
Subject: [PATCH 11/12 net-2.6.25] [NETNS]: Routing cache virtualization.

Posted by [den](#) on Wed, 23 Jan 2008 07:46:26 GMT

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

Basically, this piece looks relatively easy. Namespace is already available on the dst entry via device and the device is safe to dereference. Compare it with one of a searcher and skip entry if appropriate.

The only exception is ip_rt_frag_needed. So, add namespace parameter to it.

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

```
include/net/route.h | 2 +-
net/ipv4/icmp.c     | 2 +-
net/ipv4/route.c    | 21 ++++++++-----
3 files changed, 18 insertions(+), 7 deletions(-)
```

```
diff --git a/include/net/route.h b/include/net/route.h
```

```
index 1985d82..4eabf00 100644
```

```
--- a/include/net/route.h
```

```
+++ b/include/net/route.h
```

```
@@ -115,7 +115,7 @@ extern int __ip_route_output_key(struct net *, struct rtable **, const
struct f
```

```
extern int ip_route_output_key(struct net *, struct rtable **, struct flowi *flp);
```

```
extern int ip_route_output_flow(struct net *, struct rtable **rp, struct flowi *flp, struct sock *sk, int
flags);
```

```
extern int ip_route_input(struct sk_buff *, __be32 dst, __be32 src, u8 tos, struct net_device
*devin);
```

```
-extern unsigned short ip_rt_frag_needed(struct iphdr *iph, unsigned short new_mtu);
```

```
+extern unsigned short ip_rt_frag_needed(struct net *net, struct iphdr *iph, unsigned short
new_mtu);
```

```
extern void ip_rt_send_redirect(struct sk_buff *skb);
```

```
extern unsigned inet_addr_type(struct net *net, __be32 addr);
```

```
diff --git a/net/ipv4/icmp.c b/net/ipv4/icmp.c
```

```
index c04aac5..052b278 100644
```

```
--- a/net/ipv4/icmp.c
```

```
+++ b/net/ipv4/icmp.c
```

```
@@ -696,7 +696,7 @@ static void icmp_unreach(struct sk_buff *skb)
```

```
    "and DF set.\n",
```

```
    NIPQUAD(iph->daddr));
```

```
    } else {
```

```
-    info = ip_rt_frag_needed(iph,
```

```
+    info = ip_rt_frag_needed(&init_net, iph,
    ntohs(icmp->un.frag.mtu));
```

```
    if (!info)
```

```
        goto out;
```

```
diff --git a/net/ipv4/route.c b/net/ipv4/route.c
```

index 87076c6..07dd295 100644

--- a/net/ipv4/route.c

+++ b/net/ipv4/route.c

```
@@ -648,6 +648,11 @@ static inline int compare_keys(struct flowi *fl1, struct flowi *fl2)
    (fl1->iif ^ fl2->iif) == 0;
}
```

```
+static inline int compare_netns(struct rtable *rt1, struct rtable *rt2)
```

```
+{
```

```
+ return rt1->u.dst.dev->nd_net == rt2->u.dst.dev->nd_net;
```

```
+}
```

```
+
```

```
/*
```

```
 * Perform a full scan of hash table and free all entries.
```

```
 * Can be called by a softirq or a process.
```

```
@@ -961,7 +966,7 @@ restart:
```

```
    spin_lock_bh(rt_hash_lock_addr(hash));
```

```
    while ((rth = *rthp) != NULL) {
```

```
- if (compare_keys(&rth->fl, &rt->fl)) {
```

```
+ if (compare_keys(&rth->fl, &rt->fl) && compare_netns(rth, rt)) {
```

```
    /* Put it first */
```

```
    *rthp = rth->u.dst.rt_next;
```

```
    /*
```

```
@@ -1415,7 +1420,8 @@ static __inline__ unsigned short guess_mtu(unsigned short old_mtu)
    return 68;
```

```
}
```

```
-unsigned short ip_rt_frag_needed(struct iphdr *iph, unsigned short new_mtu)
```

```
+unsigned short ip_rt_frag_needed(struct net *net, struct iphdr *iph,
```

```
+    unsigned short new_mtu)
```

```
{
```

```
    int i;
```

```
    unsigned short old_mtu = ntohs(iph->tot_len);
```

```
@@ -1438,7 +1444,8 @@ unsigned short ip_rt_frag_needed(struct iphdr *iph, unsigned short
new_mtu)
```

```
    rth->rt_dst == daddr &&
```

```
    rth->rt_src == iph->saddr &&
```

```
    rth->fl.iif == 0 &&
```

```
-    !(dst_metric_locked(&rth->u.dst, RTAX_MTU))) {
```

```
+    !(dst_metric_locked(&rth->u.dst, RTAX_MTU)) &&
```

```
+    rth->u.dst.dev->nd_net == net) {
```

```
    unsigned short mtu = new_mtu;
```

```
    if (new_mtu < 68 || new_mtu >= old_mtu) {
```

```
@@ -2049,7 +2056,9 @@ int ip_route_input(struct sk_buff *skb, __be32 daddr, __be32 saddr,
    struct rtable * rth;
```

```
    unsigned hash;
```

```

    int iif = dev->ifindex;
+ struct net *net;

+ net = skb->dev->nd_net;
  tos &= IPTOS_RT_MASK;
  hash = rt_hash(daddr, saddr, iif);

@@ -2061,7 +2070,8 @@ int ip_route_input(struct sk_buff *skb, __be32 daddr, __be32 saddr,
    rth->fl.iif == iif &&
    rth->fl.oif == 0 &&
    rth->fl.mark == skb->mark &&
-   rth->fl.fl4_tos == tos) {
+   rth->fl.fl4_tos == tos &&
+   rth->u.dst.dev->nd_net == net) {
    dst_use(&rth->u.dst, jiffies);
    RT_CACHE_STAT_INC(in_hit);
    rcu_read_unlock();
@@ -2459,7 +2469,8 @@ int __ip_route_output_key(struct net *net, struct rtable **rp,
    rth->fl.oif == flp->oif &&
    rth->fl.mark == flp->mark &&
    !((rth->fl.fl4_tos ^ flp->fl4_tos) &
-   (IPTOS_RT_MASK | RTO_ONLINK))) {
+   (IPTOS_RT_MASK | RTO_ONLINK)) &&
+   rth->u.dst.dev->nd_net == net) {
    dst_use(&rth->u.dst, jiffies);
    RT_CACHE_STAT_INC(out_hit);
    rcu_read_unlock_bh();
--

```

1.5.3.rc5
