Subject: Re: [patch 2/2][NETNS][RFD] use dst_entries to retrieve the network namespace pointer
Posted by den on Tue, 11 Dec 2007 17:00:25 GMT
View Forum Message <> Reply to Message

Daniel Lezcano wrote:
> The objective we have is to remove the fl_net field from the struct flowi.
>
> In the previous patch, we make the network namespace to go up to the dst_entry.
> This patch makes an example in how we can use the dst_entry/rtable combined with
> a container_of to retrieve the network namespace when we have the flowi parameter.
>
> One restriction is we should pass always a flowi parameter coming from a struct
> dst_entry.
this is an obfuscation IMHO. You use rtable instead of flowi, so
- rtable obtain flowi meaning and keeps original meaning
- stack usage in increased

So, for the case, I think it is better to have an additional parameter
on IP route output path. (There is a device on the input one).

Regards,
Den

>
> ---
> include/net/ip_fib.h | 3 -
> net/ipv4/route.c | 111 ++++++++++++++++++++++++++++++----------------
> 2 files changed, 60 insertions(+), 54 deletions(-)
>
> Index: linux-2.6-netns/net/ipv4/route.c
> ===
> --- linux-2.6-netns.orig/net/ipv4/route.c
> +++ linux-2.6-netns/net/ipv4/route.c
> @@ -1788,15 +1788,18 @@ static int ip_route_input_slow(struct sk
> struct net *net = dev->nd_net;
> struct fib_result res;
> struct in_device *in_dev = in_dev_get(dev);
> - struct flowi fl = { .fl_net = net,
> - .nl_u = { .ip4_u =
> - { .daddr = daddr,
> - .saddr = saddr,
> - .tos = tos,
> - .scope = RT_SCOPE_UNIVERSE,
> - } },
> - .mark = skb->mark,
> - .iif = dev->ifindex };
> + struct rtable rt = {

```
> +  .u.dst.net = net,
> +  .fl = { .fl_net = net,
> +    .nl_u = { .ip4_u =
> +      { .daddr = daddr,
> +        .saddr = saddr,
> +        .tos = tos,
> +        .scope = RT_SCOPE_UNIVERSE,
> +      } },
> +    .mark = skb->mark,
> +    .iif = dev->ifindex },
> + };
>    unsigned flags = 0;
>    u32  itag = 0;
>    struct rtable * rth;
> @@ -1832,7 +1835,7 @@ static int ip_route_input_slow(struct sk
>    /*
>     * Now we are ready to route packet.
>     */
> - if ((err = fib_lookup(&fl, &res)) != 0) {
> + if ((err = fib_lookup(&rt.fl, &res)) != 0) {
>     if (!IN_DEV_FORWARD(in_dev))
>      goto e_hostunreach;
>     goto no_route;
> @@ -1862,7 +1865,7 @@ static int ip_route_input_slow(struct sk
>    if (res.type != RTN_UNICAST)
>     goto martian_destination;
>
> - err = ip_mkroute_input(skb, &res, &fl, in_dev, daddr, saddr, tos);
> + err = ip_mkroute_input(skb, &res, &rt.fl, in_dev, daddr, saddr, tos);
>   done:
>    in_dev_put(in_dev);
>    if (free_res)
> @@ -1898,7 +1901,6 @@ local_input:
>    rth->u.dst.flags= DST_HOST;
>    if (IN_DEV_CONF_GET(in_dev, NOPOLICY))
>     rth->u.dst.flags |= DST_NOPOLICY;
> - rth->fl.fl_net = net;
>    rth->fl.fl4_dst = daddr;
>    rth->rt_dst = daddr;
>    rth->fl.fl4_tos = tos;
> @@ -1923,7 +1925,7 @@ local_input:
>     rth->rt_flags  &= ~RTCF_LOCAL;
>    }
>    rth->rt_type = res.type;
> - hash = rt_hash(daddr, saddr, fl.iif);
> + hash = rt_hash(daddr, saddr, rt.fl.iif);
>    err = rt_intern_hash(hash, rth, (struct rtable**)&skb->dst);
>    goto done;
```

```
>
> @@ -2173,18 +2175,21 @@ static int ip_route_output_slow(struct r
>  {
>   u32 tos = RT_FL_TOS(oldflp);
>   struct net *net = oldflp->fl_net;
> - struct flowi fl = { .fl_net = net,
> -       .nl_u = { .ip4_u =
> -         { .daddr = oldflp->fl4_dst,
> -     .saddr = oldflp->fl4_src,
> -     .tos = tos & IPTOS_RT_MASK,
> -     .scope = ((tos & RTO_ONLINK) ?
> -        RT_SCOPE_LINK :
> -        RT_SCOPE_UNIVERSE),
> -          } },
> -       .mark = oldflp->mark,
> -       .iif = net->loopback_dev->ifindex,
> -       .oif = oldflp->oif };
> + struct rtable rt = {
> +  .u.dst.net = net,
> +  .fl = { .fl_net = net,
> +    .nl_u = { .ip4_u =
> +      { .daddr = oldflp->fl4_dst,
> +        .saddr = oldflp->fl4_src,
> +        .tos = tos & IPTOS_RT_MASK,
> +        .scope = ((tos & RTO_ONLINK) ?
> +           RT_SCOPE_LINK :
> +           RT_SCOPE_UNIVERSE),
> +      } },
> +    .mark = oldflp->mark,
> +    .iif = net->loopback_dev->ifindex,
> +    .oif = oldflp->oif },
> + };
>   struct fib_result res;
>   unsigned flags = 0;
>   struct net_device *dev_out = NULL;
> @@ -2234,7 +2239,7 @@ static int ip_route_output_slow(struct r
>         Luckily, this hack is good workaround.
>       */
>
> -    fl.oif = dev_out->ifindex;
> +    rt.fl.oif = dev_out->ifindex;
>     goto make_route;
>    }
>    if (dev_out)
> @@ -2256,36 +2261,36 @@ static int ip_route_output_slow(struct r
>    }
>
>    if (LOCAL_MCAST(oldflp->fl4_dst) || oldflp->fl4_dst == htonl(0xFFFFFFFF)) {
```

```
> -   if (!fl.fl4_src)
> -    fl.fl4_src = inet_select_addr(dev_out, 0,
> -               RT_SCOPE_LINK);
> +   if (!rt.fl.fl4_src)
> +    rt.fl.fl4_src = inet_select_addr(dev_out, 0,
> +           RT_SCOPE_LINK);
>     goto make_route;
>    }
> -  if (!fl.fl4_src) {
> +  if (!rt.fl.fl4_src) {
>     if (MULTICAST(oldflp->fl4_dst))
> -    fl.fl4_src = inet_select_addr(dev_out, 0,
> -               fl.fl4_scope);
> +    rt.fl.fl4_src = inet_select_addr(dev_out, 0,
> +           rt.fl.fl4_scope);
>     else if (!oldflp->fl4_dst)
> -    fl.fl4_src = inet_select_addr(dev_out, 0,
> -               RT_SCOPE_HOST);
> +    rt.fl.fl4_src = inet_select_addr(dev_out, 0,
> +           RT_SCOPE_HOST);
>    }
>   }
>
> - if (!fl.fl4_dst) {
> -  fl.fl4_dst = fl.fl4_src;
> -  if (!fl.fl4_dst)
> -   fl.fl4_dst = fl.fl4_src = htonl(INADDR_LOOPBACK);
> + if (!rt.fl.fl4_dst) {
> +  rt.fl.fl4_dst = rt.fl.fl4_src;
> +  if (!rt.fl.fl4_dst)
> +   rt.fl.fl4_dst = rt.fl.fl4_src = htonl(INADDR_LOOPBACK);
>    if (dev_out)
>     dev_put(dev_out);
>    dev_out = net->loopback_dev;
>    dev_hold(dev_out);
> -  fl.oif = net->loopback_dev->ifindex;
> +  rt.fl.oif = net->loopback_dev->ifindex;
>    res.type = RTN_LOCAL;
>    flags |= RTCF_LOCAL;
>    goto make_route;
>   }
>
> - if (fib_lookup(&fl, &res)) {
> + if (fib_lookup(&rt.fl, &res)) {
>    res.fi = NULL;
>    if (oldflp->oif) {
>     /* Apparently, routing tables are wrong. Assume,
> @@ -2306,9 +2311,9 @@ static int ip_route_output_slow(struct r
```

```
>        likely IPv6, but we do not.
>      */
>
> -   if (fl.fl4_src == 0)
> -    fl.fl4_src = inet_select_addr(dev_out, 0,
> -               RT_SCOPE_LINK);
> +   if (rt.fl.fl4_src == 0)
> +    rt.fl.fl4_src = inet_select_addr(dev_out, 0,
> +           RT_SCOPE_LINK);
>      res.type = RTN_UNICAST;
>      goto make_route;
>    }
> @@ -2320,13 +2325,13 @@ static int ip_route_output_slow(struct r
>   free_res = 1;
>
>   if (res.type == RTN_LOCAL) {
> -  if (!fl.fl4_src)
> -   fl.fl4_src = fl.fl4_dst;
> +  if (!rt.fl.fl4_src)
> +   rt.fl.fl4_src = rt.fl.fl4_dst;
>     if (dev_out)
>      dev_put(dev_out);
>     dev_out = net->loopback_dev;
>     dev_hold(dev_out);
> -  fl.oif = dev_out->ifindex;
> +  rt.fl.oif = dev_out->ifindex;
>     if (res.fi)
>      fib_info_put(res.fi);
>     res.fi = NULL;
> @@ -2336,24 +2341,24 @@ static int ip_route_output_slow(struct r
>
>  #ifdef CONFIG_IP_ROUTE_MULTIPATH
>   if (res.fi->fib_nhs > 1 && fl.oif == 0)
> -  fib_select_multipath(&fl, &res);
> +  fib_select_multipath(&rt.fl, &res);
>   else
>  #endif
> - if (!res.prefixlen && res.type == RTN_UNICAST && !fl.oif)
> -  fib_select_default(&fl, &res);
> + if (!res.prefixlen && res.type == RTN_UNICAST && !rt.fl.oif)
> +  fib_select_default(&rt.fl, &res);
>
> - if (!fl.fl4_src)
> -  fl.fl4_src = FIB_RES_PREFSRC(res);
> + if (!rt.fl.fl4_src)
> +  rt.fl.fl4_src = FIB_RES_PREFSRC(res);
>
>   if (dev_out)
```

>   dev_put(dev_out);
>   dev_out = FIB_RES_DEV(res);
>   dev_hold(dev_out);
> - fl.oif = dev_out->ifindex;
> + rt.fl.oif = dev_out->ifindex;
>
>
> make_route:
> - err = ip_mkroute_output(rp, &res, &fl, oldflp, dev_out, flags);
> + err = ip_mkroute_output(rp, &res, &rt.fl, oldflp, dev_out, flags);
>
>
>   if (free_res)
> Index: linux-2.6-netns/include/net/ip_fib.h
> ===
> --- linux-2.6-netns.orig/include/net/ip_fib.h
> +++ linux-2.6-netns/include/net/ip_fib.h
> @@ -173,7 +173,8 @@ static inline struct fib_table *fib_new_
>
> static inline int fib_lookup(const struct flowi *flp, struct fib_result *res)
> {
> - struct net *net = flp->fl_net;
> + struct rtable *rt = container_of(flp, struct rtable, fl);
> + struct net *net = rt->u.dst.net;
>   struct fib_table *local_table = net->ip_fib_local_table;
>   struct fib_table *main_table = net->ip_fib_main_table;
>   if (local_table->tb_lookup(local_table, flp, res) &&
>