, int
addr_len

  if (sll->sll_ifindex) {
    err = -ENODEV;
- dev = dev_get_by_index(&init_net, sll->sll_ifindex);
+ dev = dev_get_by_index(sk->sk_net, sll->sll_ifindex);
    if (dev == NULL)
      goto out;
  }
@@ -972,9 +970,6 @@ static int packet_create(struct net *net, struct socket *sock, int protocol)
__be16 proto = (__force __be16)protocol; /* weird, but documented */
int err;

- if (net != &init_net)
- return -EAFNOSUPPORT;
-
  if (!capable(CAP_NET_RAW))
    return -EPERM;
  if (sock->type != SOCK_DGRAM && sock->type != SOCK_RAW &&
@@ -1020,9 +1015,9 @@ static int packet_create(struct net *net, struct socket *sock, int
protocol)
  po->running = 1;
}

- write_lock_bh(&packet_sklist_lock);
- sk_add_node(sk, &packet_sklist);
- write_unlock_bh(&packet_sklist_lock);
+ write_lock_bh(&net->packet_sklist_lock);
+ sk_add_node(sk, &net->packet_sklist);
+ write_unlock_bh(&net->packet_sklist_lock);

```

```

    return(0);
out:
    return err;
@@ -1140,7 +1135,7 @@ static int packet_getname_spkt(struct socket *sock, struct sockaddr
*uaddr,
    return -EOPNOTSUPP;

    uaddr->sa_family = AF_PACKET;
- dev = dev_get_by_index(&init_net, pkt_sk(sk)->ifindex);
+ dev = dev_get_by_index(sk->sk_net, pkt_sk(sk)->ifindex);
    if (dev) {
        strlcpy(uaddr->sa_data, dev->name, 15);
        dev_put(dev);
@@ -1165,7 +1160,7 @@ static int packet_getname(struct socket *sock, struct sockaddr *uaddr,
    sll->sll_family = AF_PACKET;
    sll->sll_ifindex = po->ifindex;
    sll->sll_protocol = po->num;
- dev = dev_get_by_index(&init_net, po->ifindex);
+ dev = dev_get_by_index(sk->sk_net, po->ifindex);
    if (dev) {
        sll->sll_hatype = dev->type;
        sll->sll_halen = dev->addr_len;
@@ -1217,7 +1212,7 @@ static int packet_mc_add(struct sock *sk, struct packet_mreq_max
*mreq)
    rtnl_lock();

    err = -ENODEV;
- dev = __dev_get_by_index(&init_net, mreq->mr_ifindex);
+ dev = __dev_get_by_index(sk->sk_net, mreq->mr_ifindex);
    if (!dev)
        goto done;

@@ -1271,7 +1266,7 @@ static int packet_mc_drop(struct sock *sk, struct packet_mreq_max
*mreq)
    if (--ml->count == 0) {
        struct net_device *dev;
        *mlp = ml->next;
- dev = dev_get_by_index(&init_net, ml->ifindex);
+ dev = dev_get_by_index(sk->sk_net, ml->ifindex);
        if (dev) {
            packet_dev_mc(dev, ml, -1);
            dev_put(dev);
@@ -1299,7 +1294,7 @@ static void packet_flush_mclist(struct sock *sk)
    struct net_device *dev;

    po->mclist = ml->next;
- if ((dev = dev_get_by_index(&init_net, ml->ifindex)) != NULL) {
+ if ((dev = dev_get_by_index(sk->sk_net, ml->ifindex)) != NULL) {

```

```

    packet_dev_mc(dev, ml, -1);
    dev_put(dev);
}
@@ -1455,12 +1450,10 @@ static int packet_notifier(struct notifier_block *this, unsigned long
msg, void
    struct sock *sk;
    struct hlist_node *node;
    struct net_device *dev = data;
+ struct net *net = dev->nd_net;

- if (dev->nd_net != &init_net)
- return NOTIFY_DONE;
-
- read_lock(&packet_sklist_lock);
- sk_for_each(sk, node, &packet_sklist) {
+ read_lock(&net->packet_sklist_lock);
+ sk_for_each(sk, node, &net->packet_sklist) {
    struct packet_sock *po = pkt_sk(sk);

    switch (msg) {
@@ -1499,7 +1492,7 @@ static int packet_notifier(struct notifier_block *this, unsigned long msg,
void
    break;
    }
}
- read_unlock(&packet_sklist_lock);
+ read_unlock(&net->packet_sklist_lock);
return NOTIFY_DONE;
}

@@ -1547,6 +1540,8 @@ static int packet_ioctl(struct socket *sock, unsigned int cmd,
    case SIOCGIFDSTADDR:
    case SIOCSIFDSTADDR:
    case SIOCSIFFLAGS:
+ if (sk->sk_net != &init_net)
+ return -ENOIOCTLCMD;
    return inet_dgram_ops.ioctl(sock, cmd, arg);
#endif

@@ -1862,12 +1857,12 @@ static struct notifier_block packet_netdev_notifier = {
};

#ifdef CONFIG_PROC_FS
-static inline struct sock *packet_seq_idx(loff_t off)
+static inline struct sock *packet_seq_idx(struct net *net, loff_t off)
{
    struct sock *s;
    struct hlist_node *node;

```

```

- sk_for_each(s, node, &packet_sklist) {
+ sk_for_each(s, node, &net->packet_sklist) {
    if (!loff--)
        return s;
}
@@ -1876,21 +1871,24 @@ static inline struct sock *packet_seq_idx(loff_t off)

static void *packet_seq_start(struct seq_file *seq, loff_t *pos)
{
- read_lock(&packet_sklist_lock);
- return *pos ? packet_seq_idx(*pos - 1) : SEQ_START_TOKEN;
+ struct net *net = seq->private;
+ read_lock(&net->packet_sklist_lock);
+ return *pos ? packet_seq_idx(net, *pos - 1) : SEQ_START_TOKEN;
}

static void *packet_seq_next(struct seq_file *seq, void *v, loff_t *pos)
{
+ struct net *net = seq->private;
++*pos;
return (v == SEQ_START_TOKEN)
- ? sk_head(&packet_sklist)
+ ? sk_head(&net->packet_sklist)
: sk_next((struct sock*)v);
}

static void packet_seq_stop(struct seq_file *seq, void *v)
{
- read_unlock(&packet_sklist_lock);
+ struct net *net = seq->private;
+ read_unlock(&net->packet_sklist_lock);
}

static int packet_seq_show(struct seq_file *seq, void *v)
@@ -1926,7 +1924,26 @@ static const struct seq_operations packet_seq_ops = {

static int packet_seq_open(struct inode *inode, struct file *file)
{
- return seq_open(file, &packet_seq_ops);
+ struct seq_file *seq;
+ int res;
+ res = seq_open(file, &packet_seq_ops);
+ if (!res) {
+ seq = file->private_data;
+ seq->private = get_proc_net(inode);
+ if (!seq->private) {
+ seq_release(inode, file);

```

```

+ res = -ENXIO;
+ }
+ }
+ return res;
+}
+
+static int packet_seq_release(struct inode *inode, struct file *file)
+{
+ struct seq_file *seq= file->private_data;
+ struct net *net = seq->private;
+ put_net(net);
+ return seq_release(inode, file);
+}

static const struct file_operations packet_seq_fops = {
@@ -1934,15 +1951,37 @@ static const struct file_operations packet_seq_fops = {
    .open = packet_seq_open,
    .read = seq_read,
    .llseek = seq_lseek,
- .release = seq_release,
+ .release = packet_seq_release,
};

#endif

+static int packet_net_init(struct net *net)
+{
+ rwlock_init(&net->packet_sklist_lock);
+ INIT_HLIST_HEAD(&net->packet_sklist);
+
+ if (!proc_net_fops_create(net, "packet", 0, &packet_seq_fops))
+ return -ENOMEM;
+
+ return 0;
+}
+
+static void packet_net_exit(struct net *net)
+{
+ proc_net_remove(net, "packet");
+}
+
+static struct pernet_operations packet_net_ops = {
+ .init = packet_net_init,
+ .exit = packet_net_exit,
+};
+
+static void __exit packet_exit(void)

```



```

{
- proc_net_remove(&init_net, "packet");
  unregister_netdevice_notifier(&packet_netdev_notifier);
+ unregister_pernet_subsys(&packet_net_ops);
  sock_unregister(PF_PACKET);
  proto_unregister(&packet_proto);
}
@@ -1955,8 +1994,8 @@ static int __init packet_init(void)
    goto out;

    sock_register(&packet_family_ops);
+ register_pernet_subsys(&packet_net_ops);
  register_netdevice_notifier(&packet_netdev_notifier);
- proc_net_fops_create(&init_net, "packet", 0, &packet_seq_fops);
out:
  return rc;
}
--
1.5.3.rc5

```

Containers mailing list
Containers@lists.linux-foundation.org
<https://lists.linux-foundation.org/mailman/listinfo/containers>

Subject: Re: [PATCH 2.6.25 4/6] net: Make AF_PACKET handle multiple network namespaces

Posted by [davem](#) on Tue, 20 Nov 2007 06:33:50 GMT

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

From: "Denis V. Lunev" <den@openvz.org>

Date: Thu, 15 Nov 2007 18:59:54 +0300

> This is done by making packet_sklist_lock and packet_sklist per
> network namespace and adding an additional filter condition on
> received packets to ensure they came from the proper network
> namespace.

>

> Changes from v1:

> - prohibit to call inet_dgram_ops.ioctl in other than init_net

>

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

> Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>

Applied.

Containers mailing list

